ΚΕΦΑΛΑΙΟ 3
Συστήματα κατανεμημένης μνήμης

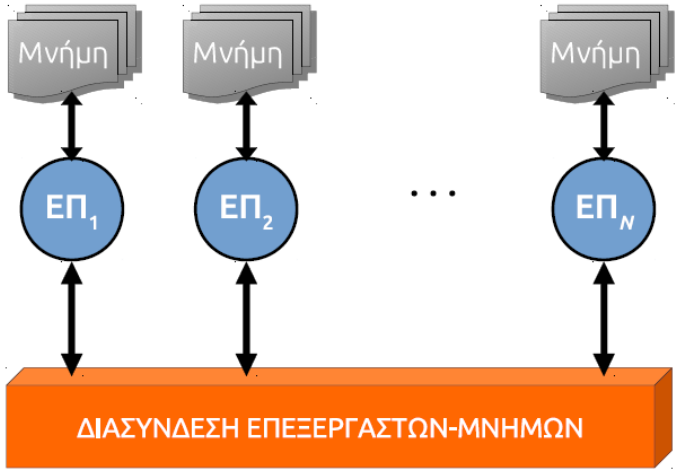
Distributed memory systems

ΜΥΕ023

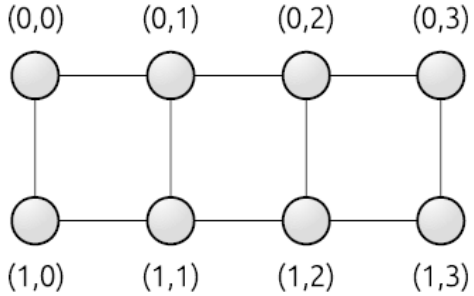
Παράλληλα Συστήματα & Προγραμματισμός

Πολυεπεξεργαστές κατανεμημένης μνήμης

- Ανεξάρτητοι επεξεργαστές, ο καθένας με την ιδιωτική του μνήμη (κόμβος = CPU + μνήμη)



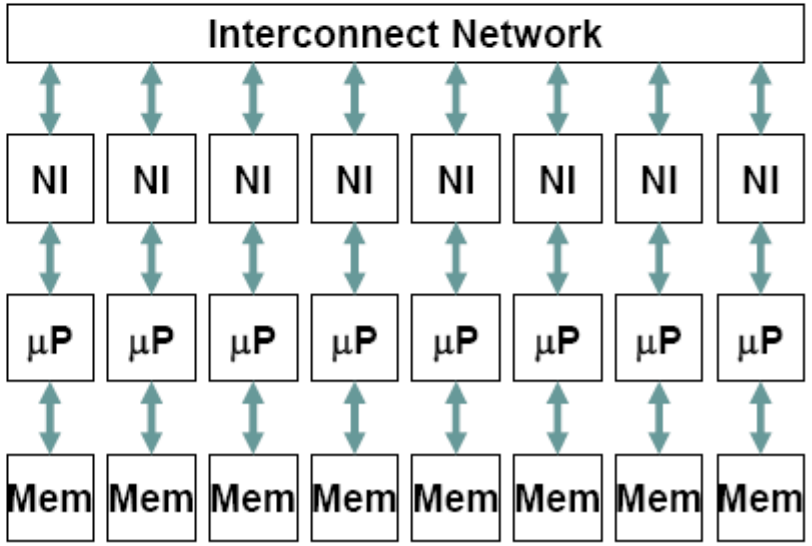
- Δίκτυο διασύνδεσης επεξεργαστών (interconnection network)
  - δίαυλος
  - δίκτυο διακοπτών
  - point-to-point, στατικό, άμεσο δίκτυο



Massively Parallel Processors (MPPs)

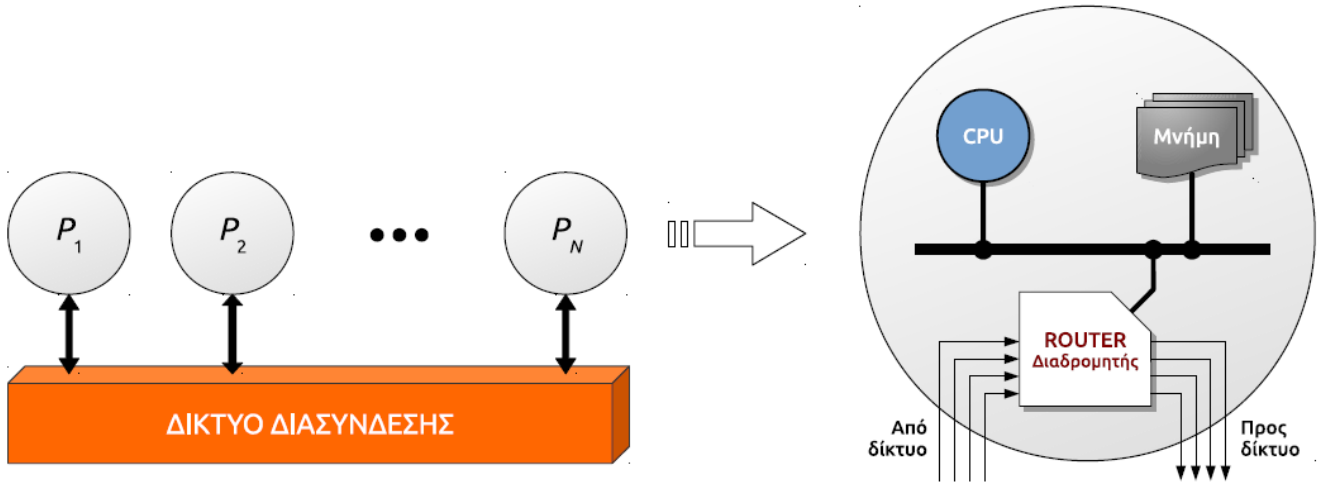
- Initial Research Projects
  - Caltech Cosmic Cube (early 1980s) using custom Mosaic processors
- Commercial Microprocessors including MPP Support
  - Transputer (1985)
  - nCube-1(1986) /nCube-2 (1990)
- Standard Microprocessors + Network Interfaces
  - Intel Paragon (i860)
  - TMC CM-5 (SPARC)
  - Meiko CS-2 (SPARC)
  - IBM SP-2 (RS/6000)
- MPP Vector Supers
  - Fujitsu VPP series

Designs scale to 100s or 1000s of nodes

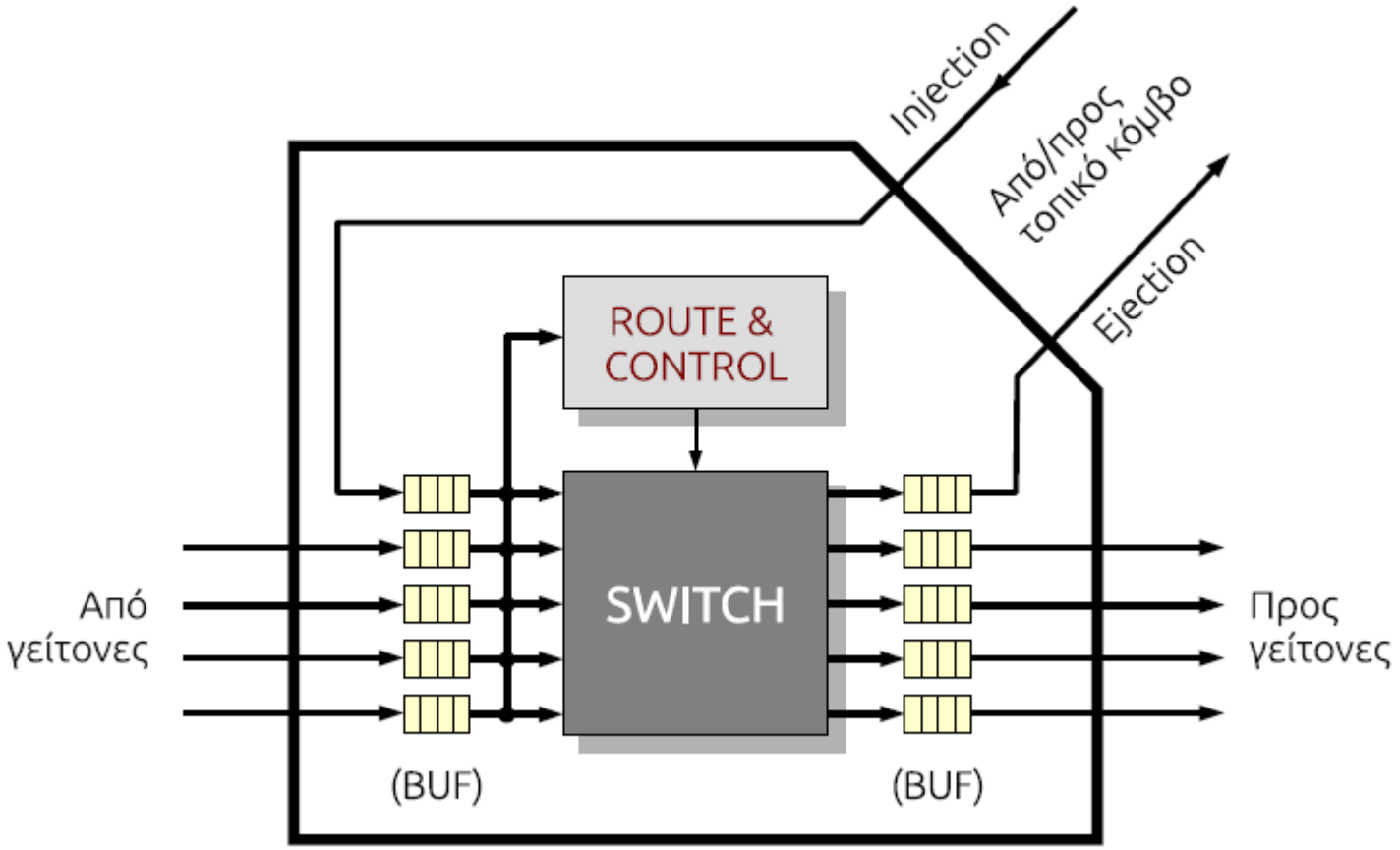


Βασική οργάνωση

- Επικοινωνία επεξεργαστών μέσω ανταλλαγής μηνυμάτων, επάνω από το δίκτυο διασύνδεσης
- Ο διαδρομητής (router) συνδέει τον κόμβο με το δίκτυο
  - Κανάλια από προς γείτονες / τοπική μνήμη



Τυπική δομή διαδρομητή



Πολυϋπολογιστές

- Λόγω του ότι κάθε κόμβος είναι ουσιαστικά ένας (σχεδόν) ολοκληρωμένος και αυτόνομος υπολογιστής, οι ΠΚΜ είναι γνωστοί και ως *πολυϋπολογιστές* (multicomputers)

- Η οργάνωση μοιάζει με δίκτυο υπολογιστων
  - Διαφορές:
    - ταχύτητα
    - τοπολογία
    - λειτουργικό σύστημα
    - ...

Ένα δίκτυο διασύνδεσης χαρακτηρίζεται από:

- Την *τοπολογία* του
  - Ποίος κόμβος συνδέεται με ποιον – χωρική διάταξη
- Τη *διαδρόμησή* του (routing)
  - Ποιο από όλα τα δυνατά μονοπάτια θα επιλεχθεί για την επικοινωνία
    - Πολλές επιλογές πολιτικών
- Τον *έλεγχο ροής* του (flow control)
  - Πώς διανέμονται οι πόροι του δικτύου (κανάλια, buffers κλπ), τι συμβαίνει σε περίπτωση συγκρούσεων
    - Αρχιτεκτονική του διαδρομητή
- Τη *μεταγωγή* του (switching)
  - Πώς μεταφέρεται εσωτερικά σε έναν διαδρομητή το μήνυμα από μία είσοδο σε μία έξοδό του
    - Κυκλώματος (circuit switching)
    - Πακέτου / μηνύματος / SAF (Store-and-Forward)
    - Virtual Cut-Through (VCT)
    - Wormhole, …

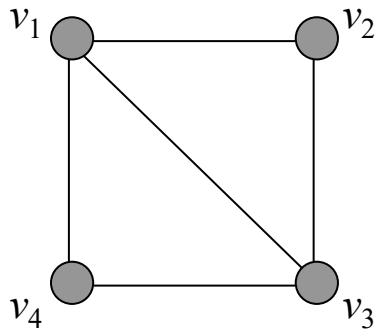
MYE023
MYE023
MYE023

Η τοπολογία του δικτύου διασύνδεσης

Τοπολογία

- Διάταξη των κόμβων στον χώρο και συνδεσμολογία μεταξύ τους
- Γράφοι ως φυσική αναπαράσταση του δικτύου
  - κόμβοι = κορυφές
  - συνδέσεις = ακμές
- Αναδρομή στην ορολογία και τις ιδιότητες των γράφων

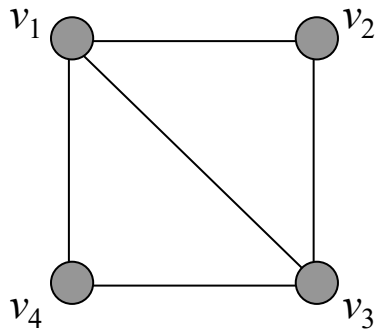
Ορολογία και ιδιότητες των γράφων



- Σύνολο κορυφών και σύνολο ακμών: G = (V,E)
  - μη κατευθυνόμενοι
- Αν e = vu ∈ E, τότε οι v, u είναι *γειτονικές (neighbors, adjacent)*
- Η ακμή είναι *προσκείμενη (incident)* στις κορυφές

- Αν η v έχει $\beta(v)$ γείτονες, τότε έχει *βαθμό* $\beta(v)$
- Αν όλες οι κορυφές έχουν τον ίδιο βαθμό β, τότε *β-regular (τακτικός)*
- Δ(G), δ(G): μέγιστος, ελάχιστος βαθμός

Ορολογία και ιδιότητες των γράφων

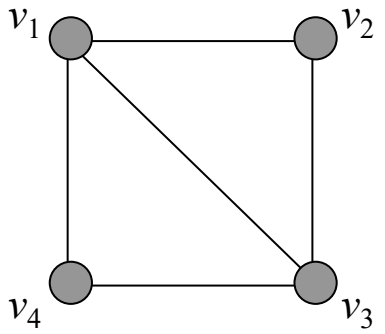


- *Περίπατος* (walk)
  - ακολουθία γειτονικών κορυφών (π.χ. από v1 στη v4: v1, v2, v3, v2, v1, v4)
- *ίχνος* (trail)
  - περίπατος χωρίς επαναλαμβανόμενες ακμές (π.χ. v1, v2, v3, v1, v4)
- *μονοπάτι* (path)
  - ίχνος χωρίς επαναλαμβανόμενες κορυφές (π.χ. v1, v2, v3, v4)

- Συνδεδεμένοι γράφοι
- *Κύκλος, Hamiltonicity*

Αποστάσεις

- Μήκος μονοπατιού
  - ο αριθμός ακμών που περιέχει



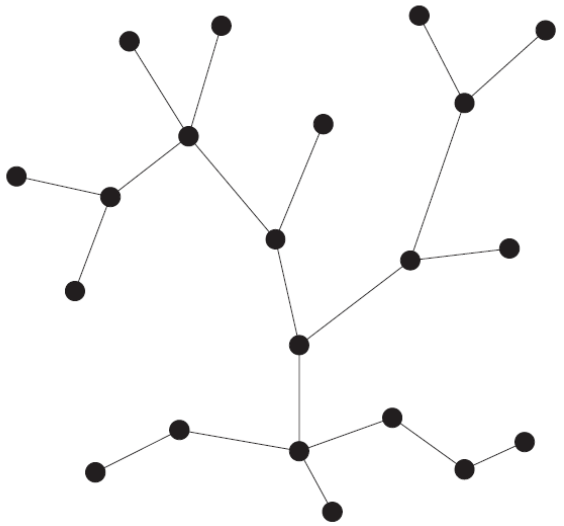
- *Απόσταση* dist(v,u)
  - το μικρότερο μήκος από όλα τα μονοπάτια v-u.

