# Αναδρομικές Συναρτήσεις και Σχέσεις I

## Ορισμός

Η ολική συνάρτηση $h(\vec{x}, y)$ είναι **κανονική** (regular):

$$\forall \vec{x}\ \exists y\ h(\vec{x}, y) = 0$$

## Παρατήρηση

Η $f(\vec{x}) = \mu y[h(\vec{x}, y) = 0]$ είναι ορισμένη για κάθε $\vec{x}$ για μια κανονική συνάρτηση $h$.

## Ορισμός

$\mathcal{R}$: η μικρότερη κλάση συναρτήσεων που

α) περιλαμβάνει τις $S$, $P$, $Z$, $U_i^n$, και

β) είναι κλειστή ως προς τη σύνθεση, την πρωταρχική αναδρομή και το $\mu$-σχήμα (εφαρμοζόμενο μόνο) σε κανονικές συναρτήσεις.

# Αναδρομικές Συναρτήσεις και Σχέσεις II

### Θεώρημα

*Η $\mathcal{R}$ περιλαμβάνει ακριβώς εκείνες τις συναρτήσεις της $\mathcal{PR}$ που είναι ολικές.*

### Απόδειξη.

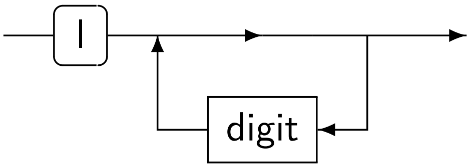
Έπεται από το **Θεώρημα Κανονικής Μορφής Kleene** (Kleene Normal Form Theorem) □

### Ορισμός

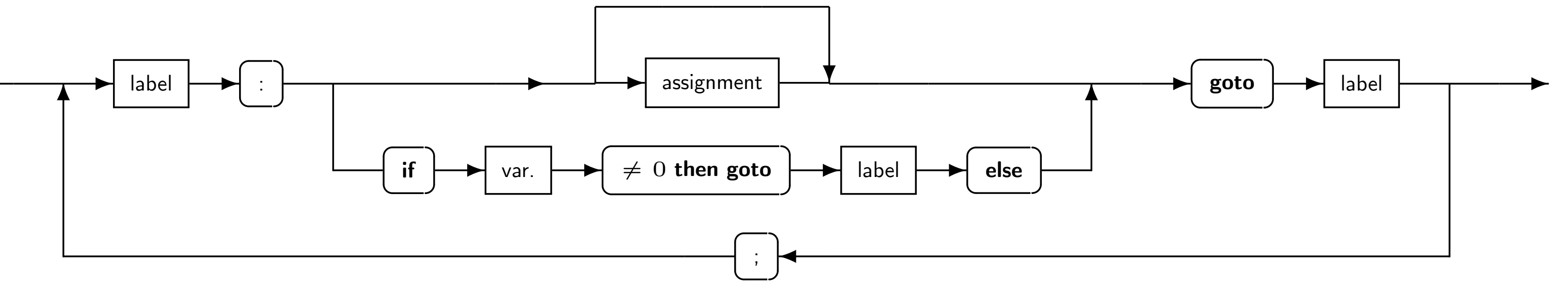
Μια **σχέση** (ή ένα **σύνολο**) είναι **αναδρομική** αν η χαρακτηριστική της (ή του) συνάρτηση είναι (ολική) αναδρομική.

# Η γλώσσα GOTO I

Μία περιορισμένη γλώσσα GOTO:

- μεταβλητές και εντολές ανάθεσης όπως στην WHILE.
- label 

- program



# Η γλώσσα GOTO II

**Παρατηρήσεις:**

- Οι εντολές των προγραμμάτων GOTO είναι αριθμημένες με διαδοχικές labels αρχίζοντας από το $l_0$.
- Η υπολογιστική εκτέλεση ενός GOTO προγράμματος σταματάει όταν εκτελεστεί ένα **goto** $l_i$ και καμία εντολή δεν είναι αριθμημένη με label $l_i$.

Άλλα *ισοδύναμα* υπολογιστικά μοντέλα αποτελούν π.χ. μία γλώσσα assembly, ή μία γλώσσα συναρτησιακή όπως η LISP.

Όλα αυτά τα μοντέλα είναι **μηχανιστικά ισοδύναμα**. Μερικά κοινά χαρακτηριστικά αυτών των μοντέλων είναι π.χ.:

- **ντετερμινιστική** υπολογιστική ακολουθία σε **διακριτά βήματα**.
- **πεπερασμένο σύνολο εντολών** που μπορούν να εκτελεστούν από κάποιον επεξεργαστή (άνθρωπο ή μη).

# Μηχανές Turing (T.M) I

Οι βασικές λειτουργίες μιας **TM** είναι:

- Διάβασε το περιεχόμενο του τρέχοντος κουτιού
- Γράψε 1 ή 0 στο τρέχον κουτάκι
- Κάνε τρέχον το αμέσως αριστερότερο ή το αμέσως δεξιότερο κουτάκι

Η ΤΜ έχει ένα πεπερασμένο αριθμό εσωτερικών **καταστάσεων (internal states)**:

$$Q = \{q_0, q_1, \ldots\}$$

**Πρόγραμμα** μιας ΤΜ: είναι ένα σύνολο από **τετράδες** της μορφής $\langle q_i, e, d, q_j \rangle$ όπου $q_i, q_j \in Q, e \in \Sigma$, $d \in A = \Sigma \bigcup \{L, R\}$ με τον εξής συναρτησιακό (ντετερμινιστικό) περιορισμό: Για κάθε $\langle q_i, e \rangle$ υπάρχει το πολύ ένα $\langle d, q_j \rangle$ έτσι ώστε η τετράδα $\langle q_i, e, d, q_j \rangle$ να ανήκει στο πρόγραμμα, δηλαδή πρόκειται για μια **συνάρτηση μετάβασης** (*transition function*) $\delta : Q \times \Sigma \rightarrow A \times Q$.

