



ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ

ΕΜ 361: Παράλληλοι Υπολογισμοί

Ενότητα #4: Παράλληλοι Αλγόριθμοι

Διδάσκων: Χαρμανδάρης Ευάγγελος
ΤΜΗΜΑ ΕΦΑΡΜΟΣΜΕΝΩΝ ΜΑΘΗΜΑΤΙΚΩΝ
ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΚΑΙ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΠΙΣΤΗΜΩΝ



Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο



ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης



ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ

Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται στην άδεια χρήσης **Creative Commons** και ειδικότερα **Αναφορά – Μη εμπορική Χρήση – Όχι Παράγωγο Έργο 3.0 Ελλάδα** (*Attribution – Non Commercial – Non-derivatives 3.0 Greece*)



[ή επιλογή ενός άλλου από τους έξι συνδυασμούς]

[και αντικατάσταση λογότυπου άδειας όπου αυτό έχει μπει (σελ. 1, σελ. 2 και τελευταία)]

- Εξαιρείται από την ως άνω άδεια υλικό που περιλαμβάνεται στις διαφάνειες του μαθήματος, και υπόκειται σε άλλου τύπου άδεια χρήσης. Η άδεια χρήσης στην οποία υπόκειται το υλικό αυτό αναφέρεται ρητώς.

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο Πανεπιστήμιο Κρήτης**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.



Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο



ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης





EM 361: Παράλληλοι Υπολογισμοί

Χαρμανδάρης Βαγγέλης, Τμήμα Εφαρμοσμένων Μαθηματικών
Πανεπιστήμιο Κρήτης, Χειμερινό Εξάμηνο 2009/10

Κεφάλαιο 4: Παράλληλοι Αλγόριθμοι

- Ταξινόμηση Παράλληλων Αλγόριθμων.
- Παράδειγμα: Υπολογισμός του Αριθμού π .
- Το Κόσκινο του Ερατοσθένη.
- Σχεδιασμός Παράλληλων Αλγορίθμων.
- Τύποι Επικοινωνίας Μεταξύ των Επεξεργαστών.



Παράλληλοι Αλγόριθμοι

- Οι παράλληλοι αλγόριθμοι μπορούν να ταξινομηθούν ανάλογα με το **πως** και **που** γίνεται ο παραλληλισμός.

Κατηγορίες παράλληλων αλγορίθμων:

- Παραλληλισμός σε επίπεδο bits (**Bit-level parallel approach**).
- Παραλληλισμός σε επίπεδο εντολών (**Control-parallel approach**).
- Παραλληλισμός σε επίπεδο δεδομένων (**Data-parallel approach**).



Παραλληλισμός σε Επίπεδο Bits

- **Bit-level parallelism:** Παραλληλισμός αυξάνοντας το μέγεθος της πληροφορίας (word size), σε bits, που μπορεί να επεξεργαστεί ένας επεξεργαστής ανά κύκλο λειτουργίας.
- Αυξάνοντας το “word size” ελαττώνεται ο αριθμός των πράξεων που πρέπει να εκτελέσει ένας επεξεργαστής για μεταβλητές με μήκος μεγαλύτερο του “word size” .
- Παράδειγμα: πρόσθεση 2 16-bit ακεραίων σε 8-bit επεξεργαστή απαιτεί 2 πράξεις ενώ σε 16-bit επεξεργαστή 1.

Ιστορικά:

- Για αρκετά χρόνια ήταν ο συνηθής τρόπος αύξησης της υπολογιστικής ισχύος: από 4-bit σε 8-bit, 16-bit και 32-bit επεξεργαστές. Οι τελευταίοι ήταν οι πιο συνηθισμένοι για περίπου 2 δεκαετίες.
- Πιο πρόσφατα (~2003) με την x86-64 αρχιτεκτονική 64-bit επεξεργαστές επικρατούν.
- Μέλλον: 128-bit επεξεργαστές;