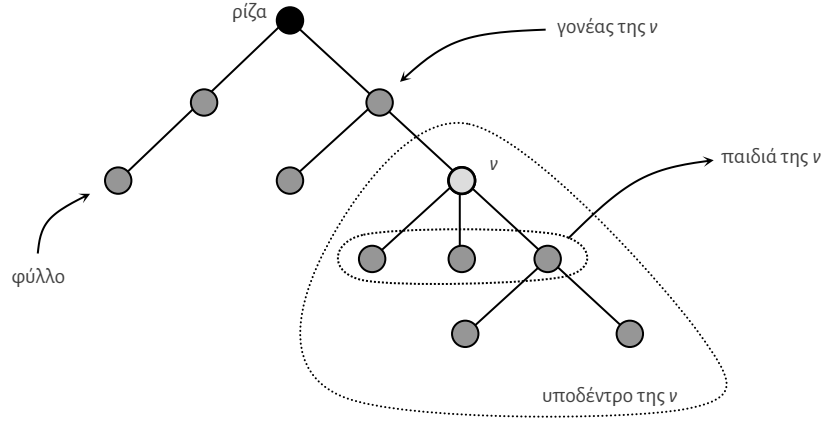
- Κορυφή u *εκκεντρική* ως προς την v: $dist(v,u) = \max_w\{v, w\}$
  - οπότε *εκκεντρικότητα* e(v) = dist(v,u)

- *Διάμετρος* = η μεγαλύτερη εκκεντρικότητα, D(G) (diameter)
- *Ακτίνα* = η μικρότερη εκκεντρικότητα, R(G) (radius)

Δέντρα

- Δέντρα (trees)
  - όχι κύκλοι
  - μοναδικά μονοπάτια
  - συνδεδεμένα, n κόμβοι, n–1 ακμές





Άλλα χαρακτηριστικά των γράφων

- *vertex-disjoint paths* (ξένα ως προς τις κορυφές)
- *edge-disjoint paths* (ξένα ως προς τις ακμές)
- *vertex connectivity*, κ(G) (συνδεσμικότητα κορυφών)
- *edge connectivity*, λ(G) (συνδεσμικότητα ακμών)

$$\kappa(G) \leq \lambda(G) \leq \delta(G)$$

- Και πολλά άλλα

Τι θέλουμε από έναν δίκτυο διασύνδεσης …

- Το δίκτυο διασύνδεσης θα πρέπει να μεταφέρει *όσο το δυνατόν περισσότερα* μηνύματα, *όσο το δυνατόν γρηγορότερα* με *ελάχιστο κόστος* και *μέγιστη αξιοπιστία*. Αυτά είναι αλληλοσυγκρουόμενα, όμως.

Τοπολογία:

- Μικρή διάμετρος, μικρή μέση απόσταση

  μικρή καθυστέρηση σε packet-switching, μικρή contention σε wormhole switching

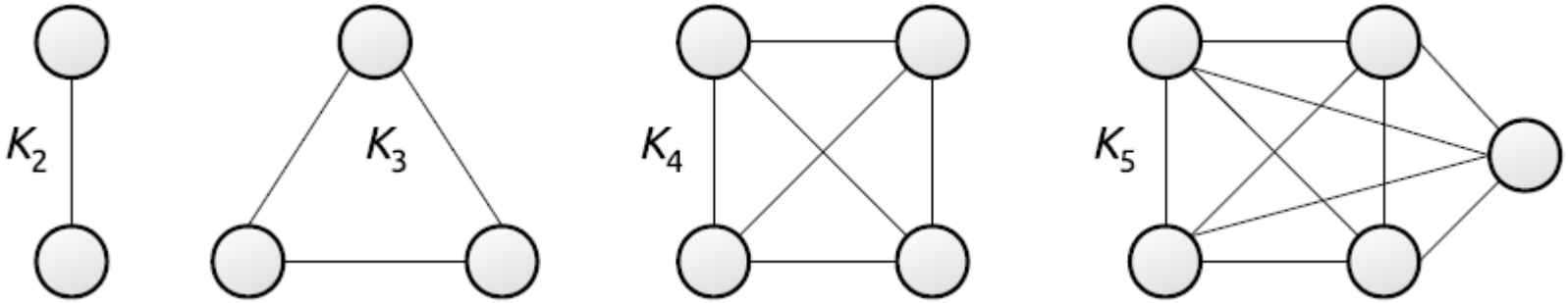
- Μικρός και σταθερός βαθμός

  απλοί και οικονομικοί routers, μικρότερη και σταθερή καλωδίωση, χαμηλότερη connectivity, μεγαλύτερες αποστάσεις

- Υψηλό connectivity
- Συμμετρία
- Εύκολη **ενσωμάτωση** άλλων γράφων και σε άλλους γράφους

  - Διότι αν ένα δίκτυο Α εμπεριέχεται σε ένα άλλο Β, τότε το δεύτερο θα έχει, εκτός των άλλων, και τις ιδιότητες του πρώτου
  - Διότι πολλές φορές έχουμε σχεδιάσει έναν αλγόριθμο για ένα δίκτυο Α (π.χ. υπάρχουν εξαιρετικοί αλγόριθμοι πολλαπλασιασμού πινάκων για tori) αλλά η παράλληλη μηχανή μας διαθέτει διασυνδετικό δίκτυο Β (π.χ. ο helios.cc.uoi.gr ήταν υπερκύβος).

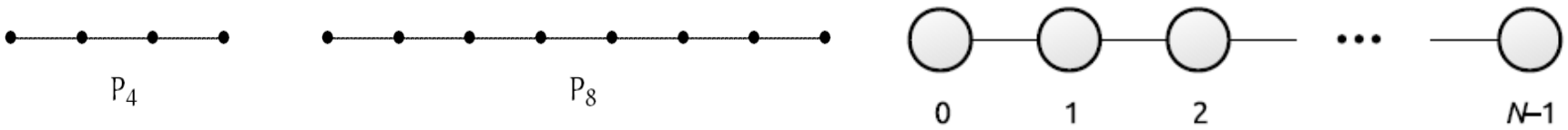
Βασικοί γράφοι: Πλήρης γράφος (complete graph)



- Όλες οι κορυφές συνδέονται με όλες
- Ο K_N έχει N κορυφές
- N(N-1)/2 ακμές
- (N-1)-regular
- $D(K_N) = 1$
- $\kappa(K_N) = N-1$
- Εμπεριέχει όλους τους γράφους με ≤ N κορυφές

- Πρακτικός μόνο για μικρό N

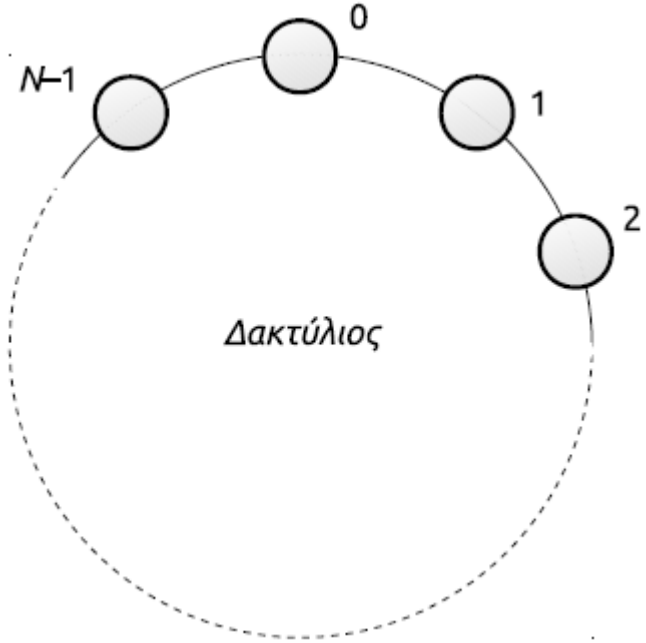
Βασικοί γράφοι: Γραμμικός γράφος (linear array)



- Απλό μονοπάτι
- Ο P_N έχει N κορυφές
- N-1 ακμές
- Μη τακτικός (βαθμοί 1 και 2)
- $D(P_N) = N-1$
- $\kappa(P_N) = 1$

- Μη πρακτικός – μόνο για εξειδικευμένες αρχιτεκτονικές (π.χ. συστολικές διατάξεις)

Βασικοί γράφοι: Δακτύλιος (ring)



- Απλός κύκλος
- Ο R_N έχει N κορυφές
- N ακμές
- 2-regular, συμμετρικός
- $D(R_N) = \text{floor}(N/2)$
- $\kappa(R_N) = 2$

- Μεγάλη διάμετρος, πολύ βασικός γράφος για κατασκευή άλλων τοπολογιών

Σχεδιασμός με βάση κριτήρια

- Συνήθως θέλουμε να βρούμε κάποιο δίκτυο που πληροί κάποια κριτήρια, π.χ. να έχει συγκεκριμένο βαθμό ή συγκεκριμένη διάμετρο με συγκεκριμένο # κόμβων / ακμών
- Θέλουμε ένα μεθοδικό τρόπο να παράγουμε τέτοια δίκτυα (π.χ. ξεκινώντας από πιο απλά δίκτυα)
- Μερικές βασικές τεχνικές
  - Γράφοι ακμών
  - Καρτεσιανό γινόμενο
  - *Η μέθοδος Cayley*

MYE023
MYE023
MYE023

Καρτεσιανό γινόμενο

Καρτεσιανό γινόμενο 2 γράφων

- Ορισμός:

$$G = (V, E) = G_1 \times G_2$$

  - Ο G_1 (G_2) ονομάζεται 1η (2η) ***διάσταση*** *(dimension)*

$$V = \{(v_1, v_2)\}$$

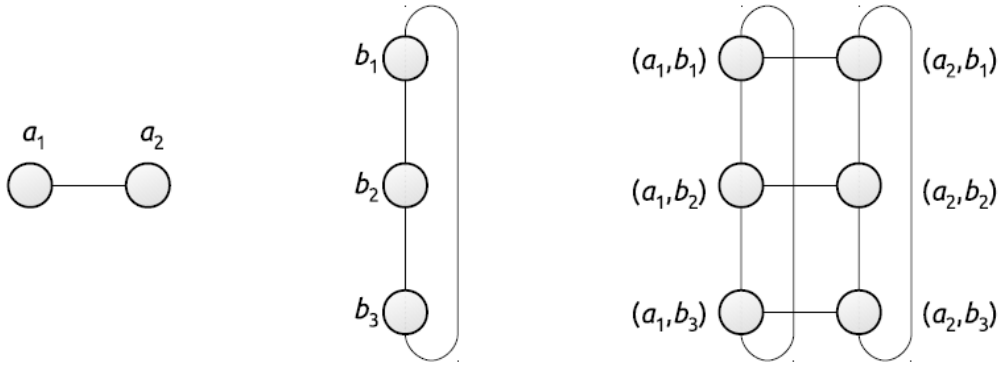
- Οι κορυφές είναι το καρτεσιανό γινόμενο των κορυφών των επιμέρους γράφων και άρα έχουν ως ετικέτα/διεύθυνση ένα ζεύγος.
  - Το πρώτο 1ο (2ο) στοιχείο του ζεύγους ονομάζεται 1η (2η) ***συντεταγμένη*** *(coordinate)*

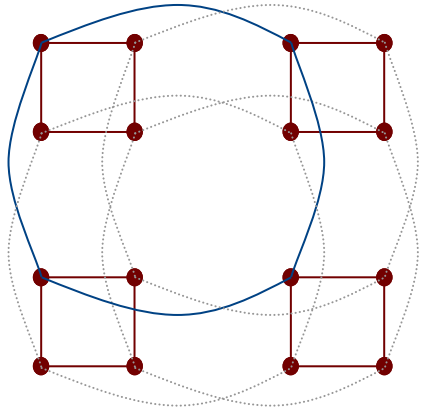
$$E = \{(v_1, v_2)(u_1, u_2) \mid v_1 = u_1 \text{ and } v_2 u_2 \in E_2 \text{ OR } v_2 = u_2 \text{ and } v_1 u_1 \in E_1\}$$

- Δύο κορυφές στον G γειτονεύουν μόνο αν έχουν στη μία διάσταση έχουν ίδια συντεταγμένη ενώ στην άλλη έχουν γειτονικές συντεταγμένες.

Παραδείγματα

$$E = \{(v_1, v_2)(u_1, u_2) \mid v_1 = u_1 \text{ and } v_2 u_2 \in E_2 \text{ OR } v_2 = u_2 \text{ and } v_1 u_1 \in E_1\}$$





Γενίκευση σε *k* διαστάσεις

$$G_i = (V_i, E_i), \qquad i = 1,2, \dots, k$$

$$G = (V, E) = G_1 \times G_2 \times \cdots \times G_k = (G_1 \times G_2 \times \cdots \times G_{k-1}) \times G_k = G' \times G_k$$

$$V = V_1 \times V_2 \times \cdots \times V_k = \{(v_1, v_2, \dots, v_k) | v_i \in V_i\}$$

$$E = \{(v_1, v_2, \dots, v_k)(u_1, u_2, \dots, u_k) | \exists j : v_j u_j \in E_j \text{ and } \forall i \neq j : v_i = u_i\}$$

- Δύο κορυφές είναι γειτονικές αν και μόνο αν έχουν τις αντίστοιχες συντεταγμένες τους ίσες, *εκτός από μία* και στη συγκεκριμένη διάσταση οι συντεταγμένες τους είναι γειτονικές ($v_j u_j \in E(G_j)$)