# Παραδείγματα TM I

**Παράδειγμα:**

$\Sigma = \{0, 1\}$, $Q = \{q_0, q_1\}$
**Πρόγραμμα**: $\{\langle q_0, 1, 0, q_1\rangle, \langle q_1, 0, R, q_0\rangle\}$
**Είσοδος**: πεπερασμένος αριθμός διαδοχικών "1", όλα τα υπόλοιπα κουτάκια περιέχουν "0".
**Αρχική κατάσταση**: $q_0$
**Αρχική θέση κεφαλής (τρέχον κουτάκι)**: το αριστερότερο "1"

# Παραδείγματα TM II

Παράδειγμα υπολογιστικής ακολουθίας, δηλαδή νόμιμης ακολουθίας από στιγμιότυπα:

$$\ldots\ 0\ 0\ \underset{\substack{\uparrow\\ q_0}}{1}\ 1\ 1\ 0\ 0$$

$$\ldots\ 0\ 0\ \underset{\substack{\uparrow\\ q_1}}{0}\ 1\ 1\ 0\ 0$$

$$\ldots\ 0\ 0\ 0\ \underset{\substack{\uparrow\\ q_0}}{1}\ 1\ 0\ 0$$

$$\ldots\ 0\ 0\ 0\ q_1\ 0\ 1\ 0\ 0$$

$$\ldots\ 0\ 0\ 0\ 0\ q_0\ 1\ 0\ 0$$

$$\ldots\ 0\ 0\ 0\ 0\ q_1\ 0\ 0\ 0$$

$$\ldots\ 0\ 0\ 0\ 0\ 0\ q_0\ 0\ 0\ 0 \qquad \textit{halt}$$

# Παραδείγματα TM III

Σε κάθε TM μπορούμε να αντιστοιχίσουμε μια μερική συνάρτηση από το $\mathbb{N}$ στο $\mathbb{N}$. Η είσοδος $n \in \mathbb{N}$ παριστάνεται με $\underline{n+1}$ διαδοχικά 1 (έτσι ο αριθμός 0 παριστάνεται με 1). Σαν αρχικό στιγμιότυπο έχουμε την κεφαλή (τρέχον κουτάκι) να δείχνει στο αριστερότερο 1 και να βρίσκεται στην κατάσταση $q_0$. Σαν έξοδο λαμβάνουμε το συνολικό αριθμό από 1 που βρίσκεται στην ταινία, όταν και εάν η μηχανή σταματήσει.

# Παραδείγματα TM IV

## Παράδειγμα

Να κατασκευαστεί μια TM που υπολογίζει το $2 * x$.

*Ιδέα:* $\begin{array}{ccccccccc} \ldots & 0 & 0 & \cancel{1} & \cancel{1} & 1 & 1 & 0 & & \\ & & & & & \uparrow & & & 1 & 1 \end{array}$

Η TM θα εργάζεται σύμφωνα με το παρακάτω πρόγραμμα (το οποίο γράφεται πάντα πρώτα σε άτυπη γλώσσα υψηλού επιπέδου):

*αρχικοποίηση; (διαγραφή του πρώτου 1)*
**while** *είσοδος<>0* **do**
  *διάγραψε ένα 1 από είσοδο;*
  *μετακίνησε κεφαλή δεξιά πέρα από είσοδο και έξοδο;*
  *πρόσθεσε δύο 1 στην έξοδο;*
  *μετακίνησε κεφαλή αριστερά πέρα από έξοδο και είσοδο*
**end**

# Παραδείγματα TM V

| | | | | |
|---|---|---|---|---|
| $< q_0$ | 1 | 0 | $q_1 >$ | |
| $< q_1$ | 0 | R | $q_2 >$ | |
| $< q_2$ | 1 | 0 | $q_3 >$ | \| halt για $< q_2 \;\; 0 >$ |
| $< q_3$ | 0 | R | $q_4 >$ | |
| $< q_4$ | 1 | R | $q_4 >$ | |
| $< q_4$ | 0 | R | $q_5 >$ | |
| $< q_5$ | 1 | R | $q_5 >$ | |
| $< q_5$ | 0 | 1 | $q_6 >$ | |
| $< q_6$ | 1 | R | $q_6 >$ | |
| $< q_6$ | 0 | 1 | $q_7 >$ | |
| $< q_7$ | 1 | L | $q_7 >$ | |
| $< q_7$ | 0 | L | $q_8 >$ | |
| $< q_8$ | 1 | L | $q_8 >$ | |
| $< q_8$ | 0 | R | $q_2 >$ | |

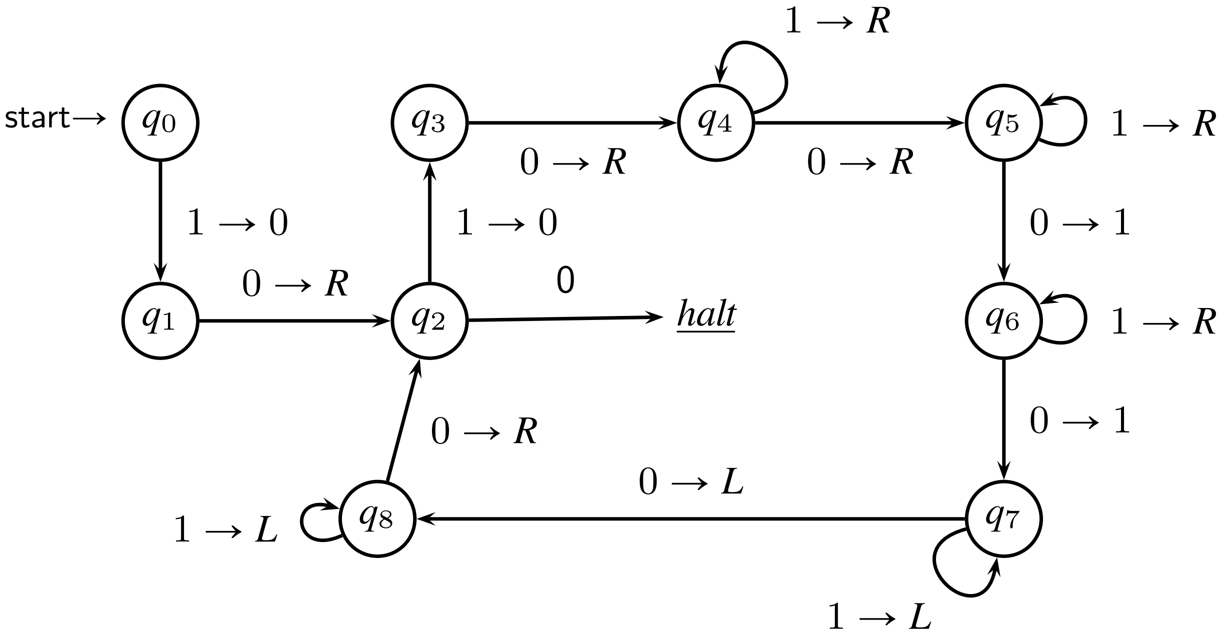
Πίνακας: Πρόγραμμα TM παραδείγματος σε μορφή τετράδων