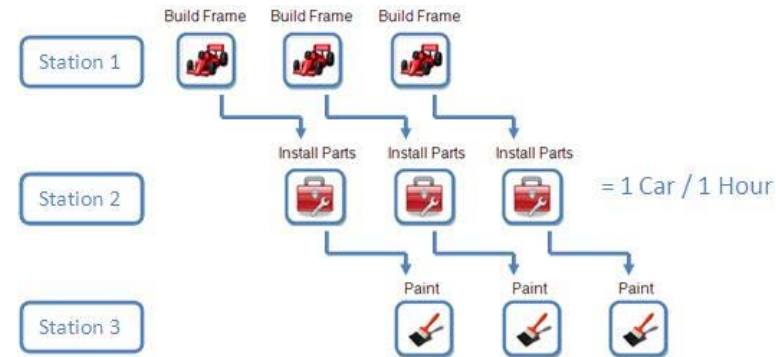


Παραλληλισμός σε Επίπεδο Εντολών

- **Control-level parallelism** ή **Παραλληλισμός Ελέγχου**: Εφαρμογή διαφορετικών πράξεων σε διαφορετικά δεδομένα ταυτόχρονα. Γνωστός και ως σωλήνωση (pipelining).
- Κατάλληλος για MIMD συστήματα.
- Θεωρούμε ένα πρόβλημα ως σύνολο από διαφορετικές διεργασίες όπου η καθεμία μπορεί να ανατεθεί σε διαφορετικό επεξεργαστή.

Παραδείγματα:

- Προσομοίωση ενός οικοσυστήματος: διαφορετικά είδη ζώων, φυτών, καιρός, κλπ. Κάθε υποσύστημα ανατίθεται σε διαφορετικό επεξεργαστή.
- Μοντελοποίηση αυτοκινήτου: τα διαφορετικά μέρη (μηχανή, σύστημα ψύξης, σύστημα θέρμανσης, κλπ.) κατανέμονται σε διαφορετικούς επεξεργαστές.





Παραλληλισμός σε Επίπεδο Δεδομένων

- **Data-level parallelism:** Εφαρμογή ίδιων πράξεων σε διαφορετικά δεδομένα ταυτόχρονα. Η ίδια διαδικασία εκτελείται σε πολλά δεδομένα ταυτόχρονα.
- Αναφέρεται και ως Κατάτμηση Χωρίου (**Domain Decomposition**).
- Κατάλληλος για SIMD και MIMD συστήματα.
- Διαφορετικές περιοχές του χώρου ανατίθενται σε διαφορετικούς επεξεργαστές.

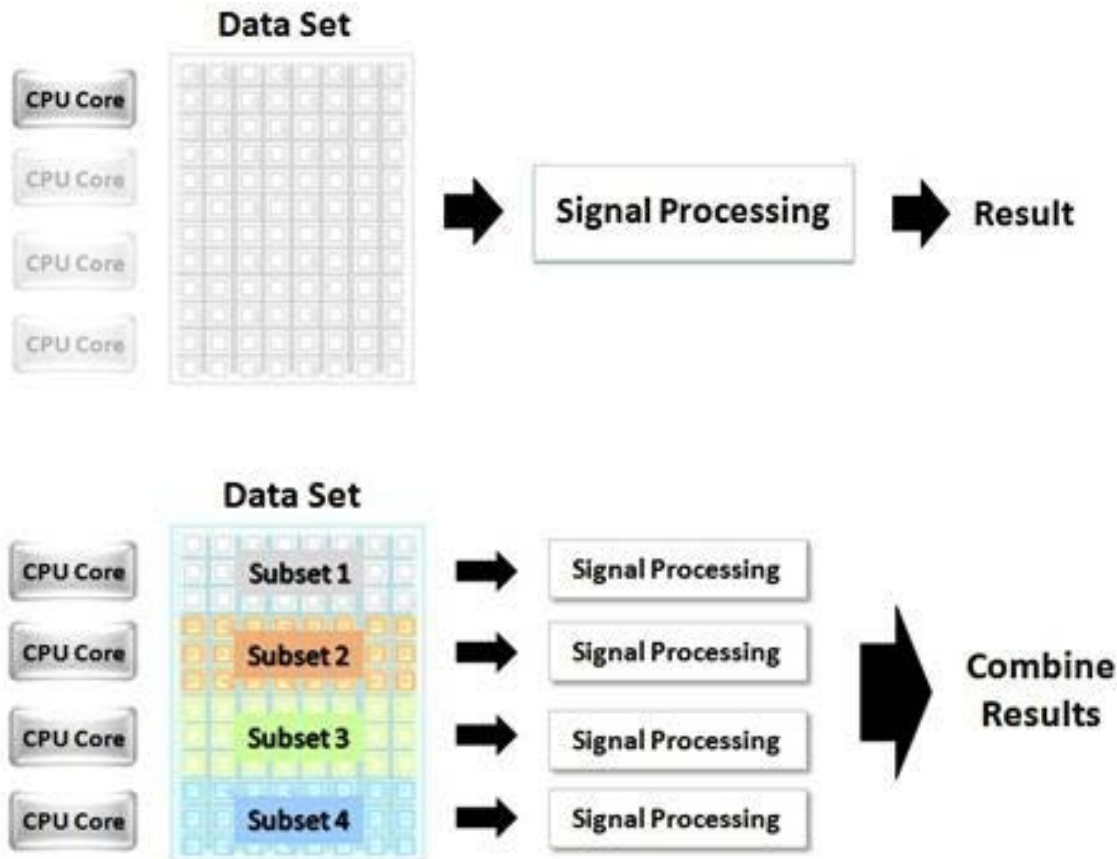
Παραδείγματα:

- Κατανομή ενός πίνακα σε διαφορετικούς επεξεργαστές.
- Μοριακή Προσομοίωση: διαφορετικά μέρη του χωρίου κατανέμονται σε διαφορετικούς επεξεργαστές.
- Αναζήτηση στοιχείων σε μια βάση δεδομένων.
- ... πολλά άλλα.



Παραλληλισμός σε Επίπεδο Δεδομένων

- **Domain Decomposition** : Ο πιο διαδεδομένος τρόπος παραλληλισμού πολύπλοκων επιστημονικών προβλημάτων.

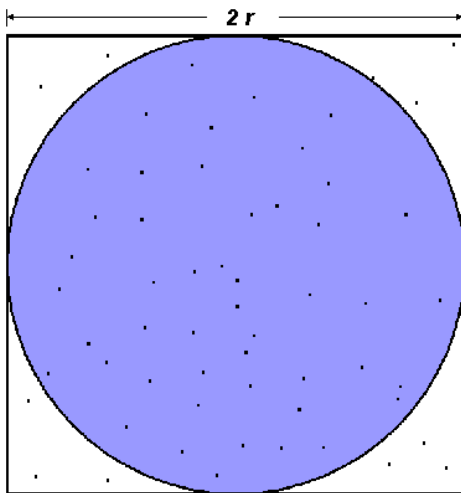




Παράδειγμα: Υπολογισμός του Αριθμού π

Υπολογισμός του αριθμού π με την ακόλουθη μέθοδο:

- Περικλείουμε κύκλο με ένα τετράγωνο. Δημιουργούμε m τυχαία σημεία μέσα στο τετράγωνο.
- Βρίσκουμε τα σημεία που εμπεριέχονται και μέσα στον κύκλο, n .
- Αν $r = n/m$, τότε ο αριθμός π προσεγγίζεται ως $\pi \approx 4r$. Όσο περισσότερα τα σημεία m τόσο μεγαλύτερη ακρίβεια του υπολογισμού.



$$A_S = (2r)^2 = 4r^2$$

$$A_C = \pi r^2$$

$$\pi = 4 \times \frac{A_C}{A_S}$$



Υπολογισμός του Αριθμού π

Σειριακός αλγόριθμος:

```
npoints = 1000000
circle_count = 0
do j = 1, npoints
    generate 2 random numbers between 0 and 1
    xcoordinate = random1
    ycoordinate = random2
    if (xcoordinate, ycoordinate) inside circle then
        circle_count = circle_count + 1
    end do
PI = 4.0*circle_count/npoints
```

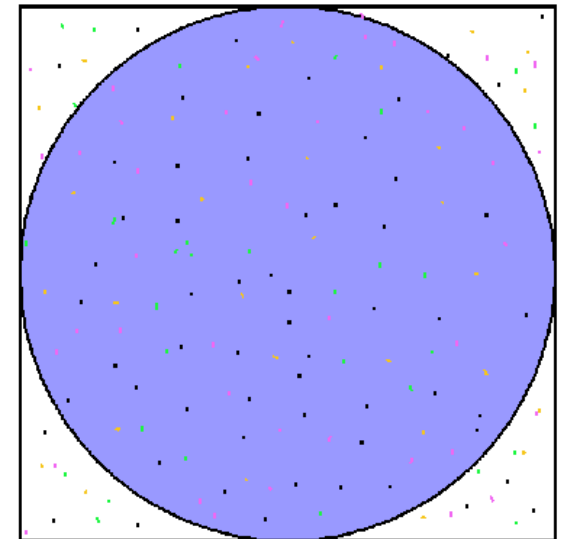
- Ο χρόνος υπολογισμού είναι κυρίως ο χρόνος εκτέλεσης της επαναληπτικής διαδικασίας (loop).
- Αυτό οδηγεί σε (σχεδόν) ‘τέλειο παραλληλισμό’ (**embarrassingly parallelism**):
 - Εντατικοί υπολογισμοί.
 - Ελάχιστη επικοινωνία, ελάχιστο I/O.