Μερικές ιδιότητες

$$G_1 \times G_2 = G_2 \times G_1$$

$$|V| = |V_1| \times |V_2|$$

$$v = (v_1, v_2)$$

$$\beta(v) = \beta(v_1) + \beta(v_2)$$

$$dist(v,u) = dist_1(v_1,u_1) + dist_2(v_2,u_2)$$

$$e(v) = e_1(v_1) + e_2(v_2)$$

$$D(G) = D(G_1) + D(G_2)$$

Για k διαστάσεις:

$$|V| = |V_1| \times |V_2| \times \cdots \times |V_k|$$

$$v = (v_1, v_2, \ldots, v_k)$$

$$\beta(v) = \sum_{i=1}^{k} \beta(v_i)$$

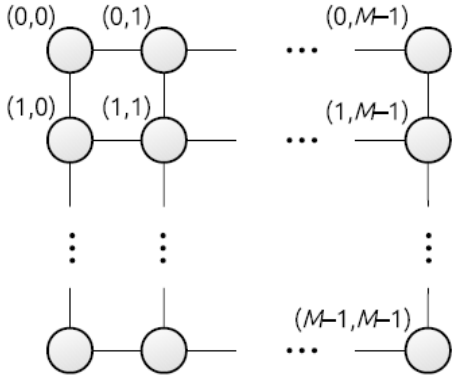
$$dist(v,u) = \sum_{i=1}^{k} dist_i(v_i,u_i)$$

$$e(v) = \sum_{i=1}^{k} e_i(v_i)$$

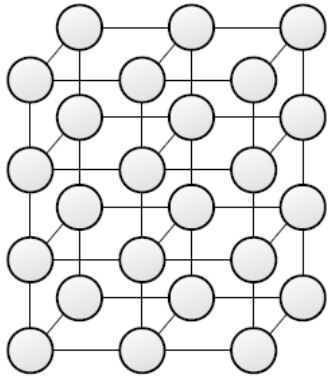
$$D(G) = \sum_{i=1}^{k} D(G_i)$$

Γνωστά δίκτυα ως καρτεσιανά γινόμενα: Πλέγματα

- Πλέγματα: γινόμενα γραμμικών γράφων



2D πλέγμα M x M

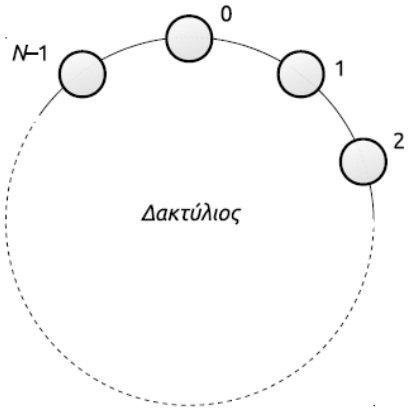


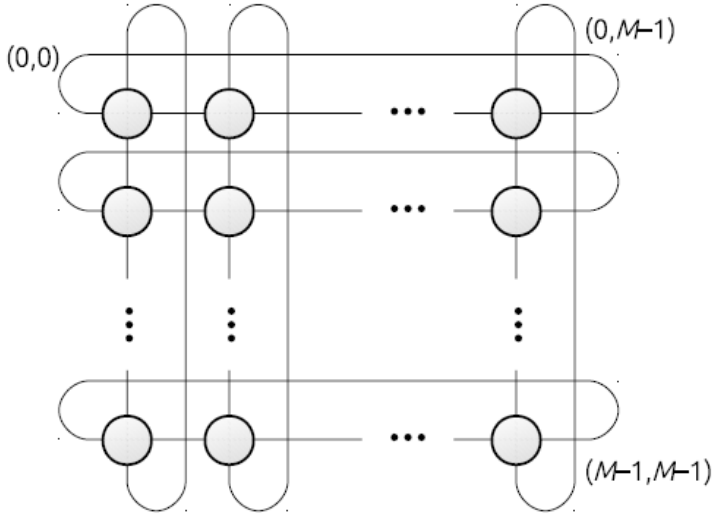
3D πλέγμα 4 x 3 x 2

- Σε τετραγωνικό πλέγμα 2D, $N = M \times M$:
  - Βαθμός: 2 (γωνίες), 3 (ακμές), 4 (όλοι οι άλλοι κόμβοι)
  - Διάμετρος: $2(M-1) \approx 2(N)^{1/2}$
- Σε τετραγωνικό πλέγμα 3D, $N = M \times M \times M$:
  - Βαθμός: 3 (γωνίες), 4 (ακμές), 5 (πλευρές), 6 (όλοι οι άλλοι κόμβοι)
  - Διάμετρος: $3(M-1) \approx 3(N)^{1/3}$

Γνωστά δίκτυα ως καρτεσιανά γινόμενα: Tori

- Tori: γινόμενα δακτυλίων



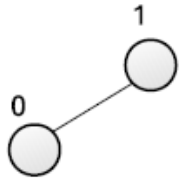


2D torus M x M

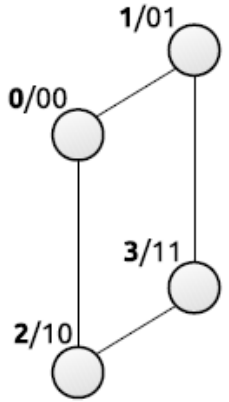
- Σε τετραγωνικό torus 2D, $N = M \times M$:
  - Τακτικός, συμμετρικός γράφος, βαθμός: 4
  - Διάμετρος: $2 \times \text{floor}(M/2) \approx N^{1/2}$

Γνωστά δίκτυα ως καρτεσιανά γινόμενα: Υπερκύβοι

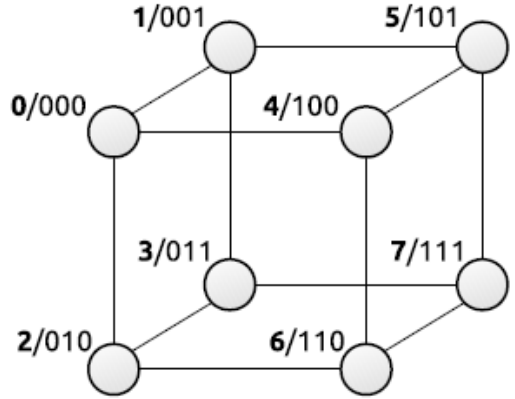
- Καρτεσιανό γινόμενο από γράφους 2 κόμβων
  - $K_2 = L_2 = R_2 = \text{whatever}_2$ 



μονοδιάστατος κύβος – Q_1



διδιάστατος κύβος – Q_2



τριδιάστατος κύβος – Q_3

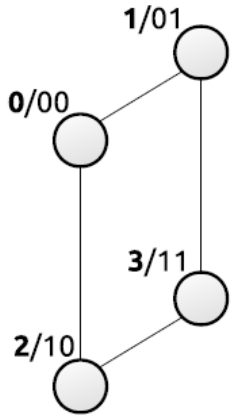
Υπερκύβος: κι άλλοι ορισμοί

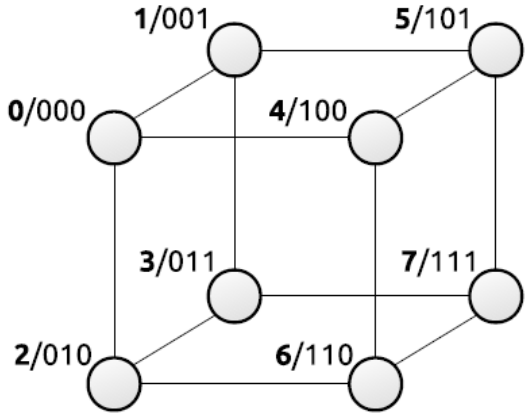
- Καρτεσιανό γινόμενο από P_2, R_2, K_2 ...

- Επίσης, είναι ιεραρχικά αναδρομικός (καρτεσιανό γινόμενο από μικρότερους κύβους)

$$Q_1 = K_2, \quad Q_d = Q_{d-1} \times Q_1 = Q_k \times Q_{d-k}$$

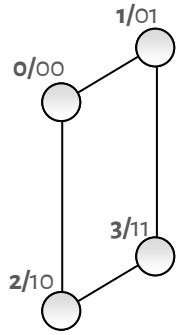
- Ισοδύναμος ορισμός, ενίοτε βολικότερος:
  - $N = 2^d$ κόμβοι με ετικέτες d-ψήφιους δυαδικούς αριθμούς.
  - *Δύο κόμβοι γειτνιάζουν αν και μόνο αν οι ετικέτες τους διαφέρουν σε 1 bit.*



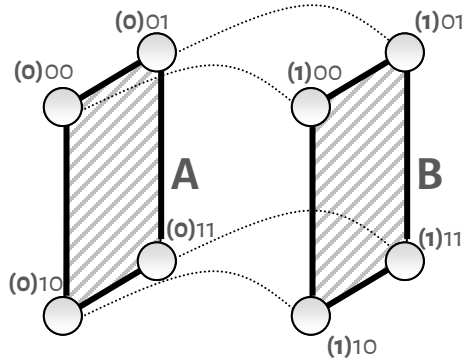


Κατασκευή με βάση τον τελευταίο ορισμό

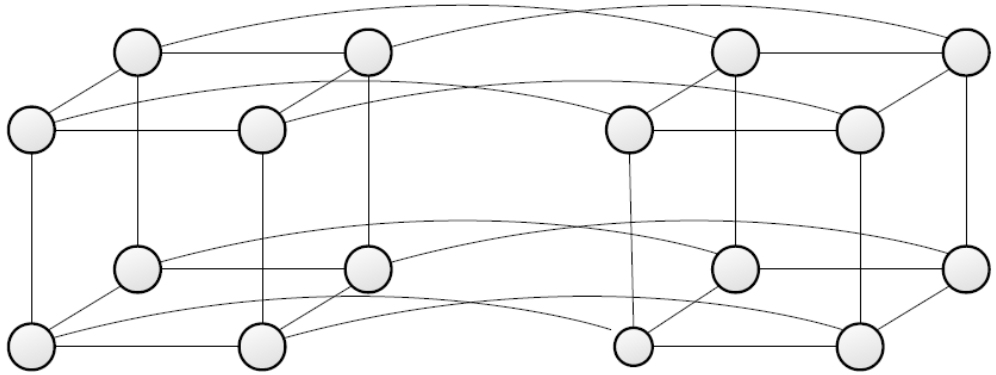
- Δύο ίδια αντίγραφα του αμέσως μικρότερου κύβου
- Συνδέω τις κορυφές με ίδια ετικέτα
- Στον πρώτο βάζω το 0 μπροστά από κάθε ετικέτα, στο δεύτερο βάζω το 1.



Διδιάστατος κύβος (Q_2)



Κατασκευή από δύο διδιάστατους κύβους



τετραδιάστατος κύβος – Q_4

Μεριικές ιδιότητες

- Τακτικός, βαθμού d ($= \log_2 N$)
- Διάμετρος $D(Q_d) = d$
- Hamiltonian, vertex symmetric, edge symmetric
- Βέλτιστη συνδεσμικότητα (d) (τόσα παράλληλα μονοπάτια)
- Γενικώς, βέλτιστες είναι πάρα πολλές από τις ιδιότητές του, βέλτιστα συμπεριφέρονται πάρα πολλοί αλγόριθμοι.
  - Γι' αυτό και ήταν από τα δημοφιλέστερα / πιο μελετημένα δίκτυα

- Σημαντικότερα μειονεκτήματα:
  - Καλή, αλλά όχι και τέλεια διάμετρος, αλλά κυρίως:
  - Μεγάλος, μη σταθερός βαθμός
  - Γι' αυτό και (σε συνδυασμό με άλλες εξελίξεις) υπερσκελίστηκε από άλλα δίκτυα χαμηλού και σταθερού βαθμού (π.χ. πλέγματα).

Συγκριτικός πίνακας

	Κόμβοι	Βαθμός	Διάμετρος	Παραδείγματα συστημάτων
Πλήρης γράφος	N	$N-1$	1	
Γραμμ. γράφος	N	$1, 2$	$N-1$	
Δακτύλιος	N	2	$\lfloor N/2 \rfloor$	Sequent Symmetry, Intel Xeon Phi
Πλέγμα 2D	$N = M \times M$	$2, 3, 4$	$2M - 2 \approx 2\sqrt{N}$	Intel Paragon Adapt. Epiphany
Πλέγμα 3D	$N = M \times M \times M$	$2, 3, \ldots, 6$	$3M - 3 \approx 3\sqrt[3]{N}$	MIT J Machine, Sandia Red Storm
Torus 2D	$N = M \times M$	4	$2\lfloor M/2 \rfloor \approx \sqrt{N}$	Cray X1E
Torus 3D	$N = M \times M \times M$	6	$3\lfloor M/2 \rfloor \approx \sqrt[3]{N}$	Cray XT3/4/5/6 Cray XE6
Υπερκύβος	$N = 2^d$	$d = \log_2 N$	$d = \log_2 N$	Intel IPSC-1/2, nCUBE-1/2/3, SGI Origin

MYE023
MYE023
MYE023

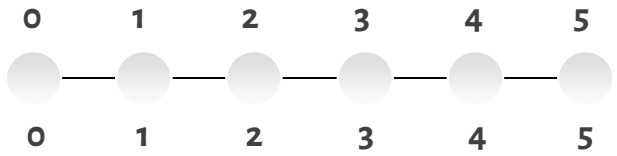
Αντιστοιχίσεις/ενσωματώσεις

(embeddings)

PARALLEL
PROCESSING
GROUP

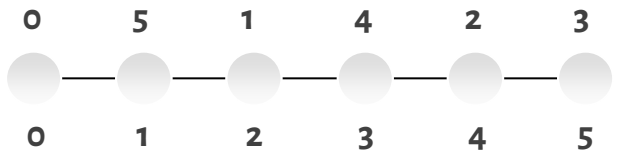
Ενσωμάτωση δακτυλίου σε ισομεγέθη γραμμικό γράφο

1ος τρόπος



Η ακμή 0-5 στον δακτύλιο γίνεται μονοπάτι μήκους 5!

2ος τρόπος



Όλες οι ακμές του δακτυλίου γίνονται το πολύ μονοπάτι μήκους 2

Με οικοδεσπότη των υπερκύβο

- Ο υπερκύβος μπορεί να προσομοιώσει ιδιαίτερα αποδοτικά σχεδόν όλα τα γνωστά δίκτυα (!)

- Βασική τεχνική για βασικούς φιλοξενούμενους:
  - Binary Reflected Gray Code (BRGC, GC για συντομία)

- GC_1 = {0,1} $GC_d = \{\mathbf{0}GC_{d-1}, \mathbf{1}GC_{d-1}^R\}$

- GC_2 = { **0**{0,1}, **1**{1,0} } = { 00, 01, 11, 10 }
  GC_3 = { **0**{00,01,11,10}, **1**{10,11,01,00} } = ...

- *Κάθε στοιχείο της ακολουθίας διαφέρει από το προηγούμενο και το επόμενό του σε ακριβώς 1 bit.*

Μερικός κώδικας Gray

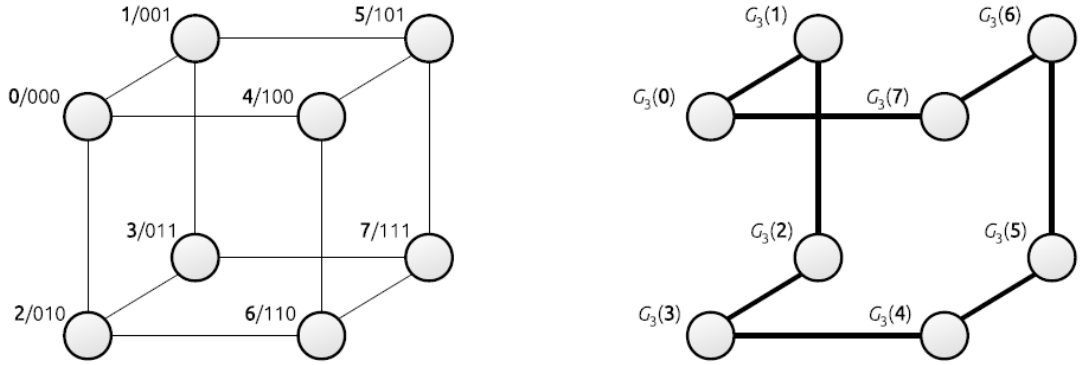
- Με λιγότερα από 2^d στοιχεία.
- Έστω ότι $GC_{d,\lambda}$ είναι τα πρώτα λ στοιχεία του GC_d
- Τότε, μερικός κώδικας Gray με k στοιχεία, είναι ο:

$$\mathrm{PGC}_{d,k} = \left\{ \mathbf{0}\mathrm{GC}_{d-1,k/2}, \mathbf{1}\mathrm{GC}^{R}_{d-1,k/2} \right\}$$

- Π.χ. έστω ότι θέλουμε τον $PGC_{3,6}$.
  - GC_2 = {00, 01 , 11, 10}, $GC_{2,3}$ = {00, 01 , 11}
  - $PGC_{3,6}$ = { **0** {00, 01 , 11}, **1** {11, 01, 00} }
    = { 000, 001, 011, 111, 101, 100 }

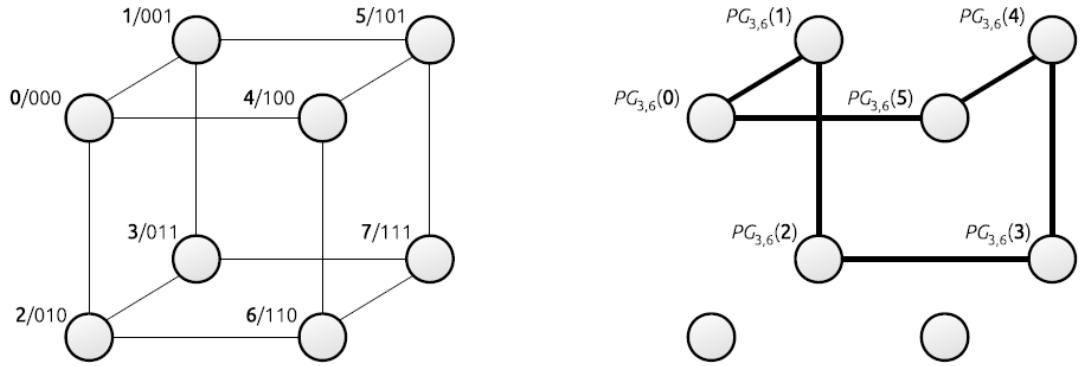
Path/Ring embedding

- Αν δακτύλιος με 2^d κόμβους, αντιστοιχίζουμε τον κόμβο i του δακτυλίου στον κόμβο GC(i) του κύβου



- Αν έχει k κόμβους (k άρτιο), τότε αντιστοιχίζουμε τον κόμβο i του δακτυλίου στον κόμβο PGC(i)

Κόμβος στον R_6	0	1	2	3	4	5
Κόμβος στον Q_3	000	001	011	111	101	100



Path/Ring embedding

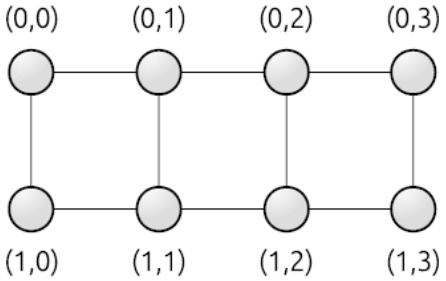
- Γιατί δουλεύει;
  - Διότι γειτονικοί κόμβοι στο δακτύλιο αντιστοιχίζονται σε γειτονικά στοιχεία του GC.
  - Όμως, γειτονικά στοιχεία του GC διαφέρουν σε ακριβώς 1 bit.
  - Επομένως, οι αντίστοιχοι κόμβοι στον υπερκύβο είναι επίσης γειτονικοί.

Mesh / torus embedding

- Είδαμε ότι $Q_d = Q_{d_1} \times Q_{d_2} \times \cdots \times Q_{d_k} \quad (d = d_1 + d_2 + \cdots + d_k)$
- Με χρήση πολλαπλών κωδίκων Gray
- Π.χ. στο κύβο Q_d, $d = d_1 + d_2$, αντιστοιχίζω torus $2^{d_2} \times 2^{d_1}$
- Χρησιμοποιώ 2 κώδικες, GC_{d_1} και GC_{d_2}
- Ο κόμβος (i, j) αντιστοιχίζεται στον $GC_{d_2}(i)||GC_{d_1}(j)$

$$x = (x_a x_{a-1} \cdots x_1)$$
$$y = (y_b y_{b-1} \cdots y_1)$$
$$x||y = (x_a x_{a-1} \cdots x_1 y_b y_{b-1} \cdots y_1)$$

Παράδειγμα

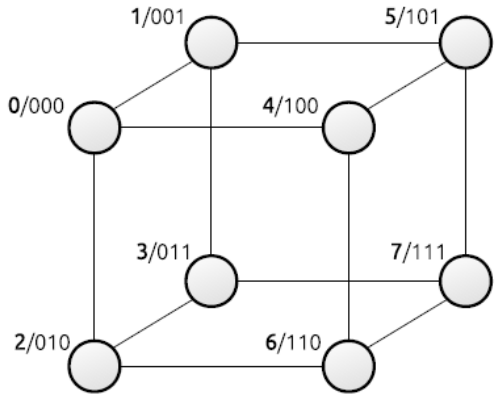


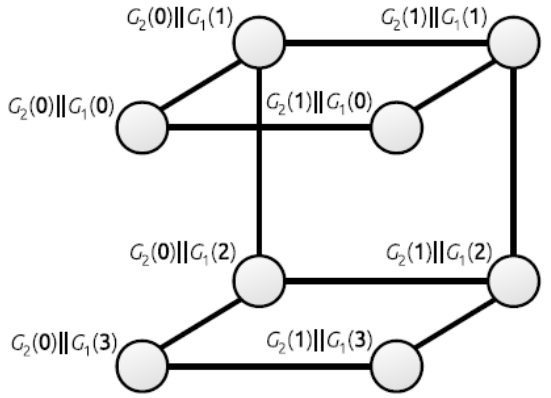
$2\times4 = 2^1 \times 2^2$, άρα 2 κώδικες: με 1 και 2 ψηφία

$G_1 = \{00, 01, 11, 10\}$

$G_2 = \{0, 1\}$.

$(0, 0) \rightarrow$ **000**	$(0, 1) \rightarrow$ **001**	$(0, 2) \rightarrow$ **011**	$(0, 3) \rightarrow$ **010**
$(1, 0) \rightarrow$ **100**	$(1, 1) \rightarrow$ **101**	$(1, 2) \rightarrow$ **111**	$(1, 3) \rightarrow$ **110**





MYEO23
MYEO23
MYEO23

Κατηγορίες διαδρόμησης

(routing)

Με βάση τις αποφάσεις διαδρόμησης (I)

- Πώς και πού παίρνονται οι αποφάσεις
  - *Κατανεμημένη* (distributed routing)
    - Σε κάθε router καθώς τον διασχίζει το μήνυμα
    - Η επικεφαλίδα έχει μόνο την ταυτότητα του παραλήπτη / προορισμού
    - Κάθε router απλά αποφασίζει σε ποιο κανάλι εξόδου θα προωθήσει το μήνυμα
    - Ιδιαίτερα βολικό σε συμμετρικά δίκτυα (όλοι οι routers χρησιμοποιούν τον ίδιο αλγόριθμο)
  - *Πηγής* (source routing)
    - Η πηγή αποφασίζει όλο το μονοπάτι και το ενσωματώνει στην επικεφαλίδα του μηνύματος (συνήθως αναφέρεται ποιο κανάλι πρέπει να χρησιμοποιήσει κάθε ενδιάμεσος κόμβος)
    - Κάθε ενδιάμεσος κόμβος μηχανικά το προωθεί στο κανάλι εξόδου που του λέει το μήνυμα και «αφαιρεί» τα bit που αναφέρονται σε αυτόν
    - Αν οι routers έχουν k κανάλια εξόδου και το μονοπάτι είναι μήκους D, θα χρειαστούν $D\log_2 k$ bits για την κωδικοποίηση του μονοπατιού
    - Ακόμα λιγότερα σε δίκτυα καρτεσιανού γινομένου, όπου μπορούμε να σημειώσουμε *μόνο την αλλαγή διάστασης* (μαζί με τη νέα κατεύθυνση) (street-sign routing)

Με βάση τις αποφάσεις διαδρόμησης (II)

- *Υβριδική / πολλαπλών φάσεων* (hybrid/multiphase routing)
  - Η πηγή προκαθορίζει μόνο μερικούς ενδιάμεσους κόμβους και όχι όλο το μονοπάτι
  - Από ενδιάμεσο σε ενδιάμεσο κόμβο η διαδρόμηση γίνεται κατανεμημένα
  - Π.χ. το randomized routing του Valiant (1 τυχαίος ενδιάμεσος κόμβος).
- *Κεντρικοποιημένη* (centralized routing)
  - Κάποιος εξωτερικός ελεγκτής παίρνει τις αποφάσεις
  - Έχει χρησιμοποιηθεί σε μηχανήματα SIMD και σε μερικά MINs

Με βάση την υλοποίηση του αλγορίθμου

- Δύο υλοποιήσεις:
  - *Αλγοριθμική*
    - Είτε σε hardware, είτε σε software υπάρχει κάποιος στοιχειώδης αλγόριθμος για τον σχηματισμό της διαδρομή (π.χ. e-cube, dimension-ordered routing).
    - Είναι αναγκαστικά «δεμένη» με συγκεκριμένη τοπολογία. Αν ο router είναι γενικός, για να φτιαχτεί οποιαδήποτε τοπολογία, η αλγοριθμική διαδρόμηση είναι δύσκολη αν όχι αδύνατη.
    - Στην περίπτωση της διαδρόμησης πηγής, μόνο η πηγή χρησιμοποιεί τον αλγόριθμο, φυσικά.
  - *Με πίνακα*
    - Κάθε κόμβος έχει πίνακα με τα κανάλια που πρέπει να χρησιμοποιήσει για κάθε δυνατό προορισμό
      - Μεγάλο μέγεθος πίνακα = Θ(# κόμβων)
      - Και ανάλογα αργή η αναζήτηση σε αυτόν
    - Interval routing: για μείωση του πίνακα, κρατάμε ένα *διάστημα διευθύνσεων* ανά κανάλι εξόδου (δηλ. 2 αριθμούς)
      - Ανοιχτό πρόβλημα για τις περισσότερες τοπολογίες
    - Στην περίπτωση της διαδρόμησης πηγής, μόνο η πηγή χρειάζεται πίνακα, φυσικά.

Με βάση την προσαρμοστικότητα του αλγορίθμου

- Δηλαδή, αν χρησιμοποιείται κι άλλη πληροφορία εκτός της διεύθυνσης του προορισμού
  - *Σταθερή* (απροσάρμοστη ☺) διαδρόμηση (deterministic)
    - Διαλέγει πάντα το ίδιο μονοπάτι μεταξύ δύο κόμβων (συνήθως κάποιο ελάχιστο).
    - Απλή, γρήγορη, η πιο συνηθισμένη στην πράξη
    - Καλή για ομοιόμορφη κίνηση και για αξιόπιστα δίκτυα, αλλιώς όχι
  - «*Τυφλή*» (oblivious)
    - Χρησιμοποιεί για τις αποφάσεις της μόνο τη δ/νση προορισμού
    - Περιλαμβάνει τη σταθερή διαδρόμηση
    - Όμως μπορεί να ΜΗΝ είναι σταθερή, αλλά τυχαία.
    - Πολλά εναλλακτικά μονοπάτια και αποφασίζει ποιο από όλα κάθε φορά
    - Καλή εναλλακτική τεχνική αντί της προσαρμοζόμενης, για ομοιόμορφη κατανομή της κίνησης στο δίκτυο

Με βάση την προσαρμοστικότητα του αλγορίθμου (II)

- *Προσαρμοζόμενη* (adaptive)
  - Χρησιμοποιεί και πληροφορία από την κίνηση του δικτύου και την κατάσταση των καναλιών για να αποφασίσει τον επόμενο κόμβο προκειμένου να αποφύγει είτε την μεγάλη κίνηση είτε ελαττωματικούς κόμβους
  - Είναι βασικά κατανεμημένη (δεν έχει πολύ νόημα για την περίπτωση της διαδρόμησης πηγής)
  - Αποτελείται από δύο λειτουργίες:
    - Routing function: επιστρέφει *σύνολο* πιθανών καναλιών εξόδου
    - Selection function: επιλέγει ένα από όλα τα κανάλια
  - Θεωρητικά έχει τις προϋποθέσεις για την καλύτερη δυνατή διαδρόμηση
  - Θέλει προσοχή να αποφεύγονται deadlocks και άλλες συναφείς, καταστροφικές καταστάσεις
  - Προφανώς απαιτεί πιο πολύπλοκο hardware

Με βάση το μήκος των μονοπατιών

- Πλησιάζουμε τον προορισμό;
  - *Ελάχιστη ή άπληστη (minimal / greedy / profitable)*
    - Με κάθε βήμα πλησιάζουμε περισσότερο στον προορισμό.
    - Συνήθως με σταθερές και τυφλές διαδρομήσεις
  - *Μη ελάχιστη (non-minimal, non-greedy, misrouting)*
    - Σε κάποιο βήμα μπορεί να μην ακολουθήσουμε τη συντομότερη πορεία
    - Σχεδόν πάντα σε προσαρμοζόμενες διαδρομήσεις
    - Προκειμένου π.χ. να αποφύγουμε τη συντομότερη διαδρομή λόγω μεγάλης κίνησης
    - Θέλει προσοχή να αποφεύγονται deadlocks και άλλες συναφείς (livelock), καταστροφικές καταστάσεις

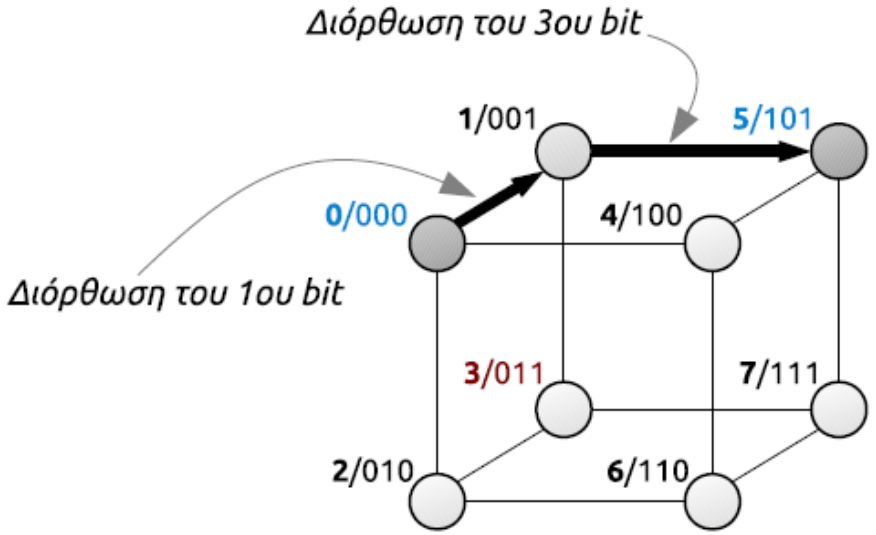
Με βάση την πρόοδο της διαδρομής (μόνο adaptive)

- Το μονοπάτι μεγαλώνει;
  - *Προοδευτική (progressive)*
    - Το μονοπάτι μεγαλώνει σε κάθε βήμα (όχι απαραίτητα πλησιάζοντας στον προορισμό)
  - *Backtracking*
    - Το μονοπάτι μπορεί και να μικρύνει κάποια στιγμή
    - Π.χ. αν συναντήσουμε κίνηση γυρνάμε πίσω και ακολουθούμε άλλη πορεία (άρα δεν μπορεί να είναι σταθερή)
    - Δύσκολη (έως αδύνατη) σε wormhole, εύκολη σε circuit switching

Παράδειγμα: διαδρόμηση σε υπερκύβο

- Σε υπερκύβο Q_d, απόσταση μεταξύ δύο κόμβων v και u
  - dist(v,u) = **# bits που διαφέρουν τα v και u**
- *Αλγόριθμος διαδρόμησης:*
  - Γνωρίζουμε ότι οι γειτονικοί κόμβοι διαφέρουν σε ακριβώς 1 bit
  - Σε κάθε βήμα, ο τρέχον κόμβος επιλέγει έναν γείτονά του ώστε να διαφέρει σε ένα bit λιγότερο από αυτά του προορισμού
  - Έτσι με κάθε βήμα «διορθώνεται» και 1 bit, μέχρι να καταλήξουμε τελικά στον προορισμό

- Ελάχιστη;
- Κατανεμημένη;
- Σταθερή;



MYE023
MYE023
MYE023

Μεταγωγή (switching)

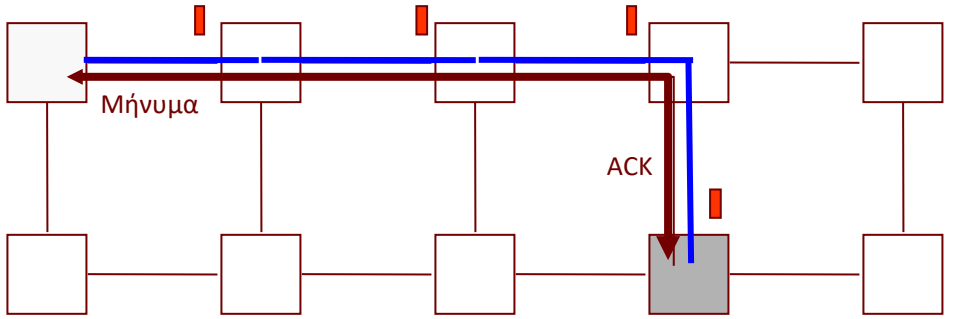
Μεταγωγή

- Ενώ ο έλεγχος ροής φυσικού μέσου μεταφέρει bits μεταξύ δύο διαδρομητών, η *μεταγωγή (switching) ενώνει εσωτερικά σε έναν διαδρομητή το κανάλι εισόδου με το επιλεγμένο κανάλι εξόδου και μεταφέρει δεδομένα.*

- Μερικές τεχνικές μεταγωγής:
  - Κυκλώματος (circuit switching)
  - Πακέτου / μηνύματος / SAF (Store-and-Forward)
  - Virtual Cut-Through (VCT)
  - Wormhole
  - Virtual channels
  - Pipelined circuit switching

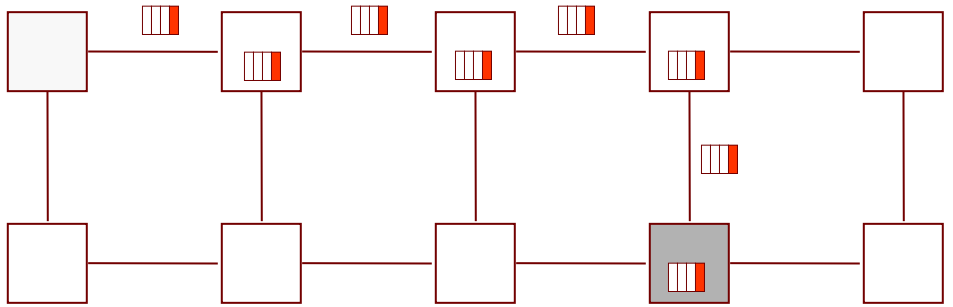
Μεταγωγή κυκλώματος

- Τρεις φάσεις:
  - σχηματισμός (και δέσμευση) του μονοπατιού από το probe
  - μεταφορά του μηνύματος
  - αποδέσμευση του μονοπατιού



Μεταγωγή SAF

- Πακέτου / μηνύματος / SAF (Store-and-Forward)
  - Το μήνυμα χωρίζεται σε πακέτα σταθερού μήκους
  - Κάθε πακέτο προωθείται ανεξάρτητα.
  - Οι κόμβοι
    - (α) το λαμβάνουν και το αποθηκεύουν σε buffer και
    - (β) το προωθούν στον επόμενο κόμβο



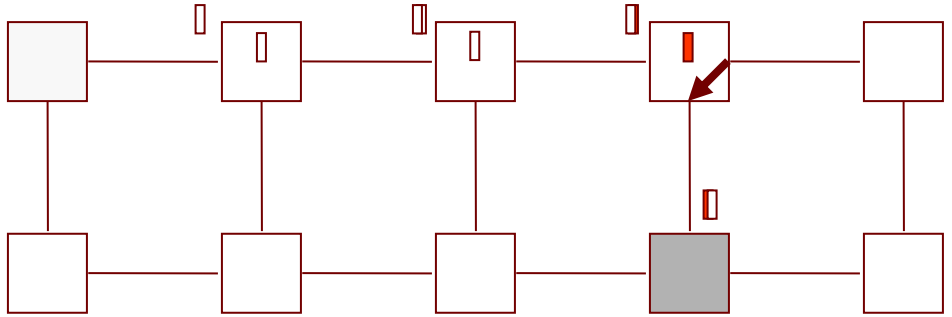
Μεταγωγή VCT

- Σαν το SAF αλλά:
  - Αν το κανάλι εξόδου είναι ελεύθερο, καθώς λαμβάνονται τα bits της επικεφαλίδας, αποφασίζεται το κανάλι εξόδου και όλο το μήνυμα διοχετεύεται κατευθείαν εκεί (άρα ελάχιστη καθυστέρηση).
  - Αν όχι, buffering όπως στο SAF.

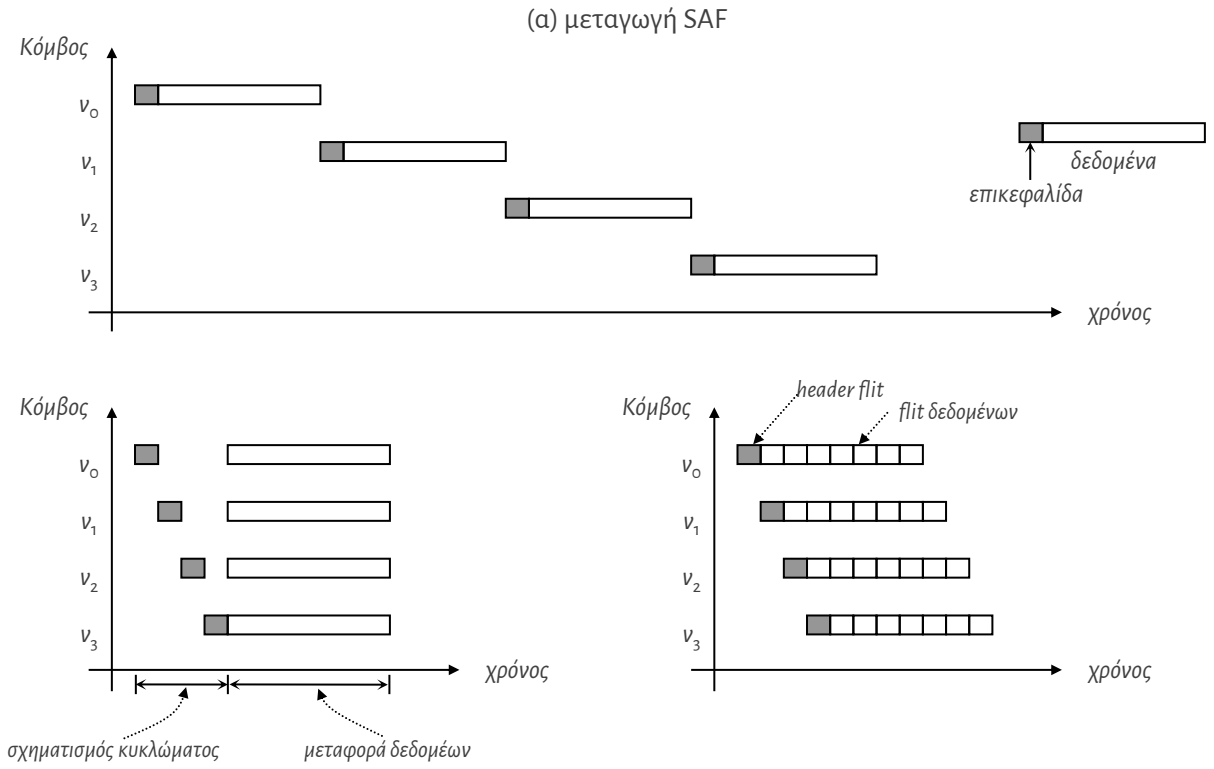
  - Ταχύτητα αν δεν υπάρξει εμπόδιο
  - Όμως, δεν εξαλείφεται η ανάγκη για buffers που έχει και το SAF.

Μεταγωγή wormhole

- Ανάμεσα σε VCT και circuit switching. Το μήνυμα χωρίζεται σε ΠΟΛΥ μικρά πακέτα, τα *flits* (1-4 bytes). Το πρώτο αποτελεί την επικεφαλίδα
- Η επικεφαλίδα προχωρά με VCT αλλά τα υπόλοιπα flits ακολουθούν (και δεσμεύουν) τους προηγούμενους κόμβους, χωρίς κενά, σαν σε pipeline.
- Αν η επικεφαλίδα μπλοκάρει κάπου, τα flits αποθηκεύονται *εκεί που βρίσκονται* (άρα πολύ μικροί buffers απαιτούνται), εμποδίζοντας με τη σειρά τους άλλα flits να περάσουν.



Χρονισμός

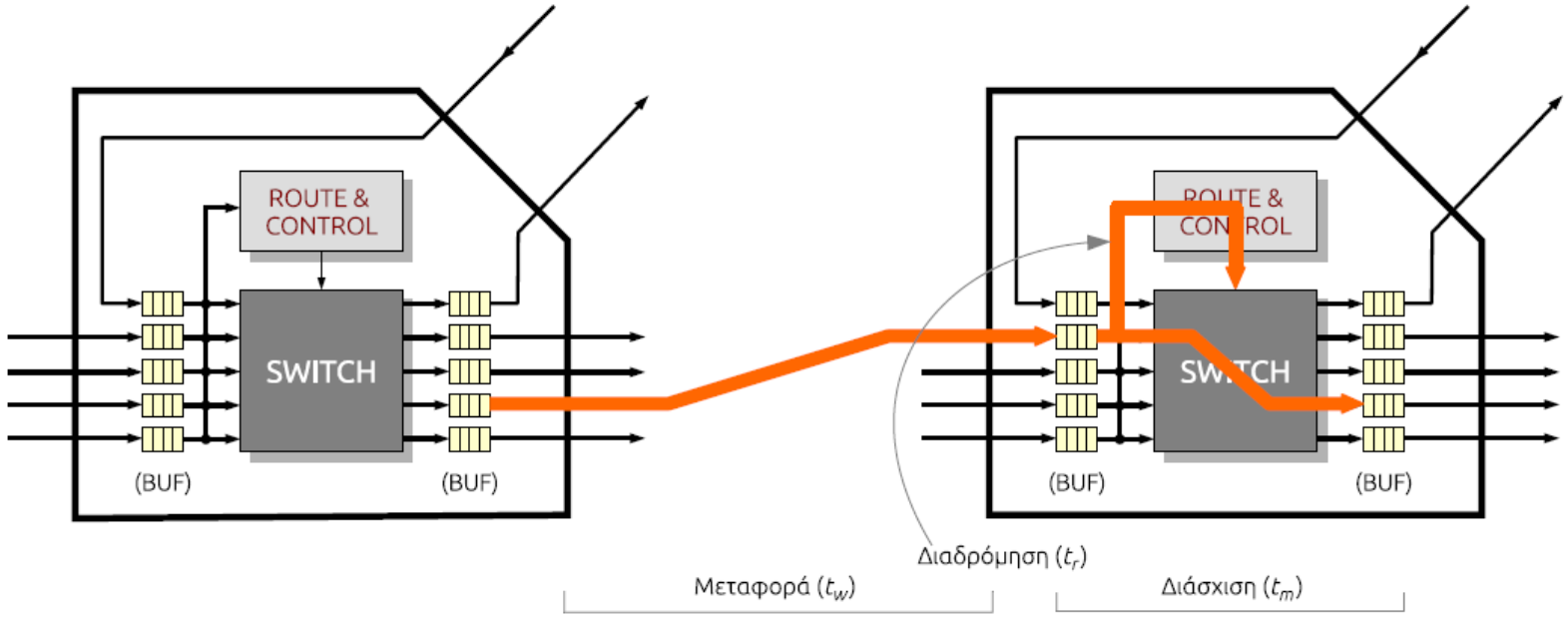

(α) μεταγωγή SAF
Κόμβος
v_0
v_1
v_2
v_3
χρόνος
δεδομένα
επικεφαλίδα
Κόμβος
v_0
v_1
v_2
v_3
χρόνος
σχηματισμός κυκλώματος
μεταφορά δεδομένων
(β) μεταγωγή κυκλώματος
Κόμβος
header flit
flit δεδομένων
v_0
v_1
v_2
v_3
χρόνος
(γ) μεταγωγή wormhole

Ας υποθέσουμε ότι ...

- Κάθε πακέτο αποτελείται από **1 flit** επικεφαλίδας και **M flits** δεδομένων
  - Σύνολο: **M+1 flits** για το πλήρες πακέτο
- Το μήνυμα πρέπει να διανύσει μονοπάτι μήκους **D** για να φτάσει στον προορισμό του
- *Δεν συναντάει κανένα εμπόδιο (δηλαδή αναμονή λόγω κατειλημμένων καναλιών) στον δρόμο του*

- Κανάλια με συχνότητα **B** Hz
- Το κανάλι έχει *πλάτος* 1 *phit* (= #bits που μεταφέρει ταυτόχρονα σε 1 κύκλο)
  - Υποθέτουμε 1 phit = 1 flit
  - Δηλαδή χωρητικότητα / ρυθμός μεταφοράς: **B flits / sec**.
  - Χρόνος ***μεταφοράς***, για να διασχίσει ένα flit το κανάλι: $\mathbf{t_w = 1/B}$ **sec**.

Χρόνοι



- Για να αποφασιστεί το κανάλι εξόδου (Route & control) σε ενδιάμεσο κόμβο, απαιτείται χρόνος $\mathbf{t_r}$
  - Χρόνος *διαδρόμησης*
- Για να μεταφερθεί από την είσοδο στην έξοδο σε ένα ενδιάμεσος κόμβος χρειάζεται χρόνος $\mathbf{t_m}$
  - Περιλαμβάνει όλες τις καθυστερήσεις (buffering, πέρασμα από switch κλπ) για να περάσει από κανάλι εισόδου σε κανάλι εξόδου
  - Χρόνος *διάσχισης*

Ο χρονισμός πιο αναλυτικά

- $T_{circuit\ switching} = \underbrace{D(t_w+t_r+t_m)}_{\text{Ο χρόνος για να σχηματιστεί το μονοπάτι από το probe}} + \underbrace{(M+1)t_w}_{\text{Χρόνος για το ack και για να μεταδοθεί το μήνυμα πάνω στο κύκλωμα}}$

- $T_{SAF} = \underbrace{D((M+1)t_w+(M+1)t_m+t_r)}_{\text{Μεταφορά ολόκληρου του μηνύματος από κόμβο σε κόμβο. Μόνο το 1ο flit χρησιμοποιείται για διαδρόμηση.}} = D(t_w+t_r+t_m) + DM(t_w+t_m)$

- $T_{VCT} = T_{WR} = \underbrace{D(t_w+t_r+t_m)}_{\text{Ο χρόνος για να φτάσει το header flit}} + Mt_w$

Τα άλλα flit ακολουθούν από πίσω και καταφθάνουν το ένα μετά το άλλο.