# Παραδείγματα TM VI

| | $q_0$ | $q_1$ | $q_2$ | $q_3$ | $q_4$ | $q_5$ | $q_6$ | $q_7$ | $q_8$ |
|---|---|---|---|---|---|---|---|---|---|
| 0 | | $R/q_2$ | | $R/q_4$ | $R/q_5$ | $1/q_6$ | $1/q_7$ | $L/q_8$ | $R/q_2$ |
| 1 | $0/q_1$ | | $0/q_3$ | | $R/q_4$ | $R/q_5$ | $R/q_6$ | $L/q_7$ | $L/q_8$ |

Πίνακας: TM σε μορφή πίνακα

# Παραδείγματα TM VII




Σχήμα: TM σε μορφή διαγράμματος καταστάσεων

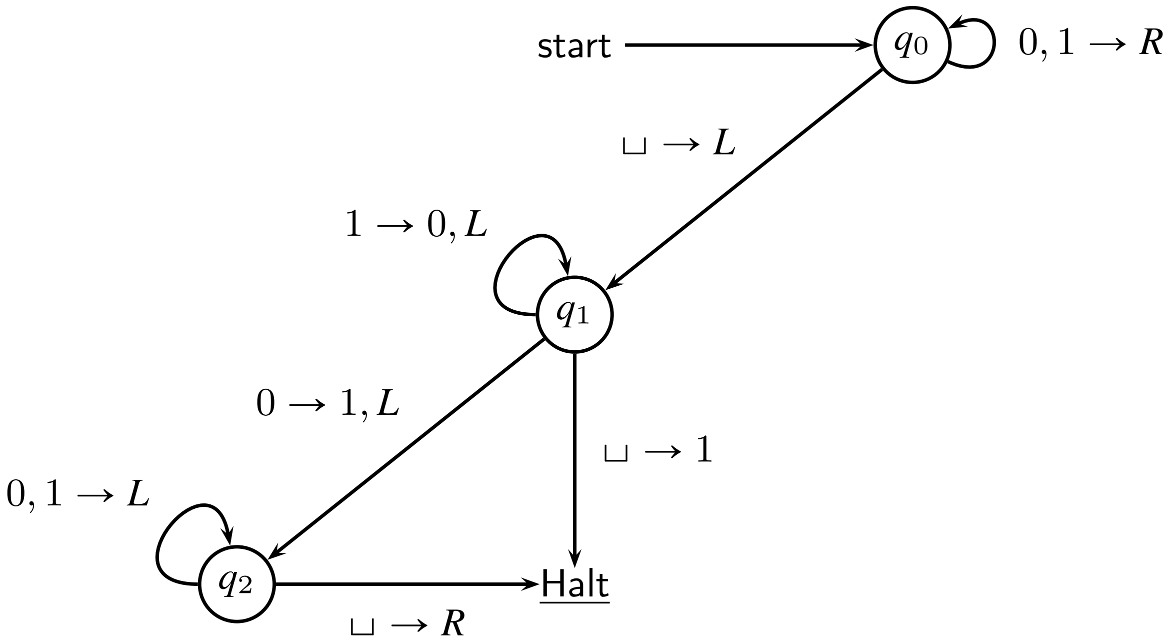
# Παραδείγματα TM I

Στο επόμενο παράδειγμα το αλφάβητο είναι $\Sigma = \{0, 1, \sqcup\}$.
Η είσοδος και η έξοδος κωδικοποιούνται στο **δυαδικό σύστημα**.
**Παράδειγμα**: Να κατασκευαστεί μια TM που υπολογίζει το $x + 1$.

Η TM θα εργάζεται σύμφωνα με το παρακάτω πρόγραμμα:

*Κάνε τρέχον το κουτάκι με τελευταίο σύμβολο της εισόδου $x$;*
**repeat**
*Αν τρέχον κουτάκι έχει $\sqcup$, γράψε 1 και σταμάτα;*
*Αν τρέχον κουτάκι έχει 1, γράψε 0, κάνε τρέχον το αμέσως αριστερότερο κουτάκι, και κρατούμενο := 1;*
*Αν τρέχον κουτάκι έχει 0, γράψε 1, κάνε τρέχον το αμέσως αριστερότερο κουτάκι, και κρατούμενο := 0;*
**until** *κρατούμενο = 0;*
*Κάνε τρέχον το κουτάκι με πρώτο σύμβολο του $x + 1$ και σταμάτα;*

# Παραδείγματα TM II




Σχήμα: TM σε μορφή διαγράμματος καταστάσεων

# Παραδείγματα TM III

Παράδειγμα λειτουργίας με είσοδο $x = 1011$:

$$\begin{array}{ccccccc}
(q_0, \underline{1}011) & \vdash & (q_0, 1\underline{0}11) & \vdash & (q_0, 10\underline{1}1) & \vdash & (q_0, 101\underline{1}) & \vdash \\
(q_0, 1011\underline{\sqcup}) & \vdash & (q_1, 101\underline{1}) & \vdash & (q_1, 10\underline{1}0) & \vdash & (q_1, 1\underline{0}00) & \vdash \\
(q_2, \underline{1}100) & \vdash & (q_2, \underline{\sqcup}1100) & \vdash & (\mathsf{HALT}, \underline{1}100) & & &
\end{array}$$

# Παραδείγματα TM IV

| | | | | |
|---|---|---|---|---|
| $< q_0$ | 0 | $q_0$ | 0 | R $>$ |
| $< q_0$ | 1 | $q_0$ | 1 | R $>$ |
| $< q_0$ | $\sqcup$ | $q_1$ | $\sqcup$ | L $>$ |
| $< q_1$ | 0 | $q_2$ | 1 | L $>$ |
| $< q_1$ | 1 | $q_1$ | 0 | L $>$ |
| $< q_1$ | $\sqcup$ | Halt | 1 | S $>$ |
| $< q_2$ | 0 | $q_2$ | 0 | L $>$ |
| $< q_2$ | 1 | $q_2$ | 1 | L $>$ |
| $< q_2$ | $\sqcup$ | Halt | $\sqcup$ | R $>$ |

Πίνακας: Πρόγραμμα TM

| | 0 | 1 | $\sqcup$ |
|---|---|---|---|
| $q_0$ | ($q_0$, 0, R) | ($q_0$, 1, R) | ($q_1$, $\sqcup$, L) |
| $q_1$ | ($q_2$, 1, L) | ($q_1$, 0, L) | (Halt, 1, S) |
| $q_2$ | ($q_2$, 0, L) | ($q_2$, 1, L) | (Halt, $\sqcup$, R) |

Πίνακας: TM σε μορφή πίνακα

# Αντιστοιχία προγραμμάτων WHILE και TM

| WHILE - πρόγραμμα | αντίστοιχη TM |
|---|---|
| succ(x) | κενή TM |
| pred(x) | $\{q_0 10 q_1,\ q_1 0 R q_2,\ q_2 10 q_2\}$ |
| zero(x) | $\{q_0 10 q_1,\ q_1 0 R q_0\}$ |
| ; | "ένωση" των δύο TM μέ αναπροσαρμογή των ονομάτων των καταστάσεων ώστε να είναι διαφορετικές και αναπροσαρμογή της παρα-στασης εξόδου της πρώτης TM που θα χρη-σημοποιηθεί ώς είσοδος για τη δέυτερη TM. |
| while | παρόμοια (βλ. επίσης το παράδειγμα παραπάνω) |

# Ντετερμινισμός και μη Ι

**Ντετερμινιστική μηχανή Turing (DTM)**:
$\delta : Q \times \Sigma \rightarrow Q \times A$, όπου $A = \Sigma \cup \{L, R\}$

**Μη-Ντετερμινιστική μηχανή Turing (NTM)**:
$\delta : Q \times \Sigma \rightarrow \mathrm{Pow}(Q \times A)$, όπου $A = \Sigma \cup \{L, R\}$

Η τριάδα που αποτελείται από τα **περιεχόμενα** της ταινίας της TM, την **θέση της κεφαλής** πάνω στην ταινία και την **τρέχουσα κατάσταση** της TM ονομάζεται **configuration** (στιγμιαία περιγραφή) της TM.

# Ντετερμινισμός και μη ΙΙ

- Σε μία **DTM** ο υπολογισμός είναι μία **γραμμική ακολουθία από configurations** (σε κάθε βήμα υπάρχει ακριβώς **μία** επόμενη νόμιμη configuration).
- Σε μία **NTM**, όμως, ο υπολογισμός περιγράφεται με ένα **δένδρο από configurations**, αφού σε κάθε βήμα είναι δυνατό να προκύψουν περισσότερες από μία επόμενες νόμιμες configurations.

Θα λέμε ότι μία συμβολοσειρά γίνεται **αποδεκτή** από μία NTM, αν γίνεται αποδεκτή **τουλάχιστον από ένα** υπολογιστικό μονοπάτι του δένδρου.

### Θεώρημα

*Για κάθε μη-ντετερμινιστική μηχανή Turing (NTM) υπάρχει* **ισοδύναμη** *ντετερμινιστική μηχανή Turing (DTM).*

# Παραλλαγές TM

Παραλλαγές Μηχανών Turing που έχουν την ίδια υπολογιστική δυνατότητα, όχι όμως και αποδοτικότητα (*efficiency*) είναι:

- πολλές ταινίες, μνήμη πλέγματος (grid memory), μνήμη περισσοτέρων διαστάσεων
- μεγαλύτερο $\Sigma$
- πολλές παράλληλες κεφαλές
- μη ντετερμινιστικές μεταβάσεις
- μίας κατευθύνσεως, απείρου μήκους ταινία
- εγγραφή και κίνηση της κεφαλής σε κάθε βήμα

# Υπολογιστικά Μοντέλα I

Μερικά υπολογιστικά μοντέλα είναι τα εξής:

- προγράμματα Pascal
- προγράμματα Pascal χωρίς αναδρομή (αφαίρεση αναδρομής με χρήση στοίβας)
- προγράμματα Pascal χωρίς αναδρομή και χωρίς άλλους τύπους δεδομένων εκτός από τους φυσικούς αριθμούς (επιτυγχάνεται με κωδικοποιήσεις)
- προγράμματα WHILE (μόνη δομή ελέγχου το WHILE)
- προγράμματα GOTO και IF
- Assembler-like RAM (random access machine), URM (universal register machine)
- SRM (single register machine) ένας καταχωρητής
- Μηχανή Turing (πρόσβαση μόνο σε ένα κουτάκι ''cell'' της ταινίας κάθε φορά)
- παραλλαγές από μηχανές Turing
- Thue: κανόνες επανεγγραφής (re-writing rules)
- Post: κανονικά συστήματα (normal systems)
- Church: λογισμός λ (λ-calculus)

# Υπολογιστικά Μοντέλα ΙΙ

- Curry: συνδυαστική λογική (combinatory logic)
- Markov: Μ. αλγόριθμοι
- Kleene: γενικά αναδρομικά σχήματα (general recursive schemes)
- Shepherdson-Sturgis, Elgott: URM, SRM, RAM, RASP
- Σχήματα McCarthy (if ... then ... else ... ⇒ LISP)

# Υπολογιστικά Μοντέλα III

Τα χαρακτηριστικά των παραπάνω μοντέλων είναι:

- ντετερμινιστική πολυπλοκότητα σε διακριτά βήματα
- πεπερασμένο σύνολο εντολών που εκτελούνται από επεξεργαστή
- απεριόριστη μνήμη

**Θεώρημα**

*$f$ είναι TM υπολογίσιμη ανν*

- *$f$ είναι WHILE-υπολογίσιμη*
- *$f$ είναι GOTO-υπολογίσιμη*
- *$f$ είναι PASCAL-υπολογίσιμη*
- *$f$ είναι μερικά αναδρομική (partial recursive)*

# Σχήματα McCarthy I

Το **σχήμα McCarthy** είναι ένα γενικότερο προγραμματιστικό σχήμα:

$$f(x, y) = \textbf{if } g(\ldots) = 0 \textbf{ then } h(\ldots) \textbf{ else } k(\ldots)$$

όπου $g(\ldots)$, $h(\ldots)$ και $k(\ldots)$ είναι όροι-συναρτήσεις που κατασκευάζονται με σύνθεση και προβολές με $x$, $y$, $f$ και άλλες "γνωστές" συναρτήσεις.

Παραδείγματα:

1. **επανάληψη**:
   $f(x) \;=\; \textbf{if } x = 0 \textbf{ then } a \textbf{ else } g(f(x-1))$
2. **πρωταρχική αναδρομή**:
   $f(x, y) \;=\; \textbf{if } y = 0 \textbf{ then } g(x) \textbf{ else } h(x, y-1, f(x, y-1))$
3. **μ-σχήμα**: $f(x) = f'(x, 0)$ όπου:
   $f'(x, y) \;=\; \textbf{if } g(x, y) = 0 \textbf{ then } y \textbf{ else } f'(x, y+1)$

# Σχήματα McCarthy II

Το McCarthy σχήμα ορίζει με μοναδικό τρόπο μία συνάρτηση $f$, στην πραγματικότητα το ελάχιστο σταθερό σημείο της $f = \tau(f)$, όπου $\tau$ είναι μία σύνθεση από συναρτησιακούς όρους και την **if-then-else** πρόταση. (Οι συναρτησιακοί όροι όπως και το **if-then-else** είναι συνεχείς!)

**Παράδειγμα:** $f(x, y) \;=\; \textbf{if } x = y \textbf{ then } y + 1 \textbf{ else } f(x, f(x \dot{-} 1, y + 1))$

Οι ακόλουθες συναρτήσεις αποτελούν όλες λύσεις (σταθερά σημεία):
(Επιβεβαιώστε το ελέγχοντας όλες τις δυνατές περιπτώσεις!)

$$f_1(x, y) = x + 1$$

$$f_2(x, y) = \begin{cases} x + 1, & \text{αν } x \geqslant y \\ y \dot{-} 1, & \text{διαφορετικά} \end{cases}$$

$$f_3(x, y) = \begin{cases} x + 1, & \text{αν } x \geqslant y \text{ και abs(x-y) είναι άρτιος} \\ \text{δεν ορίζεται}, & \text{διαφορετικά} \end{cases}$$

Ισχύουν: $f_3 \sqsubset f_1$, $f_3 \sqsubset f_2$, αλλά $f_2 \not\sqsubseteq f_1$, $f_1 \not\sqsubseteq f_2$.
Π.χ.: $f_1(2, 5) = 3$, $f_2(2, 5) = 4$, $f_3(2, 5)$: μη ορισμένο.

# Σχήματα McCarthy III

**Τι τιμή θα υπολογίσει μία υλοποίηση σε PASCAL για $f(2,5)$;**

PASCAL συμβολισμός:

**function** $f(x,y$: integer): integer;
**begin**
  **if** $x = y$ **then** $f := y+1$ **else** $f := f(x, f(x-1, y+1))$
**end**

PASCAL υπολογισμός:

$$\begin{aligned} f(2,5) &\vdash \textbf{if } 2=5 \textbf{ then } 5+1 \textbf{ else } f(2,f(1,6)) \vdash f(2,f(1,6)) \\ &\vdash f(2, \textbf{if } 1=6 \textbf{ then } 6+1 \textbf{ else } f(1,f(0,7))) \vdash f(2,f(1,f(0,7))) \\ &\vdash \ldots \text{ επ' άπειρον, δηλαδή } f(2,5) \text{ δεν ορίζεται (επίσης και το } f_3(2,5)). \end{aligned}$$

Στην παραπάνω ακολουθία υπολογισμού χρησιμοποιήσαμε δύο διαφορετικούς μηχανισμούς: **απλοποιήσεις (simplifications)**, που είναι μαθηματικοί υπολογισμοί (if-then-else) και **αντικαταστάσεις (substitutions)** μίας ή περισσοτέρων εμφανίσεων του όρου $f(\ldots)$ με $\tau(f)(\ldots)$.

# Υπολογιστικές Στρατηγικές Ι

Μία **υπολογιστική στρατηγική** καθορίζει ποιοι όροι πρέπει να αντικατασταθούν. Δίνουμε μερικά παραδείγματα στρατηγικών (υπογραμμίζουμε τους όρους που θα αντικατασταθούν):

- LI: **αριστερότερη-εσωτερικότερη (leftmost-innermost)**. Αντιστοιχία με την "κλήση με τιμή" ("call-by-value"), δηλαδή:

$$f(0, \underline{f}(0,1)) + f(f(2,0), f(3,1))$$

- LO: **αριστερότερη-εξωτερικότερη (leftmost-outermost)**. Αντιστοιχία με την "κλήση κατά όνομα" ("call-by-name"), δηλαδή:

$$\underline{f}(0, f(0,1)) + f(f(2,0), f(3,1))$$

- PI: **παράλληλη-εσωτερικότερη (parallel-innermost)**, δηλαδή:

$$f(0, \underline{f}(0,1)) + f(\underline{f}(2,0), \underline{f}(3,1))$$

# Υπολογιστικές Στρατηγικές ΙΙ

- PO: **παράλληλη-εξωτερικότερη (parallel-outermost)**, δηλαδή:

$$\underline{f}(0, f(0,1)) + \underline{f}(f(2,0), f(3,1))$$

- FA: **ελευθέρου ορίσματος (free-argument)** (υπάρχει τουλάχιστον ένα όρισμα ελεύθερο, δηλαδή δεν περιέχει την $f$):

$$\underline{f}(0, \underline{f}(0,1)) + f(\underline{f}(2,0), \underline{f}(3,1))$$

- FS: **πλήρους αντικατάστασης (full substitution)**, δηλαδή:

$$\underline{f}(0, \underline{f}(0,1)) + \underline{f}(\underline{f}(2,0), \underline{f}(3,1))$$

- **κανονική στρατηγική (normal strategy)**: βλέπε παρακάτω.