Υπολογισμός του π : Παραλληλισμός Δεδομένων

Ο παραλληλισμός αυτού του αλγόριθμου μπορεί να γίνει σε επίπεδο δεδομένων: αναθέτουμε σε κάθε επεξεργαστή μέρος της επαναληπτικής διαδικασίας.

- Κάθε επεξεργαστής εκτελεί το δικό του μέρος (task) του loop.
- Δεν χρειάζεται επικοινωνία μεταξύ των επεξεργαστών κατά τη διάρκεια εκτέλεσης της επαναληπτικής διαδικασίας.
- Χρησιμοποιούμε το μοντέλο «αφέντη/εργάτη» (master/slave).



■ task 1
■ task 2
■ task 3
■ task 4



Υπολογισμός του π : Παραλληλισμός Δεδομένων

Παράλληλος αλγόριθμος (με κόκκινο οι αλλαγές):

```
npoints = 1000000
circle_count = 0
p = number of tasks
num = npoints/p
do j = 1, num
    generate 2 random numbers between 0 and 1
    xcoordinate = random1
    ycoordinate = random2
    if (xcoordinate, ycoordinate) inside circle then circle_count = circle_count + 1
end do
find out if I am MASTER or WORKER
if I am MASTER receive from WORKERS their circle_counts
    compute PI (use MASTER and WORKER calculations)
else if I am WORKER
    send to MASTER circle_count
end if
```



Παράδειγμα: ‘Το Κόσκινο του Ερατοσθένη’

Αλγόριθμος εύρεσης πρώτων αριθμών (Sieve of Eratosthenes)

- Έστω ότι ζητάμε όλους τους πρώτους αριθμούς έως n . Ο αλγόριθμος προχωράει ως εξής:
 - Ξεκινάμε με τον αριθμό 2. Αποκλείουμε-διαγράφουμε όλα τα πολλαπλάσιά του ως n .
 - Ο επόμενος πρώτος μη-διαγραμμένος αριθμός είναι πρώτος.
 - Συνεχίζουμε με τον επόμενο πρώτο μη-διαγραμμένο αριθμό (το 3) και αποκλείουμε όλα τα πολ/σια του.
 - Επαναλαμβάνουμε την διαδικασία ως να φτάσουμε τον αριθμό n . Με τη διαδικασία αυτή βρίσκουμε όλα και λιγότερους αριθμούς προς διαγραφή. Όσοι απομένουν είναι οι πρώτοι αριθμοί.
- Δεν χρειάζεται να ελέγξουμε ως τον αριθμό n **αλλά τον $n^{1/2}$. Γιατί;**

Σειριακή εκτέλεση – Βασικά επαναλαμβανόμενα βήματα:

- (α) Βρίσκουμε τον επόμενο πρώτο.
- (β) Διαγράφουμε από την λίστα όλα τα πολλαπλάσιά του.



‘Το Κόσκινο του Ερατοσθένη’: Παραλληλισμός Ελέγχου

Sieve of Eratosthenes – A control parallel approach

- Κάθε επεξεργαστής δουλεύει (εκτελεί τα βήματα (α), (β)) σε διαφορετικό πρώτο αριθμό.
- Προβλήματα:
 1. Σε ασύγχρονη επικοινωνία δύο επεξεργαστές μπορεί να δουλεύουν στον ίδιο πρώτο.
 2. Μπορεί να εκτελούνται πράξεις που δεν χρειάζονται, π.χ. ο P1 βρίσκει αλλά δεν προλαβαίνει να διαγράψει τα πολ/σια του 2 ενώ ο P2 αφού τελειώνει με τον 3 βρίσκει ως επόμενο μη-διαγραμμένο αριθμό το 4!

