

Περιεχόμενα

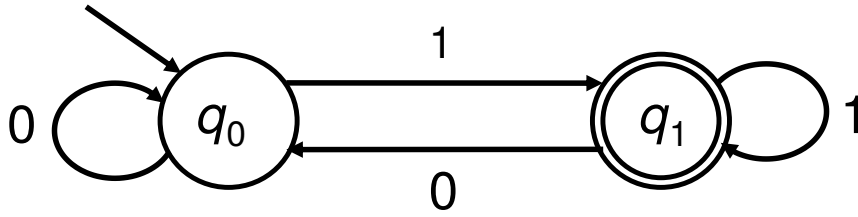
1 Υπολογισμότητα

- Ιστορία - Εισαγωγή
- Μαθηματικό Υπόβαθρο
- LOOP: Μια απλή γλώσσα προγραμματισμού
- LOOP-υπολογίσιμες και πρωταρχικές αναδρομικές συναρτήσεις
- Σταθερά σημεία
- Προγράμματα WHILE και μερικές αναδρομικές συναρτήσεις
- Μηχανές Turing και άλλα υπολογιστικά μοντέλα
- Σχήματα McCarthy. Στρατηγικές Υπολογισμού
- Θέση Church-Turing. Κανονική μορφή Kleene
- Μη επιλυσιμότητα. Θεωρία αναδρομικών συναρτήσεων
- Αναδρομικά και αναδρομικά αριθμήσιμα σύνολα
- Μαντεία. Σχετική Υπολογισμότητα. Αριθμητικές Σχέσεις

Αυτόματα και τυπικές γλώσσες

- **Τυπικές γλώσσες**: χρησιμοποιούνται για την κωδικοποίηση υπολογιστικών προβλημάτων αλλά και τον ορισμό 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 και **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

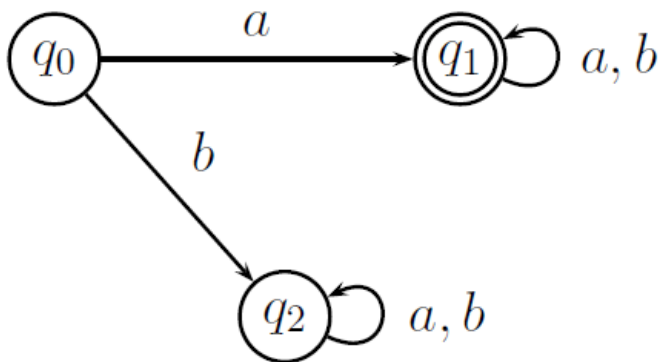
$$L_1 := \{w \in \{a, b\}^* \mid w \text{ αρχίζει με 'a'}\}$$

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

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

Παράδειγμα: DFA για L_1

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



	a	b
q ₀	q ₁	q ₂
q ₁	q ₁	q ₁
q ₂	q ₂	q ₂

εκτέλεση με είσοδο $abba$:

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

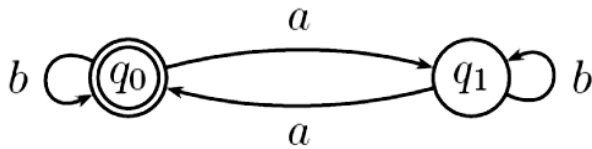
ΑΠΟΔΟΧΗ

Τυπικός 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. $Q = \{q_0, q_1, q_2\}$
 - Σ : πεπερασμένο αλφάβητο εισόδου ($\Sigma \cap Q = \emptyset$), π.χ. $\Sigma = \{a, b\}$
 - $\delta : Q \times \Sigma \rightarrow Q$: συνάρτηση μετάβασης, π.χ. $\delta(q_0, a) = q_1$
 - $q_0 \in Q$: αρχική κατάσταση
 - $F \subseteq Q$: σύνολο τελικών καταστάσεων (αποδοχής), π.χ. $F = \{q_1\}$

Γλώσσα με DFA και γλώσσα χωρίς DFA

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



	a	b
q0	q1	q0
q1	q0	q1

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

Αποδεικνύεται ότι **δεν υπάρχει** DFA που να αναγνωρίζει την L_3 !
(χρειάζεται μνήμη με μέγεθος που εξαρτάται από την είσοδο)

Αποδοχή DFA: τυπικοί ορισμοί

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

Η επεκτεταμένη δ 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 δ (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

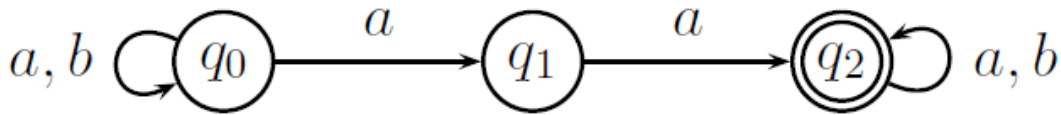
- **Ντετερμινιστικά αυτόματα:** για κάθε συνδυασμό κατάστασης / συμβόλου εισόδου υπάρχει **μοναδική** επόμενη κατάσταση
- **Μη-ντετερμινιστικά αυτόματα:**
 - για κάθε συνδυασμό κατάστασης / συμβόλου εισόδου υπάρχει **επιλογή** από σύνολο δυνατών επόμενων καταστάσεων
 - αποδοχή αν **κάποια** σειρά επιλογών 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 ϵ** (NFA με **ϵ -κινήσεις**): μπορεί να αλλάξει κατάσταση **χωρίς ανάγνωση** επόμενου συμβόλου.

Παράδειγμα NFA

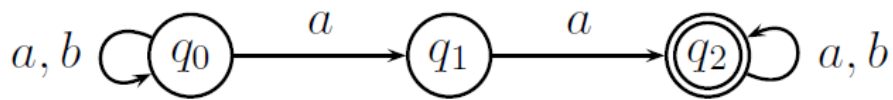
$L_4 := \{w \in \Sigma^* \mid w \text{ περιέχει δύο συνεχόμενα } a\}$



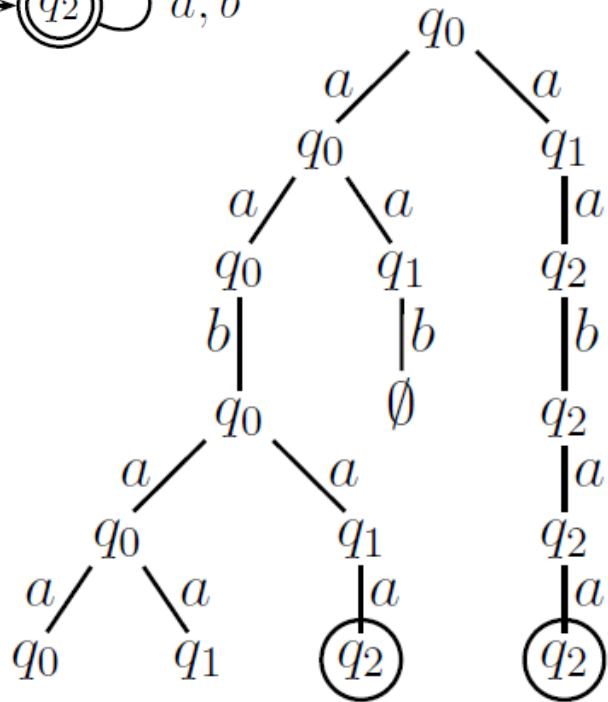
	a	b
q_0	$\{q_0, q_1\}$	$\{q_0\}$
q_1	$\{q_2\}$	\emptyset
q_2	$\{q_2\}$	$\{q_2\}$

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

Παράδειγμα NFA



Δένδρο υπολογισμού
για είσοδο *ααβαα*



ΑΠΟΔΟΧΗ

Τυπικός ορισμός NFA

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

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

Υπενθύμιση: στη συνάρτηση μετάβασης, **κενό σύνολο** σημαίνει ότι η συγκεκριμένη 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 δ 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
- Παράδειγμα: $\delta(q_0, aa) = \{q_0, q_1, q_2\}$, $\delta(q_0, ba) = \{q_0, q_1\}$

Ισοδυναμία NFA και DFA

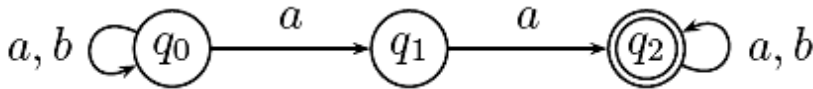
- **Θεώρημα Rabin-Scott:** για κάθε NFA υπάρχει ένα DFA που αποδέχεται την ίδια γλώσσα.
- Επομένως τα DFA και τα NFA αναγνωρίζουν ακριβώς την ίδια κλάση γλωσσών: τις **κανονικές γλώσσες (regular languages)**.
- Οι κανονικές γλώσσες αντιστοιχούν σε **κανονικές παραστάσεις (regular expressions)**:

π.χ. $(a+b)^*bbab(a+b)^*$

NFA \rightarrow 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	$\{q_2\}$	$\{q_2\}$

Κατασκευάζουμε το **δυναμοσύνολο** των καταστάσεων. **Αρχική** κατάσταση: $\{q_0\}$.

Τελικές: όσες περιέχουν τελική.

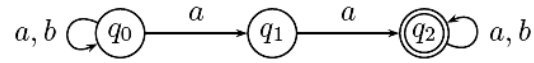
Υπόδειξη: εξετάζουμε μόνο

προσβάσιμα από $\{q_0\}$ σύνολα καταστάσεων.

NFA \rightarrow DFA

(ii)

NFA για τη γλώσσα L_4



	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

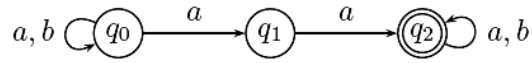
$Q' \setminus \Sigma$	a	b
$\checkmark \{q_0\}$	$\{q_0, q_1\}$	$\{q_0\}$

Υπόδειξη: εξετάζουμε
μόνο **προσβάσιμα**
από $\{q_0\}$ σύνολα
καταστάσεων.

NFA → DFA

(iii)

NFA για τη γλώσσα L_4



	a	b
q0	{q0, q1}	{q0}
q1	{q2}	∅
q2	{q2}	{q2}

DFA για τη γλώσσα L_4

$Q' \setminus \Sigma$	a	b
√ {q0}	{q0, q1}	{q0}

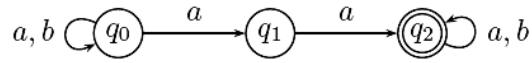
Υπόδειξη: εξετάζουμε μόνο **προσβάσιμα** από $\{q_0\}$ σύνολα καταστάσεων.

√ $\overline{\{q_0, q_1\}}$	{q0, q1, q2}	{q0}
-----------------------------	--------------	------

NFA \rightarrow DFA

(iv)

NFA για τη γλώσσα L_4



	a	b
q ₀	{q ₀ , q ₁ }	{q ₀ }
q ₁	{q ₂ }	∅
q ₂	{q ₂ }	{q ₂ }

DFA για τη γλώσσα L_4

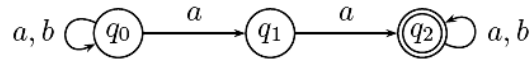
$Q' \setminus \Sigma$	a	b
✓ {q ₀ }	{q ₀ , q ₁ }	{q ₀ }

Υπόδειξη: εξετάζουμε
μόνο **προσβάσιμα**
από {q₀} σύνολα
καταστάσεων.

✓ {q ₀ , q ₁ }	{q ₀ , q ₁ , q ₂ }	{q ₀ }
✓ {q ₀ , q ₁ , q ₂ }	{q ₀ , q ₁ , q ₂ }	{q ₀ , q ₂ }

NFA \rightarrow DFA

NFA για τη γλώσσα L_4



(v)

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

DFA για τη γλώσσα L_4

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

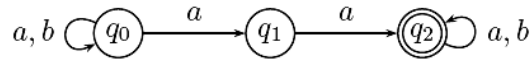
Υπόδειξη: εξετάζουμε μόνο **προσβάσιμα** από $\{q_0\}$ σύνολα καταστάσεων.

✓ {q0, q1}	{q0, q1, q2}	{q0}
✓ {q0, q2}	{q0, q1, q2}	{q0, q2}
✓ {q0, q1, q2}	{q0, q1, q2}	{q0, q2}

NFA \rightarrow DFA

(vi)

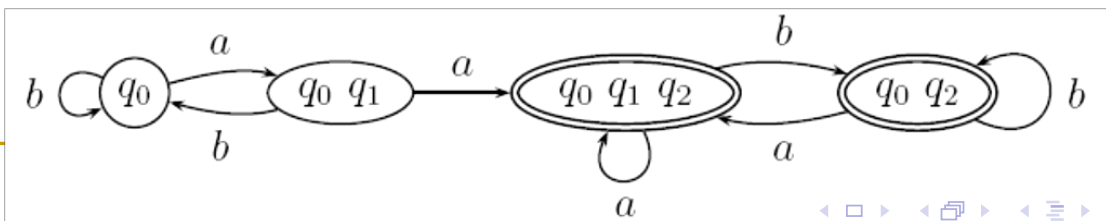
NFA για τη γλώσσα L_4



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

DFA για τη γλώσσα L_4

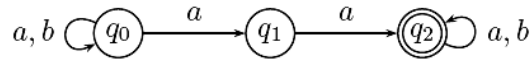
$Q' \setminus \Sigma$	a	b
$\{q_0\}$	{q0, q1}	{q0}
$\{q_0, q_1\}$	{q0, q1, q2}	{q0}
$\{q_0, q_2\}$	{q0, q1, q2}	{q0, q2}
$\{q_0, q_1, q_2\}$	{q0, q1, q2}	{q0, q2}



NFA \rightarrow DFA

(vii)

NFA για τη γλώσσα L_4

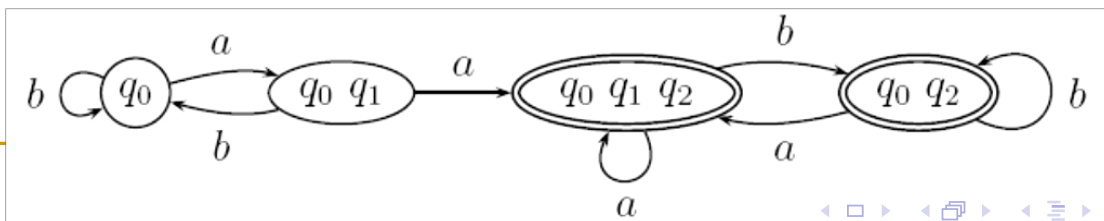


	a	b
q0	{q0, q1}	{q0}
q1	{q2}	\emptyset
q2	{q2}	{q2}

DFA για τη γλώσσα L_4

Οι μη προσβάσιμες καταστάσεις δεν παίζουν ρόλο!

$Q' \setminus \Sigma$	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}



NFA \rightarrow DFA: η μέθοδος τυπικά

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

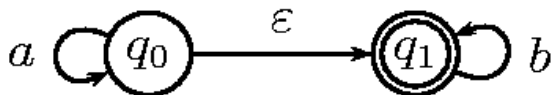
Ένα ισοδύναμο DFA $M' = (Q', \Sigma, q'_0, F', \delta')$, ορίζεται ως εξής:

- $Q' = \text{Pow}(Q)$, 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 .
- $\delta'(R, a) = \{q \in Q \mid q \in \delta(r, a) \text{ για } r \in R\}$, είναι δηλαδή το σύνολο των καταστάσεων όπου μπορεί να βρεθεί το M ξεκινώντας από οποιαδήποτε κατάσταση του συνόλου R και διαβάζοντας το σύμβολο a (a -κίνηση).

Αυτόματα με ϵ -κινήσεις: NFA_ϵ

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

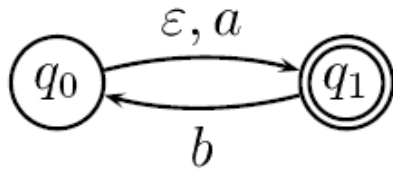
Παράδειγμα: NFA_ϵ για $L_5 := \{a^*b^*\} = \{a^n b^m \mid n, m \in \mathbb{N}\}$



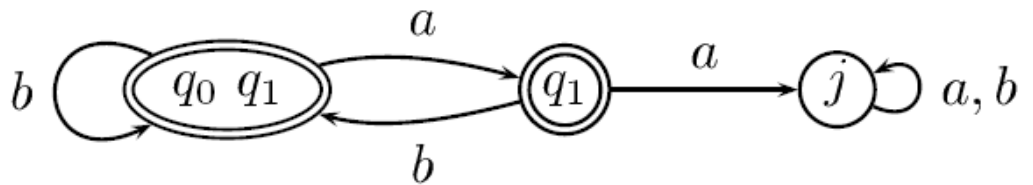
	a	b	ϵ
q_0	$\{q_0\}$	\emptyset	$\{q_0, q_1\}$
q_1	\emptyset	$\{q_1\}$	$\{q_1\}$

Ισοδυναμία NFA_ϵ με DFA: παράδειγμα

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



DFA για $\overline{L_4}$:



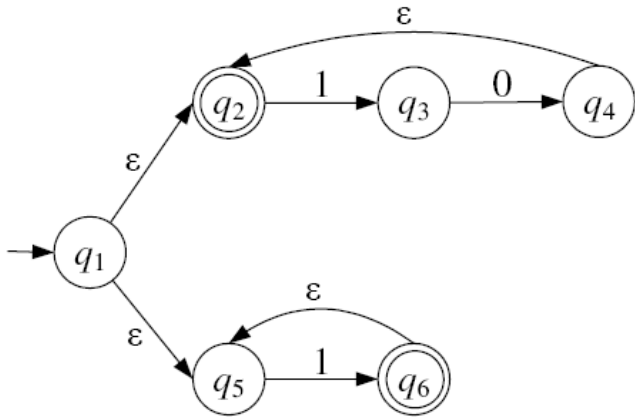
NFA_ε → DFA: η μέθοδος τυπικά

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

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 .
- $q'_0 = \varepsilon\text{-κλείσιμο}(q_0)$ ($\varepsilon\text{-κλείσιμο}(q_i) = \{p \mid p \text{ προσβάσιμη κατάσταση από την } q_i \text{ μόνο με } \varepsilon\text{-κινήσεις}\}$)
- $F' = \{R \in Q' \mid R \cap F \neq \emptyset\}$, δηλαδή μια κατάσταση του M' είναι τελική αν περιέχει μια τελική κατάσταση του M .
- $\delta'(R, a) = \{q \in Q \mid q \in \varepsilon\text{-κλείσιμο}(\delta(r, a)) \text{ για } r \in R\}$, δηλαδή $\delta'(R, a)$ είναι το σύνολο των καταστάσεων όπου μπορεί να βρεθεί το M ξεκινώντας από **οποιαδήποτε κατάσταση του R** , κάνοντας μία **a -κίνηση** και χρησιμοποιώντας στη συνέχεια οσοδήποτε **ε -κινήσεις**.

NFA_ε → DFA: παράδειγμα



- ε-κλείσιμο της $q_1 = \{q_1, q_2, q_5\}$
- ε-κλείσιμο της $q_2 = \{q_2\}$
- ε-κλείσιμο της $q_3 = \{q_4\}$
- ε-κλείσιμο της $q_4 = \{q_2, q_4\}$
- ε-κλείσιμο της $q_5 = \{q_5\}$
- ε-κλείσιμο της $q_6 = \{q_5, q_6\}$

Αρχική κατάσταση
 $\{q_1, q_2, q_5\}$
 Τελικές οι μπλέ

Q'	0	1
$\{q_1, q_2, q_5\}$	$\{\}$	$\{q_3, q_5, q_6\}$
$\{q_3, q_5, q_6\}$	$\{q_2, q_4\}$	$\{q_5, q_6\}$
$\{q_2, q_4\}$	$\{\}$	$\{q_3\}$
$\{q_5, q_6\}$	$\{\}$	$\{q_5, q_6\}$
$\{q_3\}$	$\{q_2, q_4\}$	$\{\}$
$\{\}$	$\{\}$	$\{\}$

$NFA_\epsilon \rightarrow DFA$: εναλλακτική μέθοδος (σε NFA πρώτα)

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

□ Κατασκευάζουμε πρώτα ισοδύναμο NFA $M' = (Q, \Sigma, q_0, F', \delta')$ ως εξής:

- $F' = F \cup \{q_0\}$ αν ϵ -κλείσιμο(q_0) περιέχει τελική,
 $F' = F$ αλλιώς.
- $\delta'(q, a) = \epsilon$ -κλείσιμο($\delta(\epsilon$ -κλείσιμο(q), a)), (δ επεκτ/νη)

δηλαδή $\delta'(q, a)$ είναι το σύνολο των καταστάσεων όπου μπορεί να βρεθεί το M ξεκινώντας από την κατάσταση q , χρησιμοποιώντας οσοσδήποτε ϵ -κινήσεις, μία a -κίνηση, και χρησιμοποιώντας ξανά οσοσδήποτε ϵ -κινήσεις.

□ Από το M' κατασκευάζουμε ισοδύναμο DFA.

$NFA_\epsilon \rightarrow DFA$: εναλλακτική μέθοδος (σε NFA πρώτα, ταχύτερα)

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

□ Κατασκευάζουμε πρώτα ισοδύναμο NFA $M' = (Q, \Sigma, q_0, F', \delta')$ ως εξής:

- F' : είναι το F μαζί με *κάθε* κατάσταση q για την οποία το ϵ -κλείσιμο(q) περιέχει κάποια τελική.
- $\delta'(q, a) = \delta(\epsilon\text{-κλείσιμο}(q), a)$, (δ επεκτ/νη)

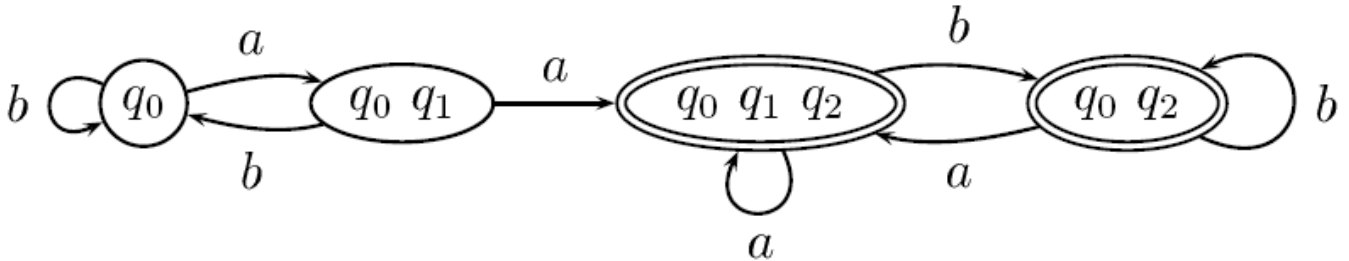
δηλαδή $\delta'(q, a)$ είναι το σύνολο των καταστάσεων όπου μπορεί να βρεθεί το M ξεκινώντας από την κατάσταση q , χρησιμοποιώντας οσοδήποτε ϵ -κινήσεις πρώτα, και μετά μία a -κίνηση (δηλ. χωρίς ϵ -κινήσεις μετά το a).

□ Από το M' κατασκευάζουμε ισοδύναμο DFA.

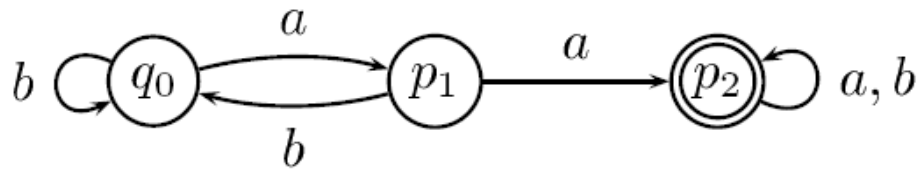
Ελαχιστοποίηση DFA: παράδειγμα

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

Αρχικό DFA

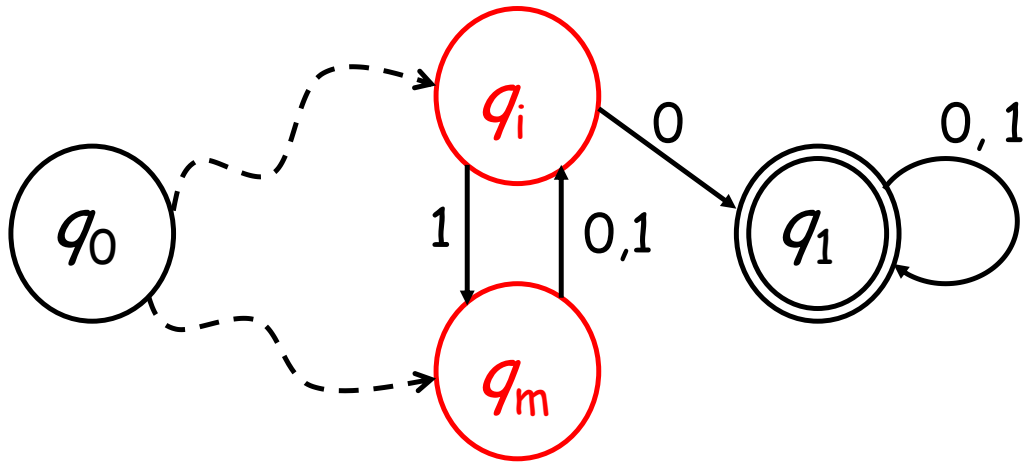


Ελάχιστο DFA



Ελαχιστοποίηση DFA (i)

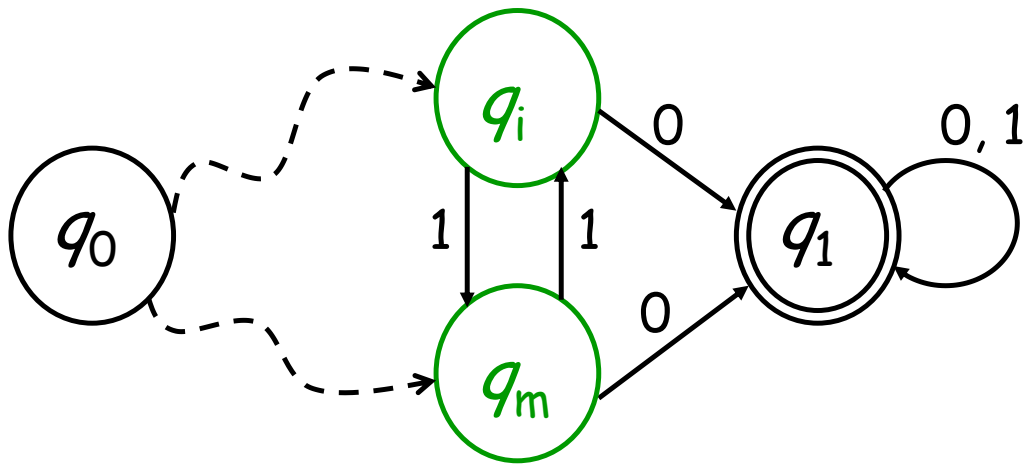
Δύο καταστάσεις DFA λέγονται *μη ισοδύναμες*, δηλαδή *διακρίσιμες*, αν *υπάρχει* συμβολοσειρά που να οδηγεί την μία από αυτές σε τελική κατάσταση, ενώ την άλλη όχι.



Ελαχιστοποίηση DFA (ii)

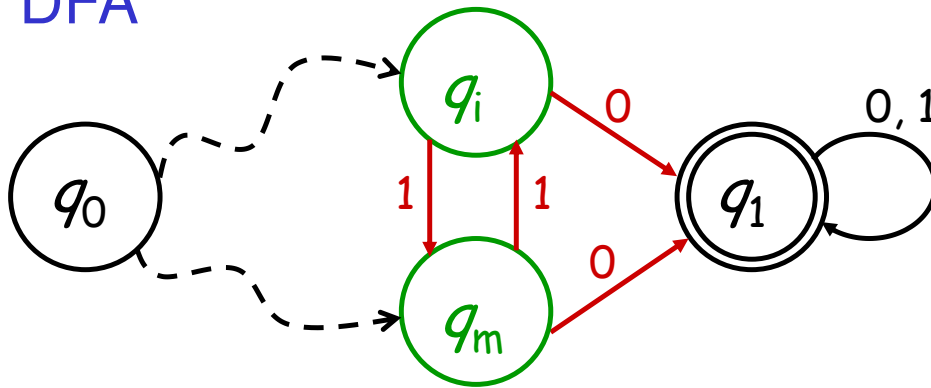
Δύο καταστάσεις μπορούν να συγχωνευτούν σε μία (είναι *ισοδύναμες*) αν:

οδηγούν με ίδιες συμβολοσειρές σε ίδιο αποτέλεσμα

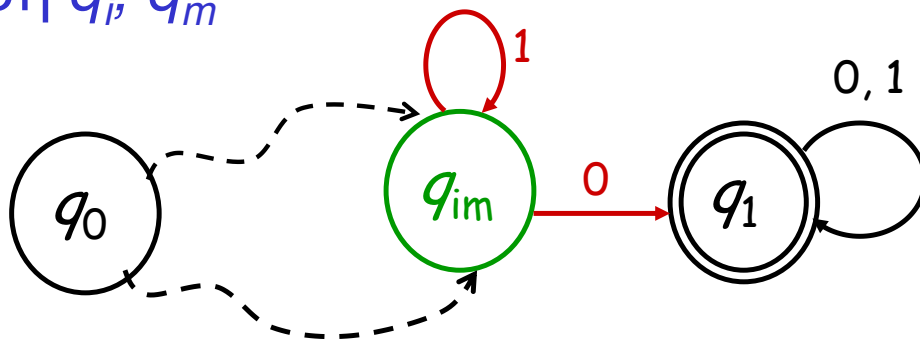


Ελαχιστοποίηση DFA (iii)

Αρχικό DFA

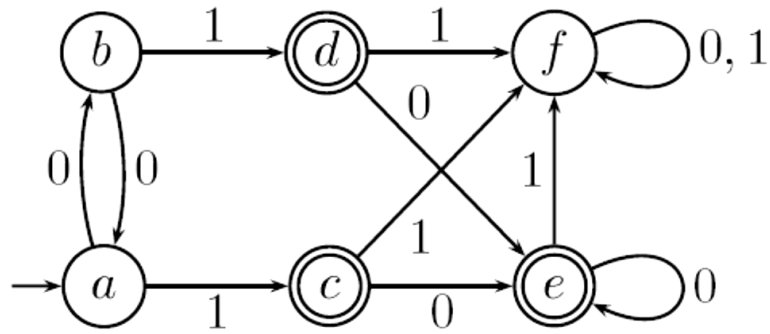


Συγχώνευση q_i, q_m

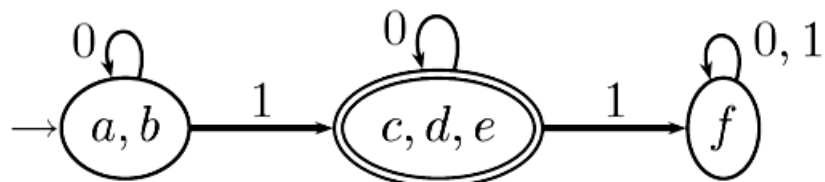


Ελαχιστοποίηση DFA: 2^ο παράδειγμα

Αρχικό DFA

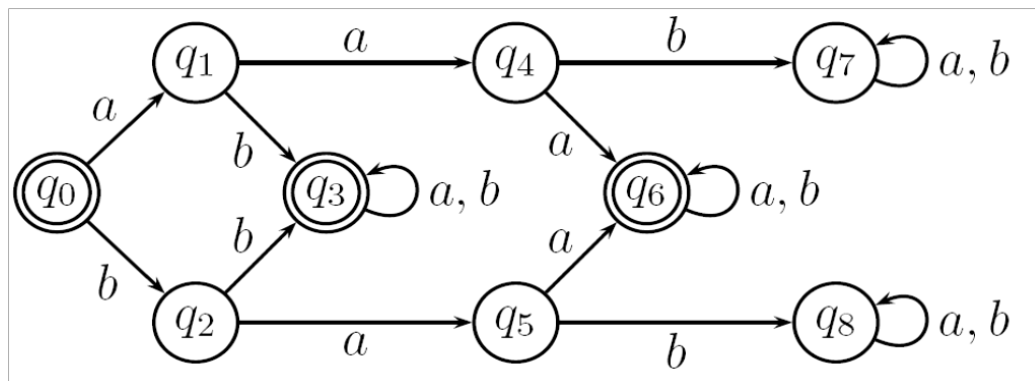


Ελάχιστο DFA

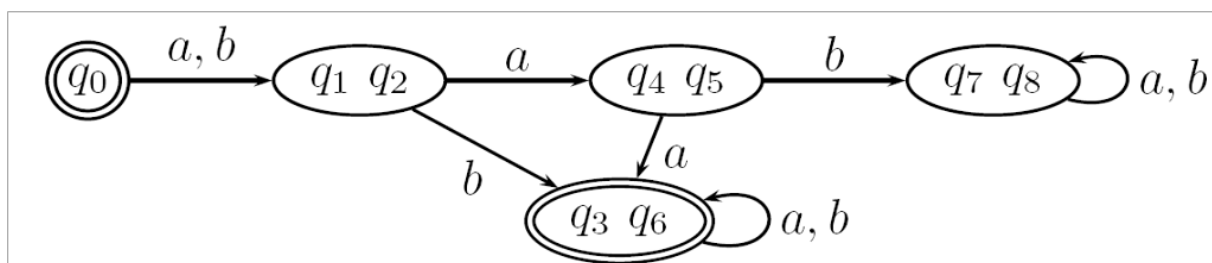


Ελαχιστοποίηση DFA: 3^ο παράδειγμα

Αρχικό DFA



Ελάχιστο DFA



Μέθοδος ελαχιστοποίησης DFA

- Δύο καταστάσεις λέγονται **k -διακρίσιμες** αν με κάποια συμβολοσειρά μήκους ακριβώς k οδηγούν σε **διαφορετικό αποτέλεσμα** (και δεν είναι i -διακρίσιμες για κανένα $i < k$). 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 .

Μέθοδος ελαχιστοποίησης DFA

- 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.
- Γιατί δουλεύει:

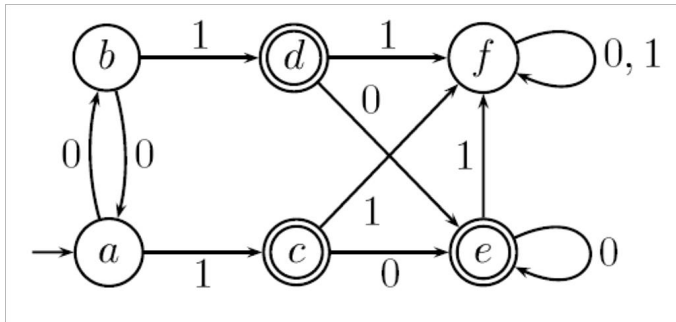
δεν υπάρχουν $(i+1)$ -διακρίσιμες καταστάσεις αν δεν υπάρχουν i -διακρίσιμες καταστάσεις

Η μέθοδος συστηματικά

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

- Αρχικά γράφουμε X_0 σε όλα τα ζεύγη κατα/σεων που είναι **0-διακρίσιμες** γιατί η μία είναι τελική και η άλλη όχι.
- Σε κάθε "γύρο" $i+1$, εξετάζουμε όλα τα μη σημειωμένα ζεύγη και γράφουμε X_{i+1} σε ένα ζεύγος αν από τις δύο καταστάσεις του με ένα σύμβολο το DFA πηγαίνει σε **i -διακρίσιμες** καταστάσεις (ήδη 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

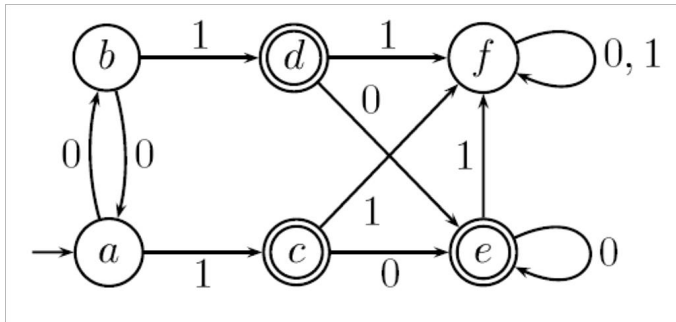


b					
\textcircled{c}	X_0	X_0			
\textcircled{d}	X_0	X_0			
\textcircled{e}	X_0	X_0			
f			X_0	X_0	X_0
	a	b	\textcircled{c}	\textcircled{d}	\textcircled{e}

Γύρος 0:

**εννέα ζεύγη
0-διακρίσιμων
καταστάσεων**

Παράδειγμα εφαρμογής της μεθόδου

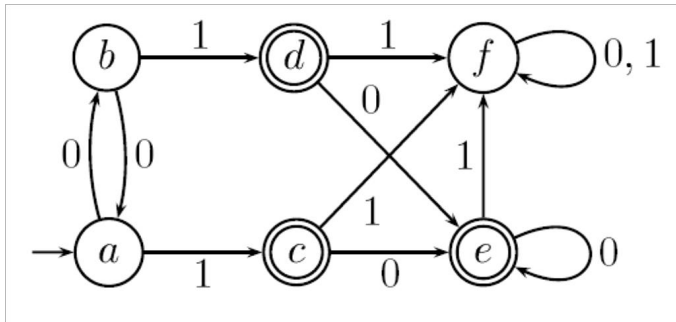


b					
c	X_0	X_0			
d	X_0	X_0			
e	X_0	X_0			
f	X_1	X_1	X_0	X_0	X_0
	a	b	c	d	e

Γύρος 1:

δύο ζεύγη
1-διακρίσιμων
ΚΑΤΑΣΤΑΣΕΩΝ

Παράδειγμα εφαρμογής της μεθόδου

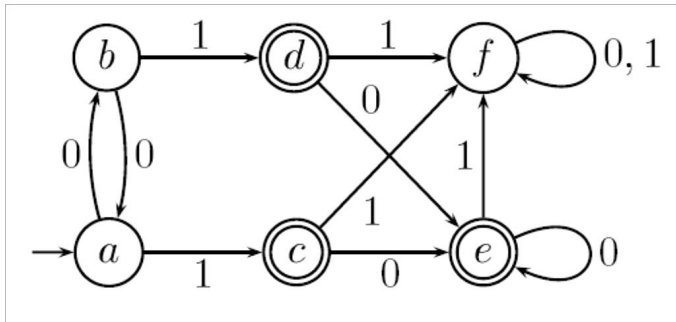


b					
c	X_0	X_0			
d	X_0	X_0			
e	X_0	X_0			
f	X_1	X_1	X_0	X_0	X_0
	a	b	c	d	e

Γύρος 2:

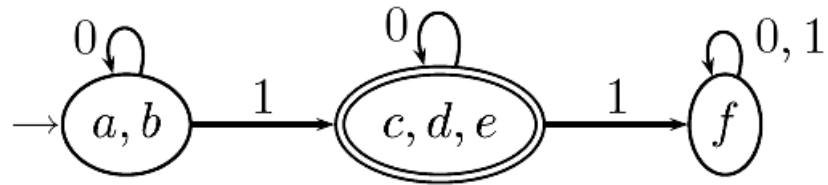
**κανένα ζεύγος
2-διακρίσιμων
καταστάσεων**

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 .
- ϵ : **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:

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

- Σ^* : το σύνολο όλων των λέξεων του αλφαβήτου Σ .
- Γλώσσα από το αλφάβητο Σ : 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 α με την συμβολοσειρά β .
- Κάθε τυπική γραμματική **παράγει** μια τυπική γλώσσα: το σύνολο των συμβολοσειρών (με **τερματικά σύμβολα** μόνο) που παράγονται 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

ε : 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.ώ. $V \cap T = \emptyset$,
- ένα πεπερασμένο σύνολο P από *κανόνες παραγωγής*, δηλαδή διατεταγμένα ζεύγη (α, β) ,
όπου $\alpha, \beta \in (V \cup T)^*$ και $\alpha \neq \varepsilon$
(σύμβαση: γράφουμε $\alpha \rightarrow \beta$ αντί για (α, β)),
- 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 οποιεσδήποτε 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 α παράγει το β ») 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)

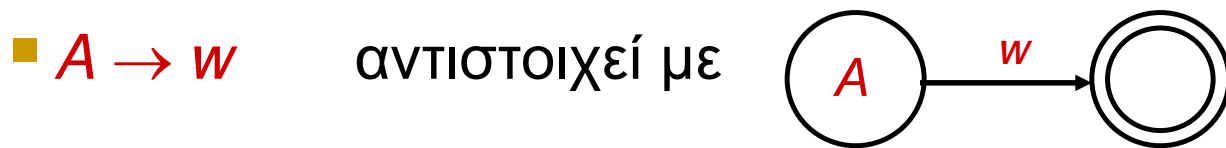
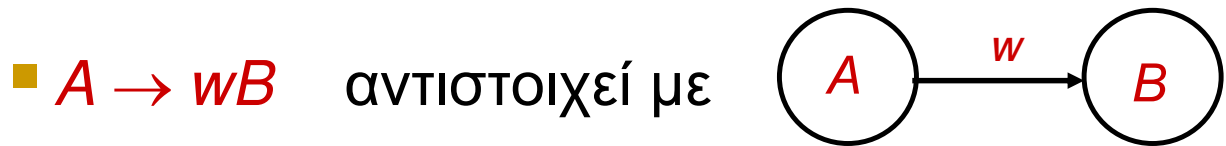
$$A \rightarrow Bw \text{ ή } A \rightarrow w$$

(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:



- 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 Σ .

- $L_1 L_2 := \{uv \mid u \in L_1 \wedge v \in L_2\}$: 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
- ε : παριστάνει $\{\varepsilon\}$
- a : παριστάνει $\{a\}$, $a \in \Sigma$
- $(r+s)$: 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:

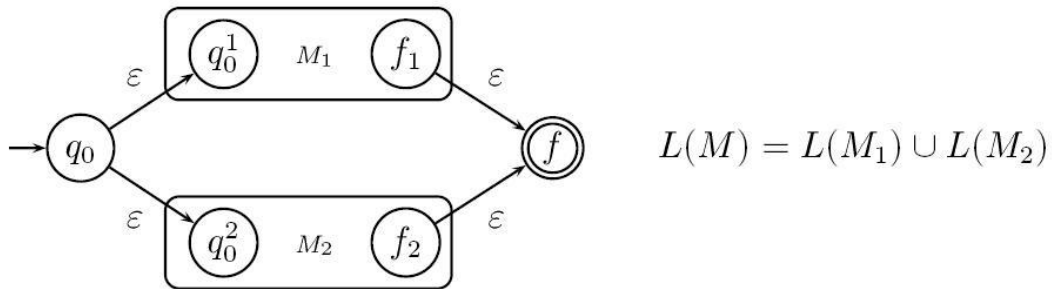
' \Rightarrow ' : Επαγωγή στη δομή της κανονικής παράστασης r :

1. Επαγωγική Βάση:

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

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

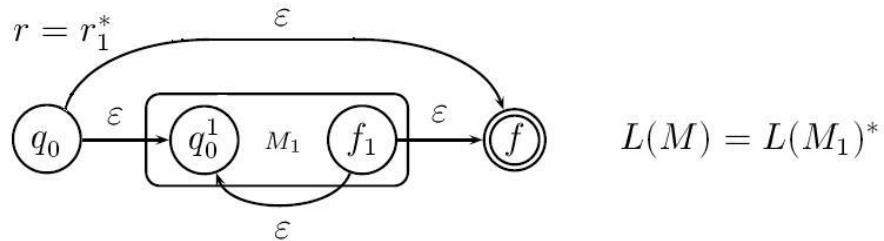
Περίπτωση α: $r = r_1 + r_2$



Περίπτωση β: $r = r_1 r_2$



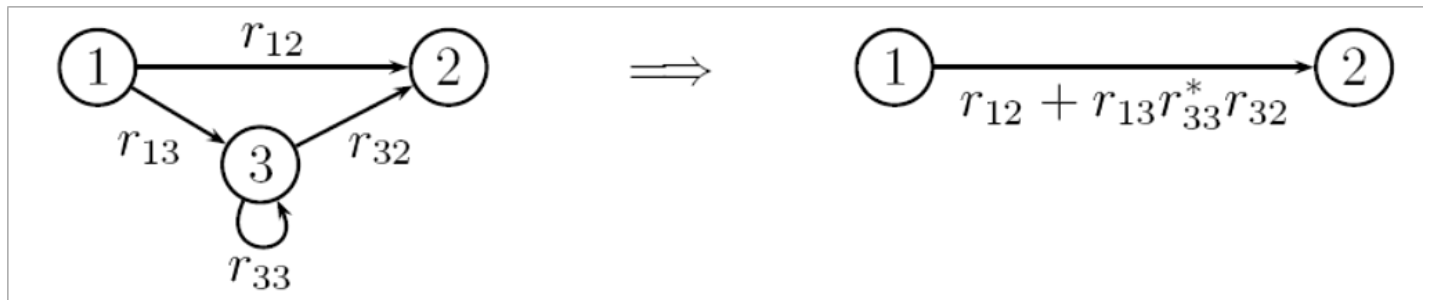
Περίπτωση γ: $r = r_1^*$



Ισοδυναμία κανονικών παραστάσεων και αυτομάτων (συν.)

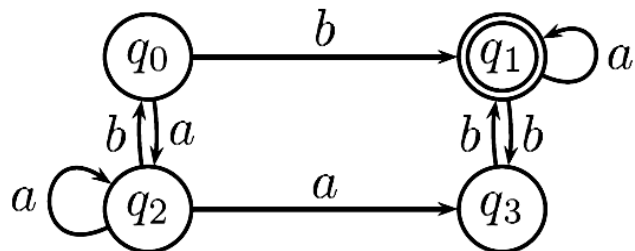
' \leq ': Κατασκευή κανονικής παράστασης από FA (GNFA).

Απαλείφουμε ενδιάμεσες καταστάσεις σύμφωνα με το σχήμα:

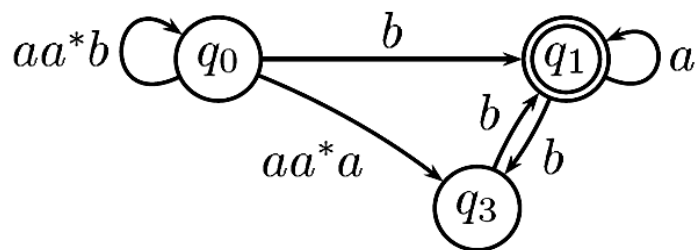


Παράδειγμα κατασκευής κανονικής παράστασης από FA

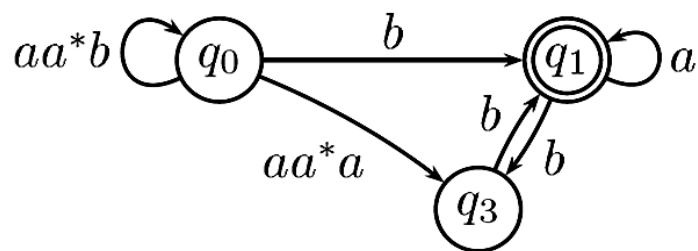
Αρχικό DFA



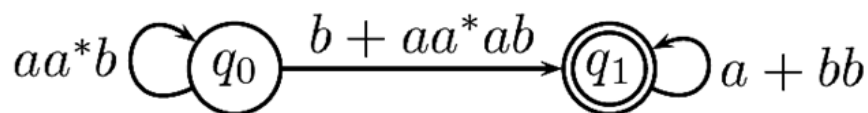
Μετά από διαγραφή q_2



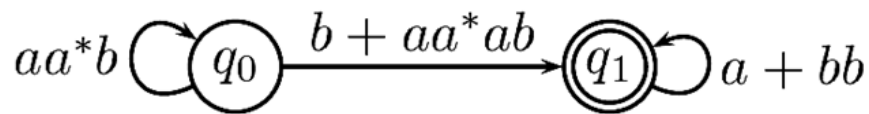
Παράδειγμα κατασκευής κανονικής παράστασης από FA



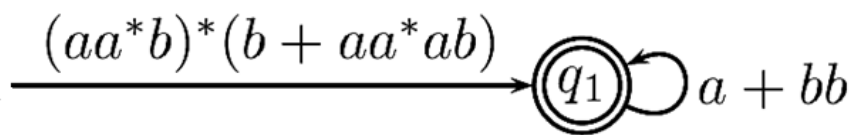
Διαγραφή q_3



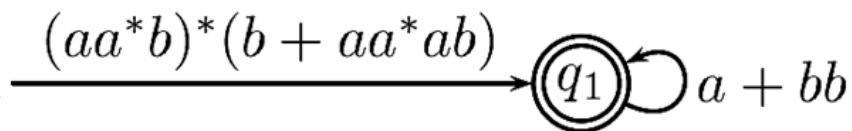
Παράδειγμα κατασκευής κανονικής παράστασης από FA



Διαγραφή q_0



Παράδειγμα κατασκευής κανονικής παράστασης από FA



Τελική παράσταση

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

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): τρόπος κατασκευής DFA για **τομή** (αλλά και **ένωση**) κανονικών γλωσσών.

Γινόμενο αυτομάτων DFA

- 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 **γινόμενο των M_1, M_2** 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: $\delta'((q_i, p_k), \sigma) = (q_{i'}, p_{k'}) \Leftrightarrow \delta(q_i, \sigma) = q_{i'} \wedge \delta(p_k, \sigma) = p_{k'}$
- **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 των τελικών καταστάσεων.

Γινόμενο αυτομάτων NFA

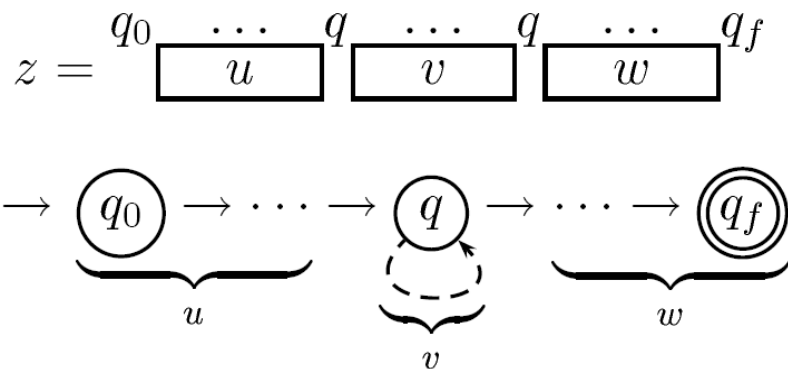
- Ορίζεται με παρόμοιο τρόπο.
- Χρειάζεται προσοχή στις ε-κινήσεις και στο συνδυασμό μεταβάσεων σε junk states με κανονικές μεταβάσεις.

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*:

Έστω η L 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» $z = uvw$, με $|uv| \leq n$ και $|v| > 0$
- *επιλέγουμε* i ώστε η λέξη $uv^i w$ να μην είναι στη γλώσσα L

ΑΤΟΠΟ

(adversary argument)

Παράδειγμα χρήσης Pumping Lemma (i)

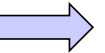
- **Θεώρημα.** Η γλώσσα $L = \{z \mid z \text{ έχει το ίδιο πλήθος } 0 \text{ και } 1\}$ δεν είναι κανονική.

- **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» $z = uvw$, με $|uv| \leq n$ και $|v| > 0$



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)

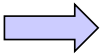
Θεώρημα. Η γλώσσα $L = \{z \mid z=0^i1^j, i > j\}$ δεν είναι κανονική.

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 $|z| = 2n+1 > n$

$$\begin{array}{c} \blacksquare z = \underbrace{000000000\dots0}_{n+1} \underbrace{1111111\dots1}_n \end{array}$$

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



Δεύτερο παράδειγμα χρήσης PL (ii)

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

$$z = \underbrace{0\dots0}_u \underbrace{0\dots0}_v \underbrace{011111111\dots1}_w$$

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

ΑΤΟΠΟ

- Επομένως η L δεν είναι κανονική.

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**.

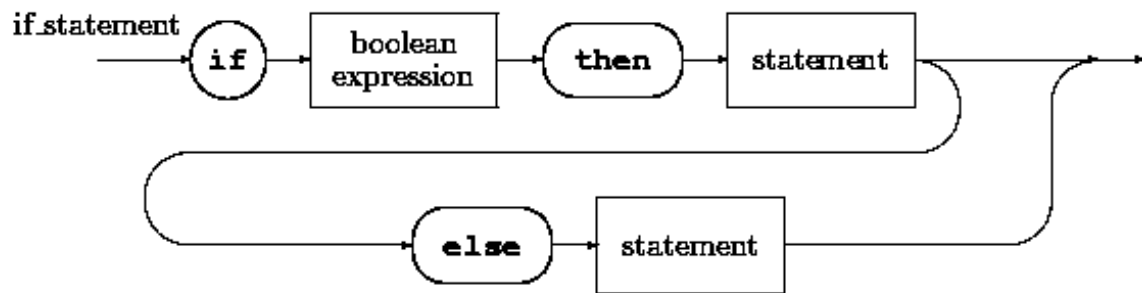
Γραμματικές για μη κανονικές γλώσσες

- **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, $\alpha \in (V \cup T)^*$
- Παράδειγμα:

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

Δυνατή ακολουθία παραγωγής:

$$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aaaSbbb \Rightarrow aaabbb$$

Γλώσσα που παράγεται:

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

Γραμματικές χωρίς συμφραζόμενα (Context Free) (iii)

- 2^ο παράδειγμα:

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

$$P: S \rightarrow S+S, \quad S \rightarrow S*S,$$

$$S \rightarrow 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$$

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^ο παράδειγμα:

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

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

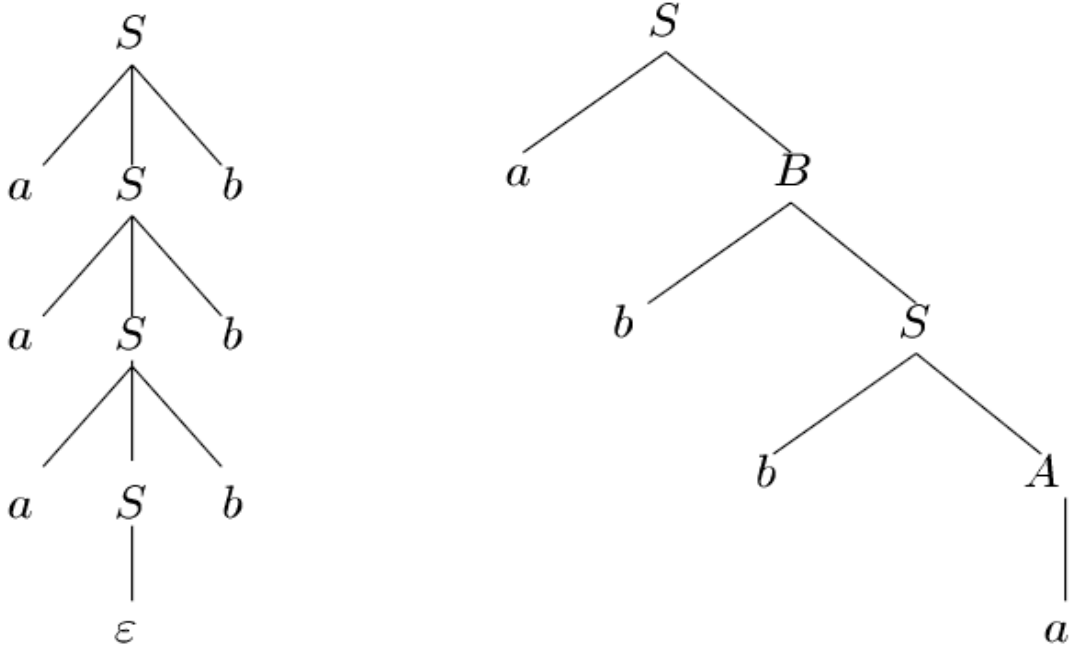
Δυνατή ακολουθία παραγωγής:

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

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): $aabbba$ και $aabbba$ 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 ϵ).
- 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 ϵ , 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 $S \xRightarrow{*} w$ αν και μόνο αν υπάρχει συντακτικό δένδρο της G με φύλλωμα w .

Απόδειξη:

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

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

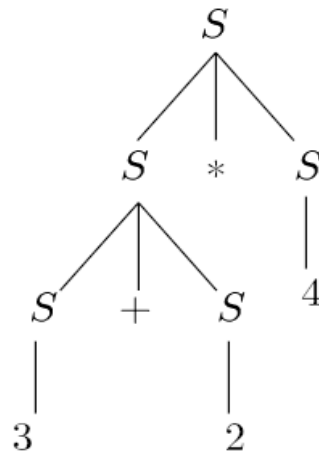
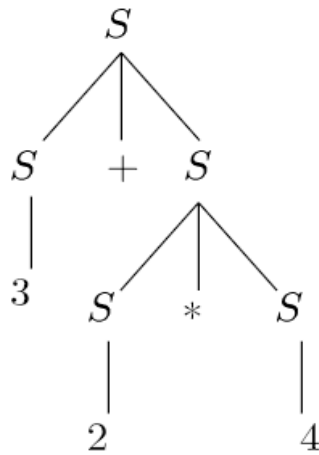
Διφορούμενες γραμματικές

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

Παράδειγμα:

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

$P: S \rightarrow S+S \mid S*S \mid 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$



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(x)**: τοποθετεί στοιχείο x στην κορυφή της στοίβας
 - **pop**: διαβάζει και αφαιρεί στοιχείο από την κορυφή της στοίβας

Αυτόματα Στοίβας (PDA) (ii)

Παράδειγμα: PDA για αναγνώριση της γλώσσας

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

Περιγραφή αυτομάτου

- **push(a)** στη στοίβα για κάθε 0 στην είσοδο, **push(b)** στη στοίβα για κάθε 1 στην είσοδο, συνέχισε μέχρι να διαβαστεί **c**
- μετά **pop**: εφόσον πάνω στοιχείο στοίβας συμφωνεί με είσοδο (**a με 0, b με 1**) συνέχισε
- Αποδοχή με **κενή στοίβα**

Τυπικός ορισμός PDA

Αυτόματο στοίβας (Pushdown Automaton, PDA):

επτάδα $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$

- Q : το σύνολο των καταστάσεων του M (πεπερασμένο)
- Σ : αλφάβητο εισόδου
- Γ : αλφάβητο **στοίβας**
- $\delta : Q \times \Sigma \cup \{\epsilon\} \times \Gamma \rightarrow \text{Pow}(Q \times \Gamma^*)$: συνάρτηση μετάβασης (μη ντετερμινισμός, ϵ -κινήσεις)
- $q_0 \in Q$: αρχική κατάσταση
- $Z_0 \in \Gamma$: **αρχικό σύμβολο στοίβας**
- $F \subseteq Q$: σύνολο τελικών καταστάσεων

Αυτόματα Στοίβας (PDA) (iv)

Είδη αποδοχής PDA

- Αν βρεθεί σε **τελική κατάσταση** (δηλ. αποδοχής) μόλις διαβαστεί όλη η είσοδος, ανεξαρτήτως περιεχομένου στοίβας
- Αν βρεθεί με **κενή στοίβα** μόλις διαβαστεί όλη η είσοδος, ανεξαρτήτως κατάστασης

Αντίστοιχα ορίζονται οι γλώσσες:

- $L_f(M)$: αποδοχή με τελική κατάσταση
- $L_e(M)$: αποδοχή με κενή στοίβα

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 :

- $L = L_f(M)$, M είναι PDA.
- $L = L_e(M')$, M' είναι PDA.
- Η L είναι γλώσσα χωρίς συμφραζόμενα (c.f.)

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 στο μάθημα «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 παρακάτω είναι ισοδύναμα:

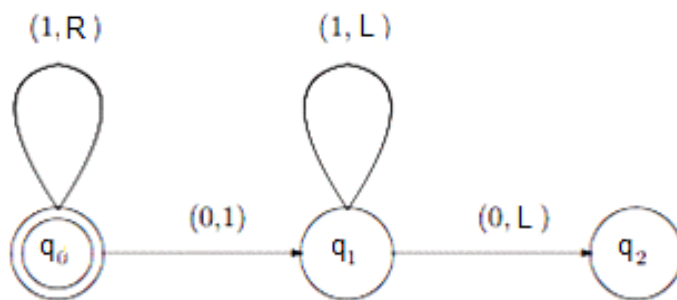
1. Η γλώσσα L γίνεται αποδεκτή από μια μηχανή Turing
2. $L=L(G)$, όπου G είναι γενική γραμματική

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 διαβάσει.

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)

Γραμματική c.s. για τη γλώσσα $1^n 0^n 1^n$:

$$\begin{aligned} S &\rightarrow 1Z1 \\ Z &\rightarrow 0 \mid 1Z0A \\ A0 &\rightarrow 0A \\ A1 &\rightarrow 11 \end{aligned}$$

Μετατροπή σε κανονική μορφή:

$$A0 \rightarrow H0 \quad H0 \rightarrow HA \quad HA \rightarrow 0A$$

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

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 παρακάτω είναι ισοδύναμα (L χωρίς ε):

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