# Υπολογιστικές Στρατηγικές ΙΙΙ

**Παρατήρηση**: Δεν υπολογίζουν όλες οι στρατηγικές το ελάχιστο σταθερό σημείο. Για παράδειγμα, έστω:

$$f(x, y) \; = \; \textbf{if } x = 0 \textbf{ then } 1 \textbf{ else } f(x-1, f(x, y)).$$

Το ελάχιστο σταθερό σημείο είναι $f^*(x, y) = 1$ (επαλήθευση αργότερα). Από την άλλη, η LI στρατηγική υπολογίζει:

$$f_{\text{LI}}(x, z) = \begin{cases} 1, & \text{αν } x = 0 \\ \text{μη ορισμένη}, & \text{διαφορετικά} \end{cases}$$

δηλαδή $f(1, 0) \vdash f(0, f(1, 0)) \vdash f(0, f(0, f(1, 0))) \vdash \ldots$

# Υπολογιστικές Στρατηγικές IV

Εύρεση του ελάχιστου σταθερού σημείου ενός McCarthy σχήματος:

$$f(x) \quad = \quad \textbf{if } x > 10 \textbf{ then } x - 5 \textbf{ else } f(x+3)$$

Ελάχιστο σταθερό σημείο:

$$f^*(x) = \begin{cases} x - 5, & \text{αν } x > 10 \\ x - 2 & \text{αν } 8 \leqslant x \leqslant 10 \\ x + 1 & \text{αν } 5 \leqslant x \leqslant 7 \\ x + 4 & \text{αν } 2 \leqslant x \leqslant 4 \\ x + 7 & \text{αν } x \leqslant 1 \end{cases}$$

Συνολοθεωρητικός συμβολισμός:

$$F = \{\langle x, x-5 \rangle \mid x > 10\} \cup \{\langle x, z \rangle \mid x \leqslant 10 \wedge \langle x+3, z \rangle \in F\}$$

Εύρεση ελάχιστου σταθερού σημείου του

$$\tau(Y) = \{\langle x, x-5 \rangle \mid x > 10\} \cup \{\langle x, z \rangle \mid x \leqslant 10 \wedge \langle x+3, z \rangle \in Y\}$$

# Υπολογιστικές Στρατηγικές V

**Παρατήρηση**: Το $\tau$ είναι συνεχές, άρα το θεώρημα σταθερού σημείου δίνει:

$$\tau^0(\varnothing) = \varnothing$$
$$\tau^1(\varnothing) = \tau(\varnothing) = \{\langle x, x-5\rangle \mid x > 10\}$$
$$\tau^2(\varnothing) = \tau(\tau(\varnothing) = \tau(\{\langle x, x-5\rangle \mid x > 10\} \cup \{\langle 10, 8\rangle, \langle 9, 7\rangle, \langle 8, 6\rangle\}$$
$$\tau^3(\varnothing) = \tau(\tau^2(\varnothing) = \tau(\varnothing) \cup \{\langle 10, 8\rangle, \langle 9, 7\rangle, \langle 8, 6\rangle, \langle 7, 8\rangle, \langle 6, 7\rangle, \langle 6, 7\rangle, \langle 5, 6\rangle\}$$
$$\tau^4(\varnothing) = \tau^3(\varnothing) \cup \{\langle 4, 8\rangle, \langle 3, 7\rangle, \langle 2, 6\rangle\}$$
$$\tau^5(\varnothing) = \tau^4(\varnothing) \cup \{\langle 1, 8\rangle, \langle 0, 7\rangle\}$$
$$\tau^6(\varnothing) = \tau^5(\varnothing)$$

και για όλα τα $i \geqslant 5$: $\tau^i(\varnothing) = \tau^5(\varnothing)$, άρα $F = \bigcup_i \tau^i(\varnothing) = \tau^5(\varnothing)$ που δίνει ακριβώς την παραπάνω ορισμένη $f^*$.

# Υπολογιστικές Στρατηγικές VI

Μέχρι στιγμής υποθέσαμε (προσποιηθήκαμε) ότι ασχολούμαστε μόνο με **αυστηρές** (strict) συναρτήσεις, δηλαδή συναρτήσεις που δεν ορίζονται όταν δεν ορίζονται κάποια από τα ορίσματά τους. Όμως, συναρτήσεις που ορίζονται χρησιμοποιώντας σύνθεση και **if**-**then**-**else** δεν έχουν πλέον αυτή την ιδιότητα! Θέλουμε **if** $x = x$ **then** $5$ **else** $y + 3$ να ορίζεται και να είναι ίσο με 5 ακόμα και αν το $y$ δεν ορίζεται.

Για να δώσουμε μία πιο συγκεκριμένη περιγραφή της κατάστασης επεκτείνουμε το βασικό πεδίο $\mathbb{N}$ (φυσικοί) με ένα νέο σύμβολο $\perp$ που συμβολίζει την *μη ορισμένη* "τιμή": $\mathbb{N}^{\perp} = \mathbb{N} \cup \{\perp\}$. Προφανώς ισχύει: $\perp \sqsubseteq x, \ \forall x \in \mathbb{N}^{\perp}$ (δηλαδή το $\perp$ είναι λιγότερο, ή εξίσου, ορισμένο σε σχέση με οποιοδήποτε στοιχείο του πεδίου μας). Συνεπώς, αντίστοιχα $\perp \sqcup x = x, \ \forall x \in \mathbb{N}^{\perp}$.

Μπορούμε να επεκτείνουμε αυτή τη γενίκευση του $\sqsubseteq$ και $\sqcup$ σε $n$-άδες, δηλαδή $\langle x, y, \perp \rangle \sqsubseteq \langle x, y, z \rangle \forall x, y, z \in \mathbb{N}^{\perp}$ και $\{\langle x, y, \perp \rangle\} \sqcup \{\langle x, y, z \rangle\} = \{\langle x, y, z \rangle\}$.

Επιπρόσθετα, έχουμε ακόμα την ακόλουθη ιδιότητα συνέπειας:

$$f(x, \perp) = y \text{ συνεπάγεται } \forall z \colon f(x, z) = y.$$

# Υπολογιστικές Στρατηγικές VII

Επιστρέφοντας στο αρχικό παράδειγμα του κεφαλαίου μπορούμε να ξεκαθαρίσουμε τον ορισμό της $f_1$:

$$f_1(x,y) = \begin{cases} x+1, & \text{αν } x \text{ και } y \text{ είναι διαφορετικά από } \perp \\ \perp, & \text{διαφορετικά} \end{cases}$$

Μπορούμε επίσης να επαληθεύσουμε ότι

$$f^*(x,y) = \begin{cases} 1, & \text{αν } x \neq \perp \\ \perp, & \text{διαφορετικά} \end{cases}$$

είναι το ελάχιστο σταθερό σημείο της

$$f(x,y) \;=\; \textbf{if } x = 0 \textbf{ then } 1 \textbf{ else } f(x-1, f(x,y)).$$

# Υπολογιστικές Στρατηγικές VIII

Συνολοθεωρητικός συμβολισμός:

$$F = \tau(F) = \{\langle 0, y, 1\rangle \mid y \in \mathbb{N}^{\perp}\} \sqcup \\ \{\langle x, y, z\rangle \mid x \neq 0 \ \wedge \ \exists u \langle x, y, u\rangle \in F \ \wedge \ \langle x-1, u, z\rangle \in F\}$$

Για να υπολογίσουμε επαναληπτικά το ελάχιστο σταθερό σημείο πρέπει να ξεκινήσουμε με $\Omega$ αντί για $\varnothing$. $\Omega$ είναι η "πουθενά ορισμένη" συνάρτηση, δηλαδή $\Omega = \{\langle x, y, \perp\rangle \mid x, y \in \mathbb{N}^{\perp}\}$.

$$\begin{aligned}
\tau^0(\Omega) &= \Omega = \{\langle x, y, \perp\rangle\} \\
\tau^1(\Omega) &= \tau(\Omega) = \{\langle 0, y, 1\rangle\} \cup \{\langle x, y, \perp\rangle \mid x \neq 0\} \\
\tau^2(\Omega) &= \tau(\tau(\Omega)) = \{\langle 0, y, 1\rangle, \langle 1, y, 1\rangle\} \cup \{\langle x, y, \perp\rangle \mid x > 1 \vee x = \perp\} \\
\tau^3(\Omega) &= \{\langle 0, y, 1\rangle, \langle 1, y, 1\rangle, \langle 2, y, 1\rangle\} \cup \{\langle x, y, \perp\rangle \mid x > 2 \vee x = \perp\} \\
F = \bigcup_i \tau_i(\Omega) &= \{\langle x, y, 1\rangle \mid x \in \mathbb{N} \wedge y \in \mathbb{N}^{\perp}\} \cup \{\langle \perp, y, \perp\rangle \mid y \in \mathbb{N}^{\perp}\}
\end{aligned}$$

# Υπολογιστικές Στρατηγικές IX

Παρόμοια μπορεί να δειχθεί ότι η $f_3$ είναι το ελάχιστο σταθερό σημείο για το αρχικό παράδειγμα του κεφαλαίου.

Συνολοθεωρητικός συμβολισμός:

$$F = \{\langle x, x, x+1 \rangle\} \cup \{x, y, z \mid x \neq y \wedge \exists u(\langle x \dot{-} 1, y+1, u \rangle \in F \wedge \langle x, u, z \rangle \in F)\}$$

# Θεώρημα Cadiou I

## Θεώρημα (Cadiou)

*Κάθε υπολογιστική στρατηγική υπολογίζει μία μερική συνάρτηση $g$ που είναι λιγότερο ή εξίσου ορισμένη με το ελάχιστο σταθερό σημείο $f^*$ δηλ. $g \sqsubseteq f^*$.*

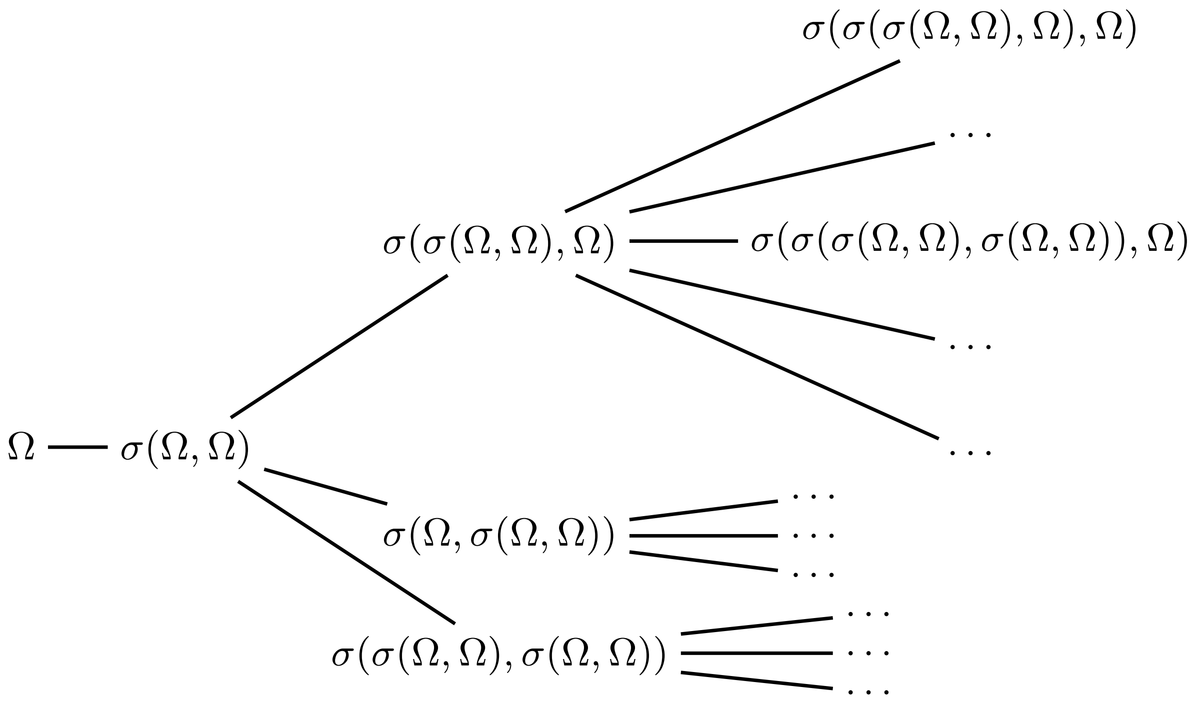
## Πόρισμα

*Αν κάποιος υπολογισμός σύμφωνος με κάποια υπολογιστική στρατηγική τερματίζει, τότε δίνει πάντοτε την σωστή τιμή της συνάρτησης ελαχίστου σταθερού σημείου $f^*$.*

**Ιδέα της απόδειξης:**
Συμβολισμός: $\sigma(f, f)$ αντί του $\tau(f)$ για να υποδηλώσουμε ακριβώς δύο εμφανίσεις του $f$ στο $\tau$.

# Θεώρημα Cadiou II

Θεωρήσατε το ακόλουθο δέντρο που έχει (ως μονοπάτια) υπολογισμούς όλων των δυνατών στρατηγικών:



# Θεώρημα Cadiou III

- Κατά μήκος κάθε μονοπατιού έχουμε μία μη φθίνουσα ακολουθία από συναρτήσεις (όλο και περισσότερο ορισμένες).
- Το χαμηλότερο μονοπάτι αντιστοιχεί στο: $\Omega - \tau(\Omega) - \tau(\tau(\Omega)) - \ldots$ και οδηγεί στο ελάχιστο σταθερό σημείο $f^*$.
- Σε κάθε επίπεδο του δέντρου όλες οι συναρτήσεις στους κόμβους είναι λιγότερο (ή εξίσου) ορισμένες από την χαμηλότερη.
- Επομένως, το όριο οποιουδήποτε μονοπατιού είναι λιγότερο (ή εξίσου) ορισμένο από το όριο του χαμηλότερου μονοπατιού, που είναι το ελάχιστο σταθερό σημείο $f^*$.

$\square$

# Θεώρημα Cadiou IV

**Παρατηρήσεις:**

- Το χαμηλότερο μονοπάτι αντιστοιχεί στην στρατηγική FS.
- Στην απόδειξη χρησιμοποιήσαμε μία τεχνική που ονομάζεται "υπολογιστική επαγωγή" (computational induction): αν θέλουμε να αποδείξουμε την ιδιότητα $A(f)$ για μια συνάρτηση $f$ ορισμένη με ένα σχήμα τύπου McCarthy
  1. δείχνουμε $A(\Omega)$
  2. υποθέτοντας $A(g)$, δείχνουμε $A(\tau(g))$
  3. υποθέτοντας $\forall i, A(\tau^i(\Omega))$, δείχνουμε $A(\bigsqcup_i \tau^i(\Omega))$
- Η LISP είναι μία γλώσσα προγραμματισμού βασισμένη σε αναδρομές τύπου McCarthy.
- Οι outermost στρατηγικές είναι στρατηγικές σταθερού σημείου (διαισθητικά, διότι το εμφωλιασμένο βάθος των υπαρχουσών εμφανίσεων δεν αυξάνεται). Για αυτό οι LO, PO, FS και FA είναι στρατηγικές σταθερού σημείου, αλλά οι LI, PI δεν είναι (παρότι πιθανώς έχουν λιγότερες αντικαταστάσεις αν τερματίζουν με τη τιμή $f^*$).

# Θεώρημα Cadiou V

- **Κανονική στρατηγική**: μία στρατηγική σταθερού σημείου με ελάχιστο αριθμό αντικαταστάσεων (όχι κατ' ανάγκη ελάχιστο αριθμό βημάτων αντικατάστασης). *Σύνθημα*: "Ποτέ μην κάνεις κάτι σήμερα που μπορείς να αναβάλεις για αύριο" δηλαδή κοίτα μπροστά: Έλεγξε αν μετά από την αντικατάσταση της L.O. εμφάνισης, μπορείς απ' ευθείας να υπολογίσεις την **if**-συνθήκη. Στην περίπτωση που αυτό γίνεται, κάνε αυτή την αντικατάσταση. Στην περίπτωση που δεν γίνεται έλεγξε με τον ίδιο τρόπο την L.O. εμφάνιση στην **if**-συνθήκη που δεν μπορεί να υπολογιστεί απευθείας, κ.τ.λ. (Τελικά θα βρεις μία "καλή" εμφάνιση, που δίνει υπολογίσιμες **if**-συνθήκες.)
  Παράδειγμα:

$$\begin{aligned} f(x,y) = \; & \textbf{if } x \leqslant 10 \textbf{ then } x+y \\ & \textbf{else } f(x-10, f(x-5, y+1)) + f(f(3, y+2), y+10) \end{aligned}$$

# Θεώρημα Cadiou VI

Κανονική στρατηγική:

$$\begin{aligned}
\underline{f}(12,5) &\vdash \underline{f}(2, f(7,6)) + f(f(3,7),15) \\
&\vdash 2 + \underline{f}(7,6) + f(f(3,7),15) \\
&\vdash 2 + 13 + f(\underline{f}(3,7),15) \\
&\vdash 15 + \underline{f}(10,15) \\
&\vdash 40 && \text{(σε 5 υπολογιστικά βήματα)}
\end{aligned}$$