Χρόνος Υπολογισμού:

- Έστω ότι ο χρόνος υπολογισμού είναι μόνο ο χρόνος υπολογισμού πολ/σιων και διαγραφής-μαρκαρίσματος κάθε κελιού.
- Έστω n ακέραιοι αριθμοί με k πρώτους $(\pi_1, \pi_2, \dots, \pi_k)$. Ο αριθμός των πράξεων (υπολογισμού πολ/σίων) είναι:

$$N_{\pi} = \left(\frac{n+1-\pi_1^2}{\pi_1} \right) + \left(\frac{n+1-\pi_2^2}{\pi_2} \right) + \dots + \left(\frac{n+1-\pi_k^2}{\pi_k} \right) = N_1 + N_2 + \dots + N_{\pi_k}$$



‘Το Κόσκινο του Ερατοσθένη’: Παραλληλισμός Ελέγχου

Χρόνος Υπολογισμού

- Έστω t_0 ο χρόνος ‘μαρκαρίσματος’ κάθε κελιού. Τότε ο σειριακός χρόνος εκτέλεσης είναι:

$$T_S = N_\pi t_0$$

Μέγιστη παράλληλη επιτάχυνση: Όταν στέλνουμε όλους τους πρώτους σε διαφορετικούς επεξεργαστές. Τότε ο χρόνος υπολογισμού αντιστοιχεί στους (περισσότερους) υπολογισμούς του αριθμού 2.

$$T_P^{\max} \equiv \lim_{P \rightarrow \infty} T_P = \left(\frac{n-3}{2} \right) t_0$$

- **Παράδειγμα:** Έστω $n=1000$. Τότε $N_\pi=1411$ και

$$S_{\max} = \frac{1411}{499} \cong 2.83$$



‘Το Κόσκινο του Ερατοσθένη’: Παραλληλισμός Δεδομένων

Sieve of Eratosthenes – A data parallel approach

- Όλοι οι επεξεργαστές δουλεύουν (εκτελούν τα βήματα (α), (β)) στον ίδιο πρώτο αριθμό.
- Έστω n ακέραιοι και σύστημα με P επεξεργαστές. Αναθέτουμε σε **κάθε επεξεργαστή n/P ακέραιους**. Θεωρούμε επίσης ότι $P \ll n^{1/2}$.
- Για σύστημα με κοινή μνήμη δεν υπάρχει κόστος επικοινωνίας.
- Για σύστημα με κατανεμημένη μνήμη υπάρχει κόστος επικοινωνίας.

Αλγόριθμος:

- Όλοι οι πρώτοι αριθμοί είναι στον $P1$. Ο $P1$ βρίσκει τον επόμενο πρώτο, π_k , και στέλνει την τιμή του στους άλλους επεξεργαστές.
- Κατόπιν όλοι οι επεξεργαστές βρίσκουν πολλαπλάσια του π_k στο δικό τους υποσύνολο των n αριθμών.
- Η διαδικασία συνεχίζεται ως ότου ο $P1$ βρει πρώτο αριθμό $> n^{1/2}$.



‘Το Κόσκινο του Ερατοσθένη’: Παραλληλισμός Δεδομένων

Χρόνος Εκτέλεσης: Ο χρόνος εκτέλεσης του αλγόριθμου είναι ο χρόνος υπολογισμού (διαγραφής-μαρκαρίσματος κάθε κελιού) και ο χρόνος επικοινωνίας.

• **Χρόνος Υπολογισμού:** Ο χρόνος υπολογισμού, θεωρώντας t_0 το χρόνο ‘μαρκαρίσματος’ ενός κελιού, είναι:

$$T_{comp} = \left[\left(\frac{n/P}{\pi_1} \right) + \left(\frac{n/P}{\pi_2} \right) + \dots + \left(\frac{n/P}{\pi_k} \right) \right] t_0$$

• **Επικοινωνία:** ο $P1$ στέλνει κάθε πρώτο αριθμό σε $(P-1)$ άλλους επεξεργαστές. Αν λ είναι ο χρόνος που χρειάζεται να στείλουμε έναν αριθμό, τότε ο συνολικός χρόνος επικοινωνίας για k πρώτους αριθμούς είναι:

$$T_{comm} = k(P-1)\lambda$$

Προσοχή: ο χρόνος υπολογισμού **μειώνεται** όσο αυξάνει ο αριθμός των επεξεργαστών ενώ ο χρόνος επικοινωνίας **αυξάνει**.



Σχεδιασμός Παράλληλων Αλγορίθμων

Στόχος: ο σχεδιασμός και ο προγραμματισμός του βέλτιστου δυνατού παράλληλου αλγόριθμου.

- Πρώτο βήμα είναι πάντα η **κατανόηση του προβλήματος και του σειριακού κώδικα**, αν υπάρχει.