Αν ο χρόνος να διασχισθεί ένας διαδρομητής (t_m) είναι μεγαλύτερος από τον χρόνο μετάδοσης στο κανάλι (t_w), τότε ο όρος πρέπει να είναι Mt_m.

Αν ο διαδρομητής ήταν μόνο input-buffered?

Σύγκριση

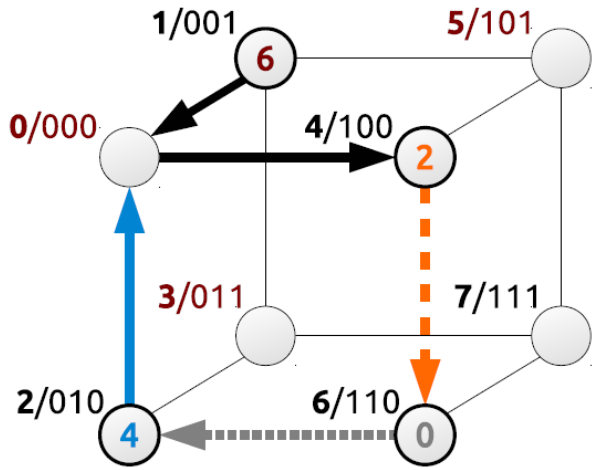
- Υπεραπλουστεύοντας, ας υποθέσουμε ότι $\mathbf{t_w \approx t_r \approx t_m = 1}$ χρονική μονάδα. Οι εκφράσεις μας απλοποιούνται ως εξής:
  - $T_{\text{circuit switching}} = T_{VCT} = T_{WR} = \Theta(D+M)$
  - $T_{SAF} = \Theta(DM)$

- Αν τα μηνύματα δεν είναι πάρα πολύ μικρά, $(M = O(D))$, τότε
  - $T_{\text{circuit switching}} = T_{VCT} = T_{WR} = \Theta(M)$
  - Επομένως οι μεταγωγές κυκλώματος, VCT και wormhole εξαρτώνται σχεδόν αποκλειστικά από το M και άρα είναι ***ανεξάρτητες της απόστασης (distance insensitive)***.

- Όλα αυτά, βέβαια, με την προϋπόθεση ότι δεν υπάρχουν συγκρούσεις / αναμονές στο μονοπάτι του μηνύματος.

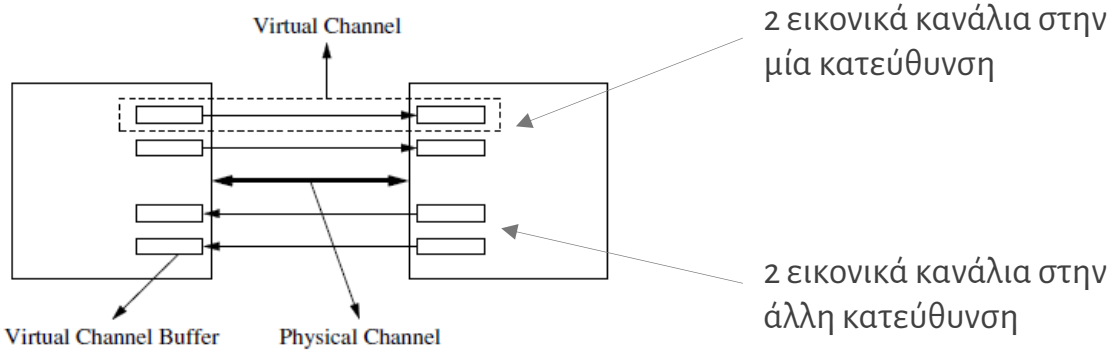
Wormhole switching

- Το wormhole switching έχει επικρατήσει διότι
  - Ταχύτητα ακόμα και σε δίκτυα μεγάλης διαμέτρου
  - Ελάχιστο buffering
  - Οπότε γίνεται δυνατή η *υλοποίηση routers σε ανεξάρτητο chip και όχι μέσω της μνήμης του κόμβου*
    - *High-speed (low latency) routers and networks*
- Σε υψηλή κίνηση είναι ιδιαίτερα επιρρεπές σε *deadlocks* αφού δεσμεύει πολλούς πόρους (κανάλια) στην πορεία του.



Τεχνική: virtual channels (εικονικά κανάλια)

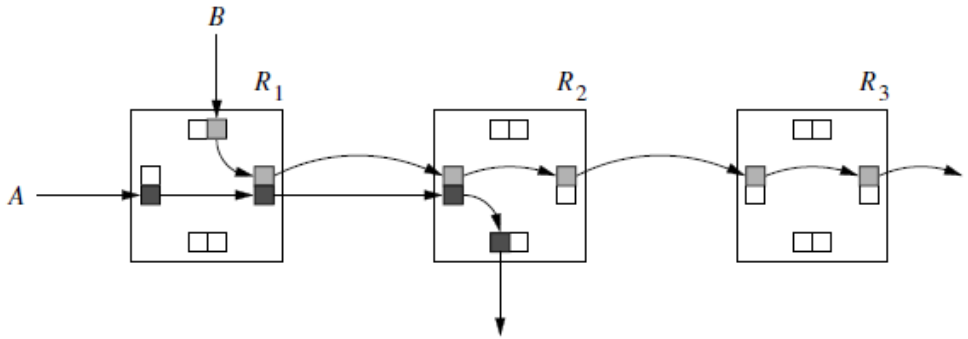
- Συνήθως οι buffers στα κανάλια είναι ουρές FIFO
  - Επομένως, αν το header flit μπλοκάρει, όλα τα προηγούμενα κανάλια *δεσμεύονται (σαν το circuit switching)*
  - *Κανένα άλλο μήνυμα δεν μπορεί να προχωρήσει*
- Τεχνική για βελτίωση της κατάστασης: virtual channels
  - Κάθε φυσικό κανάλι «υποδιαιρείται» σε πολλά εικονικά («λογικά») κανάλια.
  - Τα εικονικά κανάλια *πολυπλέκονται στο φυσικό κανάλι χρονικά*
  - Κάθε εικονικό κανάλι ορίζεται ουσιαστικά από ζεύγος buffers σε δύο γειτονικούς routers



Virtual channels

- Αν πολυπλέκονται χρονικά κ εικονικά κανάλια πάνω σε 1 φυσικό κανάλι B bits/sec, είναι σαν να έχω κ διαφορετικά φυσικά κανάλια, το καθένα (B/κ) bits/sec, δηλαδή πιο πολλά αλλά πιο αργά κανάλια.
- Αρχικά χρησιμοποιήθηκαν για το πρόβλημα του deadlock
- Όμως, μπορούν να βελτιώσουν και τις επιδόσεις μιας και πλέον το φυσικό κανάλι δεν δεσμεύεται εξ ολοκλήρου από κάποιο μπλοκαρισμένο μήνυμα
- Μπορούν έτσι να προχωρούν μαζί παραπάνω από ένα μηνύματα στο κανάλι

Virtual channels



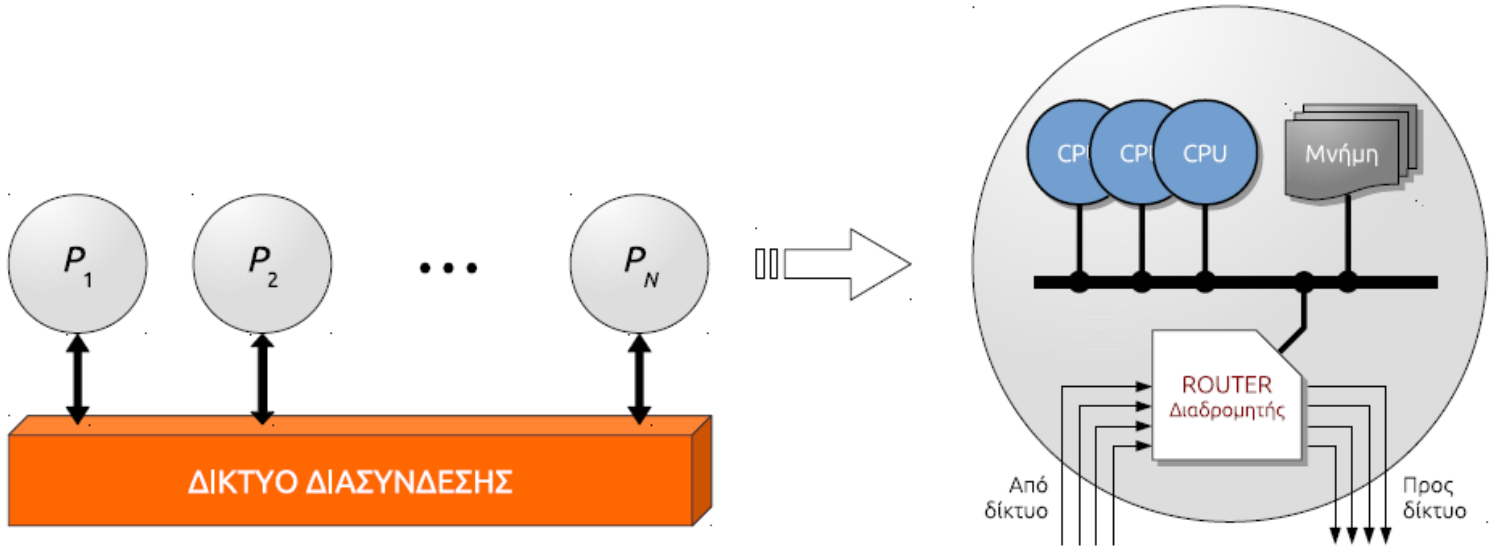
- Εδώ, αν το Α είχε μπλοκάρει στον R2, θα περίμενε αναγκαστικά και το Β (άρα 2 μηνύματα σε αναμονή, ενώ τώρα κανένα)
- Επίσης, αν το Α ήταν τεράστιο, το Β θα περίμενε για πολύ ώρα ενώ τώρα όχι.
- Όμως σίγουρα χάνουμε σε ταχύτητα και επίσης αυξάνει και η πολυπλοκότητα του διαδρομητή
- Επομένως καλό είναι να μην είναι πολλά τα virtual channels

MYE023
MYE023
MYE023

Κλιμακώσιμα συστήματα, υπολογιστικές συστάδες (clusters)

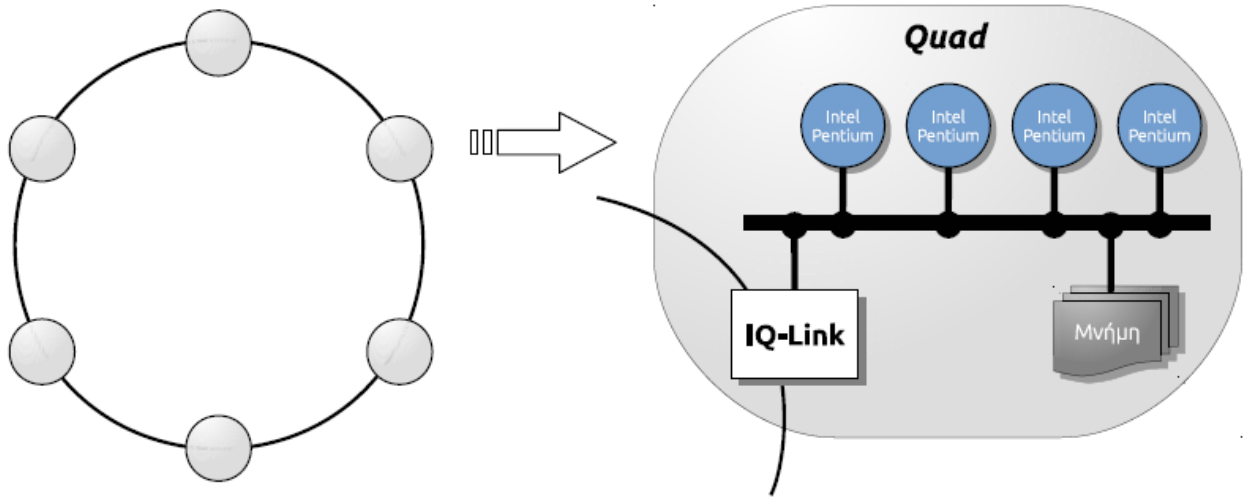
Ομαδοποιημένοι ή κλιμακώσιμοι πολυεπεξεργαστές

- Clustered ή **scalable** multiprocessors (κλιμακώσιμοι)
- Πολυεπεξεργαστές με οργάνωση κατανεμημένης μνήμης μόνο που:
  - *Κάθε κόμβος αποτελείται από ομάδα επεξεργαστών / πυρήνων που μοιράζονται την τοπική μνήμη.*
  - Άρα, κάθε ομάδα είναι ένας μικρός συμμετρικός πολυεπεξεργαστής
- Το δίκτυο διασύνδεσης συνδέει τις ομάδες



Παραδείγματα

- **SGI Origin 2000** (και μετέπειτα UV-1000, UV-2000):
  - Κόμβος =1 πλακέτα με 2 CPUs MIPS και κοινόχρηστη τοπική μνήμη
  - Σύστημα = υπερκύβος που συνδέει πλακέτες-κόμβους
- **Sequent Numa-Q**:
  - Κόμβος = Pentium Quad (4 επεξεργαστές Pentium) με κοινόχρηστη τοπική μνήμη
  - Σύστημα = δακτύλιος μεταξύ των κόμβων



Υπολογιστικές συστάδες (clusters) και πλέγματα (grids)

- Σύνολο από αυτόνομους υπολογιστές (PCs) συνδεδεμένα με ένα δίκτυο μεταξύ τους:
  - Οι *συστάδες (clusters)* είναι συνήθως ομοιογενείς (παρόμοιοι υπολογιστές, με ίδιο λειτουργικό σύστημα) και είναι τοποθετημένοι στον ίδιο χώρο
  - Τα *πλέγματα (grids)* αναφέρονται συνήθως σε ανομοιογενείς συλλογές από υπολογιστές, με διαφορετικά χαρακτηριστικά και λειτουργικά συστήματα, οι οποίο επίσης μπορεί να είναι εξαπλωμένοι γεωγραφικά σε μεγάλες αποστάσεις.
  - Μιλάμε για clusters αλλά ότι πούμε αντίστοιχα ισχύει και για πλέγματα

- Κάθε κόμβος-PC σε ένα cluster είναι ένας μικρός SMP
  - Π.χ. διαθέτει είτε πολλαπλούς επεξεργαστές είτε – πλέον το συνηθέστερο – έναν πολυπύρηνο επεξεργαστή
  - Επομένως το όλο σύστημα έχει οργάνωση είναι ομαδοποιημένου πολυεπεξεργαστή.

Υπολογιστικές συστάδες (clusters) και πλέγματα (grids)

- Δίκτυο διασύνδεσης
  - Κάρτα δικτύου = διαδρομητής
  - Από σχετικά αργό αλλά πολύ οικονομικό (π.χ. Gbit Ethernet)
  - Έως πολύ γρήγορο και ακριβό (π.χ. Myrinet, Infiniband)
- Κάθε κόμβος = αυτόνομος υπολογιστής, με δικό του χώρο μνήμης και δικό του λειτουργικό σύστημα (συνήθως Linux)
  - Υπάρχει όμως ενδιάμεσο λογισμικό ώστε να δίνεται η εντύπωση – ψευδαίσθηση ενιαίου συστήματος (SSI – Single System Image), αν χρειάζεται
- Προγραμματίζονται σαν να είναι 1 μηχανή
  - Κυρίως με μεταβίβαση μηνυμάτων

MYE023
MYE023
MYE023

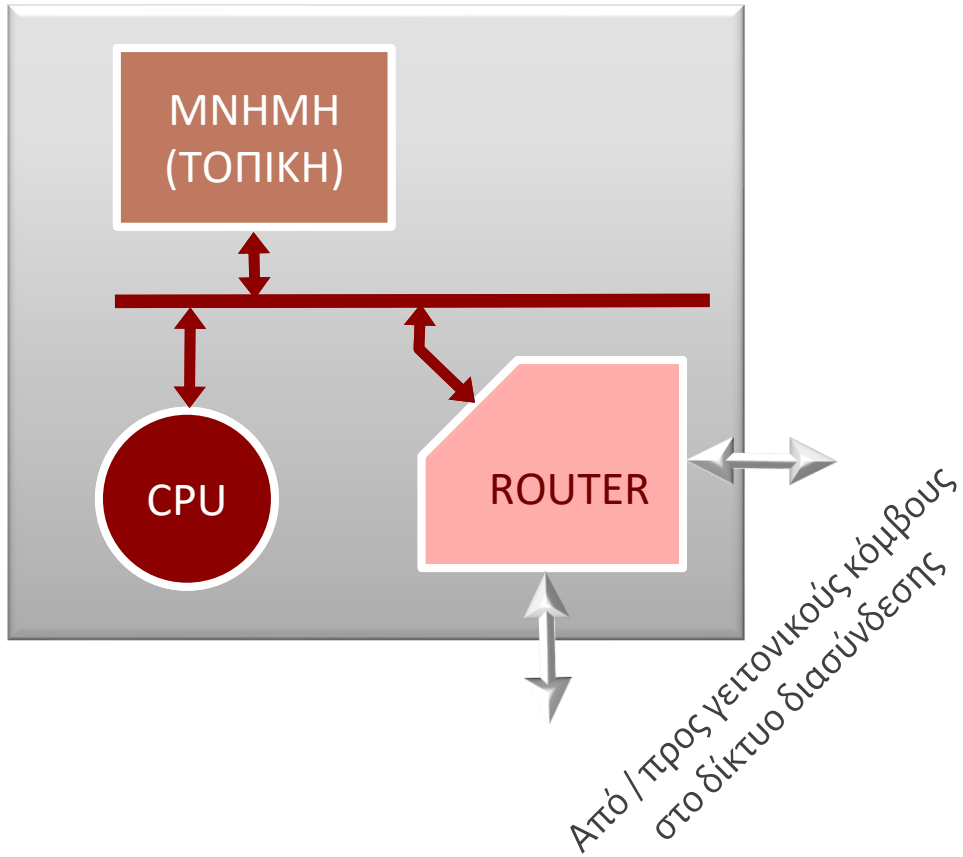
Κατανεμημένη κοινή μνήμη, NUMA

PARALLEL
PROCESSING
GROUP

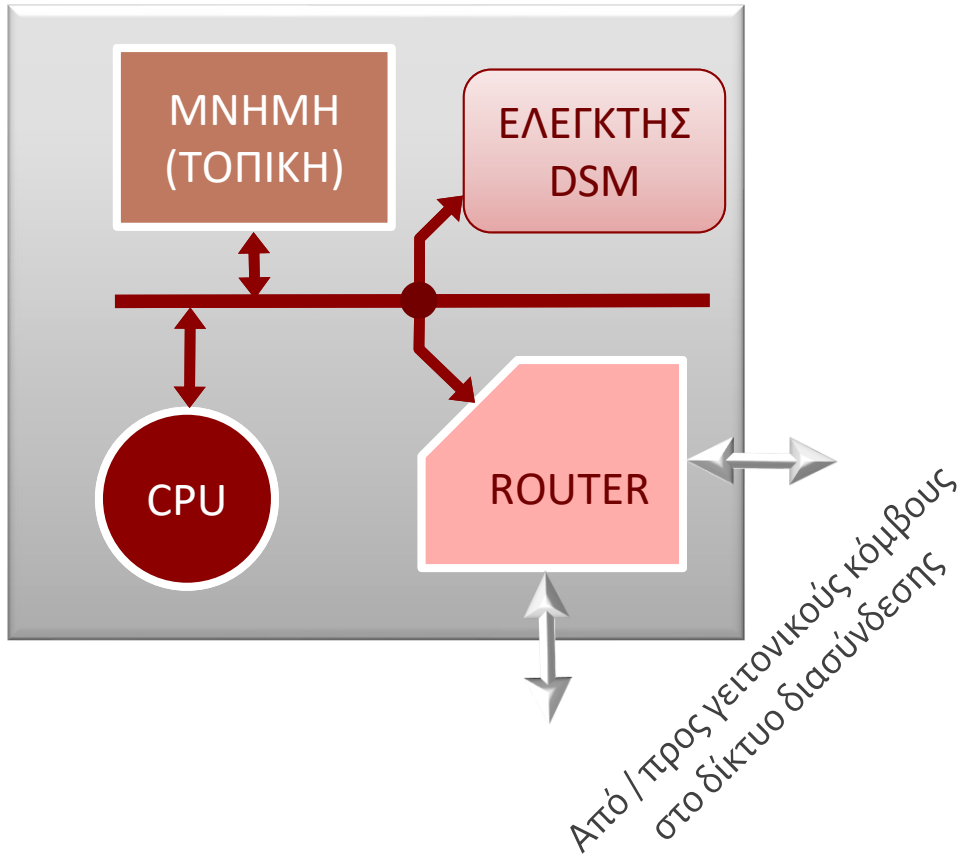
Γιατί;

- Συστήματα κατανεμημένης μνήμης:
  - Αρχιτεκτονική: **κλιμακώσιμη**
  - Προγραμματισμός (MPI): αρκετά **δύσκολος** αλλά και δυνατότητα επιδόσεων
- Συστήματα κοινόχρηστης μνήμης (SMPs):
  - Αρχιτεκτονική: **δύσκολα κλιμακώσιμη**
  - Προγραμματισμός: **οικείος / προσιτός**
- Το ιδεώδες:
  - Κλιμακώσιμες αρχιτεκτονικές που να προγραμματίζονται εύκολα (αλλά μην ξεχνάμε και τις επιδόσεις)
  - Μετά το σειριακό, το πιο εύκολο είναι ο προγραμματισμός κοινής μνήμης (π.χ. OpenMP)
- Λύση: *«emulation» της κοινόχρηστης μνήμης πάνω από το σύνολο των ιδιωτικών μνημών*
  - με hardware
  - με software (π.χ. σε clusters όπου είναι αδύνατον να αλλάξει το hardware)
  - Άρα *λογικά* κοινόχρηστη, *φυσικά* κατανεμημένη μνήμη

Hardware – κόμβος συστήματος

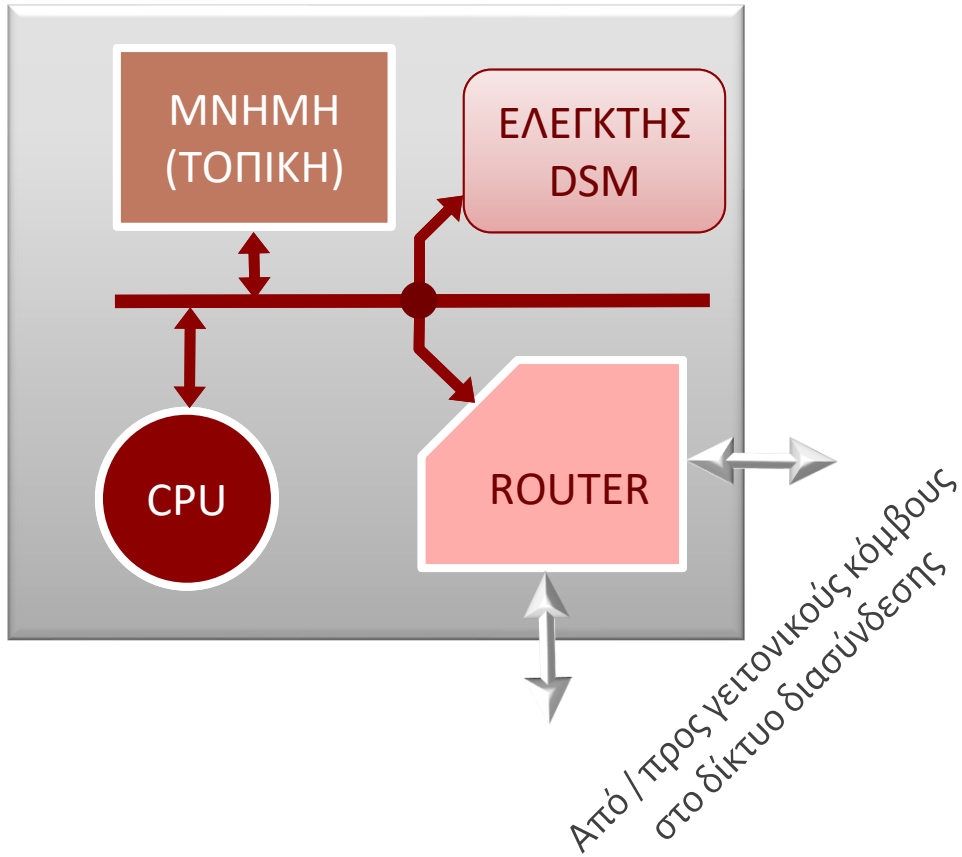


Ανομοιόμορφη προσπέλαση μνήμης (NUMA)



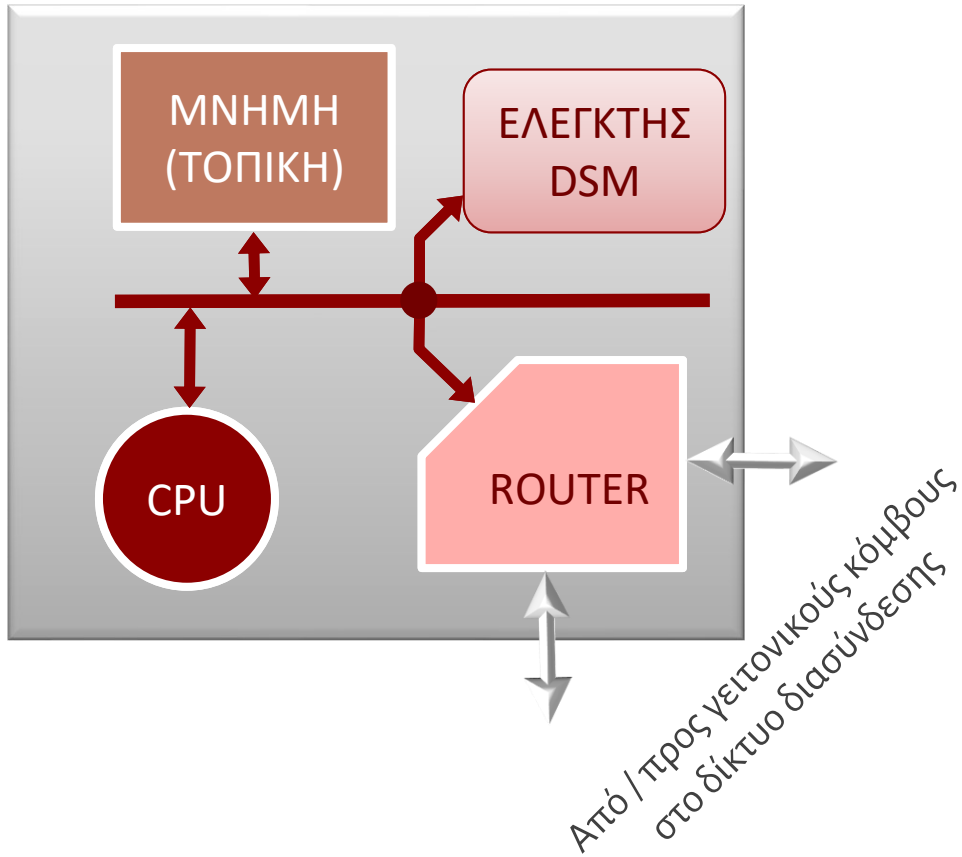
- Η CPU προσπελαύνει όλο τον χώρο διευθύνσεων ενιαία
- Η τοπική μνήμη έχει μόνο ένα μικρό κομμάτι του χώρου
- Ο ελεγκτής DSM ελέγχει κάθε διεύθυνση που προσπελαύνει η CPU
  1. Αν είναι για την τοπική μνήμη δεν κάνει τίποτε
  2. Αν όχι, αναλαμβάνει την επικοινωνία με τον κόμβο που την χειρίζεται (**home node**), στέλνοντας κατάλληλο μήνυμα. Μόλις έρθει η απάντηση, δίνει δεδομένα στη CPU σαν να ήταν αποθηκευμένο τοπικά

Ανομοιόμορφη προσπέλαση μνήμης (NUMA)



- Το μόνο που καταλαβαίνει η CPU είναι η διαφορά στην ταχύτητα προσπέλασης κάποιων δεδομένων (τα απομακρυσμένα κάνουν πολύ παραπάνω χρόνο να έρθουν)
  - NUMA (non-uniform memory access)
  - Π.χ. Cray T3D: 2 κύκλοι για τα τοπικά και περίπου 150 κύκλοι για τα απομακρυσμένα δεδομένα
  - Συστήματα με 4 επεξεργαστές AMD Opteron 8347HE: περίπου 26% (32%) επιπλέον χρόνος για απομακρυσμένη ανάγνωση (εγγραφή).

Ανομοιόμορφη προσπέλαση μνήμης (NUMA)



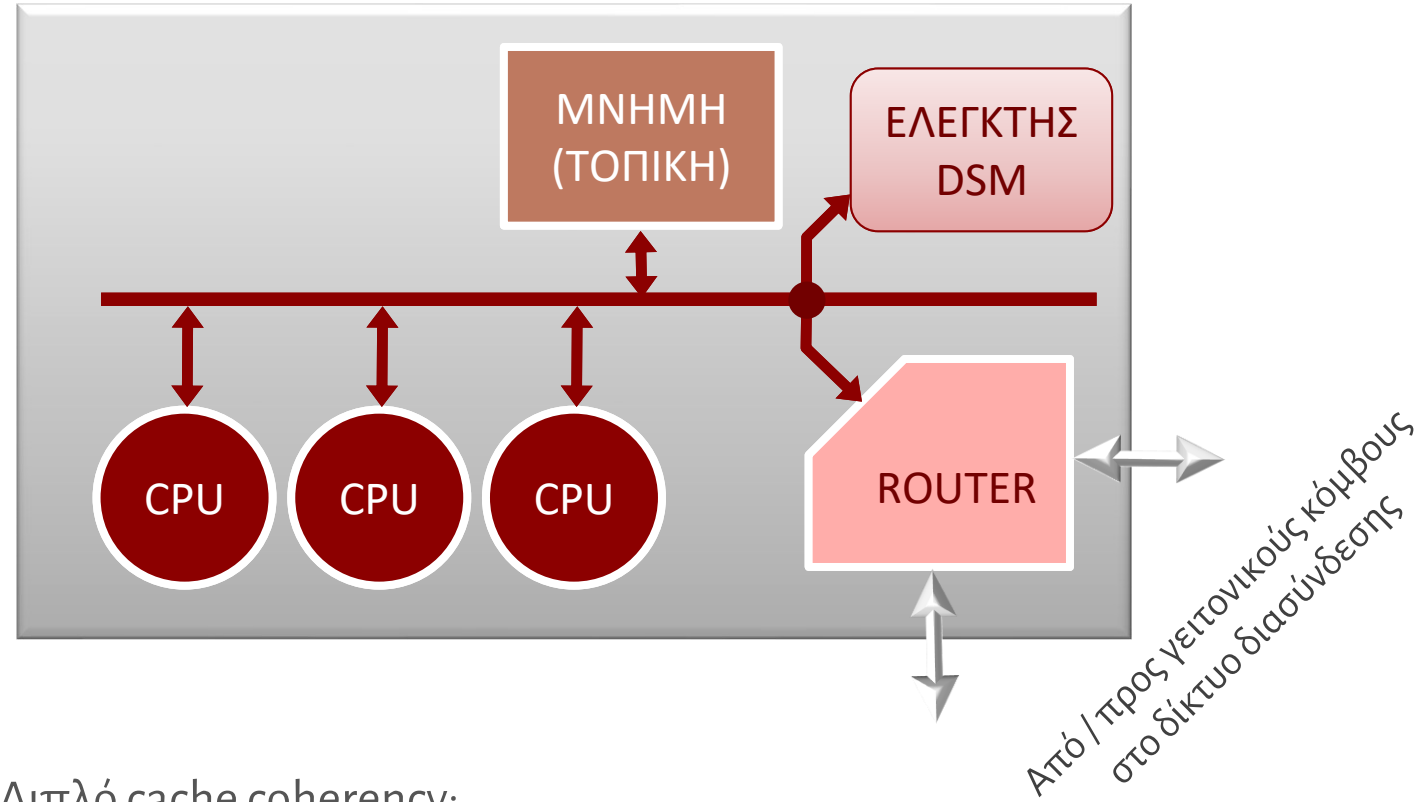
Ερώτηση: Μείωση χρόνου προσπέλασης???

- Απάντηση: caches για τα απομακρυσμένα δεδομένα
- Όμως, πάλι: πρόβλημα συνοχής για τα κοινά δεδομένα!

Λύσεις:

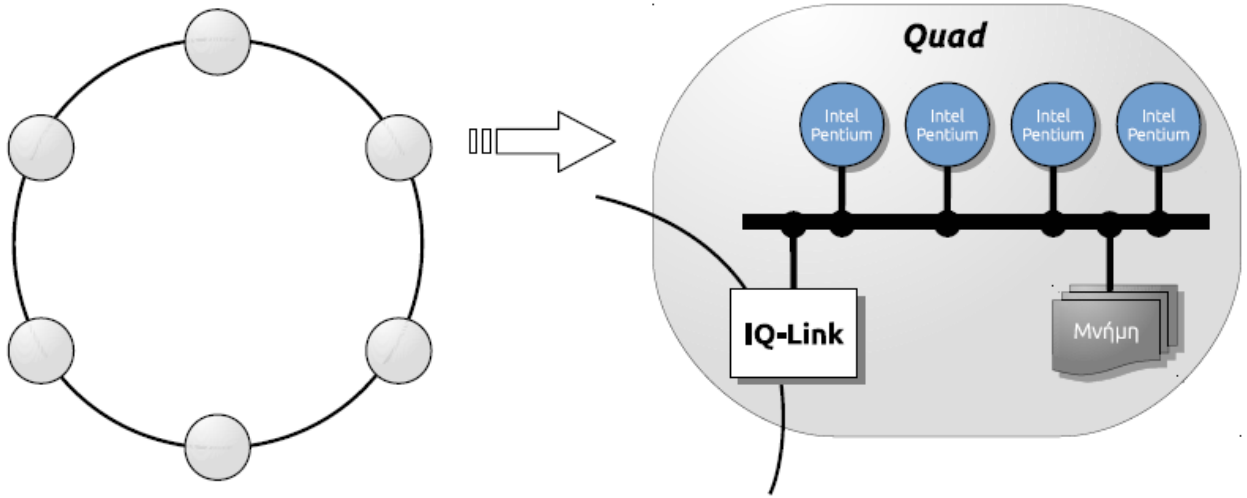
- Πρωτόκολλα συνοχής => cache-coherent NUMA (ccNUMA)
- Αποφυγή προβλήματος. Π.χ. στον Cray T3D δεν υπήρχε πρωτόκολλο συνοχής. Απλά, στις caches δεν επιτρέπονταν κοινά δεδομένα

Γενίκευση: πολυπύρηνος ή πολυεπεξεργαστικός κόμβος



- Διπλό cache coherency:
  - «εξωτερικό» πρωτόκολλο από ελεγκτή DSM για απομακρυσμένα δεδομένα
    - **Υποχρεωτικά πρωτόκολλο καταλόγων**
  - «εσωτερικό» πρωτόκολλο από caches των CPUs του κόμβου για τα τοπικά
    - Πρωτόκολλο snooping

Παράδειγμα: Sequent Numa-Q



- Το IQ-Link ενσωμάτωνε το ρόλο του διαδρομητή και του ελεγκτή DSM.
- Υλοποιούσε το εξωτερικό πρωτόκολλο
  - SCI (αλυσιδωτοί κατάλογοι)
- Εσωτερικά στην ομάδα, οι Pentium είχαν διατηρούσαν τη συνέπεια με πρωτόκολλο MESI

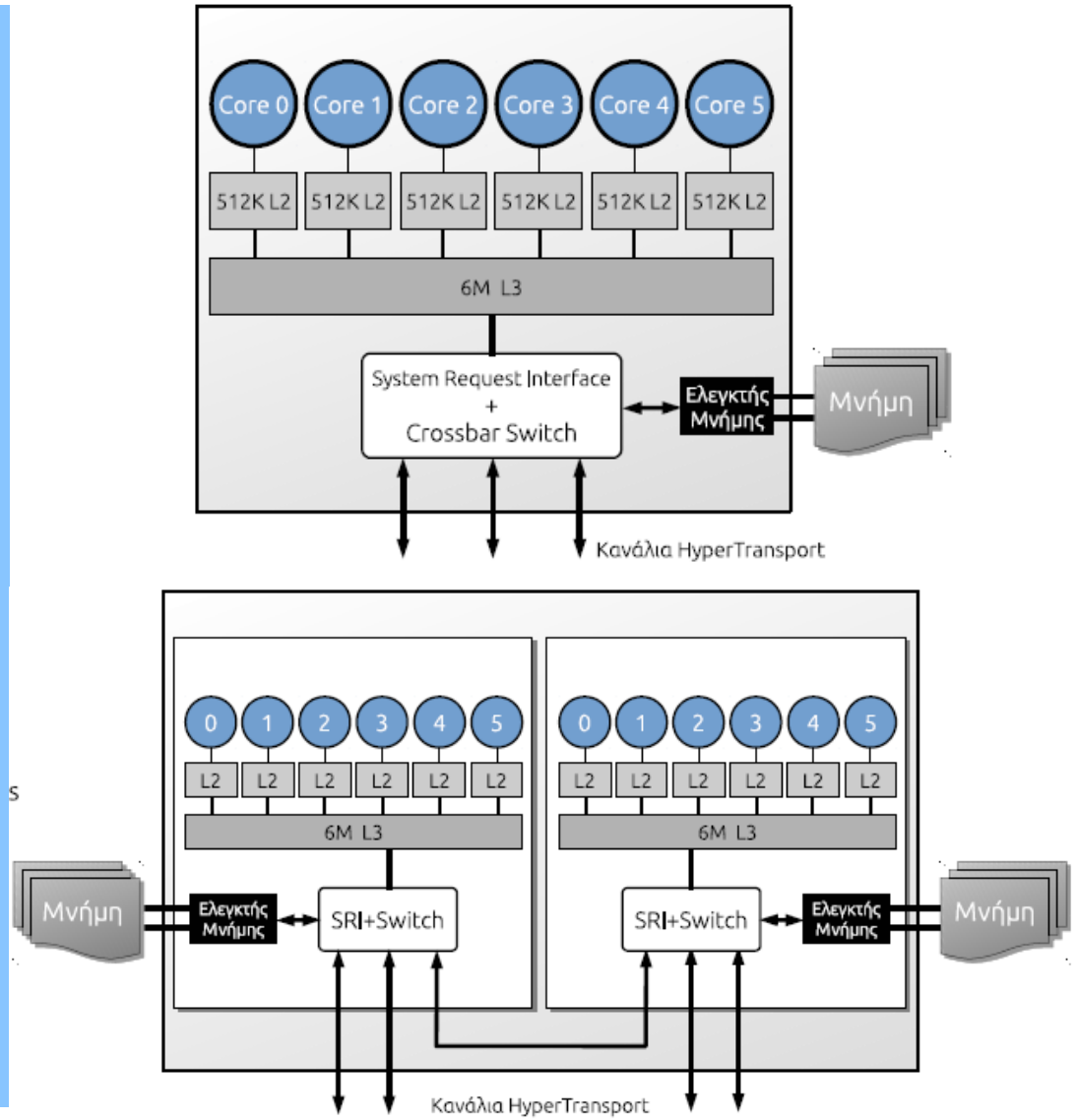
MYE023
MYE023
MYE023

Κατανεμημένη μνήμη και multicores

Πολυπύρηνοι επεξεργαστές και κατανεμημένη μνήμη

- Κυρίως δύο περιπτώσεις:
  - Χρήση multicore CPUs ως κόμβων ενός μεγαλύτερου συστήματος
    - Το σύστημα δομείται ως ένα δίκτυο από τέτοιους κόμβους
  - Χρήση multicore CPU ώστε να αποτελεί ολόκληρο το σύστημα
    - Λογικά, μιλάμε για CPU με αρκετούς πυρήνες (ίσως many-core CPU)
- Στην πρώτη περίπτωση ο κάθε επεξεργαστής έχει δικά του κανάλια για να συνδεθεί με τοπική μνήμη + επιπλέον κανάλια για να συνδεθεί με άλλους επεξεργαστές
  - AMD Opterons
  - Intel Xeon (E5, E6 series)
- Στη δεύτερη περίπτωση, μέσα στον ίδιο τον επεξεργαστή περιλαμβάνονται οι πυρήνες, οι ιδιωτικές μνήμες και το δίκτυο διασύνδεσης
  - NoC (Network-on-Chip)
  - MPSoC / MCSoc (Multiprocessor/Multicore System-on-Chip)

Παράδειγμα: AMD Opterons



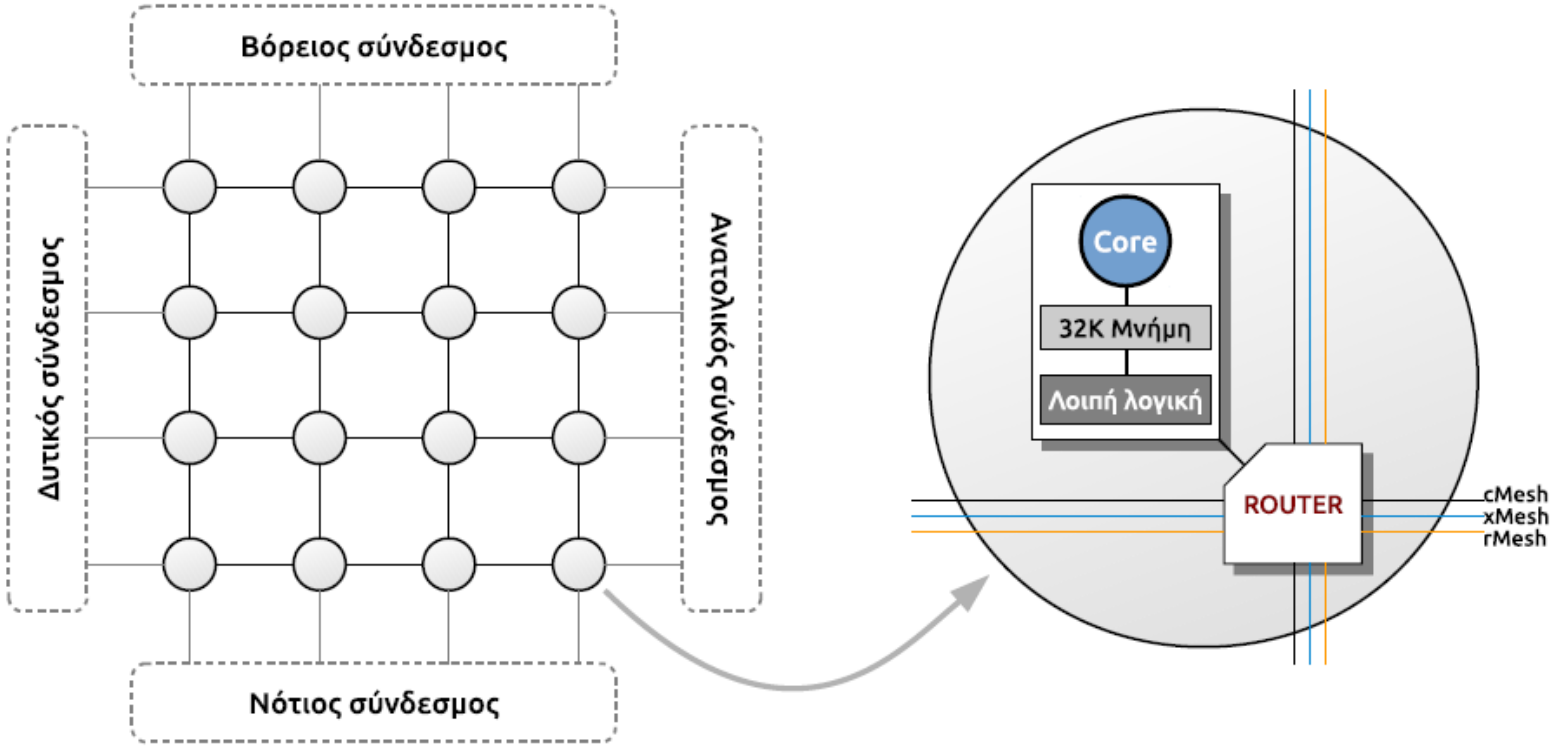
ISTANBUL (6-πύρηνος)

- 3 κανάλια HT ανά επεξεργαστή
- Μεταγωγή Virtual Cut-Through
- Μπορεί να συνδεθούν 8 κόμβοι με μέγιστη απόσταση 3 hops και να περισσέψουν και κανάλια για I/O
- HT Assist (1MiB στην L3) για υλοποίηση απλού πρωτοκόλλου καταλόγων

MAGNY-COURS (12-πύρηνος)

- 2 Istanbul ενωμένοι
- Μέχρι 4 επεξεργαστές στο σύστημα
- Επομένως μέχρι 48 πυρήνες (χωρίς προσθήκη επιπλέον υλικού)
- *NUMA factor:*
  - ≈1.2 για 1 hop (20% πιο αργή)
  - ≈1.5 για 2 hop (50% πιο αργή)

Παράδειγμα: Epiphany-16



- Για εγγραφές, 1.5 κύκλος per hop
  - για εγγραφή σε μνήμη που είναι σε απόσταση D => 1.5xD κύκλοι
- Epiphany-64 (δεν κυκλοφοράει στο εμπόριο προς το παρόν):
  - πλέγμα 8x8