# ΠΑΡΑΛΛΗΛΟΙ ΑΛΓΟΡΙΘΜΟΙ

*ΕΔΙΠ* Μαρία Λουκά

Τμήμα Πληροφορικής και Τηλεπικοινωνιών

ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ

**Εθνικόν και Καποδιστριακόν Πανεπιστήμιον Αθηνών**

20 Μαρτίου 2020

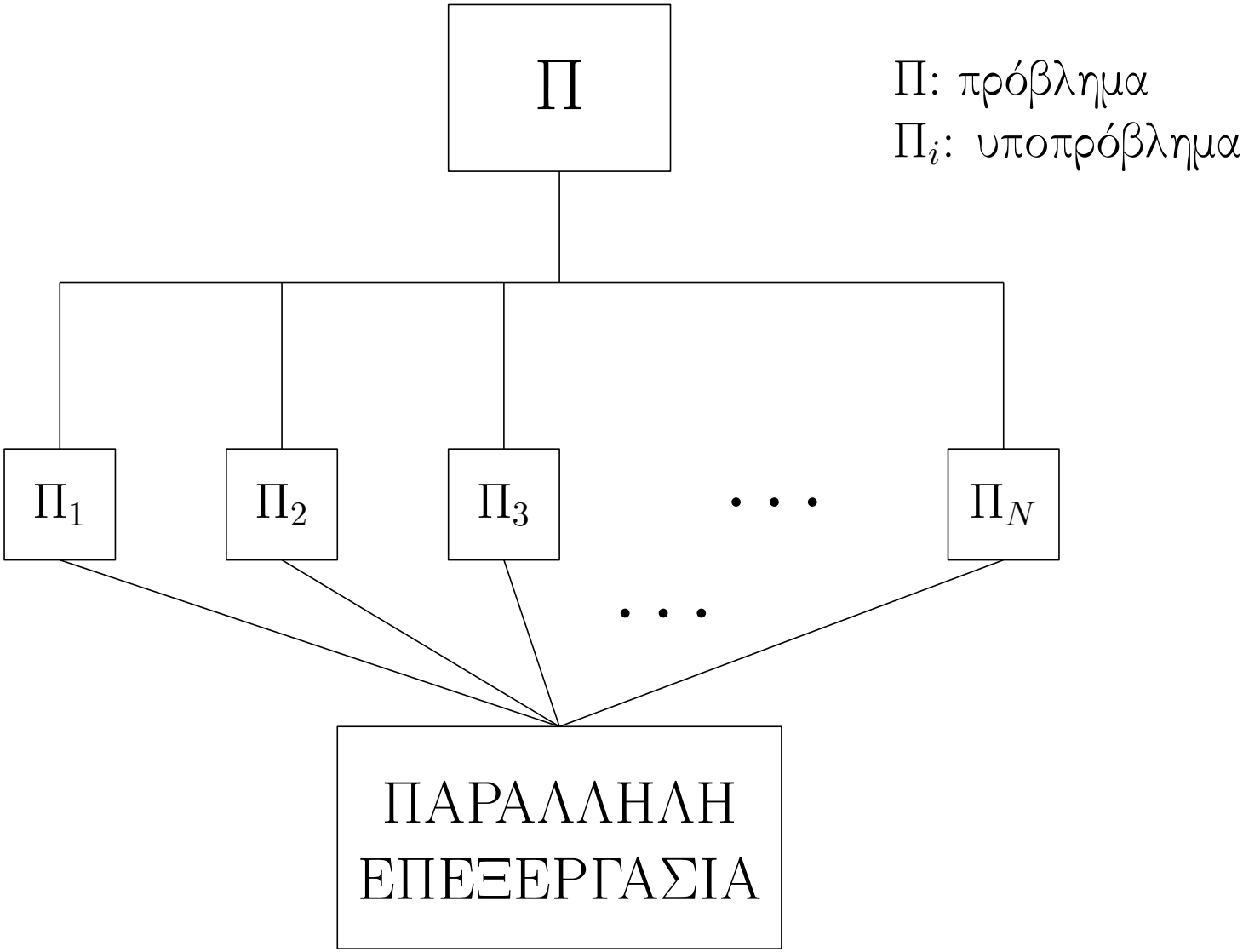
# Κεφάλαιο 1
# Εισαγωγή (Introduction)

# Η ανάγκη για παράλληλους υπολογιστές

- Οι εφαρμογές απαιτούν χρονοβόρους υπολογισμούς:
  - σχεδιασμός αεροπλάνων
  - αριθμητική προσομοίωση σε προβλήματα Υπολογιστικής Δυναμικής των Ρευστών
- Η αύξηση των υπολογισμών θα κορεστεί.

**Λύση:** Χρήση της παραλληλίας

# Η ανάγκη για παράλληλους υπολογιστές




Σχήμα: Διάγραμμα block παράλληλης επεξεργασίας

# Η ανάγκη για παράλληλους υπολογιστές

**Παράλληλος Αλγόριθμος**

Μια μέθοδος λύσης για ένα δεδομένο πρόβλημα που πρόκειται να υλοποιηθεί σε ένα παράλληλο υπολογιστή

**Παράλληλος Υπολογιστής**

Ένας υπολογιστής με πολλές μονάδες επεξεργασίας ή επεξεργαστές

**Μοντέλα Υπολογισμού**

- SISD — S: Single
- MISD — I: Instruction
- SIMD — D: Data
- MIMD — M: Multiple

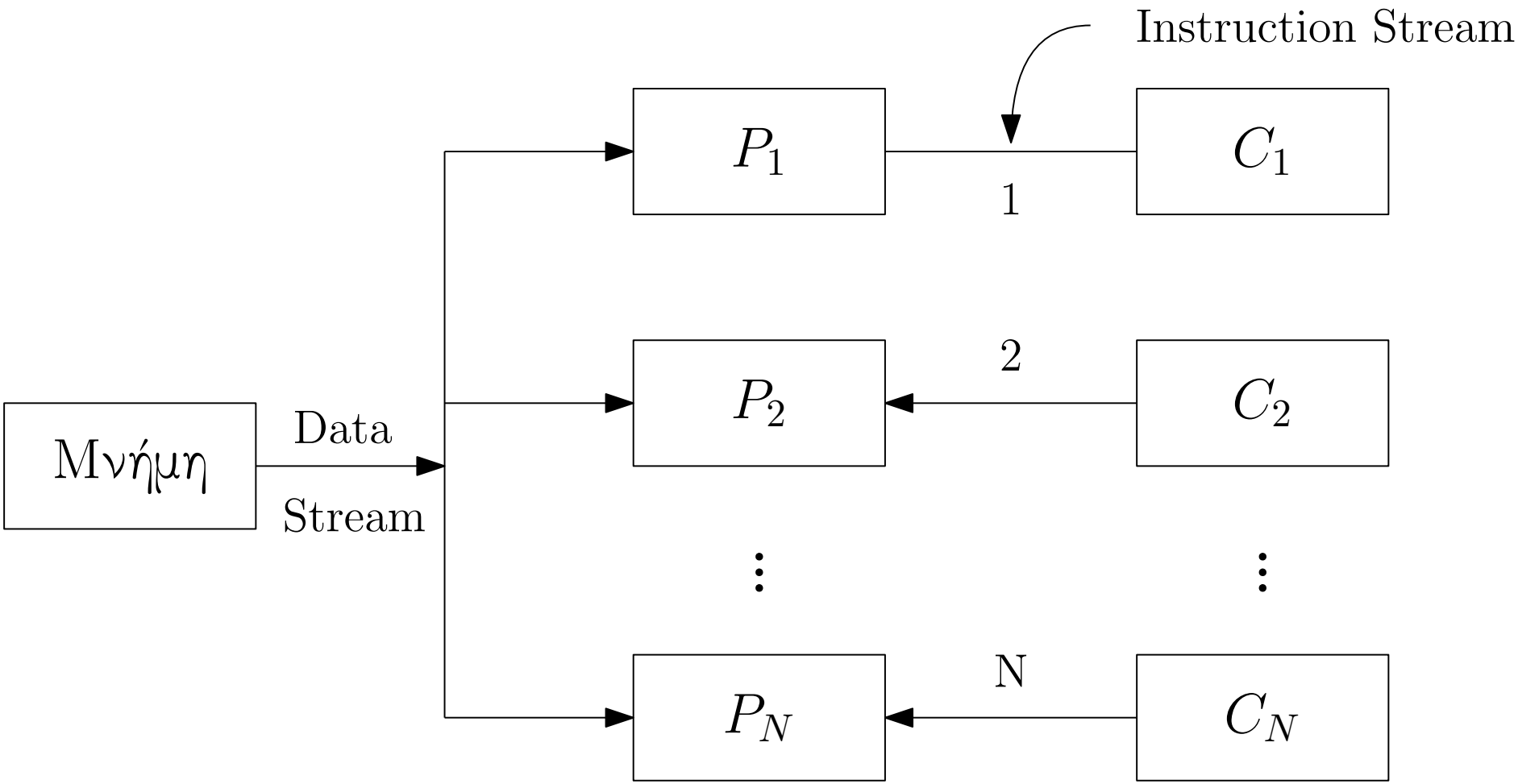
# Η ανάγκη για παράλληλους υπολογιστές

**SISD**

- John von Neumann 1940
- Ακολουθιακός (σειριακός) Αλγόριθμος


Σχήμα: Single Instruction Single Data

# MISD




Σχήμα: Multiple Instruction Single Data

# MISD

**Παράδειγμα:** Έλεγχος εάν ένας αριθμός $z$ είναι πρώτος

**Παράλληλος Αλγόριθμος**

- Έστω ότι υπάρχουν τόσοι επεξεργαστές όσοι είναι και οι υποψήφιοι διαιρέτες του $z$.
- Έλεγχος από κάθε επεξεργαστή αν ο υποψήφιος διαιρέτης διαιρεί το $z$.

**Παράδειγμα:** Εύρεση του συνόλου (κατηγορίας) στην οποία ανήκει ένα αντικείμενο

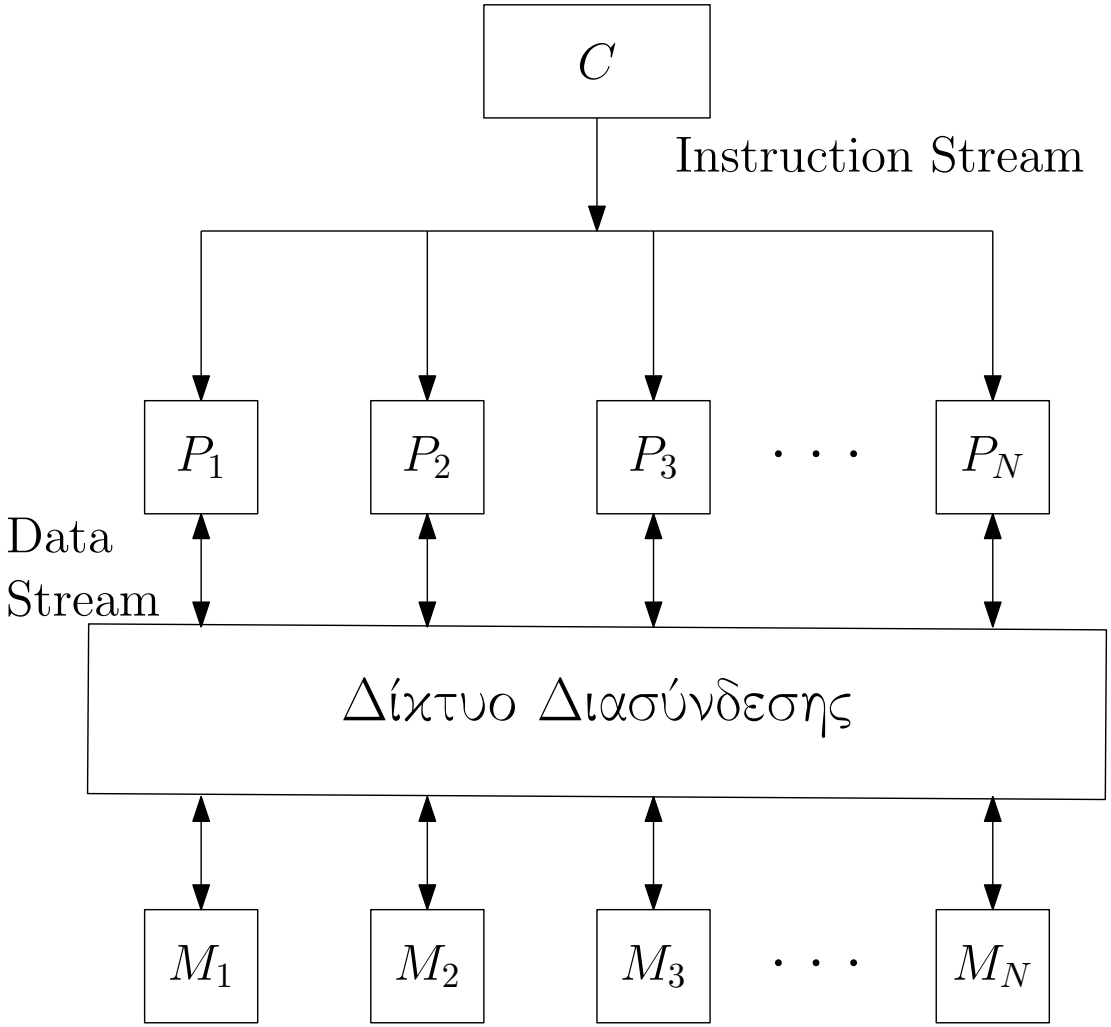
**Παράλληλος Αλγόριθμος**

- Υποθέτουμε ότι υπάρχουν τόσοι επεξεργαστές όσες είναι οι κατηγορίες.
- Στέλνεται το αντικείμενο σε κάθε επεξεργαστή, ο οποίος ελέγχει αν ανήκει στην κατηγορία του.

**Μειονέκτημα**

- Ειδικής μορφής υπολογισμοί

# SIMD




Σχήμα: Single Instruction Multiple Data

**αν** $(B = 0)$
$C = A$
**αλλιώς**
$C = A/B$

| | Επεξεργαστής 0 | Επεξεργαστής 1 | Επεξεργαστής 2 | Επεξεργαστής 3 |
|---|---|---|---|---|
| $A$ | 5 | 4 | 1 | 0 |
| $B$ | 0 | 2 | 1 | 0 |
| $C$ | 0 | 0 | 0 | 0 |

Αρχικές τιμές

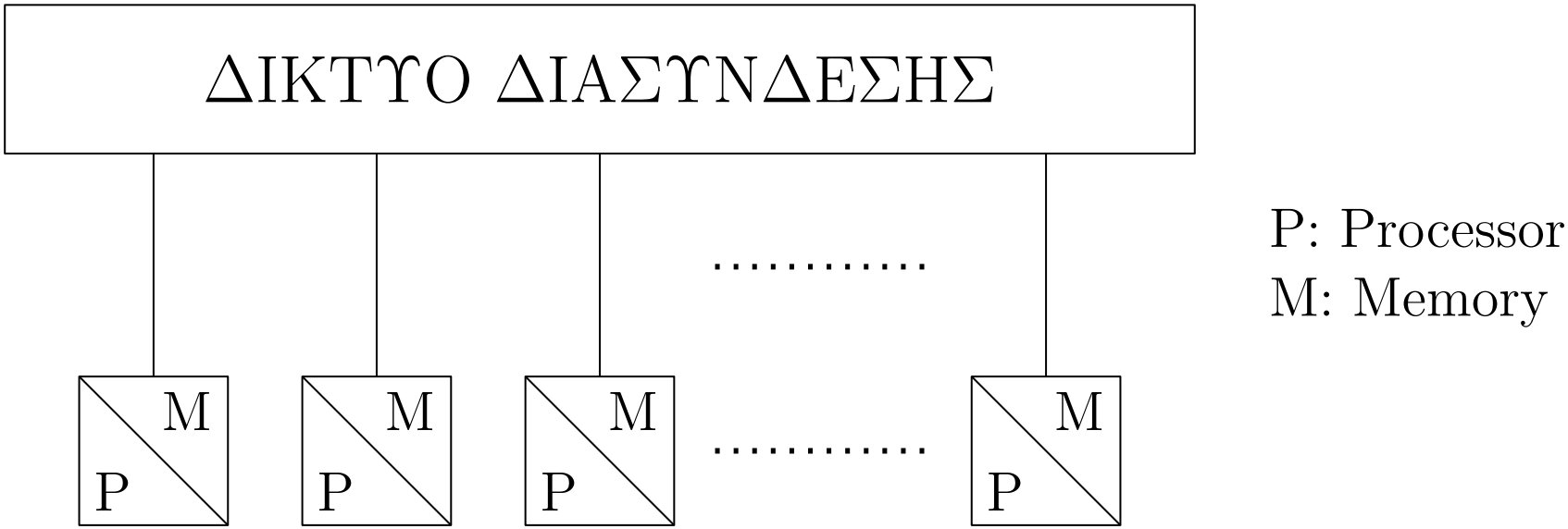
| | Επεξεργαστής 0 | Επεξεργαστής 1 | Επεξεργαστής 2 | Επεξεργαστής 3 |
|---|---|---|---|---|
| | | Αδρανής | Αδρανής | |
| $A$ | 5 | 4 | 1 | 0 |
| $B$ | 0 | 2 | 1 | 0 |
| $C$ | 5 | 0 | 0 | 0 |

Βήμα 1

| | Επεξεργαστής 0 | Επεξεργαστής 1 | Επεξεργαστής 2 | Επεξεργαστής 3 |
|---|---|---|---|---|
| | Αδρανής | | | Αδρανής |
| $A$ | 5 | 4 | 1 | 0 |
| $B$ | 0 | 2 | 1 | 0 |
| $C$ | 5 | 2 | 1 | 0 |

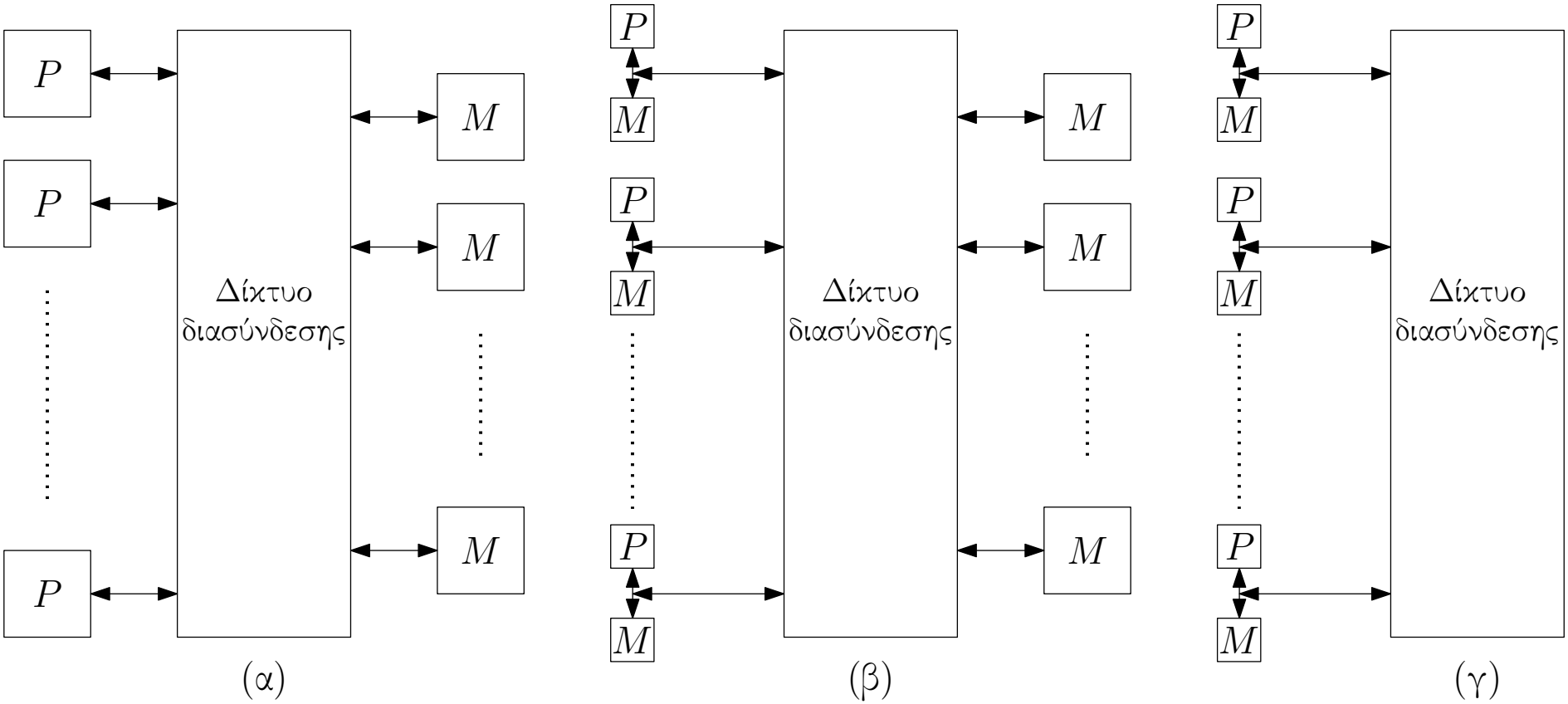
Βήμα 2

# SIMD




Σχήμα: Μία τυπική αρχιτεκτονική διαβίβασης μηνυμάτων (message passing)

# SIMD




Σχήμα: Κοινή Μνήμη: (α) UMA, (β) NUMA με τοπική και ολική μνήμη, (γ) NUMA με τοπική μνήμη μόνο

# SIMD

**Διαμοιραζόμενης Μνήμης Υπολογιστές**

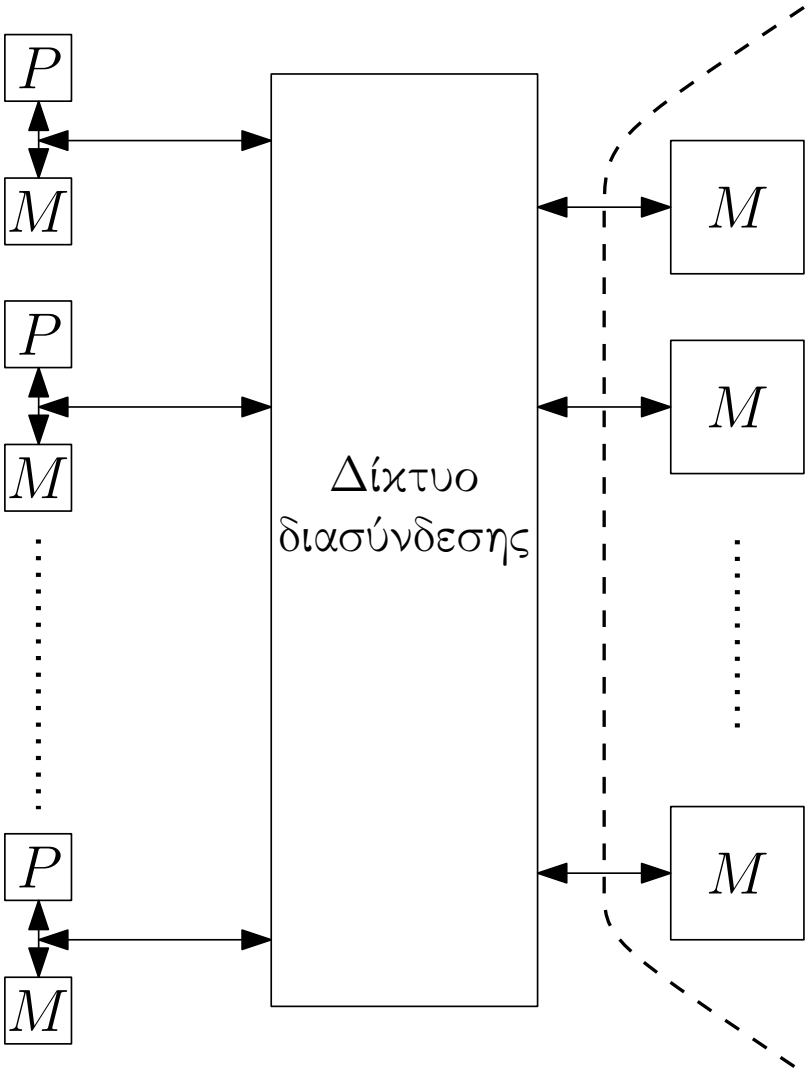
- C. mmp
- NYU Ultracomputer

**Μειονεκτήματα**

- μεγάλο εύρος δικτύου διασύνδεσης
- αργή προσπέλαση μνήμης

Θεραπεία $\rightarrow$ τοπική μνήμη

# SIMD




Διαμοιραζόμενης
Μνήμης

**UMA**

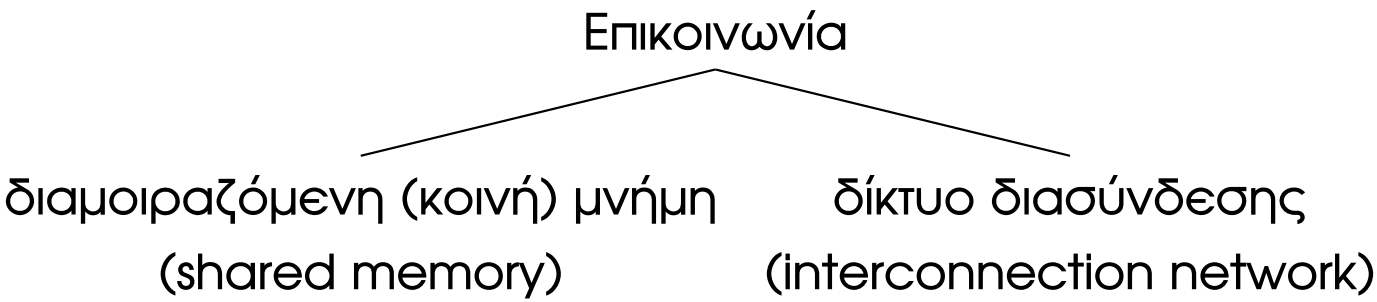
**NUMA**
TC-2000
KSR-1

Σχήμα: Υπολογιστής Διαμοιραζόμενης Μνήμης

# SIMD

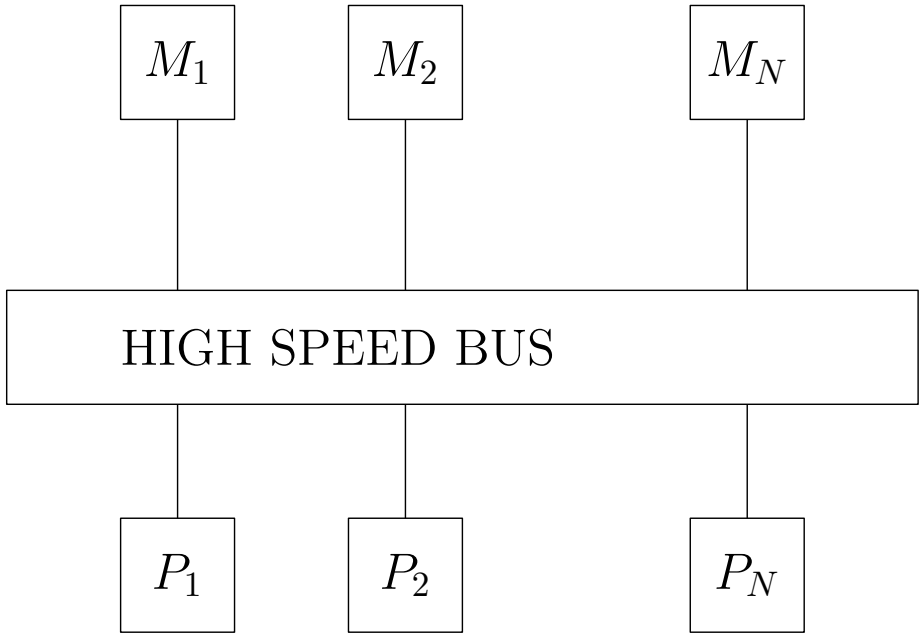
- Κάθε επεξεργαστής έχει τη δική του τοπική μνήμη (πρόγραμμα και δεδομένα)
- Οι επεξεργαστές λειτουργούν σύγχρονα: σε κάθε βήμα, όλοι οι επεξεργαστές εκτελούν την ίδια εντολή, ο κάθε ένας σε διαφορετικά δεδομένα
- Κατάλληλοι για παράλληλα-δεδομένα (data-parallel)

distributed memory ή/και message-passing



# Διαμοιραζόμενης Μνήμης (PRAM) SIMD

- EREW
- CREW
- ERCW
- CRCW



# Διαμοιραζόμενης Μνήμης (PRAM) SIMD

**Παράδειγμα**

Να προσδιοριστεί αν ένα δεδομένο στοιχείο x βρίσκεται σε ένα αρχείο με n θέσεις.

Ακολουθιακός αλγόριθμος $\longrightarrow$ απαιτεί n βήματα.

Παράλληλος αλγόριθμος:

EREW SM SIMD με $N \leq n$ επεξεργαστές $P_1, P_2, \ldots P_N$.

# Διαμοιραζόμενης Μνήμης (PRAM) SIMD

- Μετάδοση (Broadcasting) του x στους επεξεργαστές
  1. $P_1$ διαβάζει το x και το μεταδίδει στον $P_2$
  2. Οι $P_1$ και $P_2$ μεταδίδουν ταυτόχρονα το x στους $P_3$ και $P_4$ αντίστοιχα, κ.ο.κ.
  3. Οι $P_1$, $P_2$, $P_3$ και $P_4$ μεταδίδουν ταυτόχρονα το x στους $P_5$, $P_6$, $P_7$ και $P_8$ αντίστοιχα, κ.ο.κ.

  Η μετάδοση του x σε όλους τους επεξεργαστές απαιτεί $\log_2 N$ βήματα
- Χωρισμός του αρχείου σε υποαρχεία με $n/N$ στοιχεία
  $P_i$ αναζητεί σε $n/N$ στοιχεία άρα $n/N$ βήματα

Σύνολο: $\log N + n/N$ βήματα

# Διαμοιραζόμενης Μνήμης (PRAM) SIMD

Αν χρησιμοποιηθεί μια λογική μεταβλητή $F$ για τη δήλωση ότι βρέθηκε το $x$, τότε

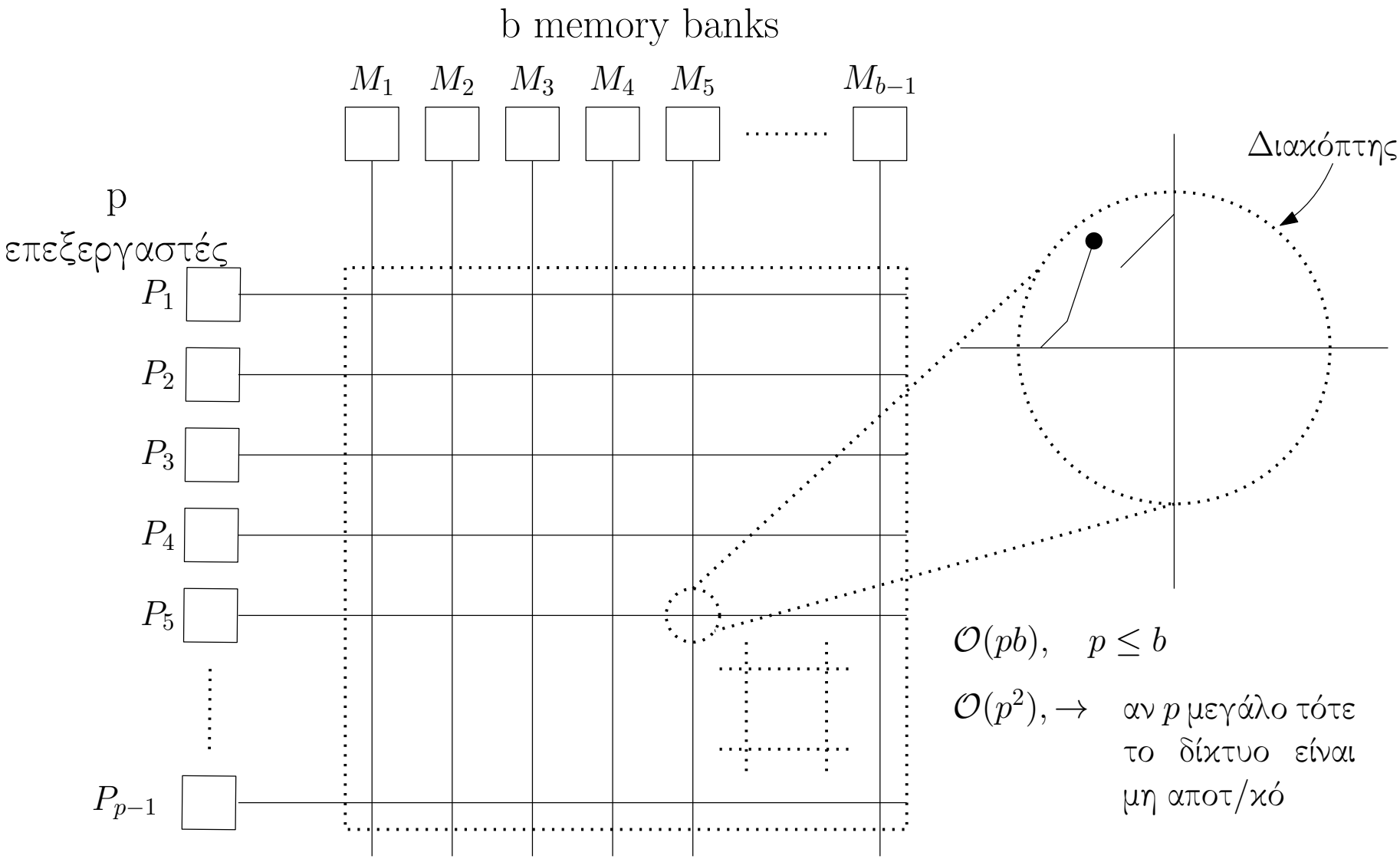
$$\log N + (n/N) \log N$$

γιατί σε κάθε βήμα της αναζήτησης απαιτείται η γνώση της $F$ σε όλους τους επεξεργαστές (broadcasting $\log N$).
Για CREW SM SIMD:

- 1 βήμα για μετάδοση του x
- 1 βήμα για μετάδοση της $F$ (τέλος κάθε αναζήτησης)

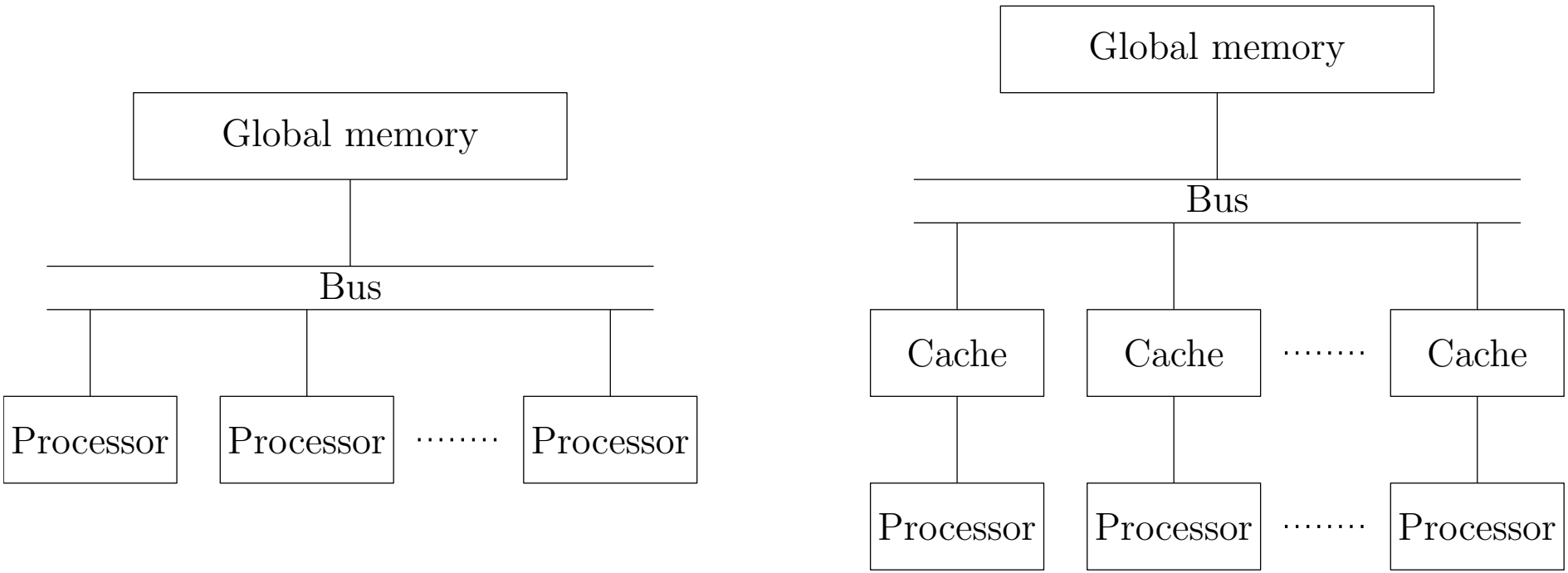
Άρα μόνο $n/N$ βήματα για αναζήτηση.

# Crossbar Δίκτυα Διασύνδεσης



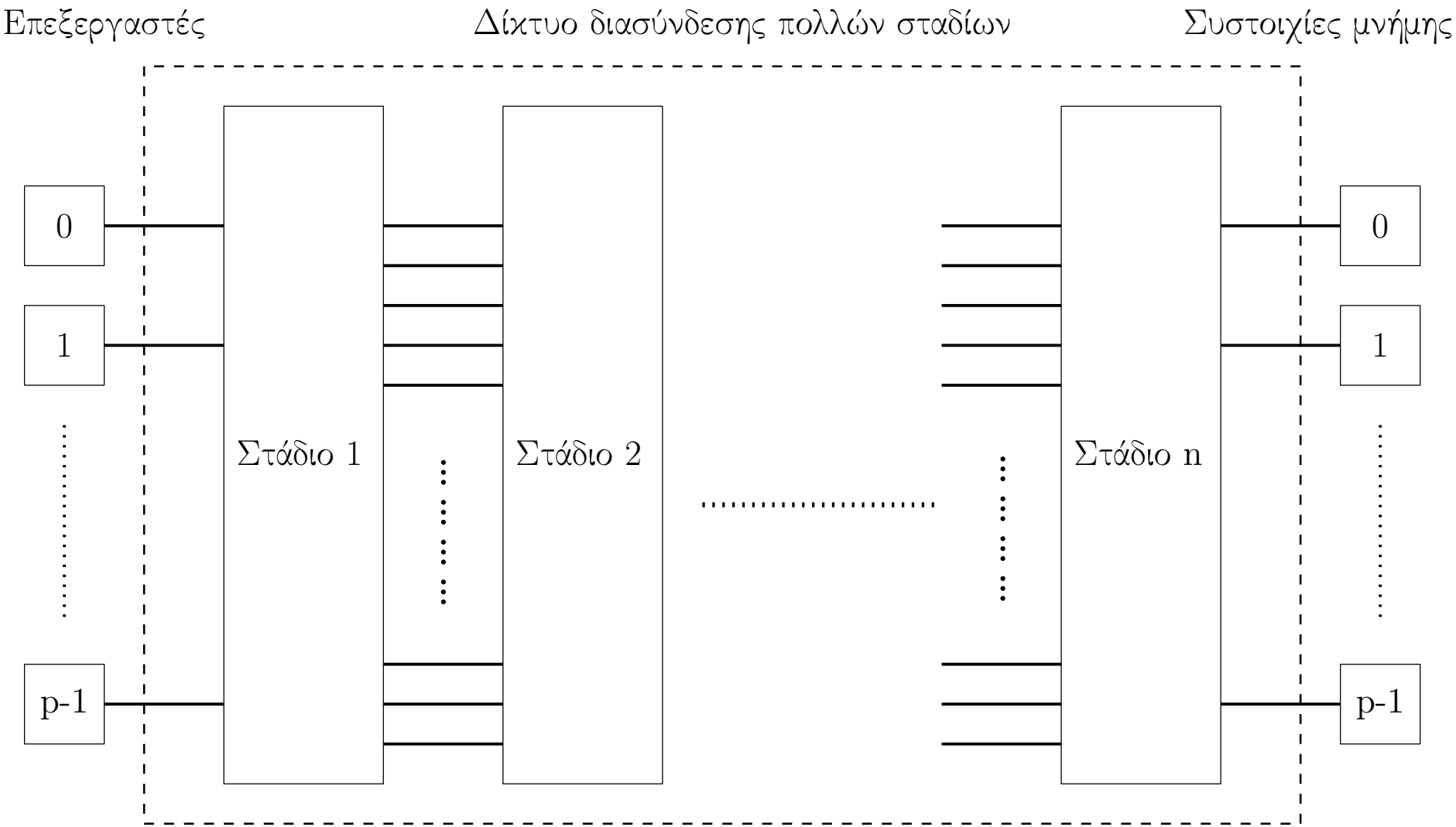

Σχήμα: Crossbar Switch: Cray Y-MP, Fujitsu VDD 500

# Bus-based Δίκτυα Διασύνδεσης

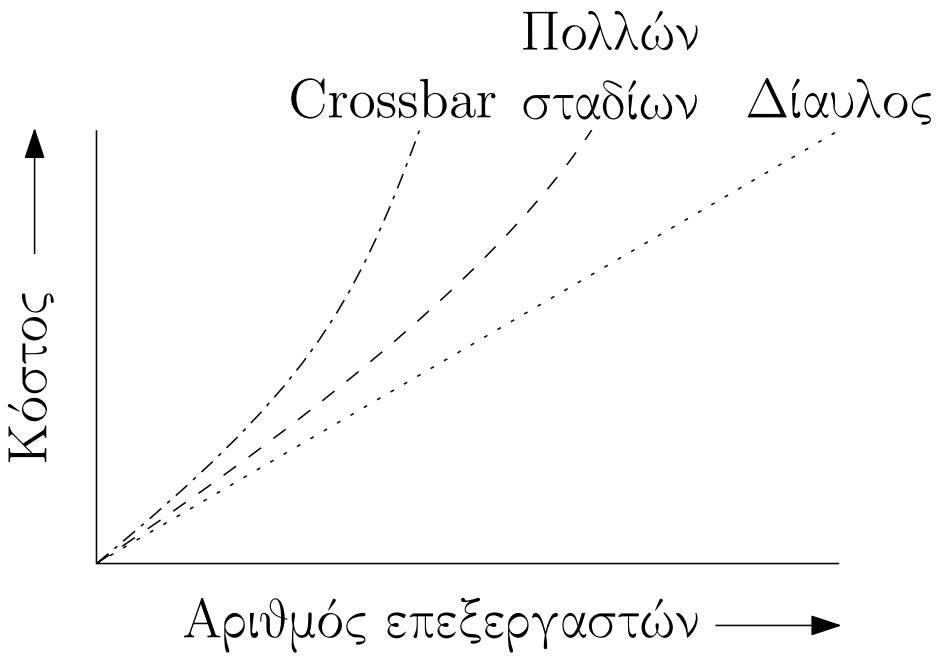


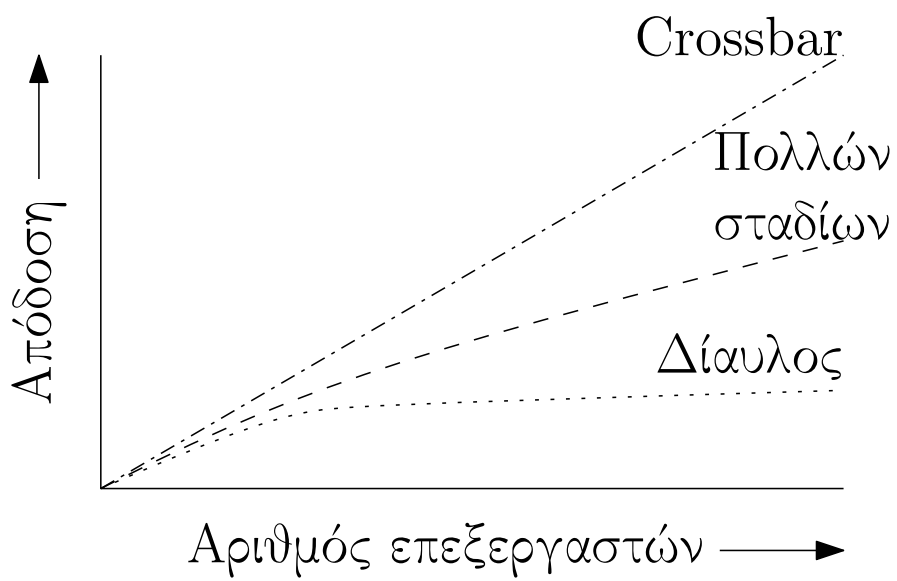

Σχήμα: Αρχιτεκτονική με bus: Symmetry, Multimax - κορεσμός με μικρό πλήθος επεξεργαστών

# Bus-based Δίκτυα Διασύνδεσης



# Bus-based Δίκτυα Διασύνδεσης



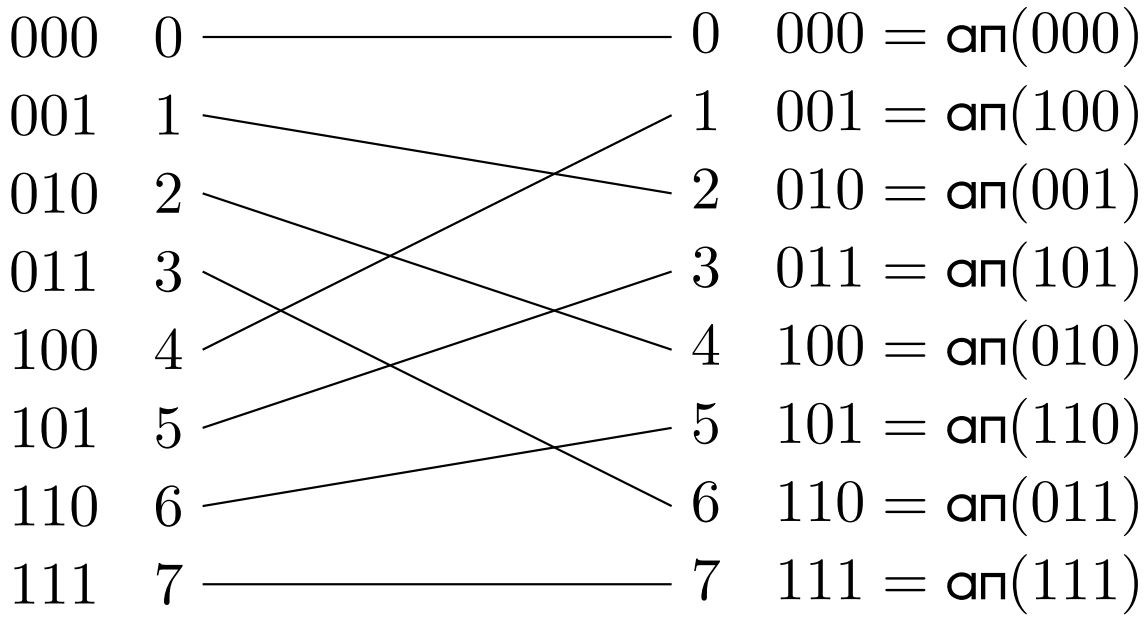



# Πολυφασικά Multistage Δίκτυα Διασύνδεσης

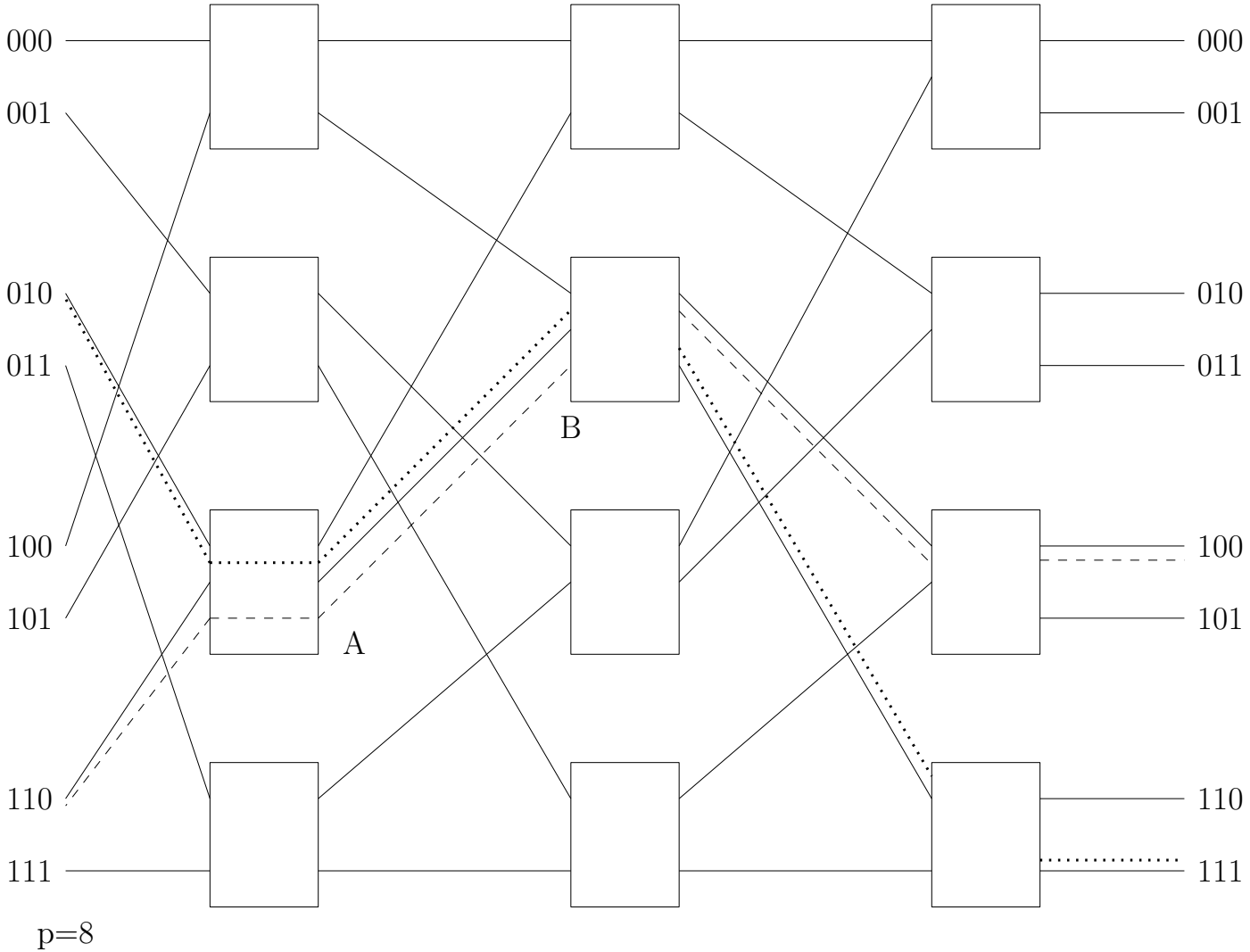
- Δίκτυο ωμέγα

$$j = \begin{cases} 2i, & 0 \leq i \leq p/2 - 1 \\ 2i + 1 - p, & p/2 \leq i \leq p - 1 \end{cases}$$

Αριστερή περιστροφή στη δυαδική παράσταση i για να ληφθεί το j



# Πολυφασικά Multistage Δίκτυα Διασύνδεσης




$p/2 \times \log p$ στοιχεία διακόπτες
AB: blocking network

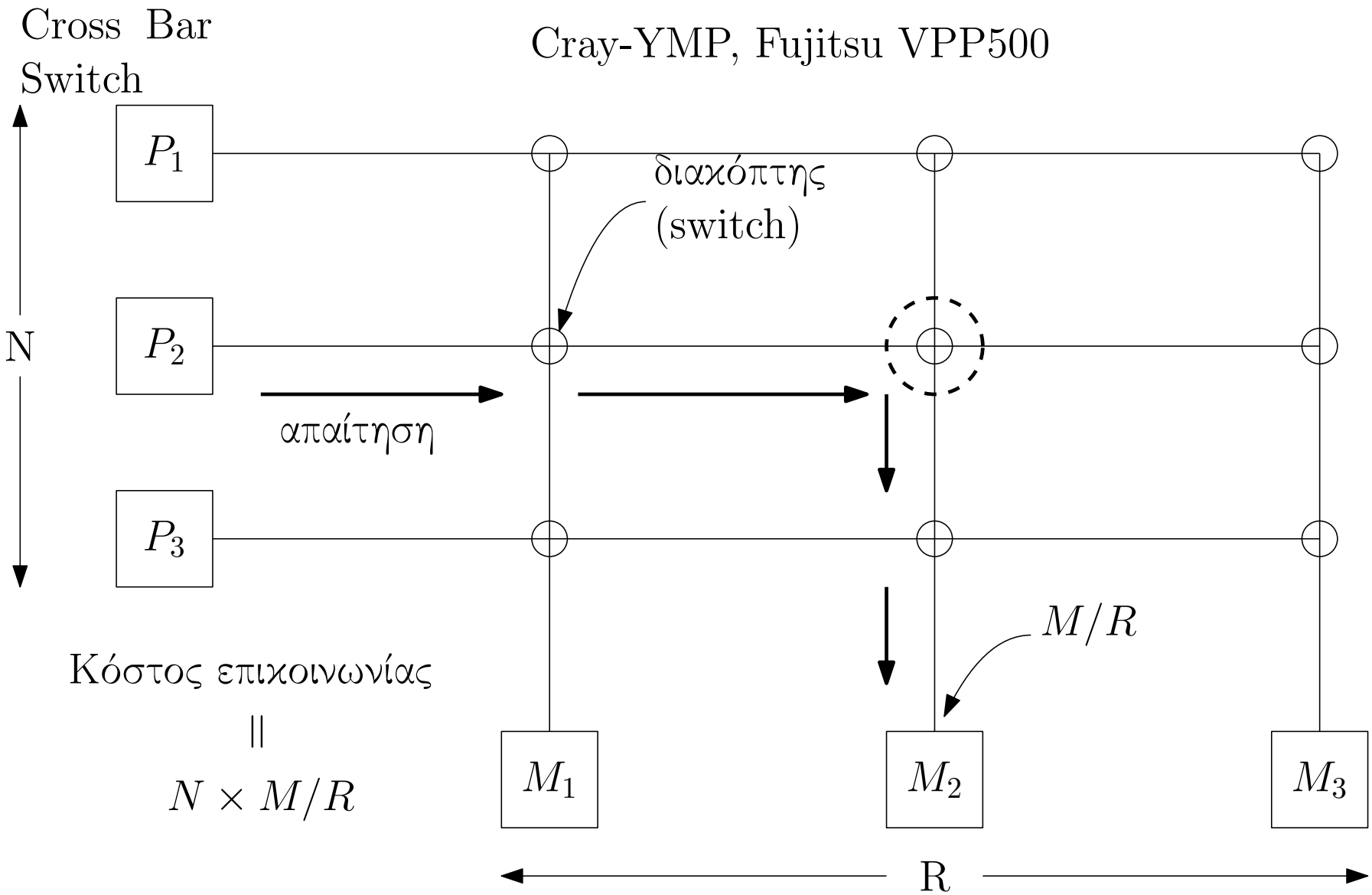
Σχήμα: Δίκτυο Ωμέγα

# Πολυφασικά Multistage Δίκτυα Διασύνδεσης

Παράλληλοι υπολογιστές: BBN Butterfly, IBM RP-3, NYU Ultracomputer

**Παρατήρηση:** Μόνο ένας επεξεργαστής μπορεί να διαβάσει ή να γράψει σε ένα τμήμα μνήμης.

# Πολυφασικά Multistage Δίκτυα Διασύνδεσης

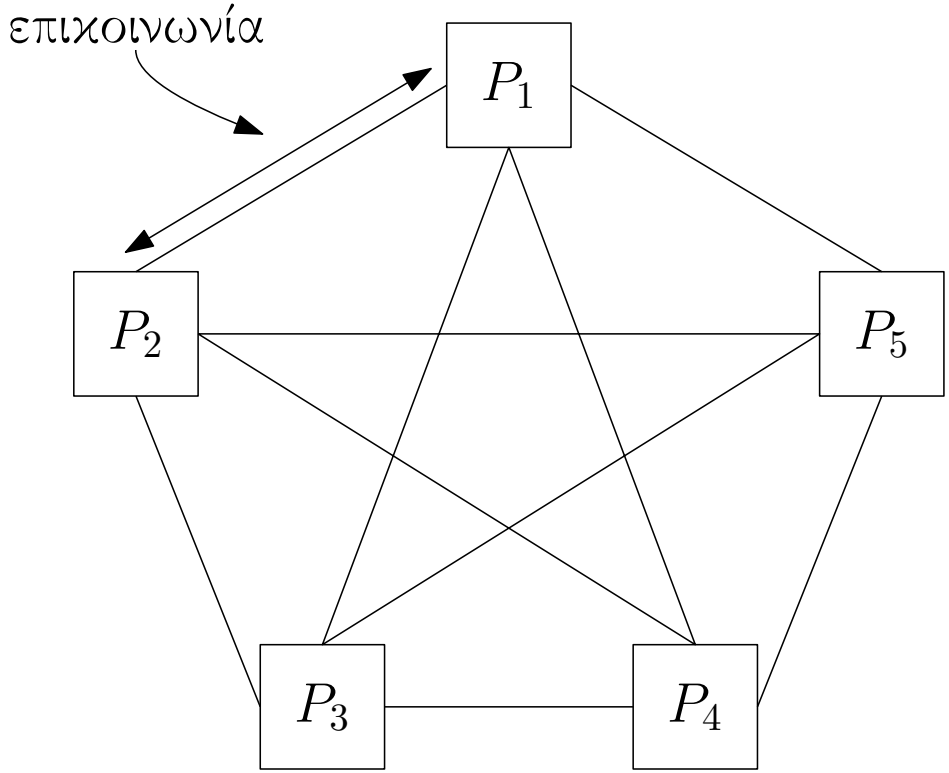


# Δίκτυο Διασύνδεσης

- Οι μνήμες $M_i$ είναι κατανεμημένες μεταξύ των $N$ επεξεργαστών ($M_i = M/N$ θέσεις μνήμης)
- Διπλής κατεύθυνσης γραμμή επικοινωνίας μεταξύ των επεξεργαστών (μηνύματα)

# Δίκτυο Διασύνδεσης

## Κλίκα (clique)



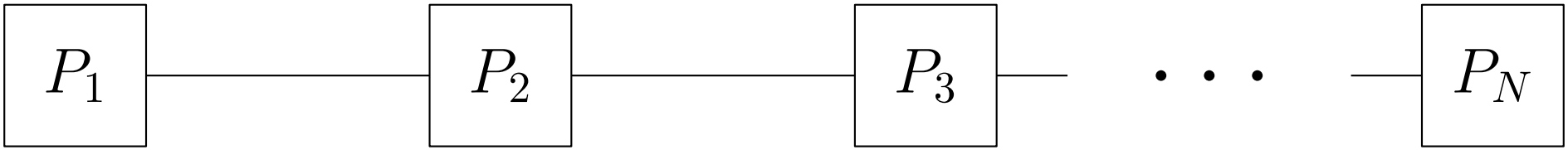

Σχήμα: Κλίκα (clique)

Σύνολο $\frac{N(N-1)}{2}$ γραμμές.

Εξωπραγματικό για μεγάλο $N$

# Δίκτυο Διασύνδεσης

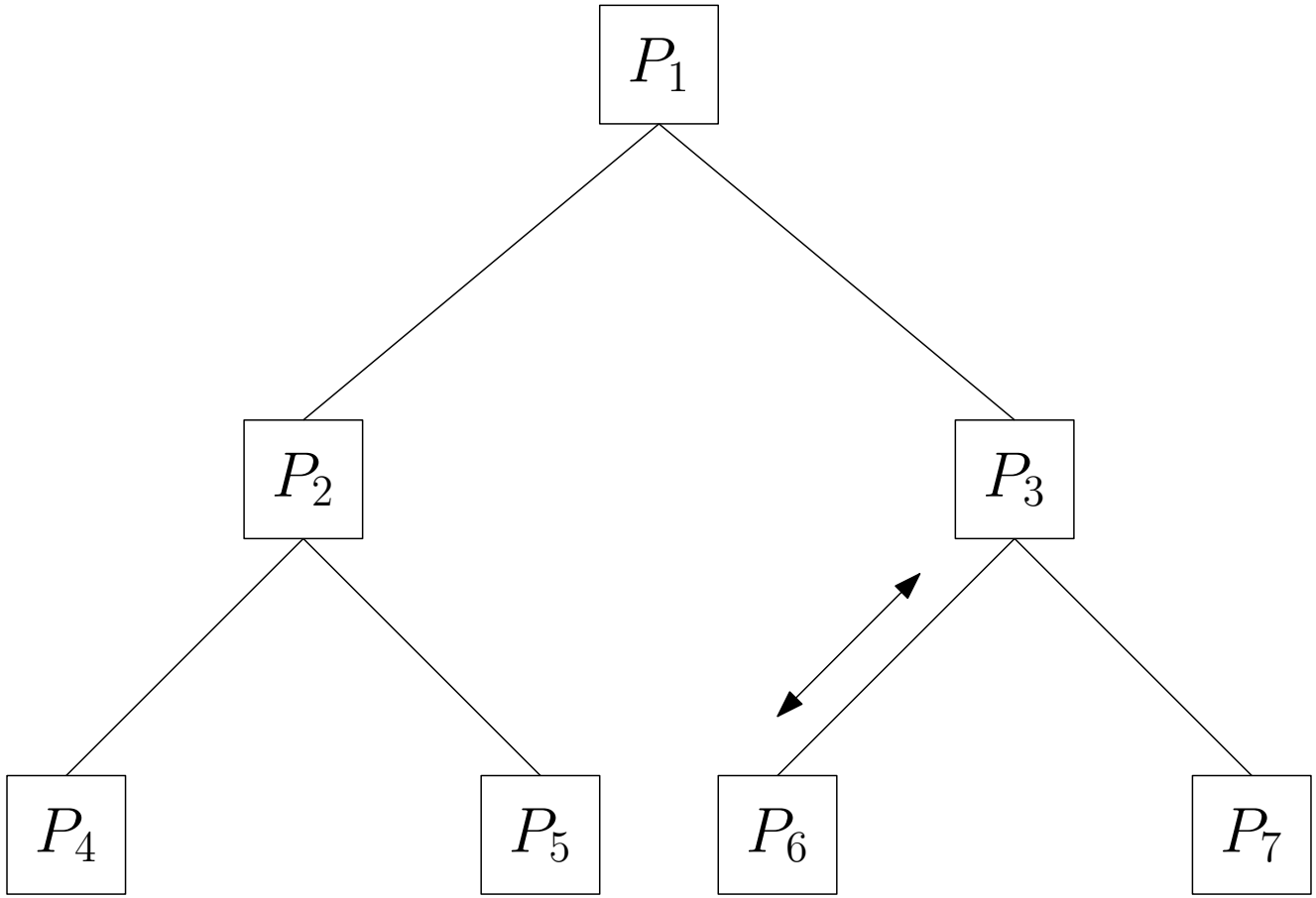
**Γραμμικός πίνακας**




Σχήμα: Γραμμικός πίνακας

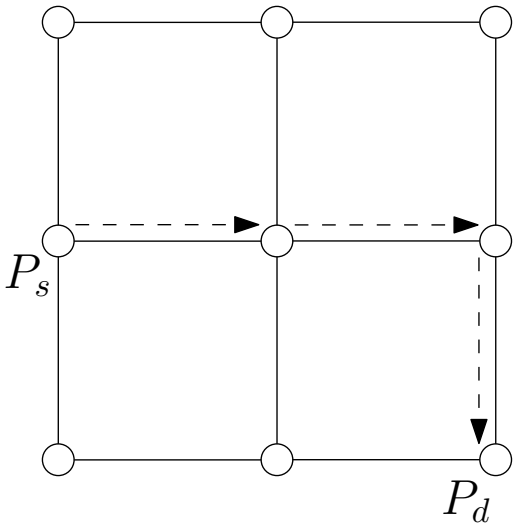
# Δίκτυο Διασύνδεσης

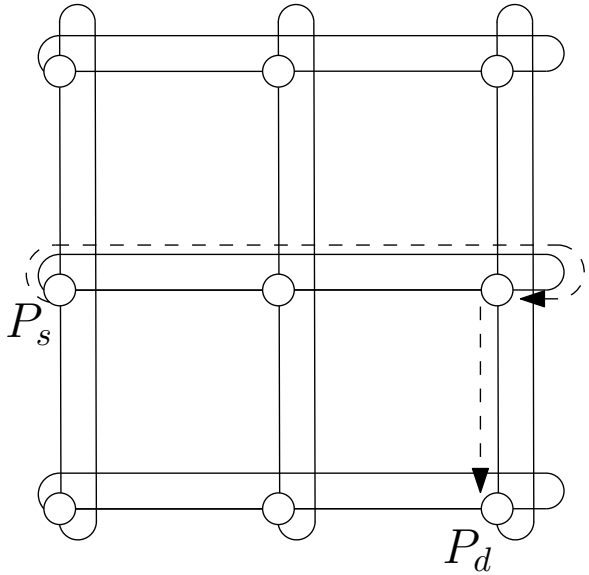
## Δέντρο



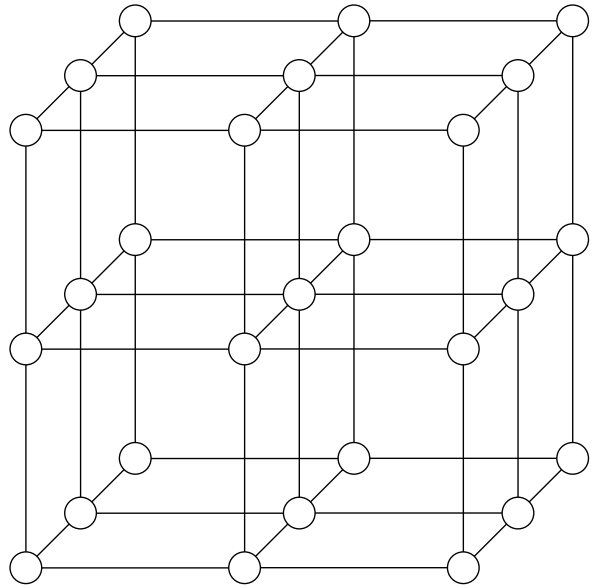

Σχήμα: Δέντρο

# Δίκτυο Διασύνδεσης



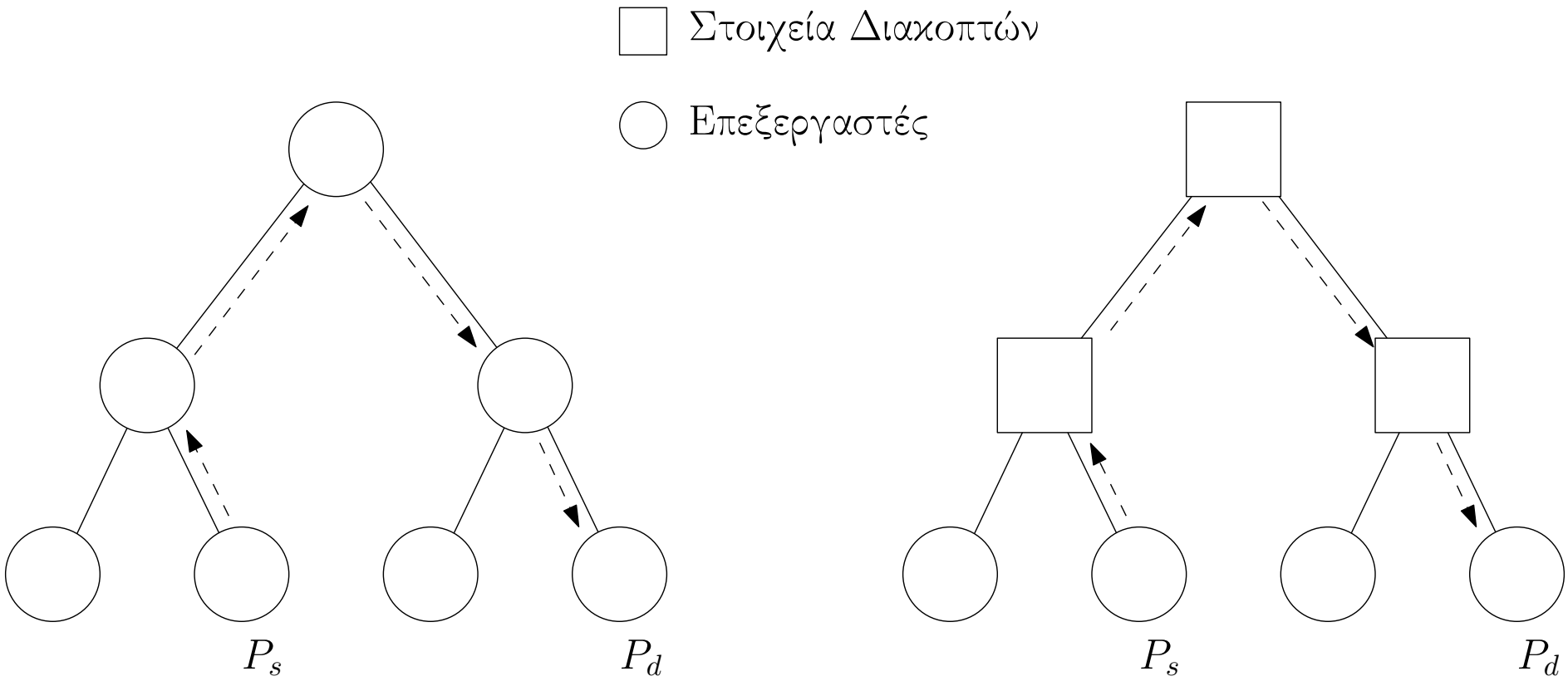

DAP, Paragon XP/S



Cray T3D

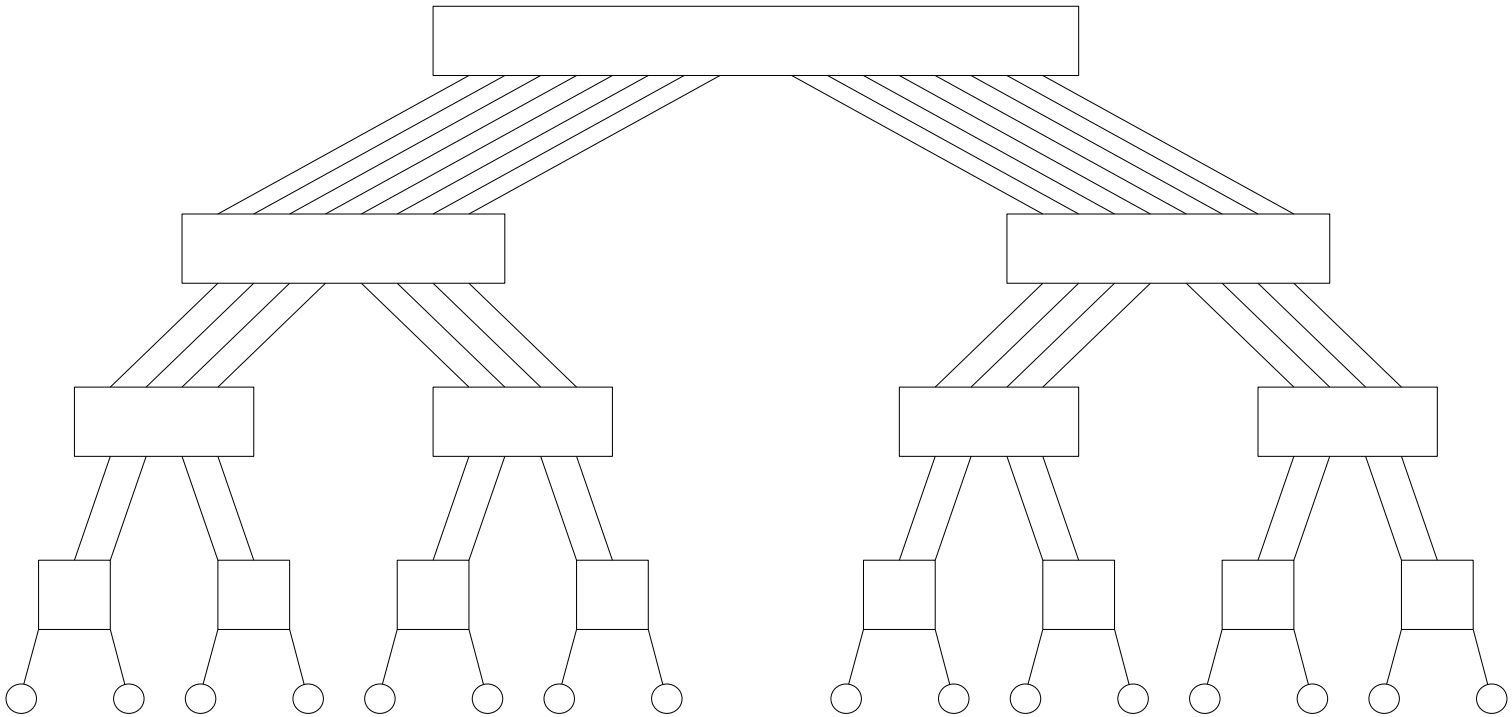
# Δίκτυο Διασύνδεσης

## Δίκτυο Δέντρου




- Δίκτυα Πλήρους Δυαδικού Δέντρου
- Διαδρομή Μηνύματος

# Δίκτυο Διασύνδεσης



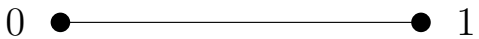
Σχήμα: fat tree network

# Δίκτυο Διασύνδεσης

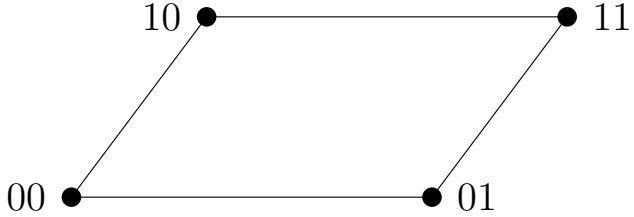
## Κύβος

$N = 2^q, \quad q \geq 1$

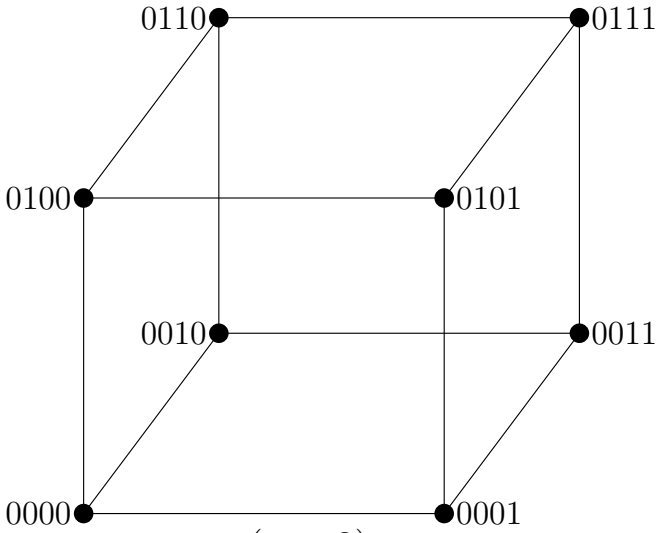

$(q = 0)$



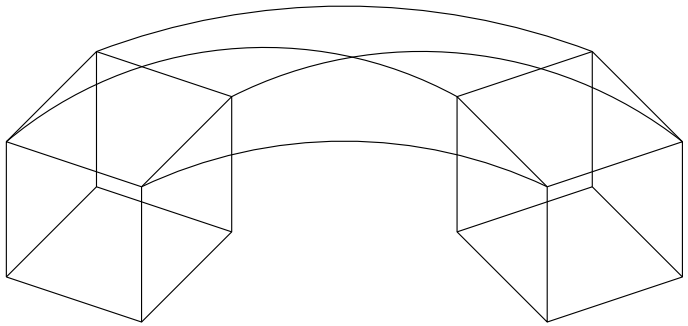

$(q = 1)$




$(q = 2)$

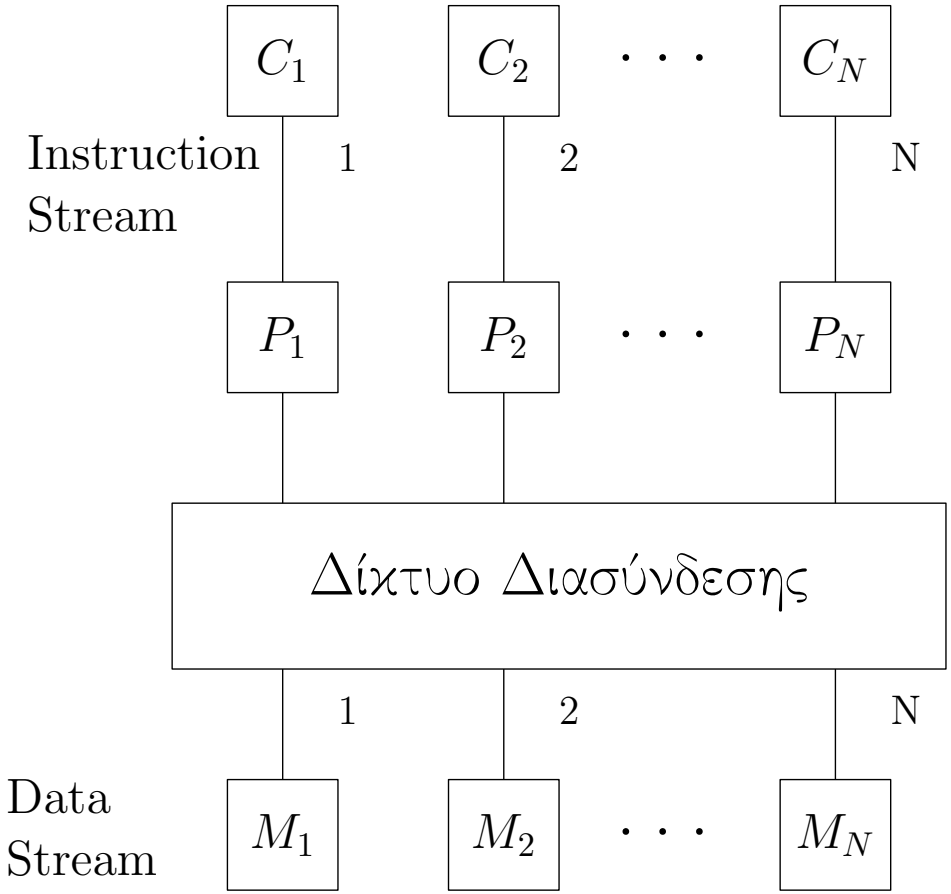



$(q = 3)$



$(q = 4)$

# MIMD Υπολογιστές

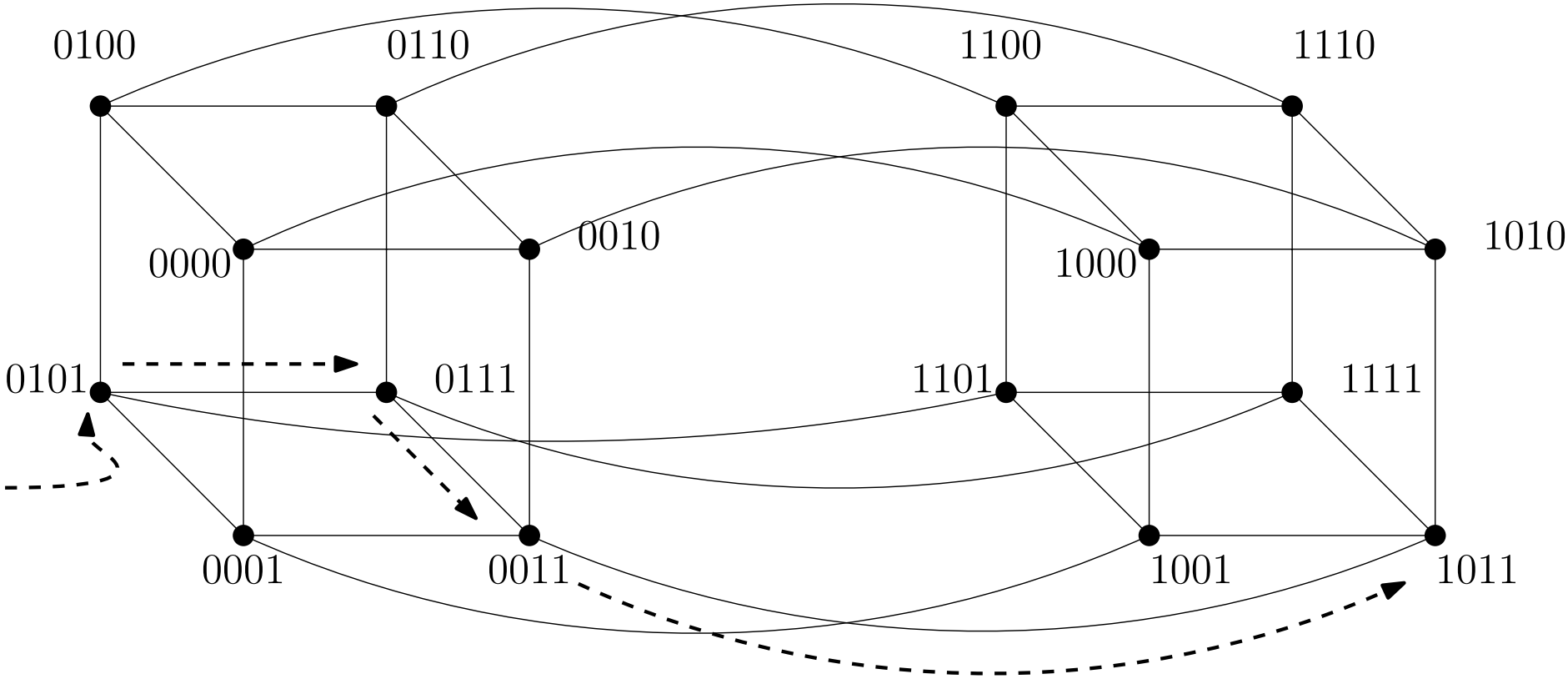



Σχήμα: Multiple Instruction Multiple Data

- Σύγχρονη λειτουργία
- Ασύγχρονη λειτουργία

# MIMD Υπολογιστές

## Δίκτυο Υπερκύβου




Σχήμα: Δίκτυο Υπερκύβου

d-διάστασης υπερκύβος

$p = 2^d$ επεξεργαστές

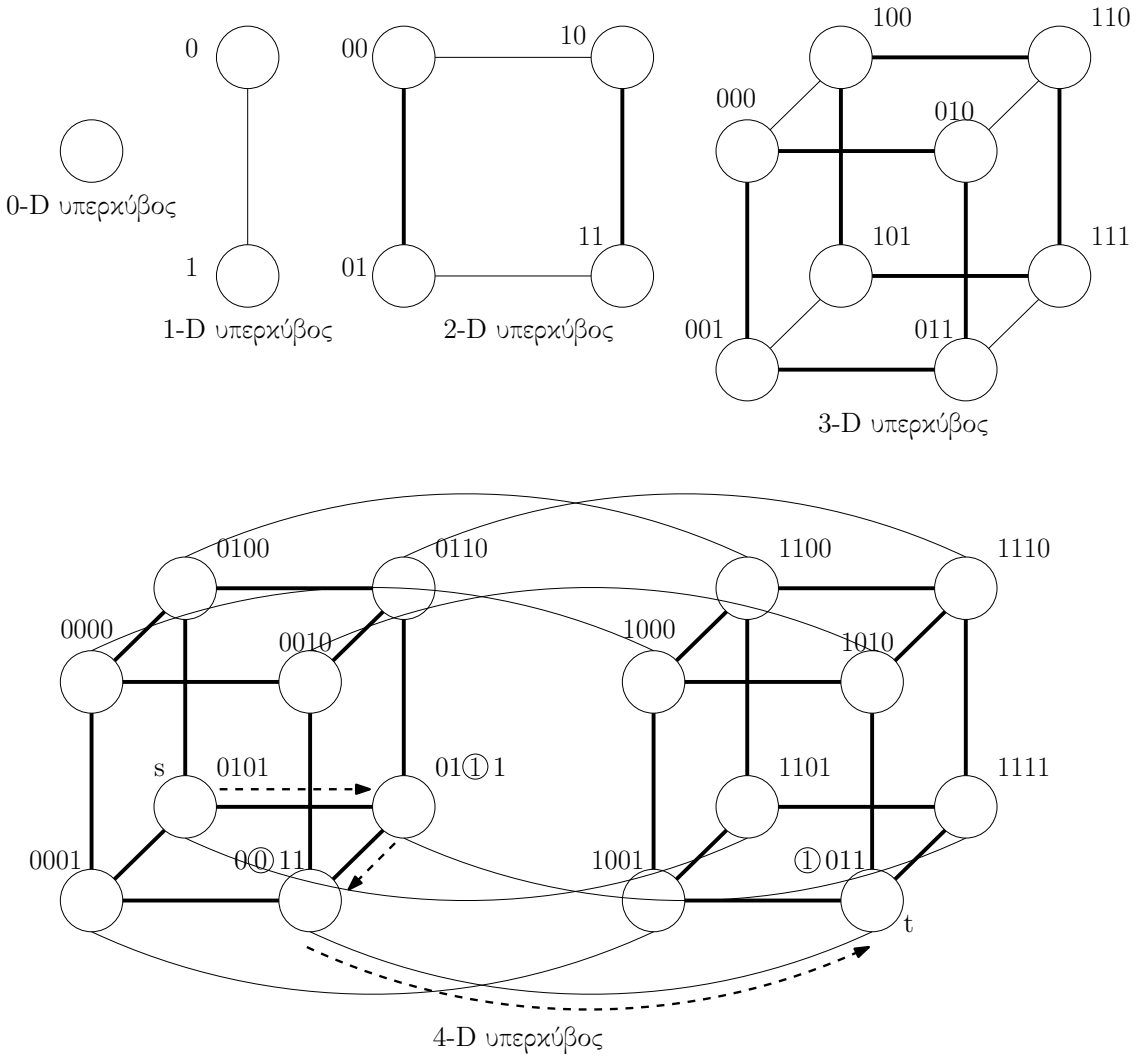
# MIMD Υπολογιστές

**Ιδιότητες**

- Δύο επεξεργαστές συνδέονται τότε και μόνο τότε αν διαφέρουν κατά ένα ακριβώς bit.
- Σε έναν d-διάστασης υπερκύβο ο κάθε επεξεργαστής συνδέεται άμεσα με d άλλους επεξεργαστές.
- Ο συνολικός αριθμός των bit θέσεων στα οποία διαφέρουν δύο επεξεργαστές λέγεται απόσταση του Hamming.

# MIMD Υπολογιστές

## Αρίθμηση των επεξεργαστών




Σχήμα: Απόσταση Hamming 0101 και 1011 = 3 ($s \oplus t = 1110$)

# MIMD Υπολογιστές

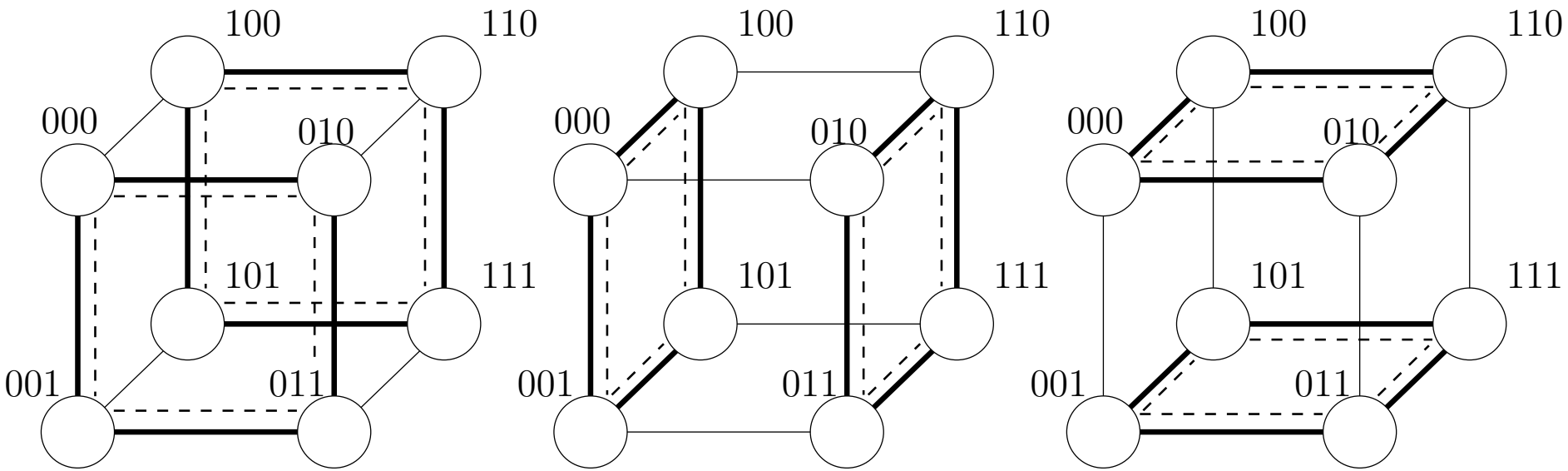
Όταν ένας υπερκύβος d+1-διάστασης κατασκευάζεται με τη σύνδεση δύο υπερκύβων d-διάστασης, στις ετικέτες του ενός υπερκύβου τίθεται το πρόθεμα 0 και στον άλλο υπερκύβο το πρόθεμα 1.
Ο μόνος επεξεργαστής (γειτονικός) του s που έχει bit 1 στη θέση ένα είναι αυτός που δείχνεται στο Σχήμα, κ.ο.κ.

# MIMD Υπολογιστές

- Η απόσταση του Hamming μεταξύ δύο επεξεργαστών s και t είναι ο συνολικός αριθμός των bits τα οποία είναι 1 στην $s \oplus t$ (όπου $\oplus$ συμβολίζει xor).
- Ο αριθμός των συνδέσμων επικοινωνίας στο συντομότερο μονοπάτι μεταξύ δύο επεξεργαστών είναι ίσος με την απόσταση του Hamming. Ένα μήνυμα δρομολογείται από τον επεξεργαστή s στον επεξεργαστή t διαμέσου διαστάσεων οι οποίες αντιστοιχούν σε θέσεις bit που έχουν 1 στη δυαδική παράσταση του $s \oplus t$. Επειδή η δυαδική παράσταση του $s \oplus t$ μπορεί να περιέχει το πολύ d μονάδες, το συντομότερο μονοπάτι μεταξύ δύο επεξεργαστών σε ένα υπερκύβο δε μπορεί να έχει περισσότερους από d συνδέσμους.

Παράλληλοι Υπολογιστές βασισμένοι σε δίκτυο υπερκύβου είναι οι: n CUBE, Cosmic Cube και iPSC.

# MIMD Υπολογιστές

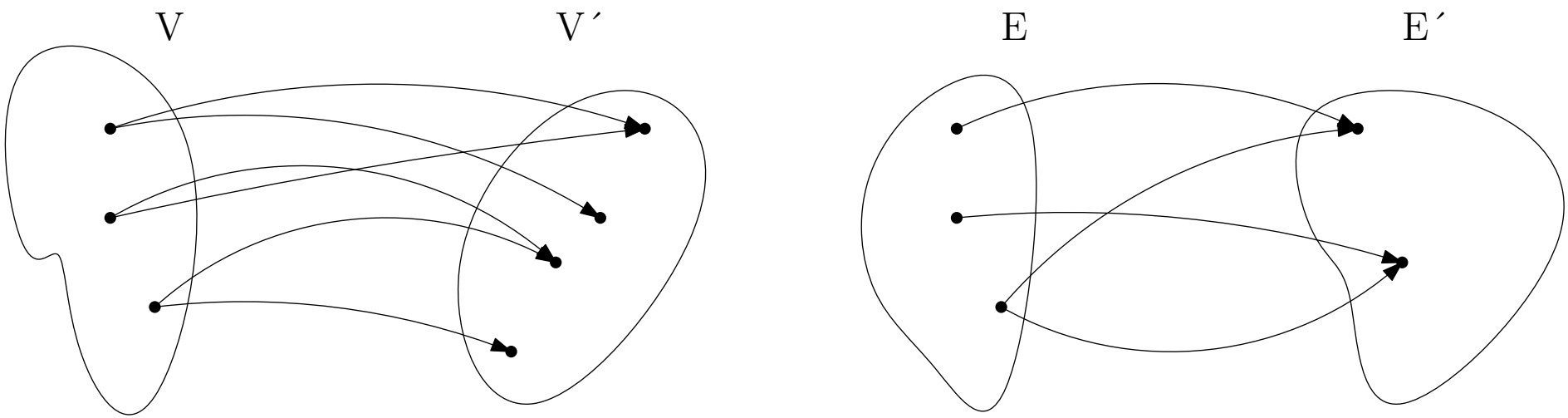


# MIMD Υπολογιστές

| **Δίκτυο** | **Διάμετρος** | **Εύρος Διχοτόμησης** | **Συνδεσιμότητα Τόξου** | **Κόστος (αριθ. συνδέσεων)** |
|---|---|---|---|---|
| Πλήρως συνδεδεμένο | 1 | $p^2/4$ | $p-1$ | $p(p-1)/2$ |
| Αστέρας | 2 | 1 | 1 | $p-1$ |
| Πλήρες Δυαδικό Δέντρο | $2\log((p+1)/2)$ | 1 | 1 | $p-1$ |
| Γραμμικός πίνακας | $p-1$ | 1 | 1 | $p-1$ |
| Δακτύλιος | $\lfloor p/2 \rfloor$ | 2 | 2 | $p$ |
| 2-D πλέγμα χωρίς κύρτωση | $2(\sqrt{p}-1)$ | $\sqrt{p}$ | 2 | $2(p-\sqrt{p})$ |
| 2-D πλέγμα με κύρτωση | $2\lfloor \sqrt{p}/2 \rfloor$ | $2\sqrt{p}$ | 4 | $2p$ |
| Υπερκύβος | $\log p$ | $p/2$ | $\log p$ | $(p\log p)/2$ |
| d-κύβος k-τάξης με κύρτωση | $d\lfloor k/2 \rfloor$ | $2k^{d-1}$ | $2d$ | $dp$ |

# Εμφύτευση Δικτύων στον Υπερκύβο

$$G(V, E) \xrightarrow{\text{εμφύτευση}} G'(V', E')$$



# Εμφύτευση Δικτύων στον Υπερκύβο

Η εμφύτευση ενός γραφήματος εντός ενός άλλου είναι σημαντική διότι ένας αλγόριθμος που έχει σχεδιαστεί για ένα συγκεκριμένο δίκτυο διασύνδεσης μπορεί να είναι αναγκαίο να προσαρμοστεί σε ένα άλλο δίκτυο.

- συσσώρευση (congestion): Το μέγιστο πλήθος ακμών που απεικονίζονται σε μια ακμή στο $E'$.
- διαστολή (dilation): Το πλήθος συνδέσμων στο $E'$ στους οποίους απεικονίζεται μία ακμή του $E$.
- επέκταση (expansion): Επεξεργαστές στο $V'$ / Επεξεργαστές στο $V$.

Στη συνέχεια θα μελετηθούν οι εμφυτεύσεις διαφόρων δικτύων διασύνδεσης εντός του υπερκύβου. Η μελέτη θα περιοριστεί στην περίπτωση όπου το πλήθος των επεξεργαστών είναι το ίδιο στα δύο δίκτυα (επέκταση = 1). Επιπλέον το πολύ μία ακμή του $E$ απεικονίζεται σε μία ακμή του $E'$ (συσσώρευση = 1).

# Εμφύτευση Δικτύων στον Υπερκύβο

**Εμφύτευση γραμμικού πίνακα σε υπερκύβο**

Ένας γραμμικός πίνακας (ή ένας δακτύλιος) που αποτελείται από $2^d$ επεξεργαστές μπορεί να εμφυτευθεί σε ένα υπερκύβο $d$ διάστασης απεικονίζοντας τον $i-$ οστό επεξεργαστή του γραμμικού πίνακα στον επεξεργαστή $G(i,d)$ του υπερκύβου, όπου

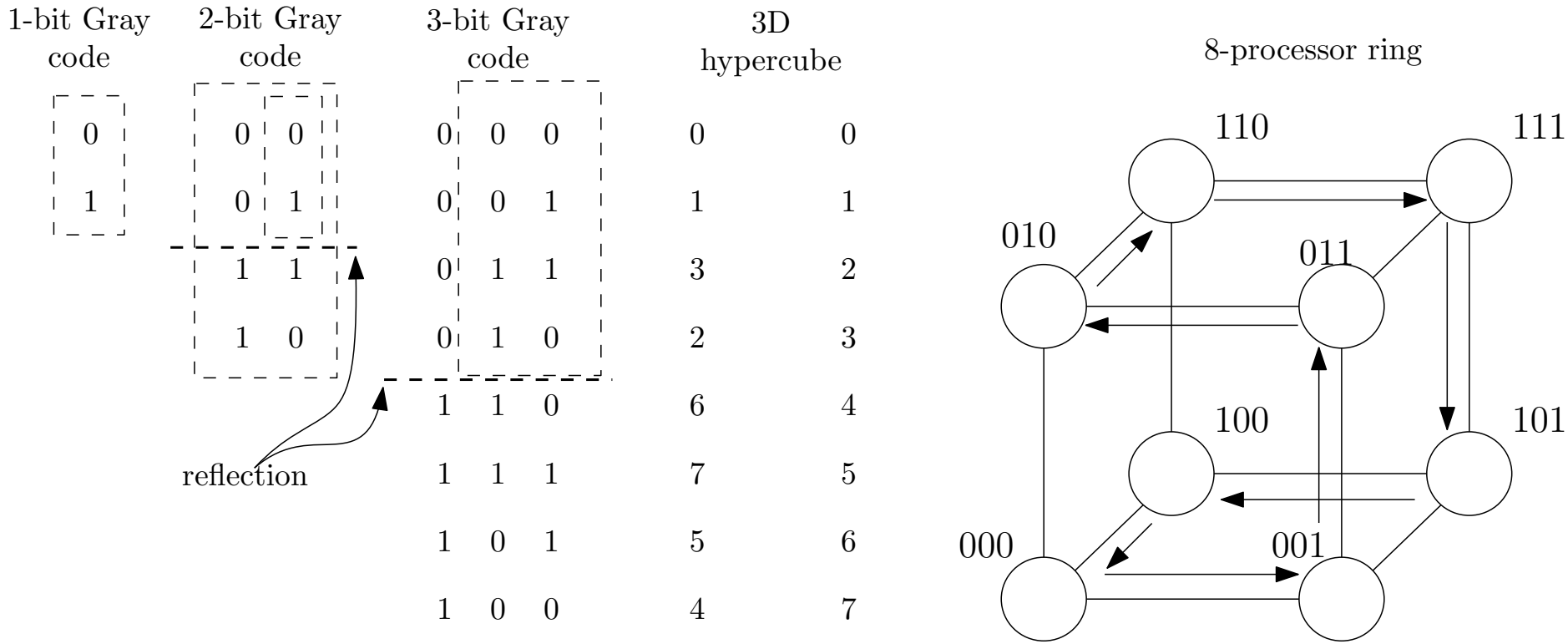
$$G(0,1) = 0$$

$$G(1,1) = 1$$

$$G(i,x+1) = \begin{cases} G(i,x), & i < 2^x \\ 2^x + G(2^{x+1} - 1 - i, x), & i \geq 2^x \end{cases}$$

# Εμφύτευση Δικτύων στον Υπερκύβο

Η συνάρτηση $G$ λέγεται binary reflected Gray code (RGC). Η $G(i, d)$ συμβολίζει το $i-$ οστό στοιχείο στην ακολουθία των κωδίκων Gray με d bits. Οι κώδικες Gray d+1 bits παράγονται από ένα πίνακα των κωδίκων Gray με d bits ως εξής: Παίρνοντας το συμμετρικό πίνακα και τοποθετώντας στα στοιχεία του συμμετρικού πίνακα το πρόθεμα 1 και στα αρχικά στοιχεία το πρόθεμα 0.




Σχήμα: Εμφύτευση ενός δακτυλίου οκτώ επεξεργαστών σε ένα υπερκύβο διάστασης 3

# Εμφύτευση Δικτύων στον Υπερκύβο

- Τα στοιχεία $G(i, d)$ και $G(i + 1, d)$ διαφέρουν μόνο σε ένα bit.
- Ο επεξεργαστής $i$ στο γραμμικό πίνακα απεικονίζεται στον επεξεργαστή $G(i, d)$ του υπερκύβου και ο επεξεργαστής $i + 1$ του γραμμικού πίνακα στον επεξεργαστή $G(i + 1, d)$ του υπερκύβου.
- Συνεπώς υπάρχει ένας άμεσος σύνδεσμος στον υπερκύβο, ο οποίος αντιστοιχεί σε κάθε άμεσο σύνδεσμο στο γραμμικό πίνακα (οι ετικέτες δύο γειτονικών επεξεργαστών διαφέρουν μόνο κατά ένα bit).
- Η απεικόνιση που ορίζεται από την $G$ έχει συσσώρευση ένα και διαστολή ένα.

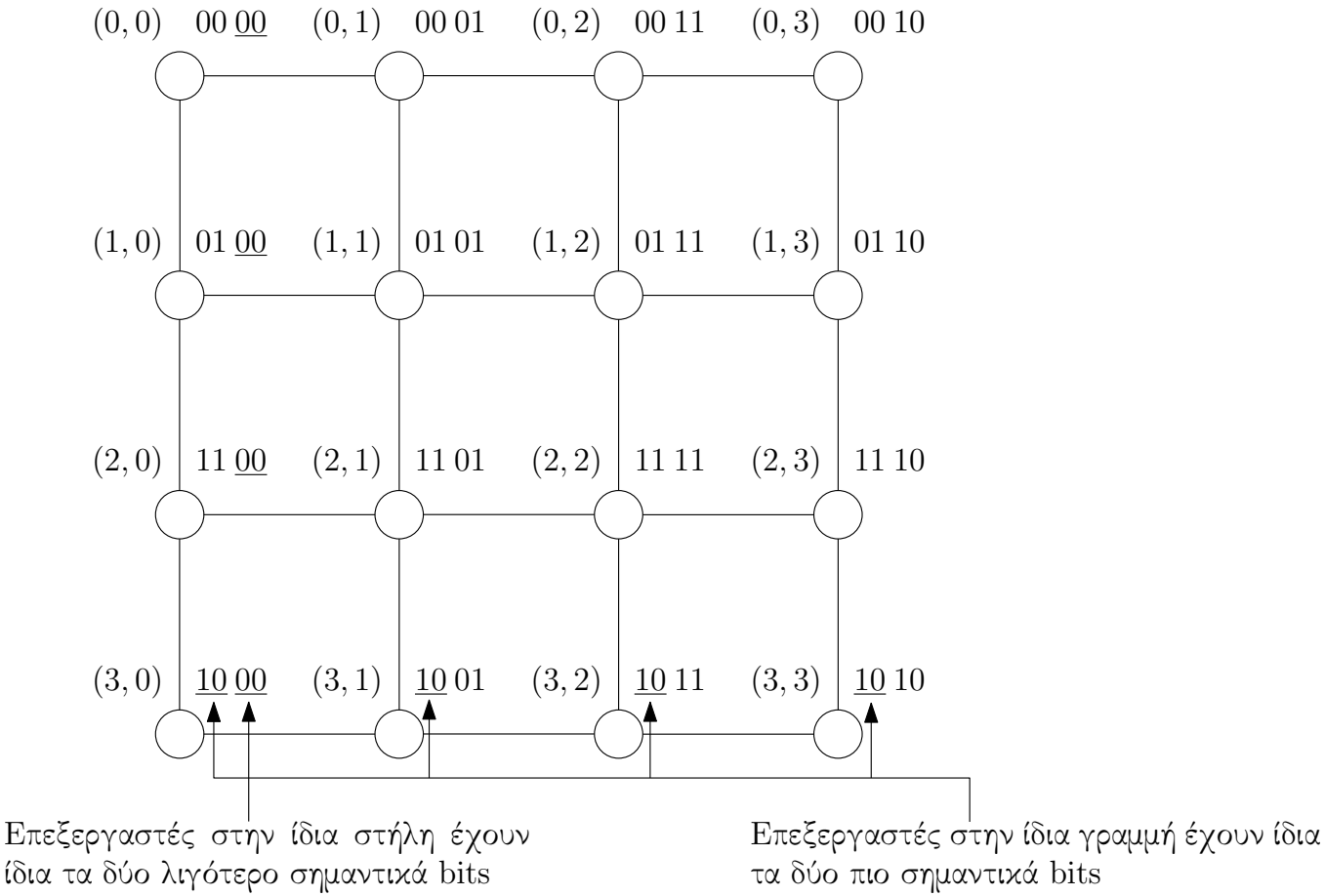
# Εμφύτευση Δικτύων στον Υπερκύβο

**Εμφύτευση δυδιάστατου δικτύου (mesh) σε υπερκύβο**

- Μπορούμε να εμφυτεύσουμε ένα $2^r \times 2^s$ wraparound mesh σε έναν $2^{r+s}$-επεξεργαστών υπερκύβο απεικονίζοντας τον επεξεργαστή $(i, j)$ του mesh στον επεξεργαστή $G(i, r)||G(j, s)$ του υπερκύβου, όπου $||$ συμβολίζει τη συνένωση των δύο κωδίκων Gray.
- Οι άμεσοι γείτονες στο mesh απεικονίζονται σε επεξεργαστές του υπερκύβου των οποίων οι ετικέτες διαφέρουν μόνο σε ένα bit). Συνεπώς η απεικόνιση αυτή έχει μία διάδοση = 1 και μία συσσώρευση = 1.

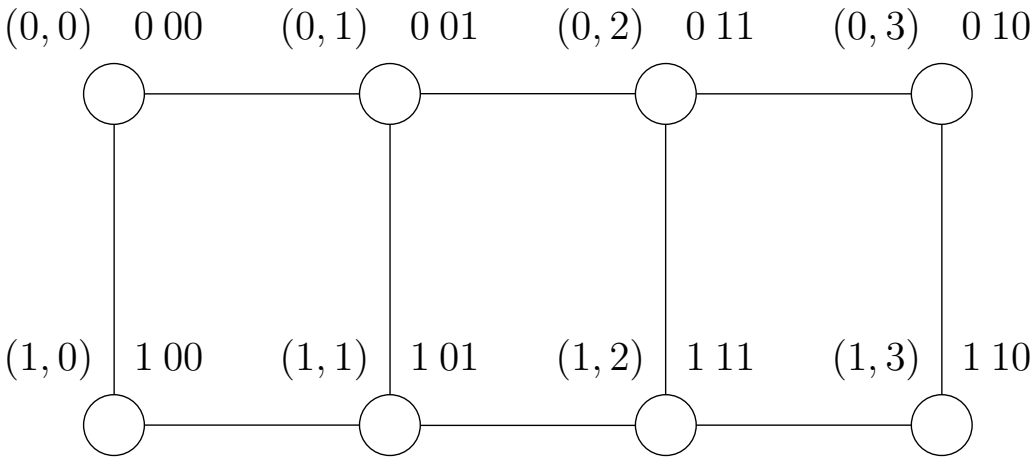
# Εμφύτευση Δικτύων στον Υπερκύβο

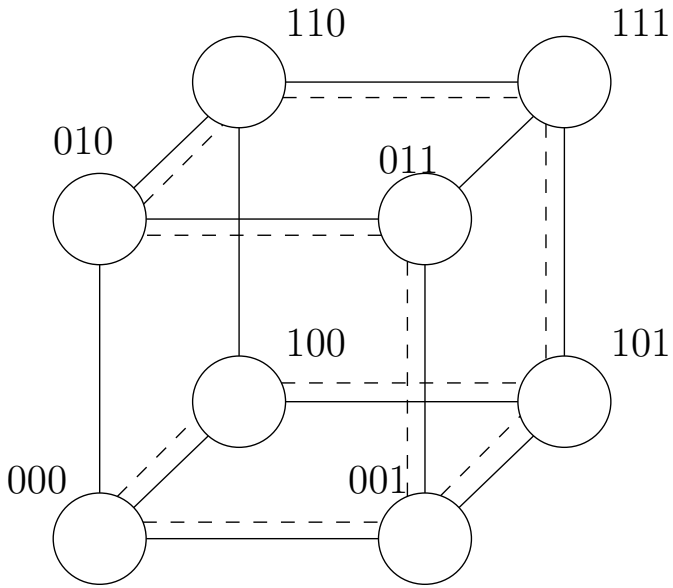
$$r = 1, s = 2, \qquad (i, j) \longrightarrow G(i, 1)||G(j, 2)$$
$$(0, 0) \longrightarrow G(0, 1)||G(0, 2) = 0||00 = 000$$
$$(0, 1) \longrightarrow G(0, 1)||G(1, 2) = 0||01 = 001$$




Σχήμα: Εμφύτευση $2 \times 4$ mesh

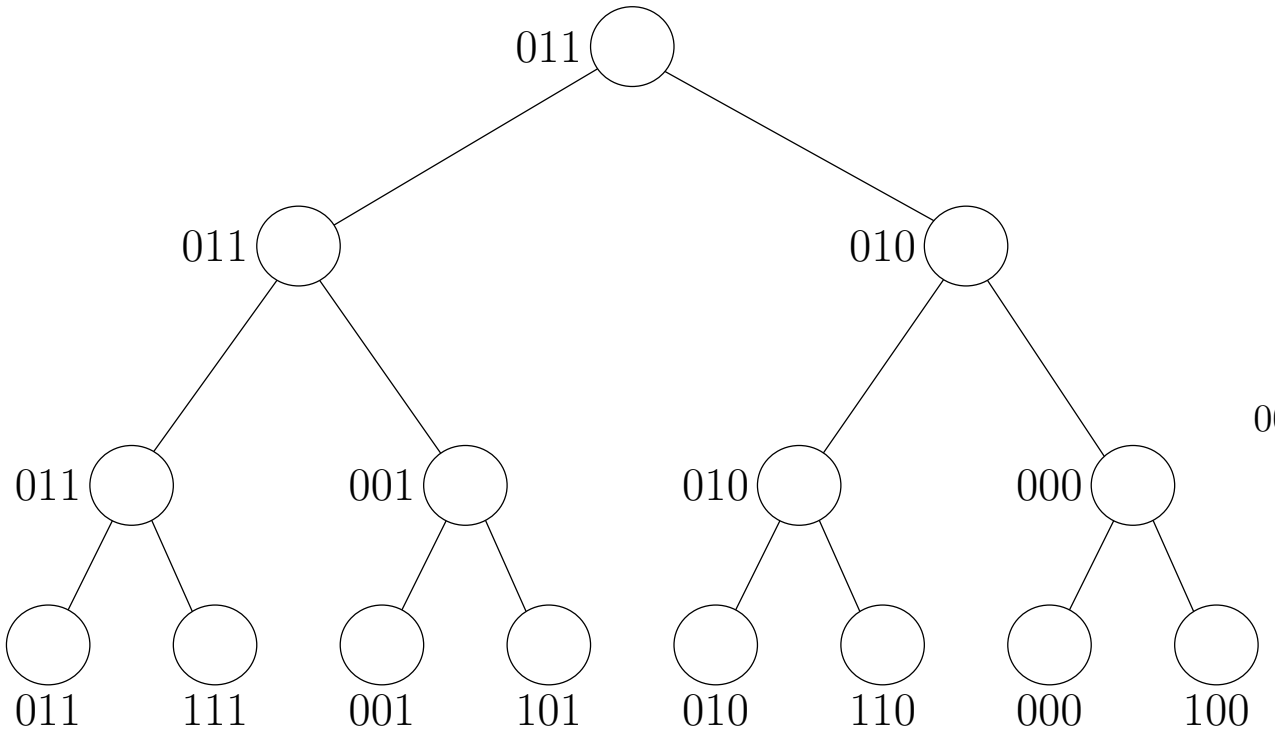
# Εμφύτευση Δικτύων στον Υπερκύβο

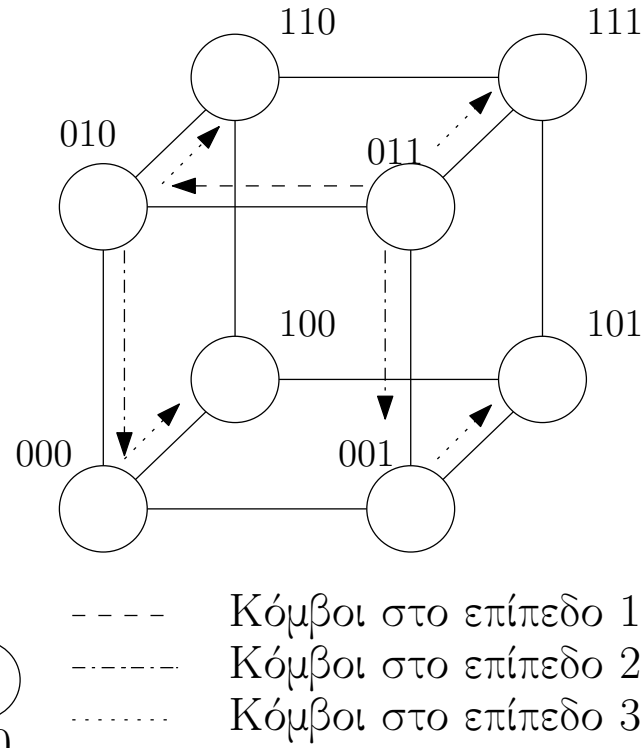



Σχήμα: Εμφύτευση $2 \times 4$ mesh

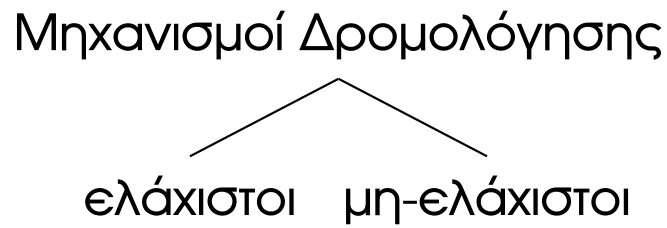
# Εμφύτευση Δικτύων στον Υπερκύβο



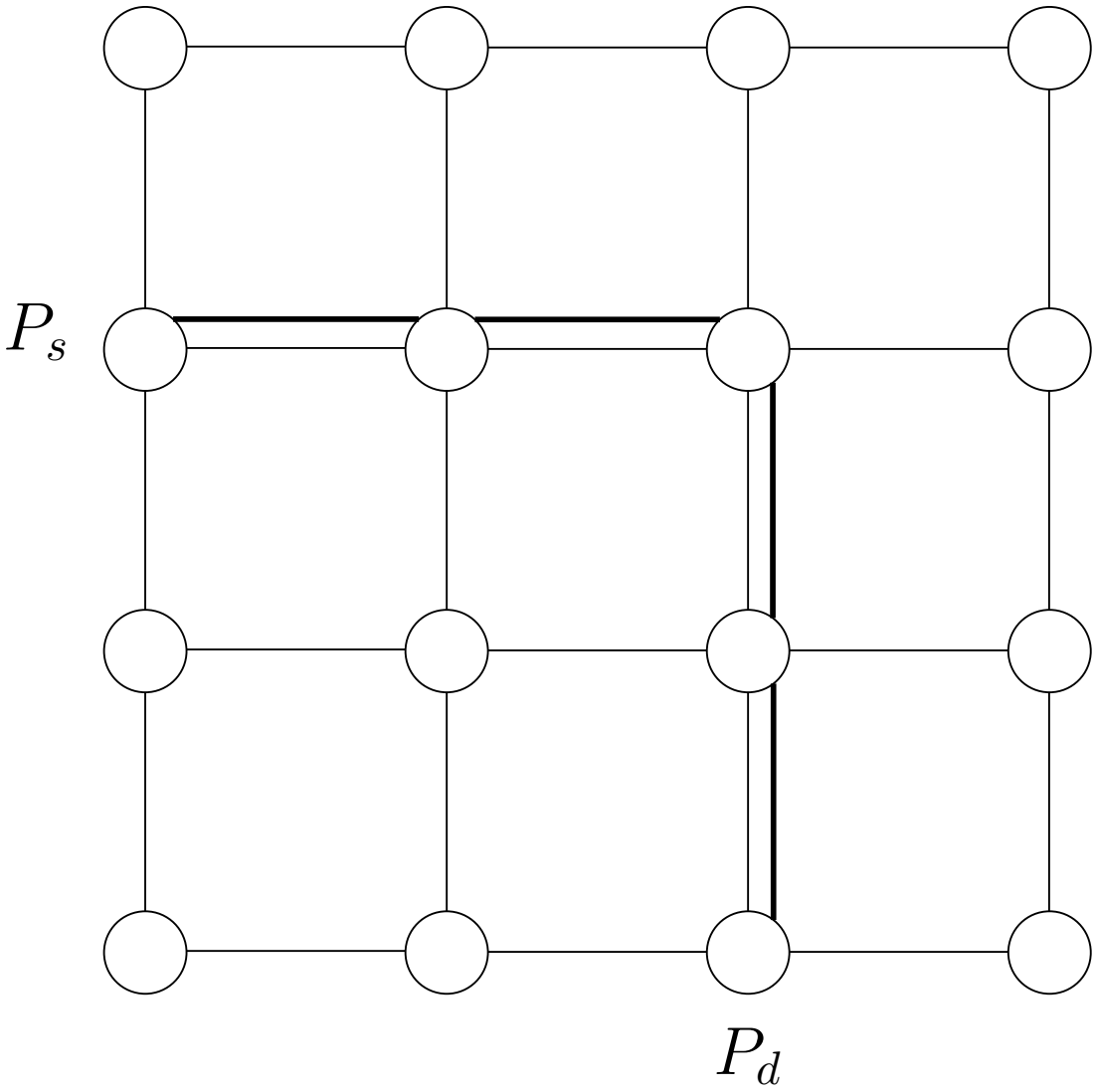
# Εμφύτευση Δικτύων στον Υπερκύβο

## Δρομολόγηση Μηνυμάτων





# Εμφύτευση Δικτύων στον Υπερκύβο

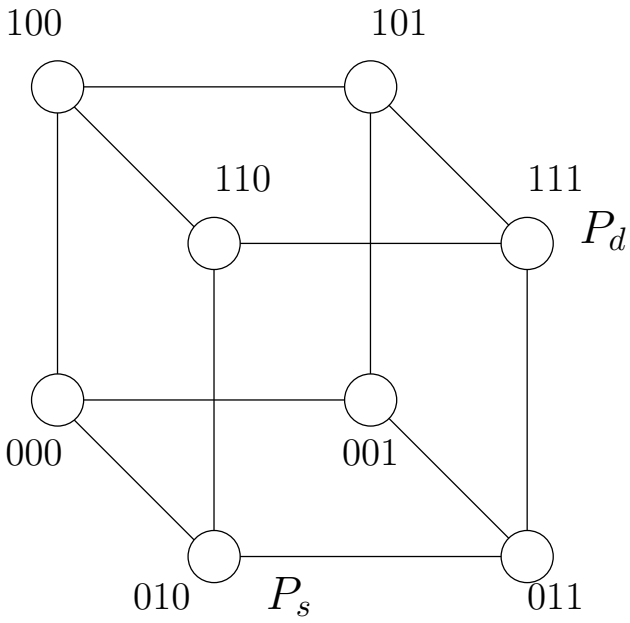



$\sqrt{p} \times \sqrt{p}$

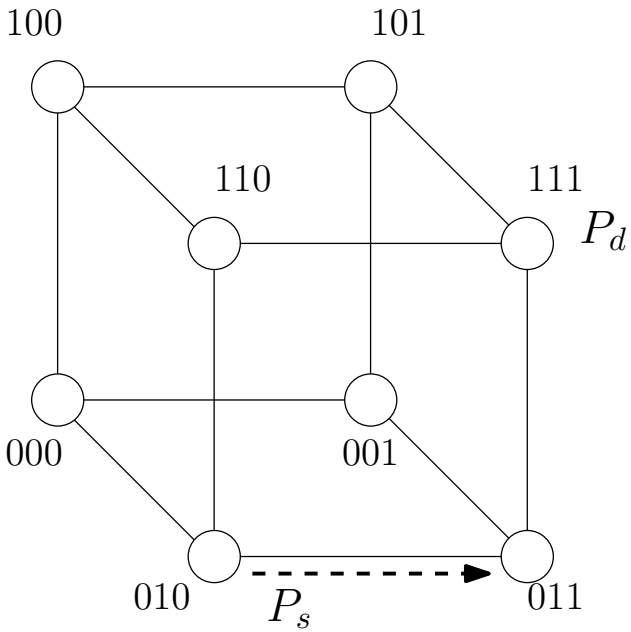
$2(\sqrt{p} - 1)$

Σχήμα: XY - δρομολόγηση σε mesh

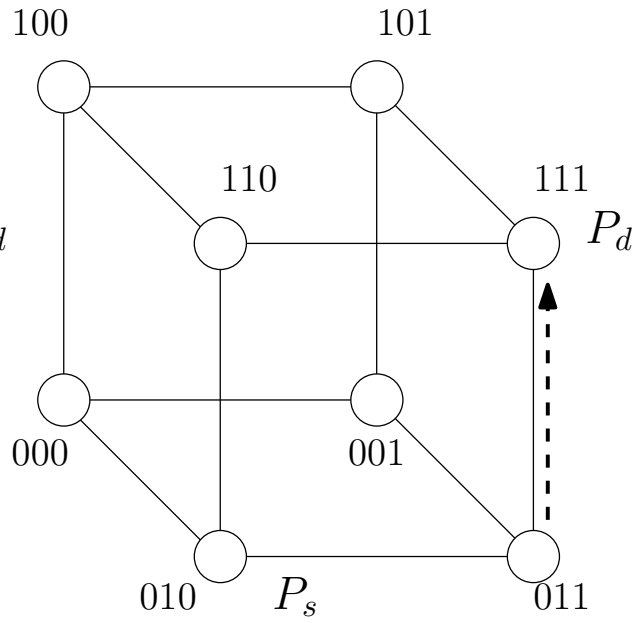
# Εμφύτευση Δικτύων στον Υπερκύβο




$\mathbf{010 \oplus 110 = 1\textcircled{0}\textcircled{1}}$
$\mathbf{010 \oplus 111 = 101}$
$010 \oplus 011 = 001$
$010 \oplus 000 = 010$
$010 \oplus 110 = 100$




Βήμα 1 $(010 \rightarrow 01\textcircled{1})$
$\mathbf{010 \oplus 111 = 0\textcircled{1}0}$
$011 \oplus 111 = 100$




Βήμα 2 $(011 \rightarrow 111)$
$\mathbf{011 \oplus 111 =}$

Σχήμα: E-cube δρομολόγηση

# Εμφύτευση Δικτύων στον Υπερκύβο

**Κόστος Επικοινωνίας**

- *καθυστέρηση επικοινωνίας (latency)*

Ο απαιτούμενος χρόνος για την επικοινωνία ενός μηνύματος μεταξύ δύο επεξεργαστών.

- *χρόνος ξεκινήματος (startup time)* ($t_s$)

Ο απαιτούμενος χρόνος για τη μεταχείριση του μηνύματος (προετοιμασία).

- *per-hop χρόνος* ($t_h$)

Ο απαιτούμενος χρόνος για να φθάσει η επικεφαλίδα του μηνύματος στον προορισμό της.

- *χρόνος ανά word* ($t_w$)

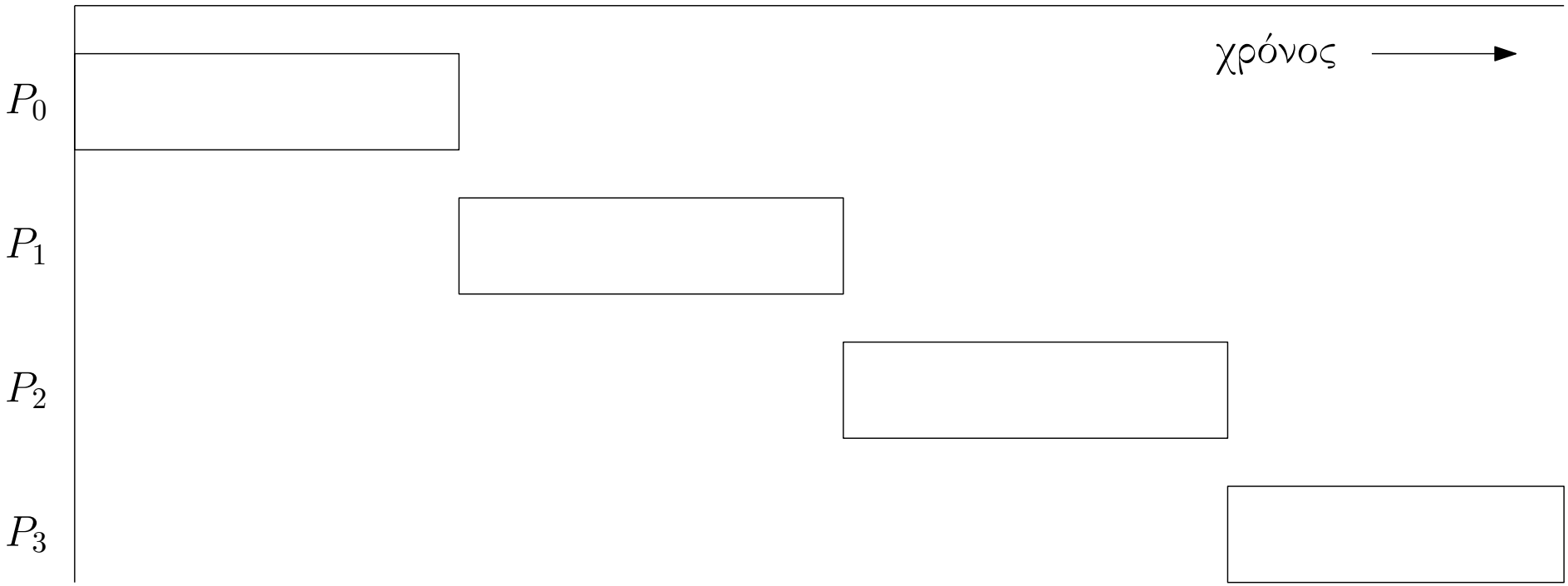
$$t_w = 1/r$$

όπου $r$ = αριθμός words ανά δευτερόλεπτο

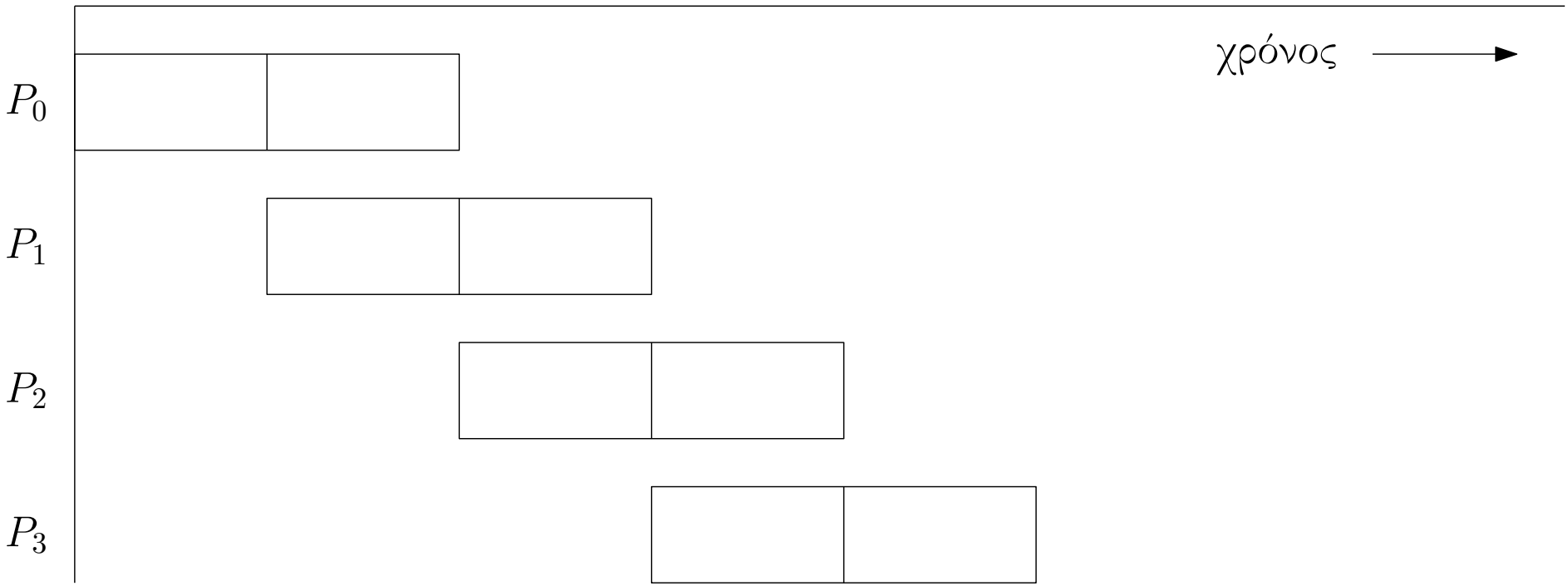
# Εμφύτευση Δικτύων στον Υπερκύβο

Καθυστέρηση Επικοινωνίας

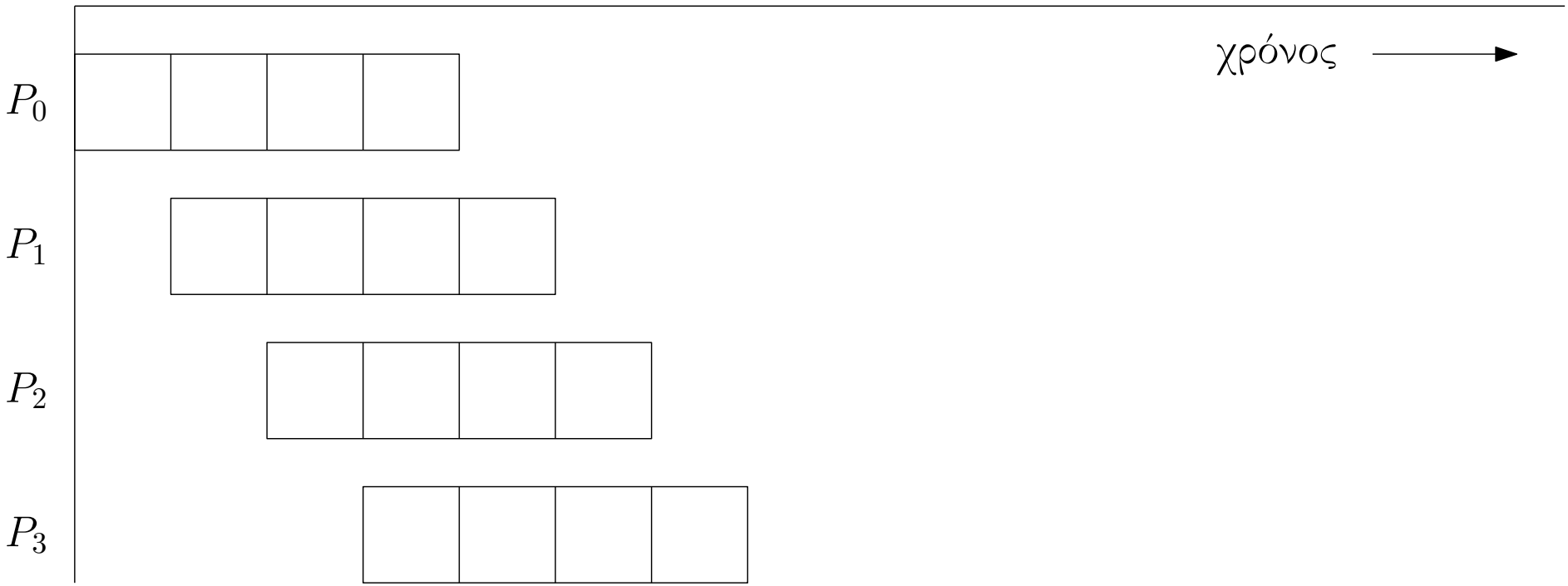
τοπολογία δικτύου  τεχνικές μεταφοράς μηνυμάτων



# Εμφύτευση Δικτύων στον Υπερκύβο



# Εμφύτευση Δικτύων στον Υπερκύβο



# Υπολογισμός κόστους επικοινωνίας

$$t_{comm} = t_s + (m\, t_w + t_h)\, l$$

όπου $m$ = πλήθος words, $l$ = πλήθος συνδέσμων

$$t_{comm} = t_s + m\, t_w\, l, \quad \mathcal{O}(ml) \quad \text{store and forward}$$

$$t_{comm} = t_s + l\, t_h + m\, t_w, \quad \mathcal{O}(l+m) \quad \text{cut through}$$

$l = 1$ τοπικότητα

# Βασικές Λειτουργίες Επικοινωνίας

- Οι σύνδεσμοι επικοινωνίας είναι διπλής κατεύθυνσης.
- Ο κάθε επεξεργαστής μπορεί να στείλει ένα μήνυμα σε ένα μόνο από τους συνδέσμους του κάθε φορά.
- Όμοια μπορεί να λάβει ένα μήνυμα από ένα σύνδεσμο κάθε φορά.
- Μπορεί να λάβει ένα μήνυμα ενώ στέλνει ένα άλλο την ίδια στιγμή στον ίδιο ή διαφορετικό σύνδεσμο.

**SF δρομολόγηση**

- $t_s + t_w m \lfloor p/2 \rfloor$ δακτύλιο
- $t_s + 2t_w m \lfloor p/2 \rfloor$ δίκτυο
- $t_s + t_w m \log p$ υπερκύβο

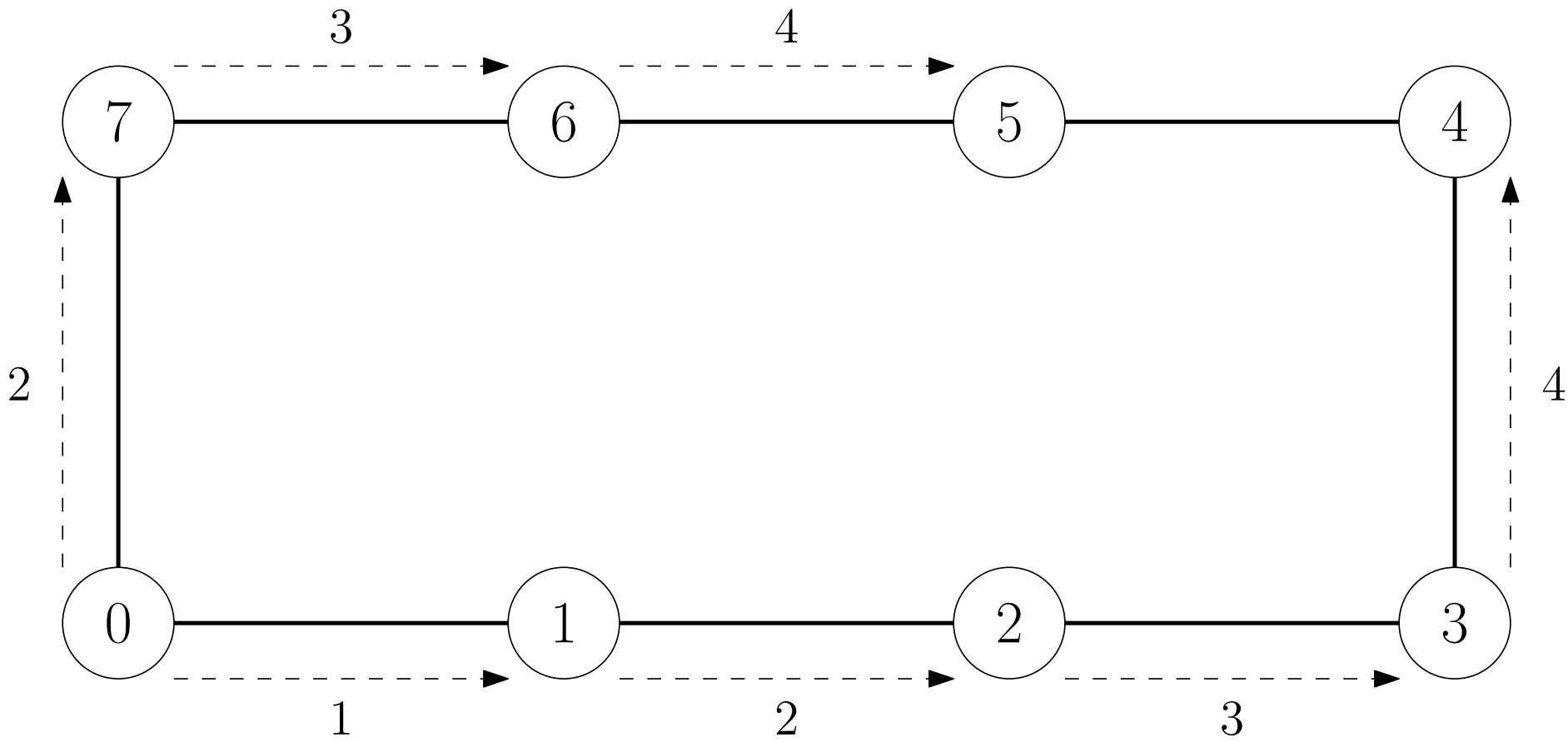
**CT δρομολόγηση**

$t_s + m\, t_w + t_h\, l$

Ένας προς όλους Μετάδοση

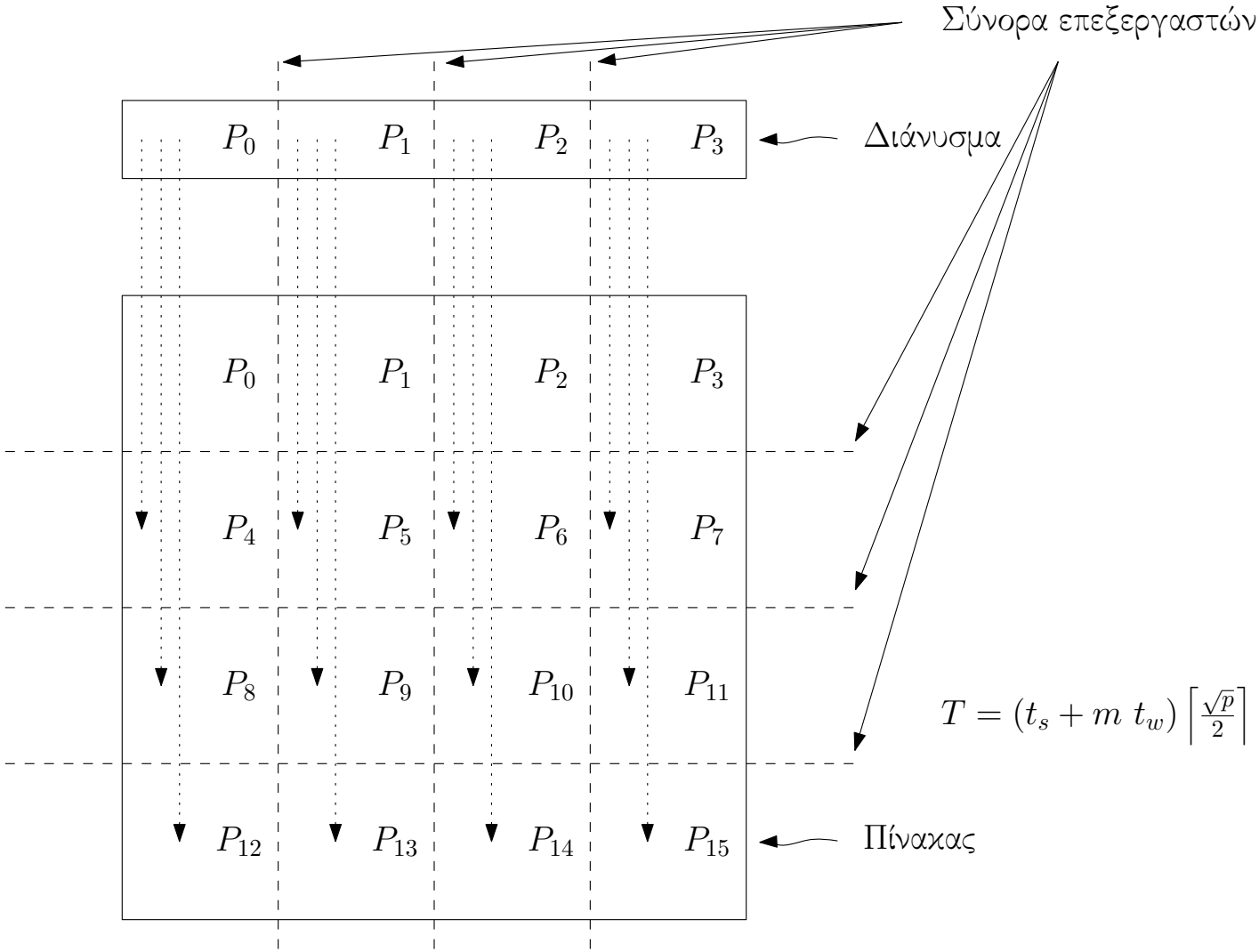
Δακτύλιος SF

# Βασικές Λειτουργίες Επικοινωνίας




Σχήμα: Δακτύλιος, $T = t_s + t_w m\lceil p/2 \rceil$

# Βασικές Λειτουργίες Επικοινωνίας

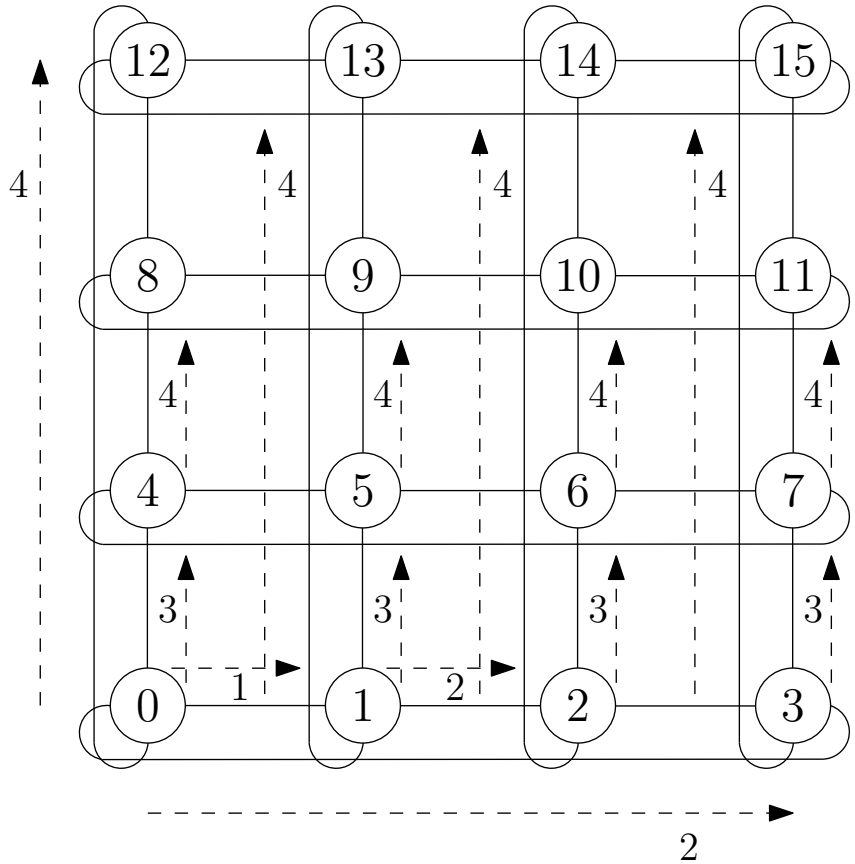



Σχήμα: Ένας προς όλους μετάδοση για τον πολλαπλασιασμό ενός $4 \times 4$ πίνακα με ένα $4 \times 1$ διάνυσμα

# Βασικές Λειτουργίες Επικοινωνίας

$$T = 2(t_s + t_w\ m)\left\lceil \frac{\sqrt{p}}{2} \right\rceil$$

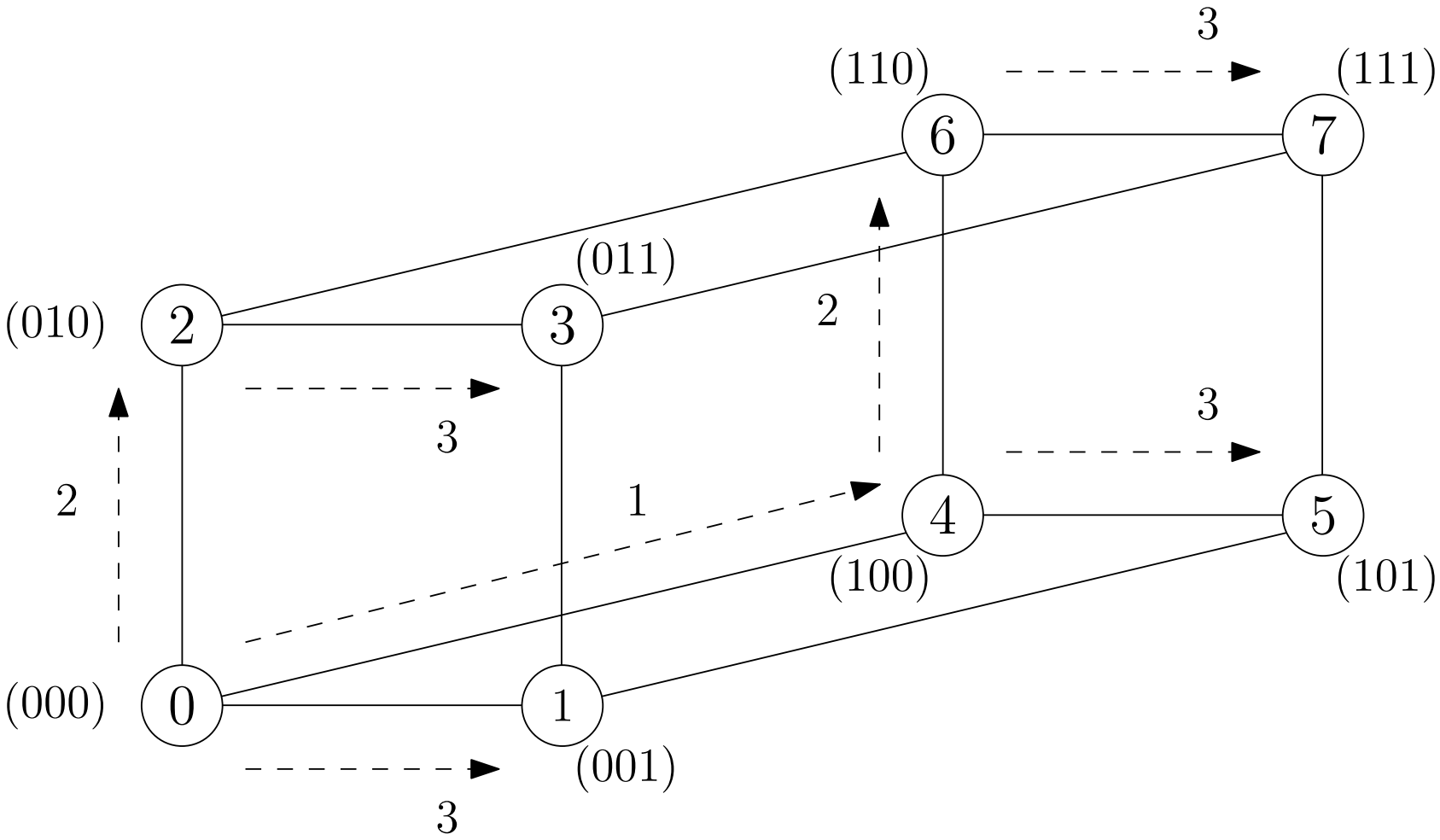
$$T = 3(t_s + t_w\ m)\left\lceil \frac{p^{1/3}}{2} \right\rceil \quad 30 \text{ mesh}$$




Σχήμα: Ένας προς όλους μετάδοση σε πλέγμα 16 επεξεργαστών με store and forward δρομολόγηση

# Βασικές Λειτουργίες Επικοινωνίας

$$T = (t_s + t_w\ m) \log p$$




Σχήμα: Ένας προς όλους μετάδοση σε 3-D υπερκύβο

# Βασικές Λειτουργίες Επικοινωνίας

**Διαδικασία 1:** ONE TO ALL BC $(d, my\_id, X)$

$\overbrace{mask}^{111} = 2^d - 1 \quad (d = 3)$ · όλα τα d bits της μάσκας ίσα με 1

**για** $i = d - 1$ **έως** $0$ · Εξωτερικό loop $(i = 2)$

**κάνε**

  $mask = mask$ XOR $2^i$ · i bit της μάσκας ίσο με 0

  **αν** $my\_id$ *AND* $mask = 0$ · αν τα μικρότερα i bits του $my\_id$ είναι $0$

  **τότε**

    **αν** $my\_id$ *AND* $2^i = 0$ · $000 \xrightarrow{\texttt{send}} 100$

    **τότε**

      $msg\_destination = my\_id$ XOR $2^i$

      **Στείλε** $Q$ στο $msg\_destination$

    **τέλος**

  **αλλιώς**

    $msg\_source = my\_id$ XOR $2^i$ · $000 \xrightarrow{\texttt{receive}} 100$

    **Λάβε** $X$ από $msg\_source$

  **τέλος**

**τέλος**

**Διαδικασία 2:** GENERAL ONE TO ALL BC $(d, my\_id, source, X)$

$my\_virtual\_id = my\_id$ XOR $source$ $mask = 2^d - 1$
**για** $i = d - 1$ **έως** $0$ **κάνε**
  $mask = mask$ XOR $2^i$ · i bit της μάσκας ίσο με 0
  **αν** $my\_virtual\_id$ *AND* $mask = 0$ **τότε**
    **αν** $my\_virtual\_id$ *AND* $2^i = 0$ **τότε**
      $virtual\_dest = my\_virtual\_id$ XOR $2^i$
      **Στείλε** $Q$ στο $virtual\_dest$ XOR $source$ · μετατροπή
       $virtual\_dest$ στην ετικέτα του φυσικού προορισμού
    **τέλος**
  **αλλιώς**
    $virtual\_source = my\_virtual\_id$ XOR $2^i$
    **Λάβε** $X$ στο $virtual\_source$ XOR $source$ · μετατροπή
     $virtual\_source$ στην ετικέτα της φυσικής πηγής
  **τέλος**
**τέλος**

# Βασικές Λειτουργίες Επικοινωνίας

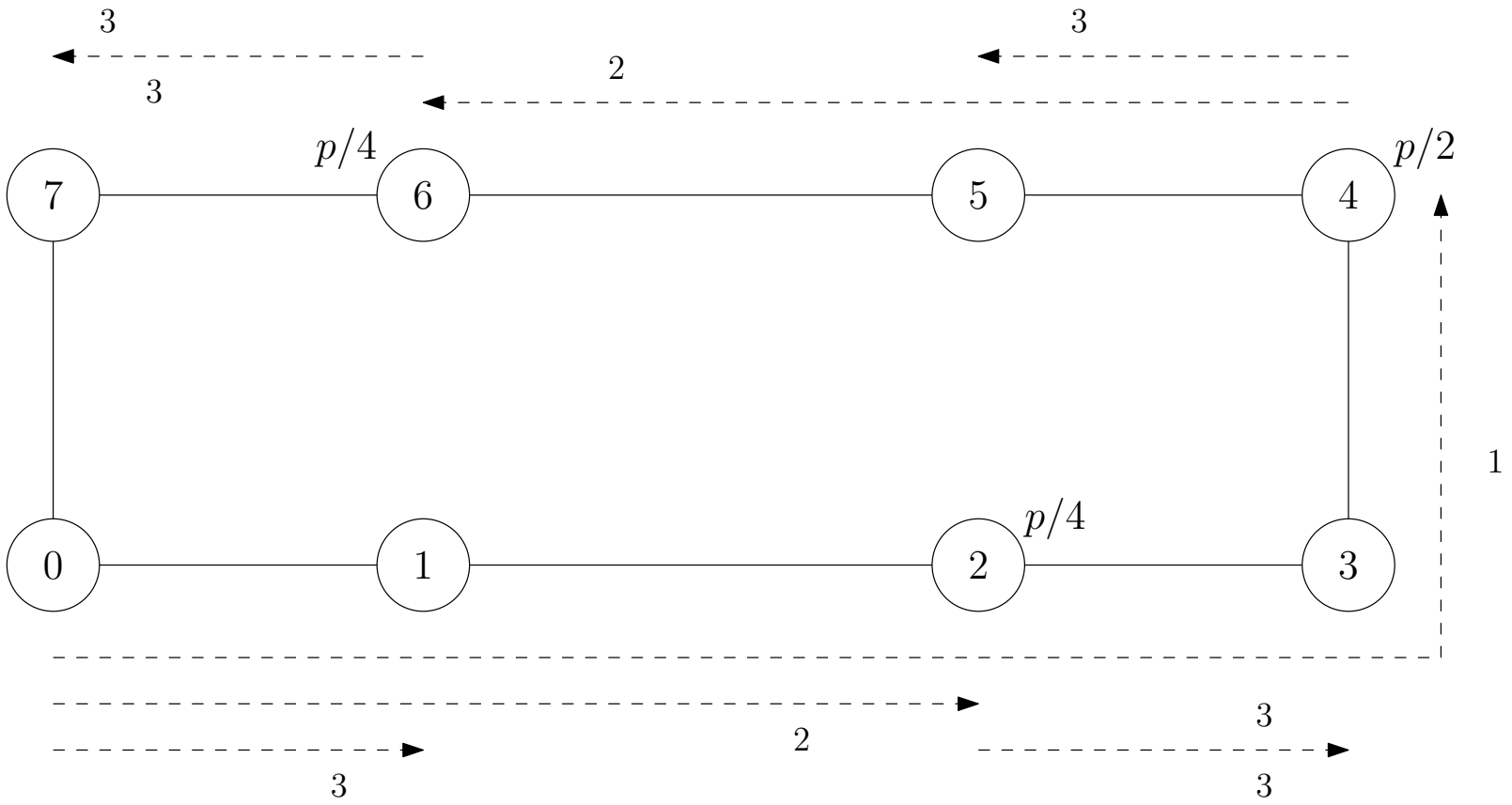
**Διαδικασία 3:** SINGLE NODE ACC $(d, my\_id, m, X, sum)$

**για** $j = 0$ **έως** $m - 1$ **κάνε**
  $sum[j] = X[j]$ $mask = 0$
**τέλος**
**για** $i = 0$ **έως** $d - 1$ **κάνε**
  Επιλογή επεξεργαστών με μικρότερα i bits ίσα με 0
  **αν** $my\_id$ *AND* $mask = 0$ **τότε**
    **αν** $my\_id$ *AND* $2^i \neq 0$ **τότε**
      $msg\_destination = my\_id$ XOR $2^i$
      **Στείλε** $sum$ στο $msg\_destination$
    **τέλος**
  **αλλιώς**
    $msg\_source = my\_id$ XOR $2^i$
    **Λάβε** $X$ από $msg\_source$ **για** $j = 0$ **έως** $m - 1$ **κάνε**
      $sum[j] = sum[j] + X[j]$
    **τέλος**
  **τέλος**
  $mask = mask$ XOR $2^i$ · i bit της μάσκας ίσο με 0
**τέλος**

# Βασικές Λειτουργίες Επικοινωνίας

- Ένας σε όλους μετάδοση με cut through δρομολόγηση
- Στο i-οστό βήμα κάθε επεξεργαστής που έχει τα δεδομένα τα στέλνει σε έναν επεξεργαστή απόστασης $p/2^i$

# Βασικές Λειτουργίες Επικοινωνίας



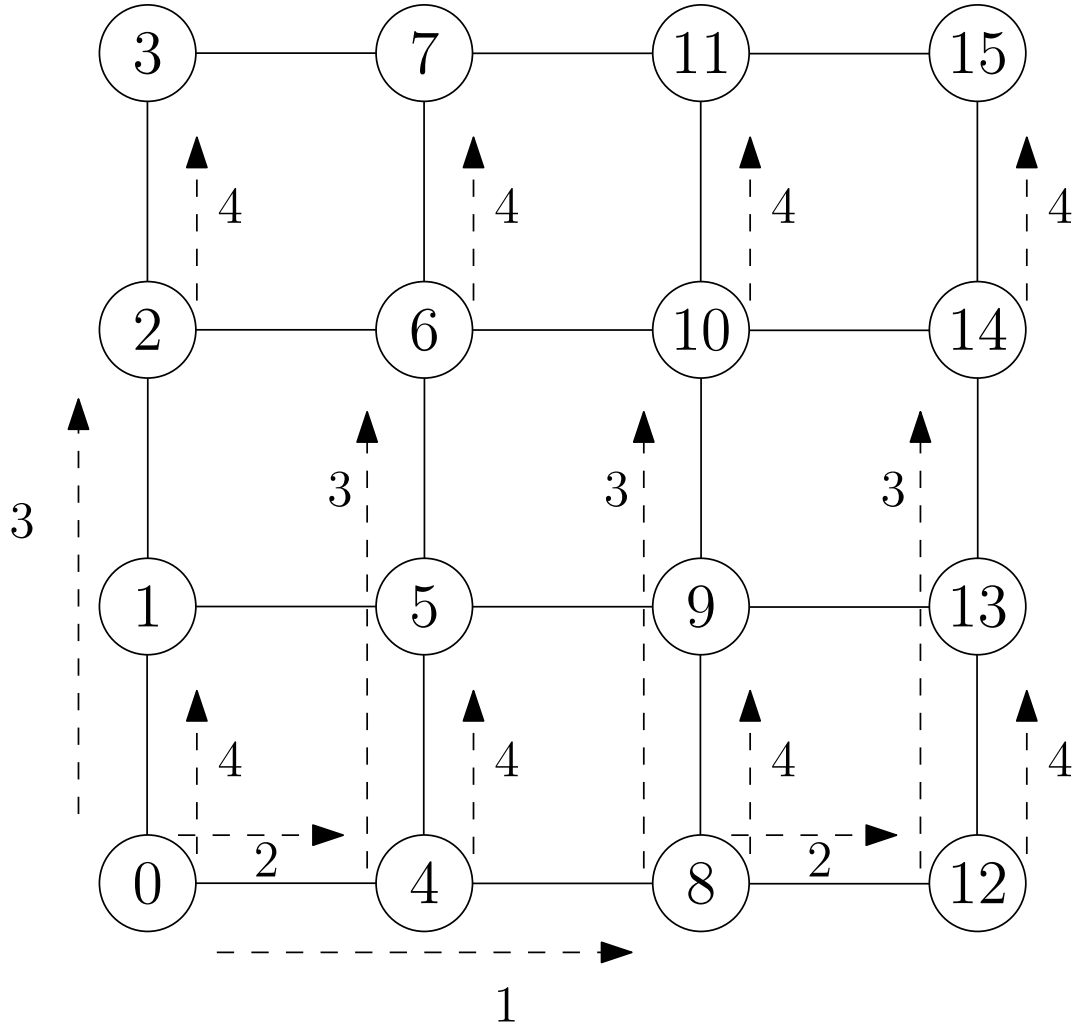

Σχήμα: Ένας σε όλους μετάδοση με CT δρομολόγηση

$$T_{\text{ένας-προς-όλους}} = \sum^{\log p} (t_s + t_w\, m + t_h\, p/2^i) = (t_s + t_w\, m) \log p + t_h(p-1)$$

# Βασικές Λειτουργίες Επικοινωνίας

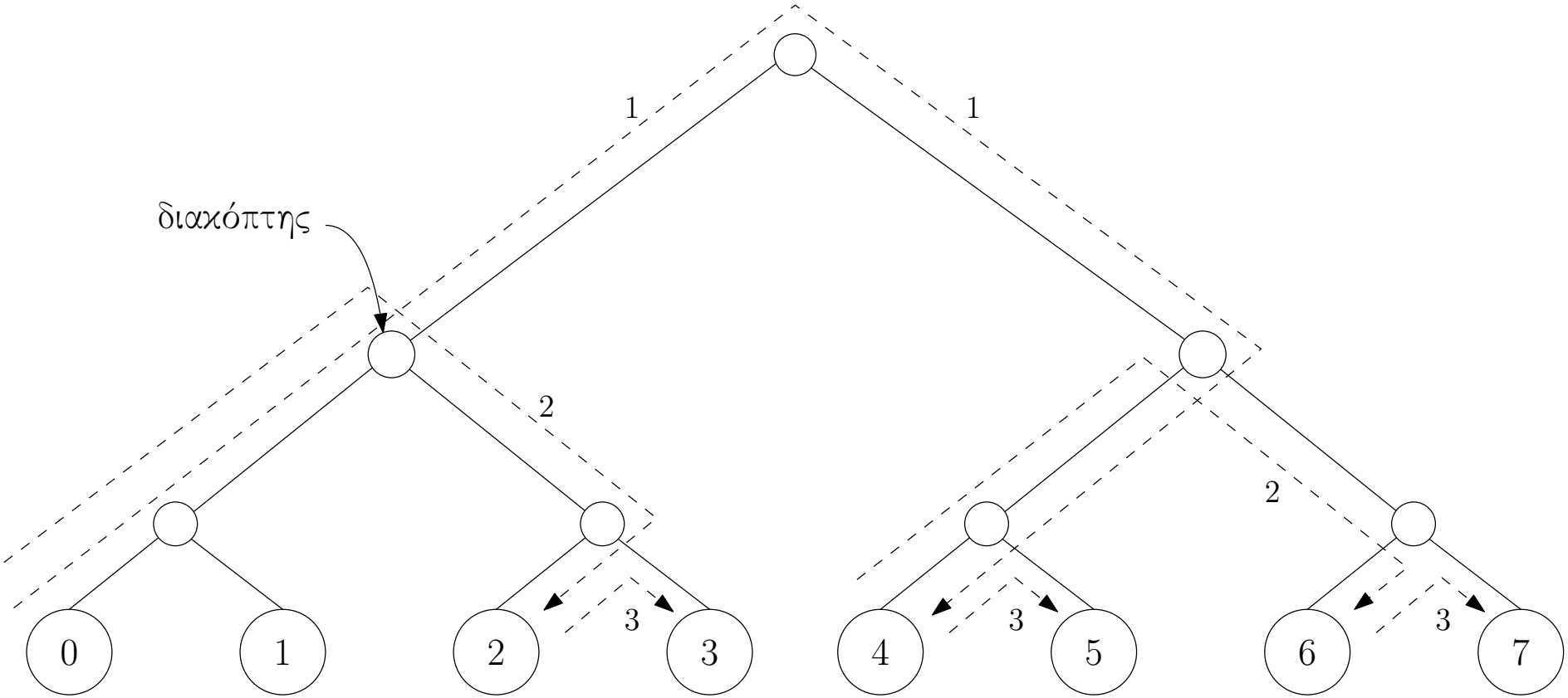
- Μετάδοση ένας προς όλους με CT δρομολόγηση
- $t_s + m\, t_w \log \sqrt{p} + (\sqrt{p} - 1) t_h$ κάθε φάση

# Βασικές Λειτουργίες Επικοινωνίας




$$T_{\text{ένας-προς-όλους}} = (t_s + m\, t_w) \log p + 2(\sqrt{p} - 1) t_h$$

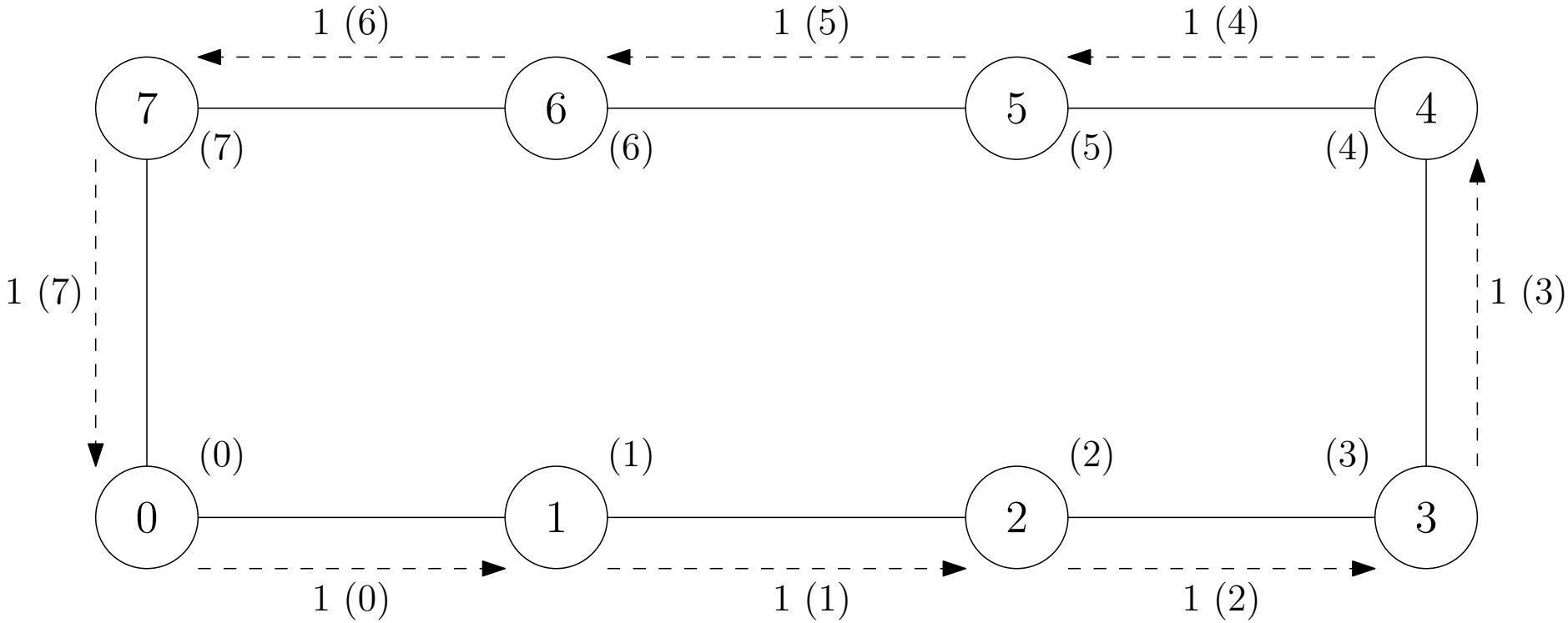
# Βασικές Λειτουργίες Επικοινωνίας



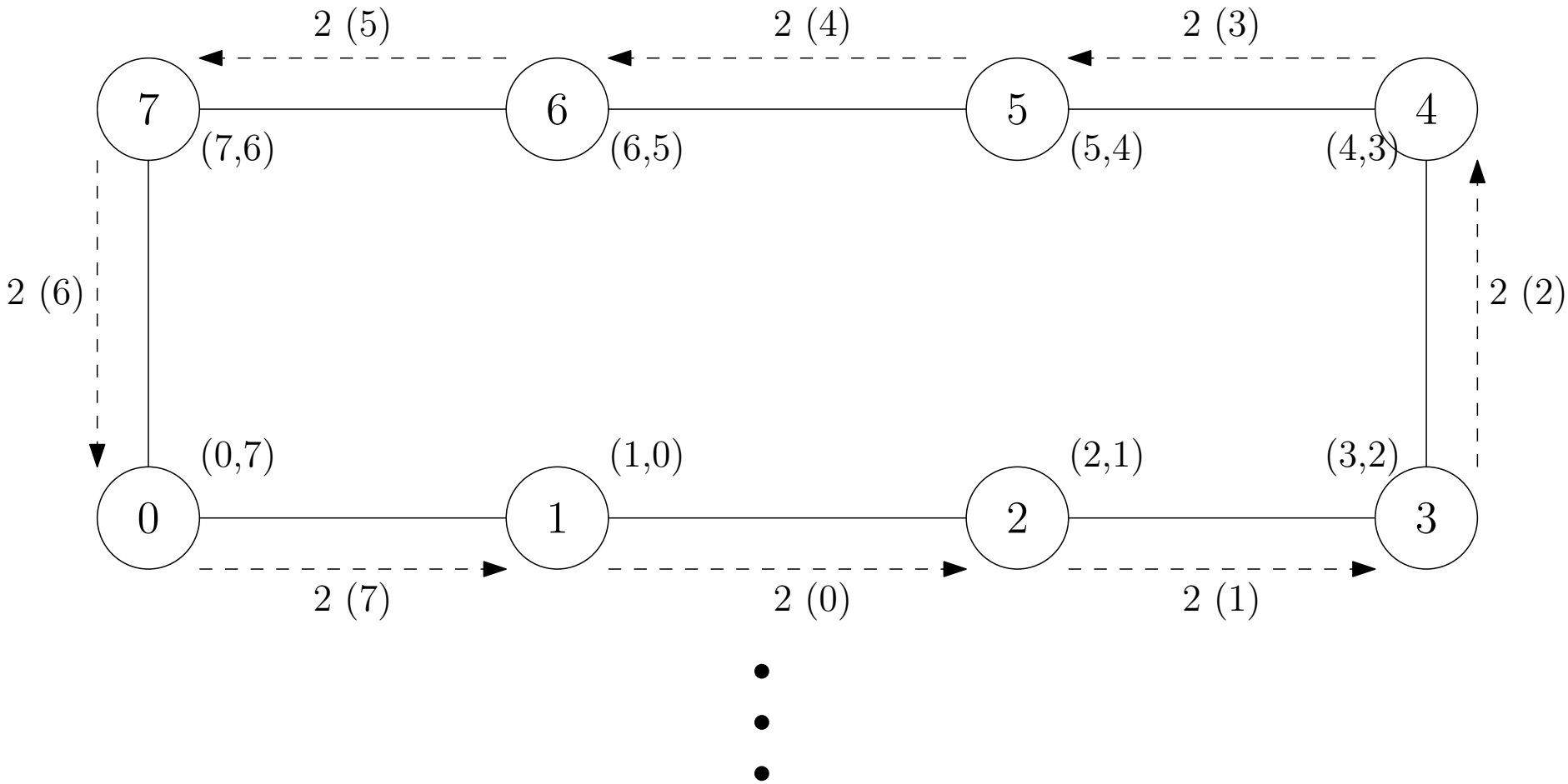

Σχήμα: Ισοζυγισμένο Δυαδικό Δέντρο

$$T_{\text{ένας-προς-όλους}} = (t_s + m\, t_w + (\log p + 1)) \log p$$

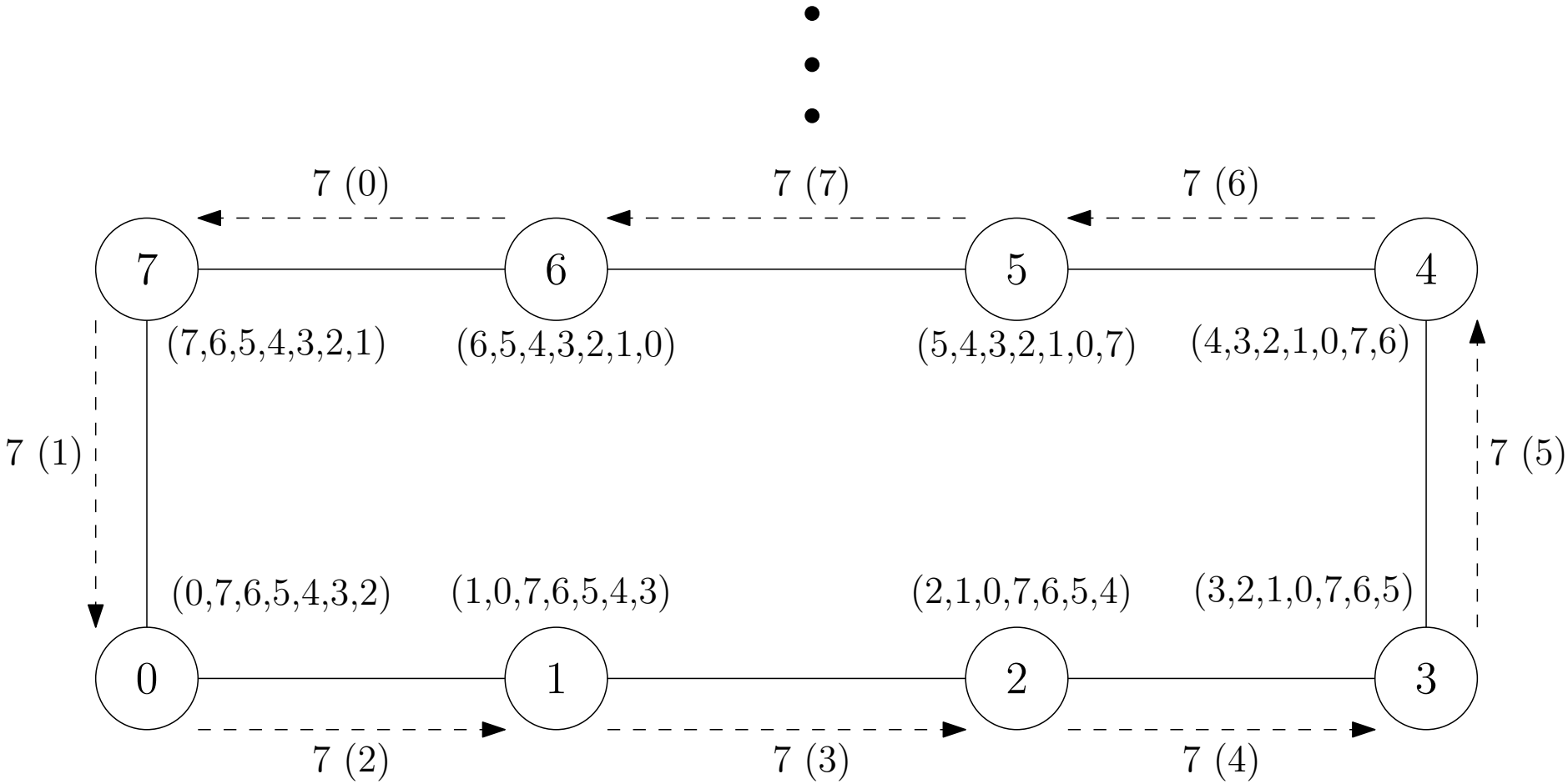
# Βασικές Λειτουργίες Επικοινωνίας



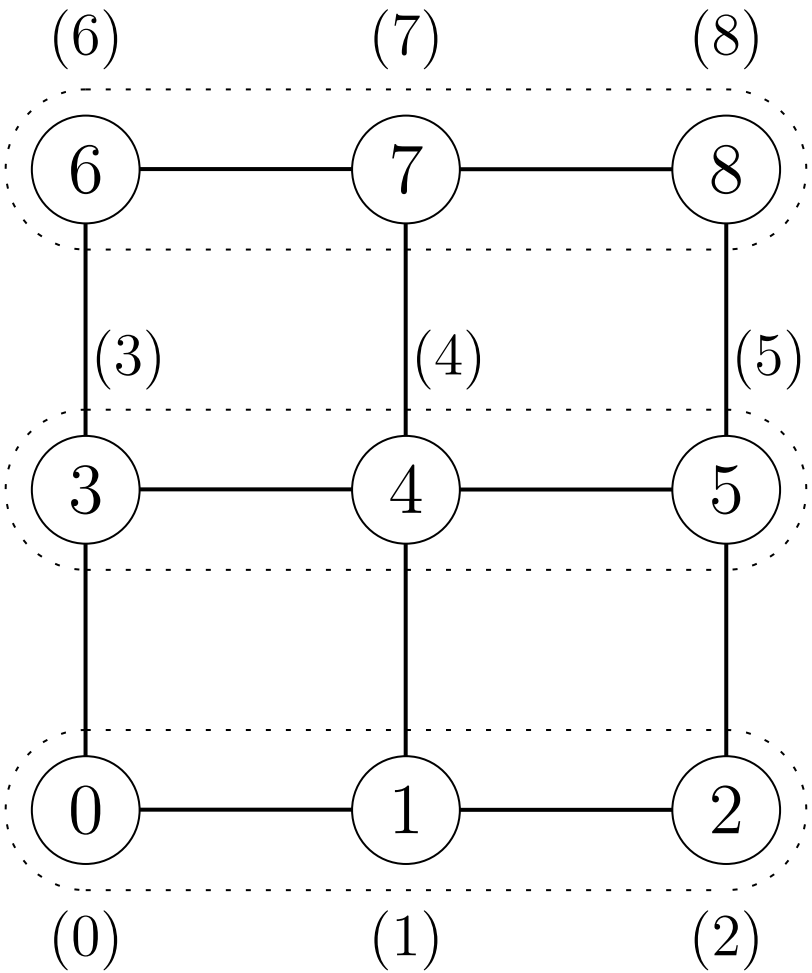
# Βασικές Λειτουργίες Επικοινωνίας

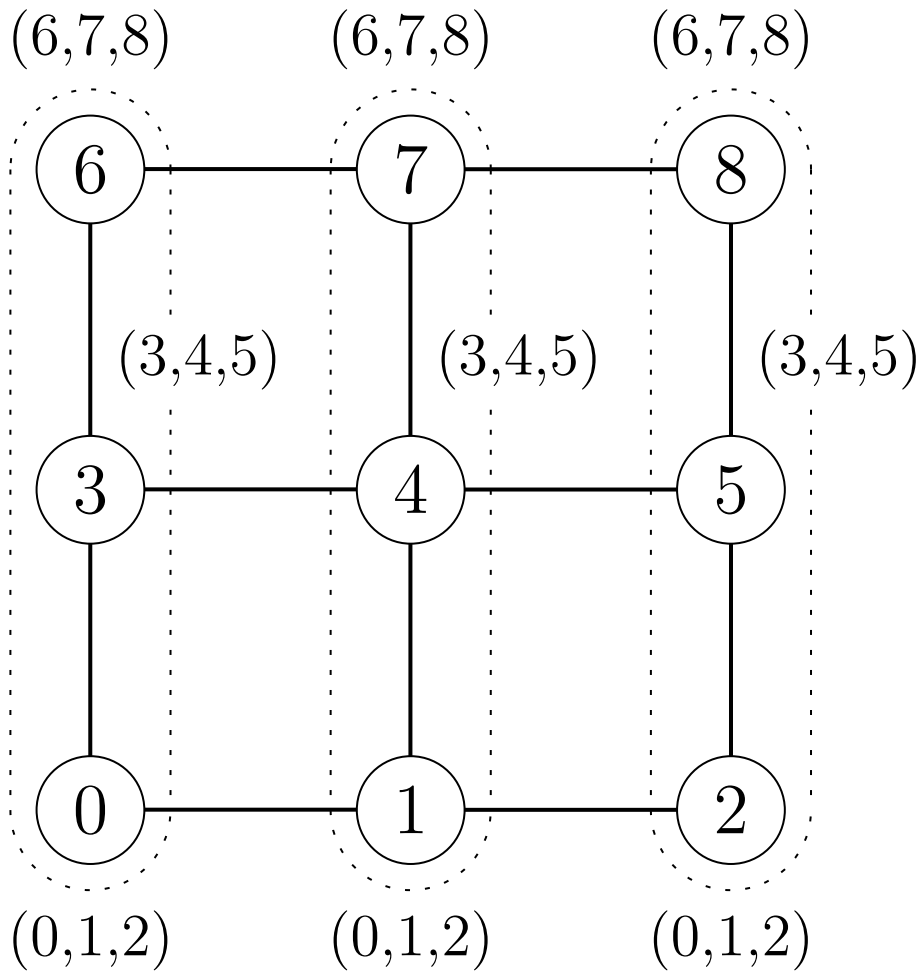


# Βασικές Λειτουργίες Επικοινωνίας



# Βασικές Λειτουργίες Επικοινωνίας



**Διαδικασία 4:** ALL TO ALL BC RING $(my\_id, my\_msg, p, result)$

$left = (my\_id - 1) \mod p$
$right = (my\_id + 1) \mod p$
$result = my\_msg$
$msg = result$
**για** $i = 1$ **έως** $p - 1$ **κάνε**
  **Στείλε** $msg$ από $left$
  $result = result \cup msg$
**τέλος**

# Βασικές Λειτουργίες Επικοινωνίας

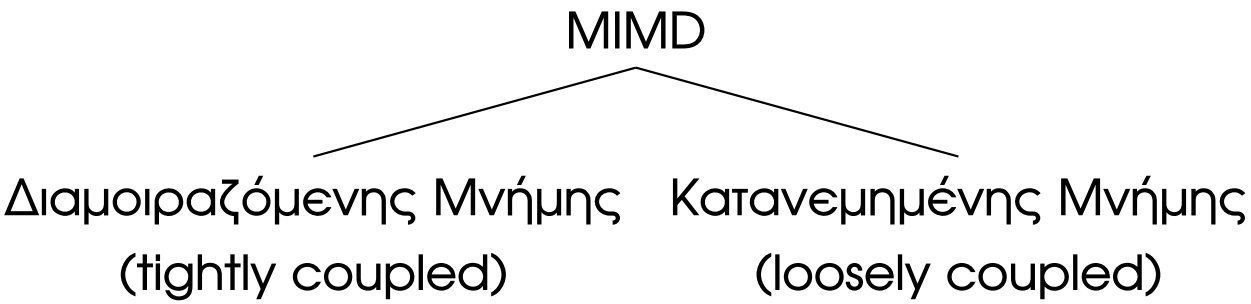
**Διαδικασία 5:** ALL TO ALL BC MESH $(my\_id, my\_msg, p, result)$

Επικοινωνία κατά μήκος γραμμής
$left = (my\_id - 1) \mod p$
$right = (my\_id + 1) \mod p$
$result = my\_msg$
$msg = result$
**για** $i = 1$ **έως** $\sqrt{p} - 1$ **κάνε**
  **Στείλε** $msg$ στο $right$
  **Λάβε** $msg$ από $left$
  $result = result \cup msg$
**τέλος**

Επικοινωνία κατά μήκος στήλης
$up = (my\_id - \sqrt{p}) \mod p$
$down = (my\_id + \sqrt{p}) \mod p$
$msg = result$
**για** $i = 1$ **έως** $\sqrt{p} - 1$ **κάνε**
  **Στείλε** $msg$ στο $down$
  **Λάβε** $msg$ από $up$
  $result = result \cup msg$
**τέλος**

# Αξιολόγηση MIMD Παράλληλων Συστημάτων

- Scheduling




**Ανάλυση Παράλληλων Αλγορίθμων**

$$T_p = T_{comp} + T_{comm}$$

# Αξιολόγηση MIMD Παράλληλων Συστημάτων

**Ταχύτητα και Αποδοτικότητα**

$$S_p = \frac{T_1}{T_p}$$

$S_p$ : ταχύτητα
$T_1$ : σειριακός χρόνος
$T_p$ : παράλληλος χρόνος σε $p$ επεξεργαστές

$$E_p = \frac{S_p}{p} \qquad \text{αποδοτικότητα}$$

# Αξιολόγηση MIMD Παράλληλων Συστημάτων

Είναι φανερό ότι $S_p \geq 1$. Επίσης ένα βήμα ενός παράλληλου αλγορίθμου χρησιμοποιώντας $p$ επεξεργαστές χρειάζεται $p$ βήματα το πολύ όταν υλοποιηθεί σειριακά. Άρα

$$T_1 \leq p\, T_p$$

Συνεπώς

$$1 \leq S_p \leq p \quad \text{και} \quad 0 \leq E_p \leq 1$$

**Ο νόμος του Amdahl**

$$T_1 = T_{seq} + T_{par}, \quad T_{seq} = f\, T_1 \quad \text{και} \quad T_{par} = (1-f)\, T_1$$

$f$ : Ποσοστό σειριακού χρόνου

# Αξιολόγηση MIMD Παράλληλων Συστημάτων

Με $p$ επεξεργαστές

$$T_p \geq T_{seq} + T_{par}/p$$

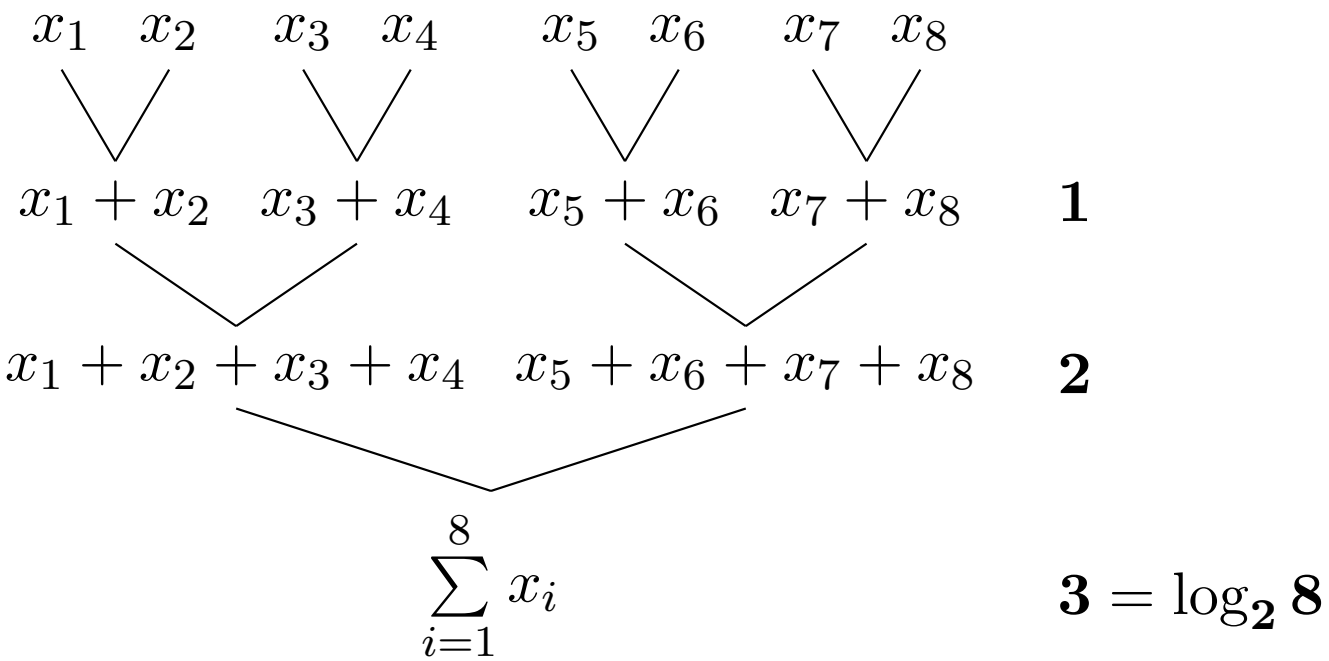
$$S_p = \frac{T_1}{T_f} \leq \frac{1}{f + \frac{1-f}{p}} \leq \frac{1}{f}$$

Αν $f = 10\% \rightarrow S_p \leq 10$ για οποιοδήποτε $p$!

# Αξιολόγηση MIMD Παράλληλων Συστημάτων

**Παράδειγμα (SIMD)**

Υπολογισμός του $\sum_{i=1}^{N} x_i$



# Αξιολόγηση MIMD Παράλληλων Συστημάτων

Αριθμός Επεξεργαστών $p = \frac{N}{2}$ CREW

$\log N$ βήματα

$$S_p = \frac{T_1}{T_p} = \frac{N-1}{\log N} \approxeq \frac{N}{\log N}$$

$$E_p \approxeq \frac{2}{\log N} \to 0 \quad \text{για} \quad N \to \infty$$

**Ανάλογος αλγόριθμος για προβλήματα**

- εσωτερικο γινόμενο διανυσμάτων
- εύρεση μέγιστου - ελάχιστου αριθμού κ.α.

# Αξιολόγηση MIMD Παράλληλων Συστημάτων

Αριθμός Επεξεργαστών $p = \frac{N}{2}$ CREW

$\log N$ βήματα

$$S_p = \frac{T_1}{T_p} = \frac{N-1}{\log N} \approxeq \frac{N}{\log N}$$

$$E_p \approxeq \frac{2}{\log N} \to 0 \quad \text{για} \quad N \to \infty$$

**Ανάλογος αλγόριθμος για προβλήματα**

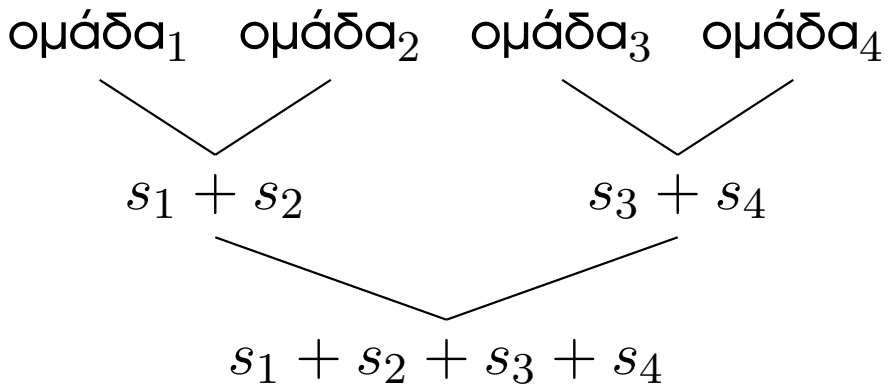
- εσωτερικο γινόμενο διανυσμάτων
- εύρεση μέγιστου - ελάχιστου αριθμού κ.α.

# Αξιολόγηση MIMD Παράλληλων Συστημάτων

**Παράδειγμα**

Υπολογισμός του $\sum_{i=1}^{N} x_i$ με τη χρήση $p$ επεξεργαστών $1 \leq p \leq N$.

1. Χωρισμός των $N$ αριθμών σε $p$ ομάδες. Κάθε ομάδα περιέχει το πολύ $\left\lceil \frac{N}{p} \right\rceil$ αριθμούς
2. Καταχώρηση μιας ομάδας σε κάθε ένα από τους $p$ επεξεργαστές
3. Υπολογισμός του αθροίσματος των αριθμών σε κάθε ομάδα σειριακά σε $\left\lceil \frac{N}{p} \right\rceil - 1$ βήματα
4. Υπολογισμός του αθροίσματος των επιμέρους αθροισμάτων (σε μορφή δυαδικού δέντρου)



# Αξιολόγηση MIMD Παράλληλων Συστημάτων

$$T_p = \left\lceil \frac{N}{p} \right\rceil - 1$$

$$S_p = \frac{\overbrace{N-1}^{T_1}}{\lceil N/p \rceil + \lceil \log p \rceil - 1} \qquad \text{Αν} \qquad N = L\, p\, \log p$$

$$S_p = \frac{L\, p}{L+1} \rightarrow \lim_{L \rightarrow \infty} S_p = p \qquad \text{γραμμική στο p.}$$

Επίσης

$$E_p = \frac{1}{L+1} \rightarrow \lim_{L \rightarrow \infty} E_p = 1$$

# Αξιολόγηση MIMD Παράλληλων Συστημάτων

**Παράδειγμα**

Δίνονται τα $X = (x_1, x_2, \ldots, x_N), \quad Y = (y_1, y_2, \ldots, y_N)$. Να υπολογιστεί το $\sum_{i=1}^{N} x_i y_i$.

Έστω ότι το $p$ διαιρεί το $N$. Τότε

$$\sum_{i=1}^{N} x_i y_i = \sum_{i=1}^{N/p} x_i y_i + \sum_{i=N/p+1}^{2(N/P)} x_i y_i + \cdots + \sum_{i=(N/p)(k-1)+1}^{k(N/P)} x_i y_i + \cdots + \sum^{N} x_i y_i$$

Άρα ο $k$ επεξεργαστής υπολογίζει

$$\sum_{i=(N/p)(k-1)+1}^{k(N/P)} x_i y_i, \quad k = 1, 2, \ldots, p$$

# Αξιολόγηση MIMD Παράλληλων Συστημάτων

Για τον ανωτέρω υπολογισμό απαιτούνται

$$\frac{N}{p} \qquad \text{πολ/σμοί}$$

$$\frac{N}{p} - 1 \qquad \text{προσθέσεις}$$

# Αξιολόγηση MIMD Παράλληλων Συστημάτων

Τα μερικά αθροίσματα προστίθενται με τη μορφή του δυαδικού δέντρου σε $\lceil \log p \rceil$ βήματα. Άρα

$$T_p = 2(\frac{N}{p} - 1) + \lceil \log p \rceil$$

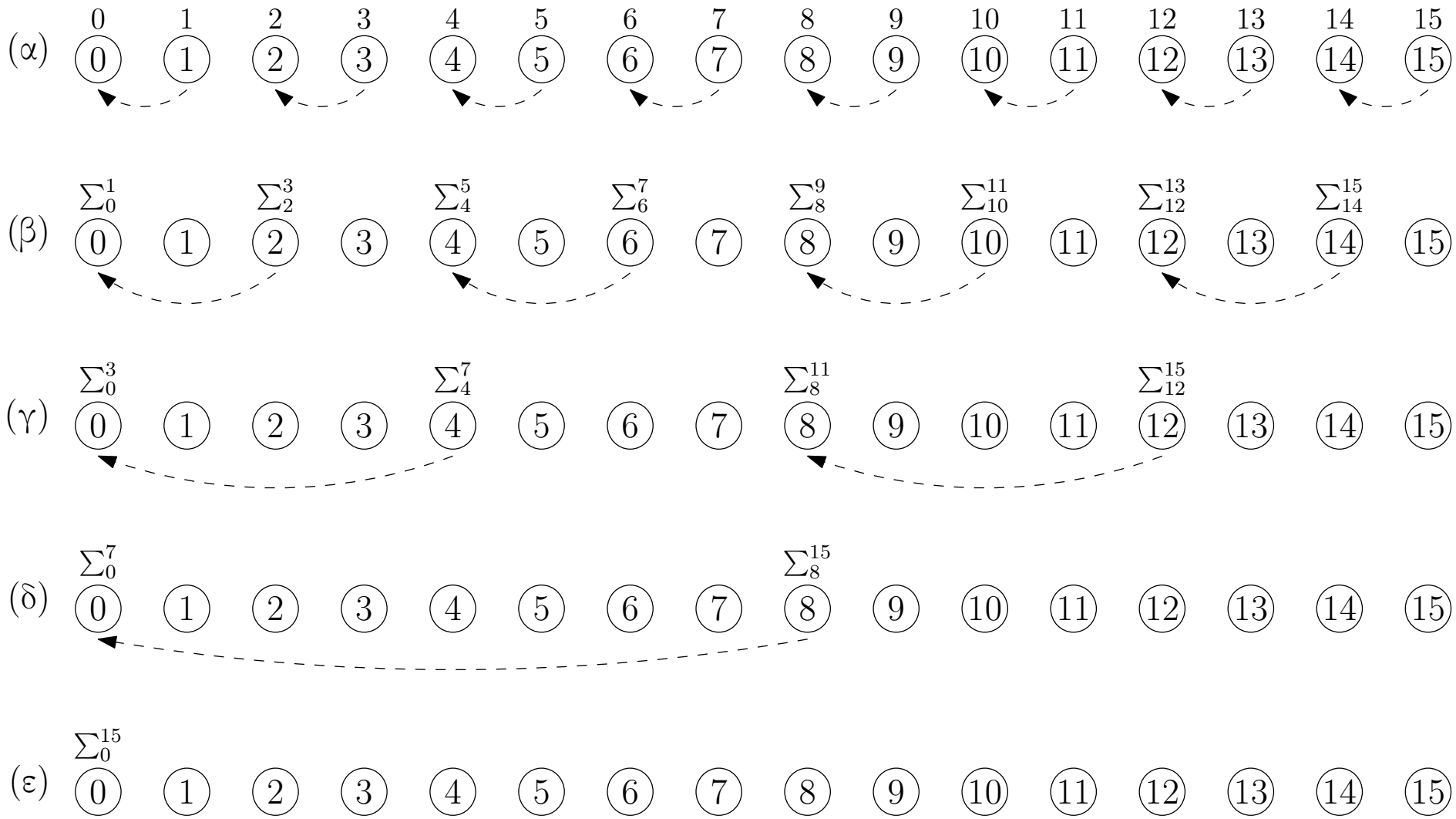
και

$$S_p = \frac{2N - 1}{2(\frac{N}{p} - 1) + \lceil \log p \rceil}$$

Αν $p = 2^k$ και $N = L\ p\ \log p$ τότε

$$S_p = \frac{2L}{2L + 1} p \rightarrow \lim_{L \rightarrow \infty} S_p = p \quad !$$

# Αξιολόγηση MIMD Παράλληλων Συστημάτων




Σχήμα: Πρόσθεση $n$ αριθμών σε ένα Υπερκύβο με $n$ επεξεργαστές

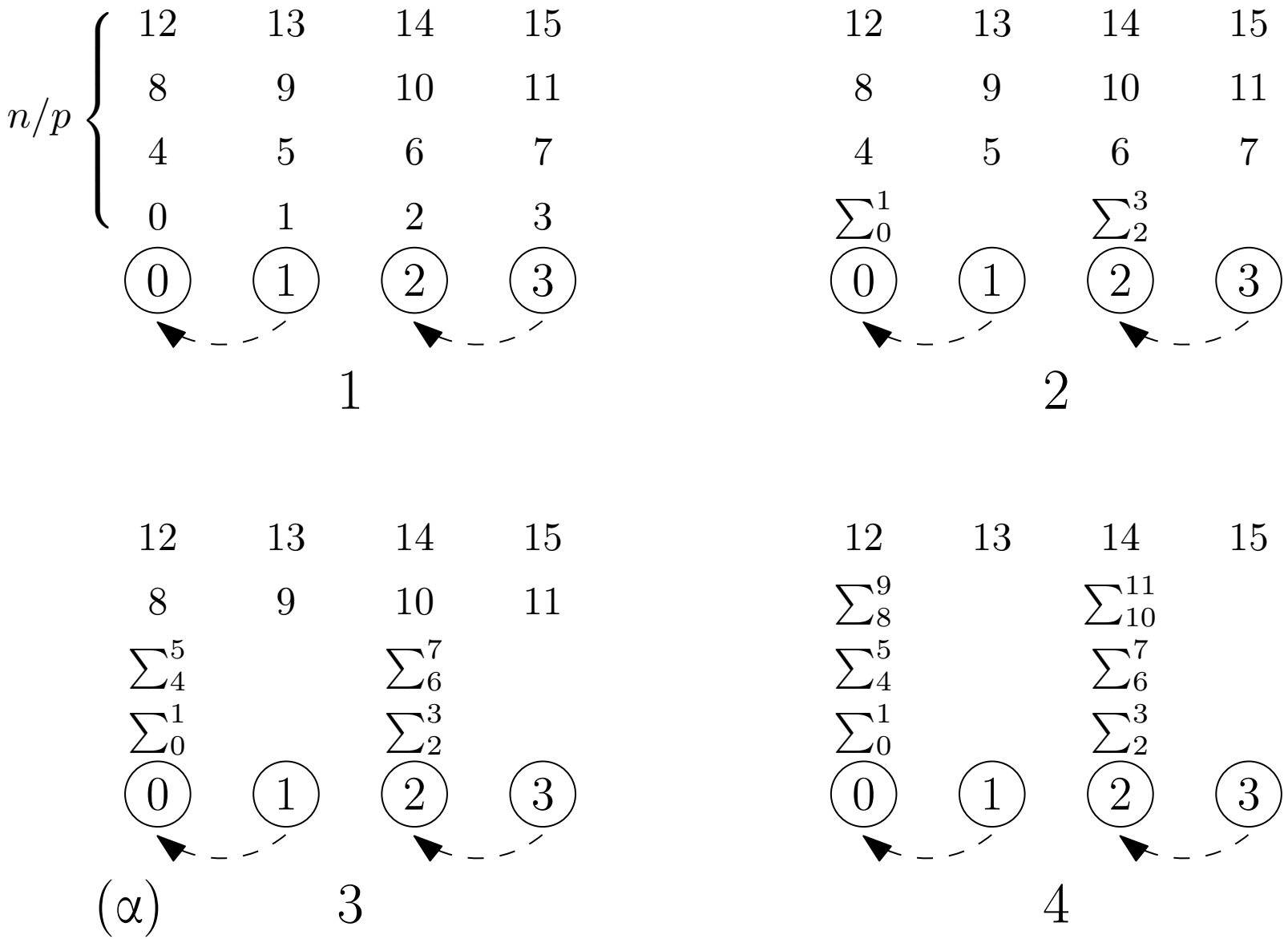
# Αξιολόγηση MIMD Παράλληλων Συστημάτων

$$T_p = \Theta(\log n), \quad S_p = \Theta(\frac{n}{\log n})$$

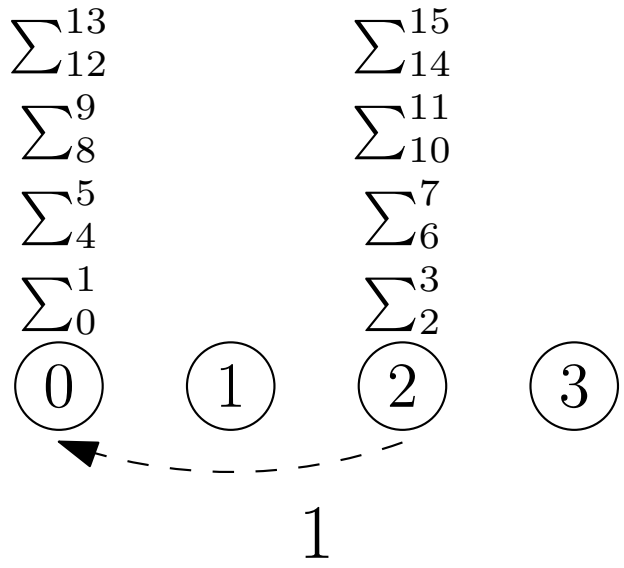
$$E_p = \Theta(\frac{1}{\log n}), \quad c_p = \Theta(n \log n)$$

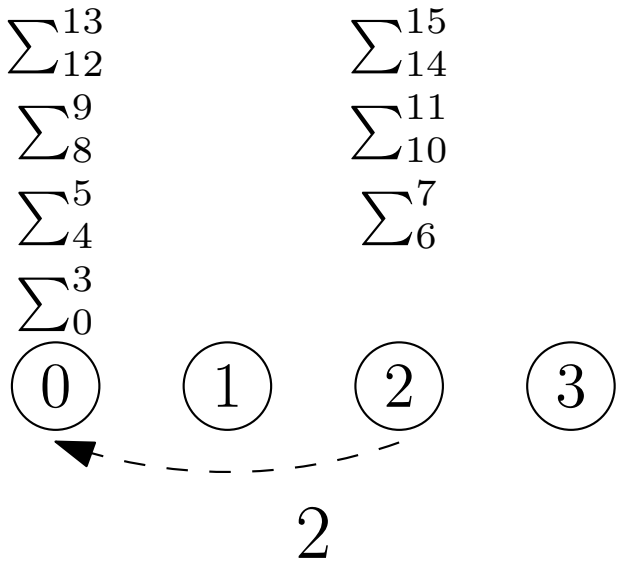
όχι βέλτιστο κόστος

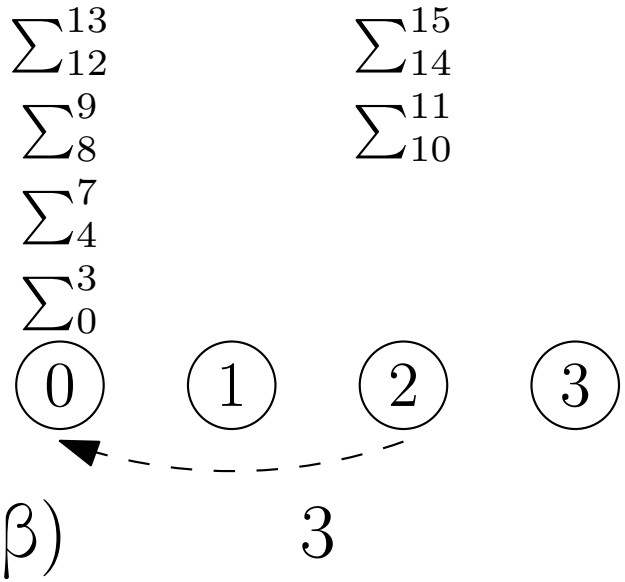
# Αξιολόγηση MIMD Παράλληλων Συστημάτων

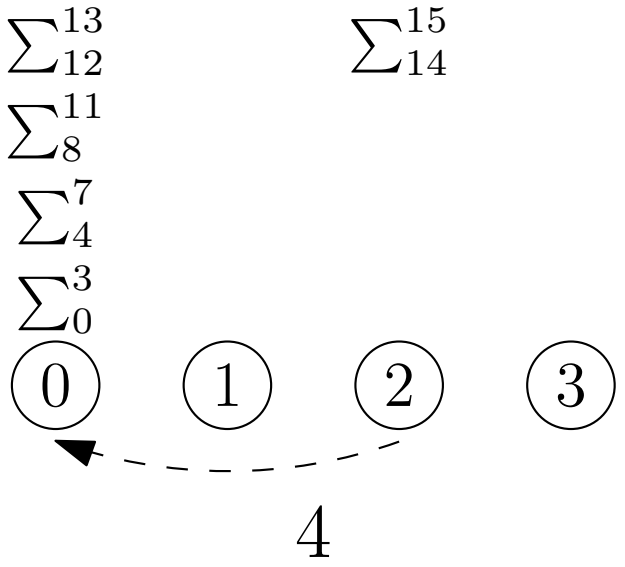


# Αξιολόγηση MIMD Παράλληλων Συστημάτων

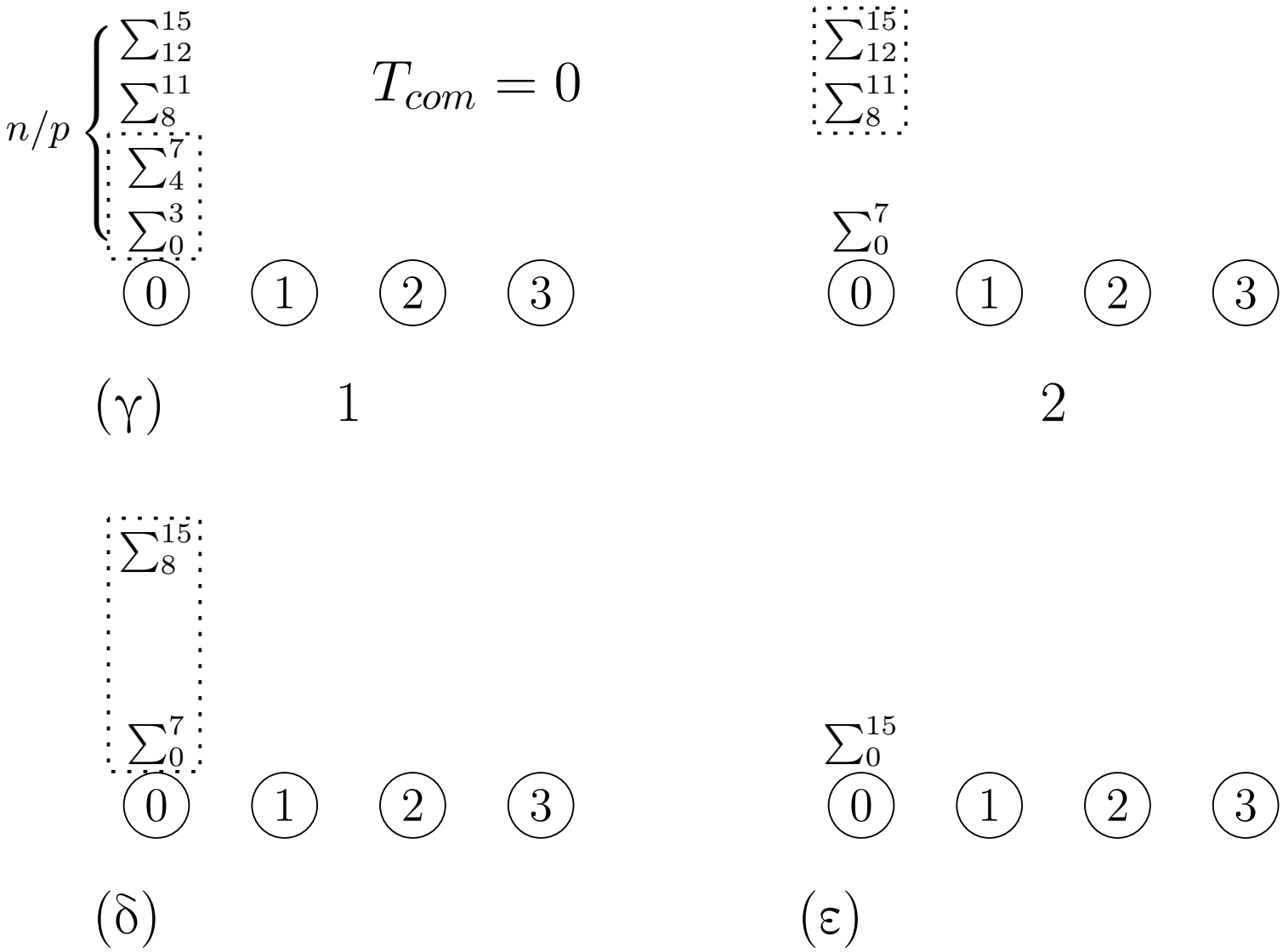


# Αξιολόγηση MIMD Παράλληλων Συστημάτων




Σχήμα: Πρόσθεση n αριθμών σε υπερκύβο με p επεξεργαστές, όπου $p < n$. Τα $n, p$ είναι δυνάμεις του 2

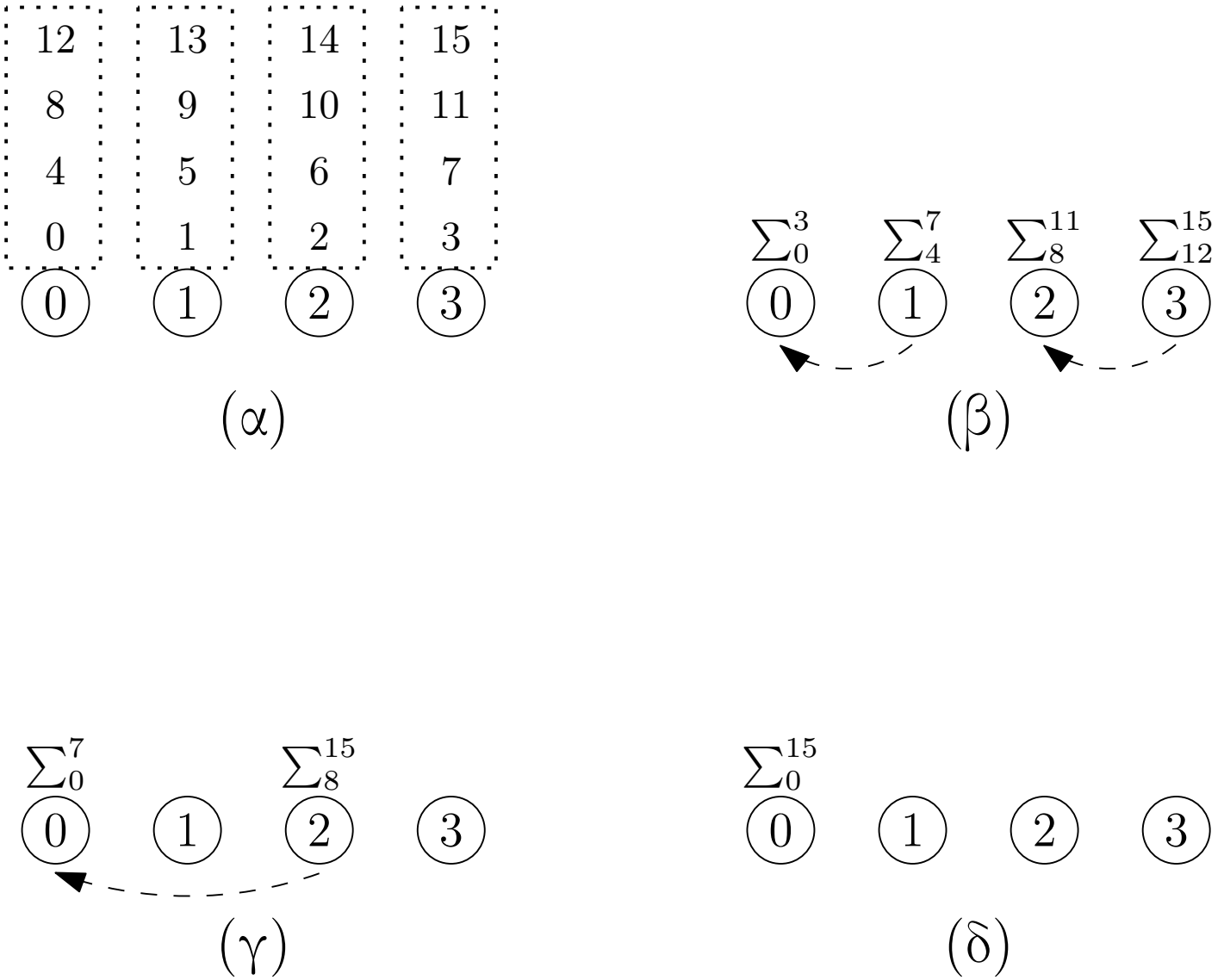
# Αξιολόγηση MIMD Παράλληλων Συστημάτων

$$T_p = \Theta\left(\left(\frac{n}{p}\right)\ \log p\right)$$

$$c_p = \Theta(n \log p) > \Theta(n) = c_1$$

όχι βέλτιστο κόστος

# Αξιολόγηση MIMD Παράλληλων Συστημάτων




Σχήμα: Διαφορετικός τρόπος για την εύρεση του προηγούμενου αθροίσματος

# Αξιολόγηση MIMD Παράλληλων Συστημάτων

$$T_p = \Theta\left(n/p + \log p\right), \quad c_p = \Theta(n + p\log p)$$

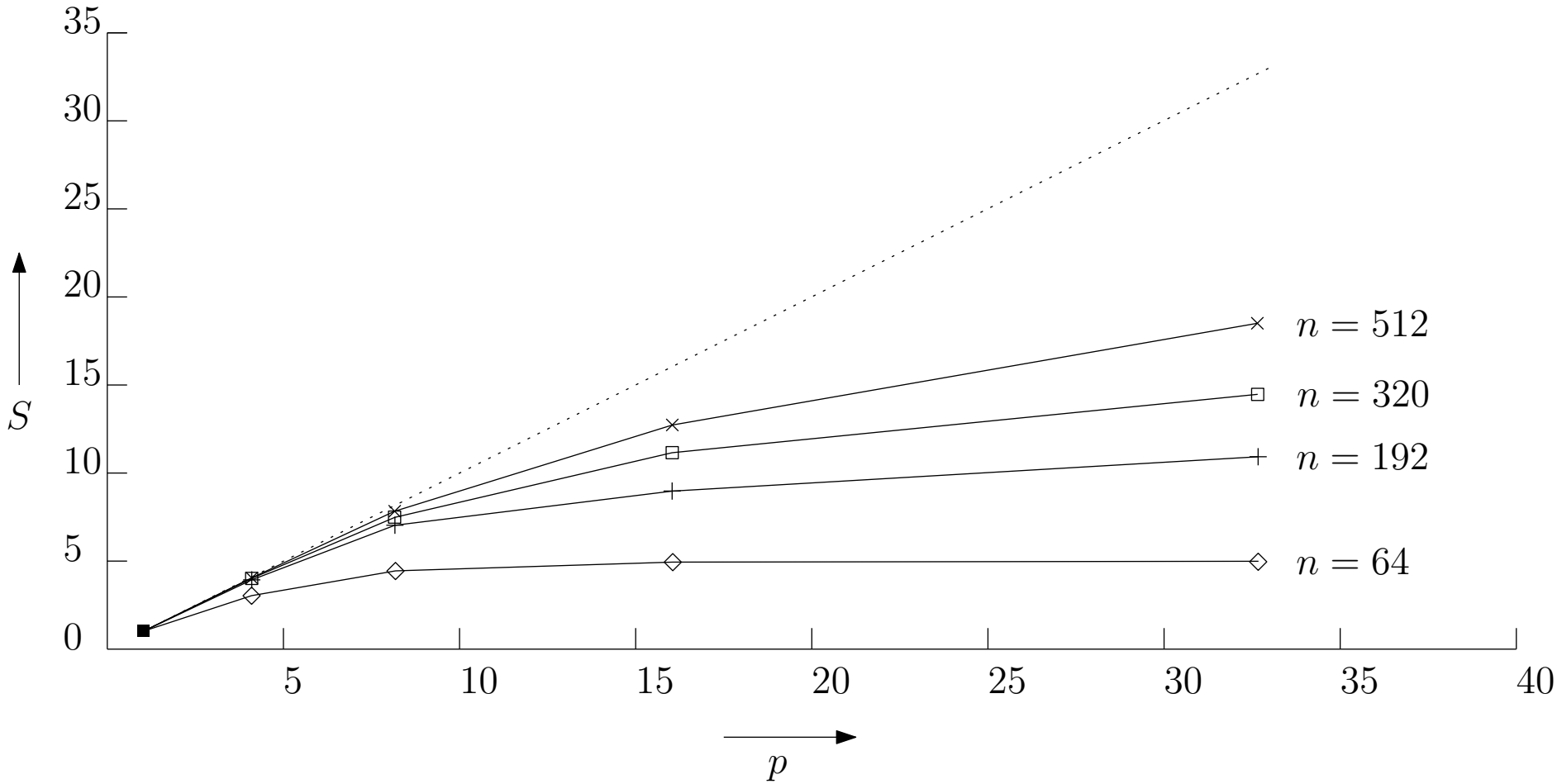
Αν $n = \Omega(p\log p)$ τότε $c_p = \Theta(n)$ !

$$T_p = \frac{n}{p} + 2t_{com}\log p, \quad S = \frac{np}{n + 2p\log p}$$

$$E_p = \frac{n}{n + 2p\log p} \qquad \begin{cases} c_p = \mathcal{O}(n + 2p\log p) \\ n = \Omega(p\log p) \end{cases}$$

βέλτιστο κόστος

# Αξιολόγηση MIMD Παράλληλων Συστημάτων




$$n = 8p \log p$$

# Αξιολόγηση MIMD Παράλληλων Συστημάτων

| $n$ | $p = 1$ | $p = 4$ | $p = 8$ | $p = 16$ | $p = 32$ |
|---|---|---|---|---|---|
| 64 | 1.0 | .80 | .57 | .33 | .17 |
| 192 | 1.0 | .92 | .80 | .60 | .38 |
| 320 | 1.0 | .95 | .87 | .71 | .50 |
| 512 | 1.0 | .97 | .91 | .80 | .62 |

Πίνακας: $E_p$

# Αξιολόγηση MIMD Παράλληλων Συστημάτων

- Η ταχύτητα δεν αυξάνει γραμμικά όταν αυξάνει το πλήθος των επεξεργαστών
- Για μεγαλύτερα προβλήματα η ταχύτητα και η αποδοτικότητα αυξάνουν

$N = 2^n,\ n \geq 1.\ x_1, x_2, \ldots, x_N \rightarrow M_1, M_2, \ldots, M_N$

Το τελικό αποτέλεσμα στην $M_1$.

# Αξιολόγηση MIMD Παράλληλων Συστημάτων

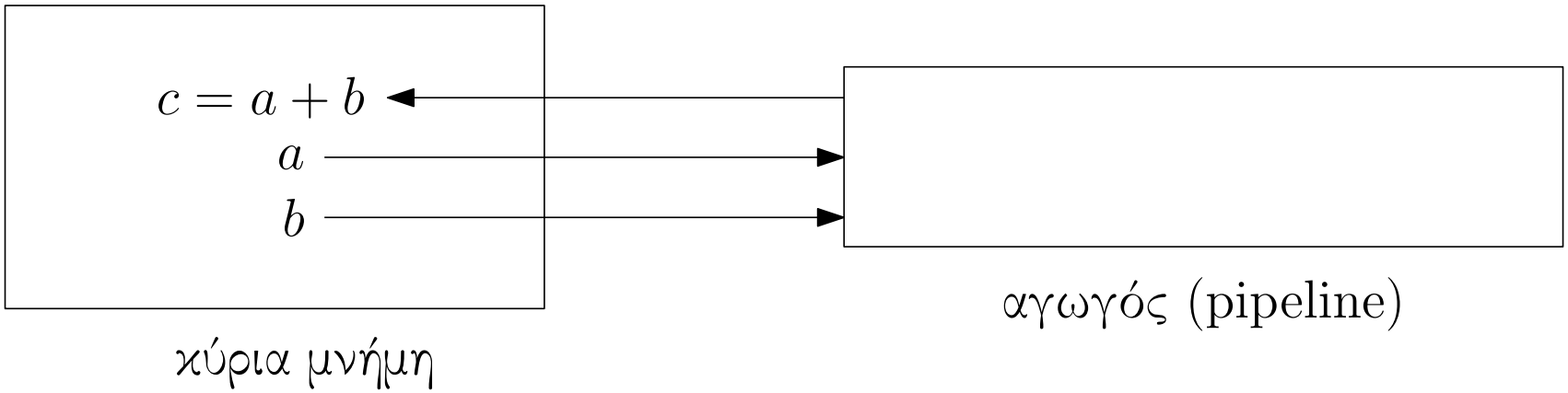
**Διαδικασία 6:** ASSOCIATIVE FAN-IN $(inc, N)$

inc $= 1$ **για** $j = 1$ **έως** $\log N$ **κάνε**
  **για** $i \in \{1 + 2k \cdot inc \mid k = 0, 1, 2, \ldots, \frac{N}{2^j} - 1\}$ **κάνε παράλληλα**
    // επεξεργαστής $p_i$
    **διάβασε** $M_i$ **και** $M_{i_{time}}$
    **πρόσθεσε** περιεχόμενα $M_i$ **και** $M_{inc+i}$
    **γράψε** το άθροισμα στο $M_i$
    inc $=$ 2inc
  **τέλος**
**τέλος**

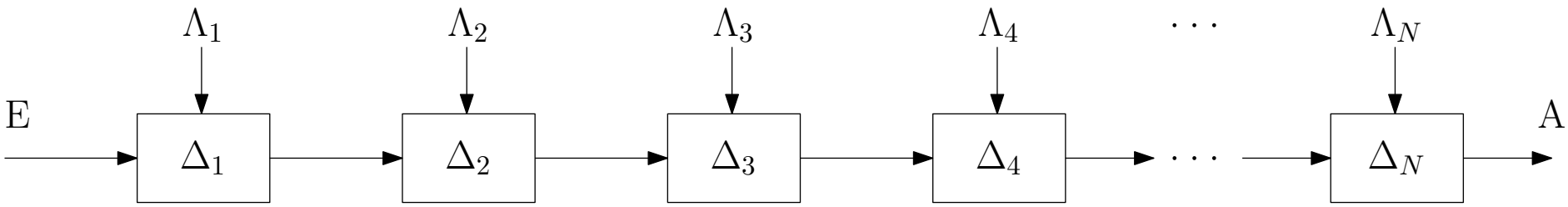
# Αξιολόγηση MIMD Παράλληλων Συστημάτων

| $\mathbf{j}$ | $\mathbf{M_1}$ | $\mathbf{M_2}$ | $\mathbf{M_3}$ | $\mathbf{M_4}$ | $\mathbf{M_5}$ | $\mathbf{M_6}$ | $\mathbf{M_7}$ | $\mathbf{M_8}$ |
|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| 1 | 3 | | 7 | | 11 | | 15 | |
| 2 | 10 | | | | 26 | | | |
| 3 | 36 | | | | | | | |

# Διανυσματικοί Υπολογιστές



# Διανυσματικοί Υπολογιστές




$E = \text{εντολή}_i \qquad \Delta_i = \text{δεδομένο}_i$

$\Lambda_i = i - \text{οστή λειτουργία}, \qquad \text{A} = \text{αποτέλεσμα}$

Σωλήνωση τριών τμημάτων

# Διανυσματικοί Υπολογιστές

| **Περίοδος χρόνου** | **1** | **2** | **3** | **4** | | **n** | **n + 1** |
|---|---|---|---|---|---|---|---|
| πάρε | $a_1$ | $a_2$ | $a_3$ | $a_4$ | ... | $a_n$ | - |
| πάρε και πρόσθεσε | - | $a_1 + b_1$ | $a_2 + b_2$ | $a_3 + b_3$ | | $a_{n-1} + b_{n-1}$ | |
| Εκτύπωσε | - | - | $a_1 + b_1$ | $a_2 + b_2$ | | $a_{n-2} + b_{n-2}$ | |