- Επιθυμητά χαρακτηριστικά: Η ελάχιστη δυνατή επικοινωνία, Επεκτασιμότητα, Τοπικότητα, Επιμεριστικότητα.

- **Γενικές προσεγγίσεις:**

- Επιμερισμός εντολών-διεργασιών → Παραλληλισμός Ελέγχου

- Επιμερισμός χωρίου → Παραλληλισμός Δεδομένων

- Πρέπει να λάβουμε υπ' όψιν:

- Αριθμό διεργασιών – αριθμό επεξεργαστών.

- Διεργασίες συγκρίσιμου μεγέθους.

- Πως αλλάζει το μέγεθος και ο αριθμός των διεργασιών με το μέγεθος του προβλήματος.

- **Βασική Ερώτηση:** Είναι ο παράλληλος αλγόριθμος μοναδικός; Αν όχι ποιες είναι οι εναλλακτικές λύσεις.



Σχεδιασμός Παράλληλων Αλγορίθμων

Επικοινωνία μεταξύ των επεξεργαστών:

- Εκτίμηση του κόστους επικοινωνίας: της αλληλεξάρτησης μεταξύ των διεργασιών.
- Σχεδιασμός της επικοινωνίας μεταξύ των διεργασιών: Πότε, Πως, Που και Τι θα σταλεί-ληφθεί.
- Διαγράμματα επικοινωνίας-μεταφοράς δεδομένων.
- Όγκος των μεταφερόμενων πληροφοριών μεταξύ των επεξεργαστών.
- Καθορισμός του τρόπου επικοινωνίας: blocking vs. non-blocking.
- Τύποι επικοινωνίας:
 - Τοπικά/Καθολικά.
 - Δομημένα/Μη Δομημένα.
 - Στατικά/Δυναμικά.
 - Συγχρονισμένα/Ασύγχρονα.



Τύποι-Σχήματα Επικοινωνίας

- **Τοπική:** η επικοινωνία επικεντρώνεται μεταξύ μικρού αριθμού διεργασιών.
- **Καθολική:** κάθε διεργασία επικοινωνεί με μεγάλο αριθμό διεργασιών.
- **Δομημένη:** η επικοινωνία ακολουθεί κάποια συγκεκριμένη δομή-τοπολογία, π.χ. δομή δέντρου, αστεριού, κλπ.
- **Μη Δομημένη:** η επικοινωνία δεν ακολουθεί κάποια συγκεκριμένη δομή-τοπολογία.
- **Στατική:** η επικοινωνία είναι σταθερή κατά τη διάρκεια εκτέλεσης του προγράμματος.
- **Δυναμική:** η επικοινωνία αλλάζει κατά τη διάρκεια εκτέλεσης του προγράμματος.
- **Συγχρονισμένη:** η αποστολή και λήψη των πληροφοριών-δεδομένων γίνεται ταυτόχρονα.
- **Ασύγχρονη:** η αποστολή και λήψη των πληροφοριών-δεδομένων είναι ανεξάρτητες μεταξύ τους.



Σχεδίαση Επικοινωνίας

Βασικές ερωτήσεις που τίθενται στη σχεδίαση της επικοινωνίας:

- Εκτελούν όλες οι διεργασίες ίδιας τάξης αριθμό εντολών επικοινωνίας;
- Υπάρχει διεργασία που επικοινωνεί με πολλές από (ή όλες) τις άλλες διεργασίες; Υπάρχει κίνδυνος συμφόρησης επικοινωνίας (bottleneck);
- Ποιος είναι ο βαθμός τοπικότητας του σχήματος επικοινωνίας;
- Είναι δυνατόν οι υπολογισμοί να γίνονται ταυτόχρονα με την επικοινωνία;
- Μπορούν πολλές διεργασίες να εκτελούνται ταυτόχρονα;



Βιβλιογραφία

- *Parallel Programming*, B. Wilkinson, M. Allen, Prentice Hall, 2nd Ed. 2005.
- *Designing and Building Parallel Programs*, Ian Foster, Addison-Wesley 1994.
- *Parallel Computing: Theory and Practice*, M. J. Quinn, McGraw-Hill, 1994.
- http://en.wikipedia.org/wiki/Sieve_of_Eratosthenes
- *Parallel Scientific Computing in C++ and MPI*, G. Karniadakis and R.M. Kirby II, Cambridge, 2003.

Τέλος Ενότητας



Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης