Ιόνιο Πανεπιστήμιο – Τμήμα Πληροφορικής

ΠΜΣ «Ερευνητικές Κατευθύνσεις στην Πληροφορική»

Παράλληλη και Κατανεμημένη Υπολογιστική Επεξεργασία

2019-20

Εισαγωγή

(Βασικές έννοιες παράλληλου υπολογισμού)

http://mixstef.github.io/courses/pms-parcomp/

Μ.Στεφανιδάκης

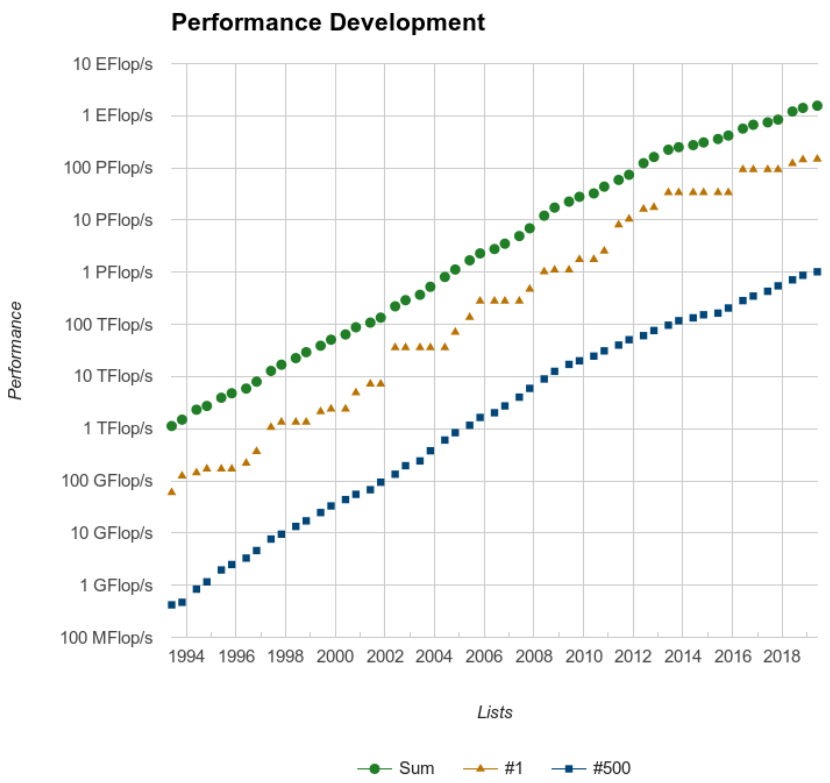
Παράλληλη επεξεργασία

- Ταυτόχρονη εκτέλεση διεργασιών
  - Κώδικας που εκτελείται την ίδια στιγμή σε διαφορετικές υπολογιστικές μονάδες (πόρους)
- Γιατί είναι επιθυμητή;
  - Επίλυση υπολογιστικά δύσκολων προβλημάτων
  - Αλλά και απλούστερων προβλημάτων με πολύ μεγάλα σύνολα δεδομένων εισόδου
    - Μοντελοποίηση φυσικών φαινομένων
    - Τεχνητή νοημοσύνη
    - Βιοιατρική
    - κ.λ.π.

Μόνο για υπερυπολογιστές;

- High Performance Computing (HPC)

https://www.top500.org/statistics/perfdevel/



Και στους «καθημερινούς» υπολογιστές μας

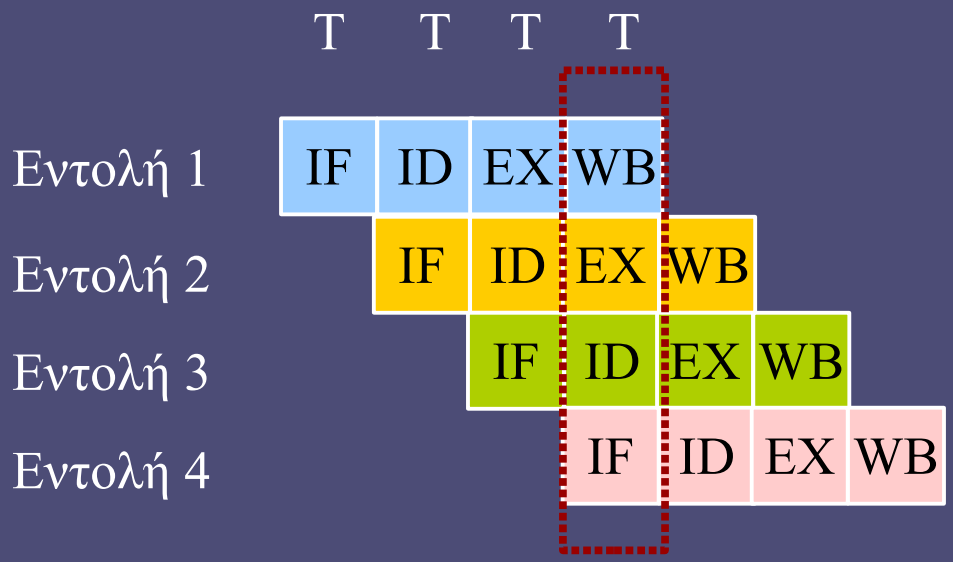
- Desktop, laptop, smartphones...
  - Οι επεξεργαστές που περιέχουν διαθέτουν άφθονες πηγές παράλληλης επεξεργασίας
  - Βασίζονται στη συνεχή πρόοδο της τεχνολογίας
    - Ο «νόμος» του Moore – η συνεχής συρρίκνωση του τρανζίστορ
  - Και στις αρχιτεκτονικές βελτιώσεις
    - Αποδοτικότερη εκτέλεση υπολογιστικών λειτουργιών

Η αναγκαιότητα της παράλληλης επεξεργασίας

- Το τέλος της «κούρσας των GHz»
  - Στις αρχές της δεκαετίας του 2000
  - Εμπόδια στα οφέλη από την αύξηση της συχνότητας του ρολογιού
    - Υπέρμετρη κατανάλωση ενέργειας – αδυναμία απαγωγής θερμότητας
    - Οι αλληλοεξαρτήσεις μεταξύ εντολών τονίζονται – μείωση της προσδοκώμενης αύξησης της απόδοσης
  - Το σειριακό πρόγραμμα δεν γίνεται πλέον γρηγορότερο «αυτόματα» με την πάροδο του χρόνου
    - "Free ride is over!"
- Πώς θα χρησιμοποιηθεί η αφθονία τρανζίστορ;
  - Παράλληλη επεξεργασία σε χαμηλότερες συχνότητες

Παρεχόμενη παραλληλία: pipelines

- «Παραλληλισμός σε επίπεδο εντολών» (ILP)
  - Μια βασική τεχνική παράλληλης επεξεργασίας
    - Την ίδια στιγμή εκτελούνται λειτουργίες πολλαπλών εντολών μηχανής
    - Ιδανικά, σε κάθε κύκλο ρολογιού (περίοδος Τ) ολοκληρώνεται μια εντολή



Παρεχόμενη παραλληλία: superscalar CPU

- Εκκίνηση περισσότερων από μια εντολή σε κάθε κύκλο ρολογιού
  - Την εποχή της «κούρσας των GHz»
  - Απαιτούνται πολλαπλά pipelines
    - Η επιλογή γίνεται αυτόματα από την ΚΜΕ που παρακολουθεί τις εντολές σε ορισμένο βάθος χρόνου (“window”)
  - Τεχνικές για την αύξηση των εντολών που μπορούν να εκτελεστούν παράλληλα
    - Εκτέλεση εκτός σειράς (out of order execution)
    - Μετονομασίες καταχωρητών (register renaming)
    - Πρόβλεψη διακλαδώσεων (branch prediction)

Παρεχόμενη παραλληλία: vector instructions

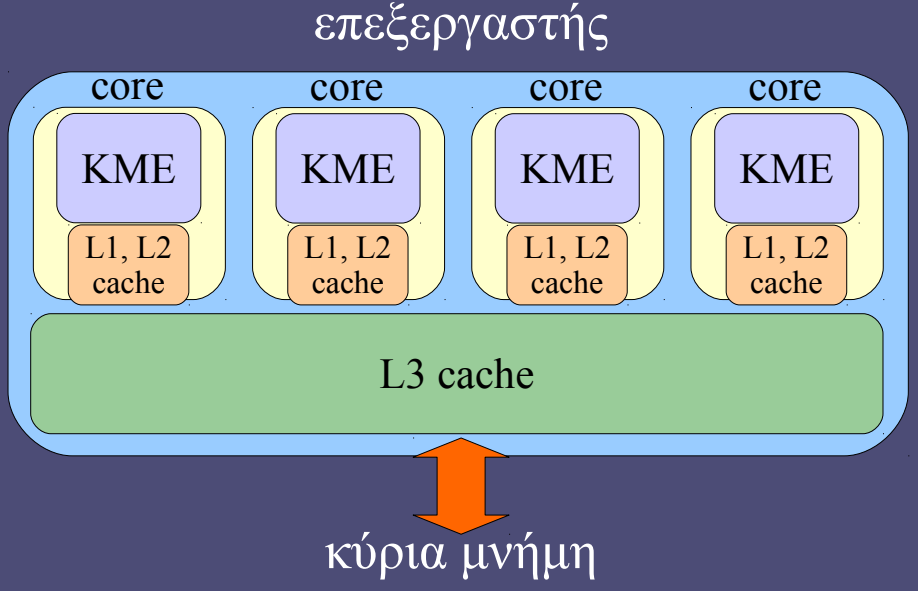
- Εντολές με πολύ μεγάλο εύρος δεδομένων
  - Την εποχή της «κούρσας των GHz»
  - Η ίδια λειτουργία σε πολλαπλά δεδομένα (SIMD)
  - Streaming instructions
    - Αρχικά για δεδομένα multimedia

Παρεχόμενη παραλληλία: SMT

- Simultaneous Multithreading
  - Το ξέρουμε καλύτερα με τον όρο marketing: “hyperthreading”
  - Παραλληλισμός σε επίπεδο thread (TLP)
  - Η «κούρσα των GHz» φτάνει στο τέλος της
  - Η ΚΜΕ μοιράζει τις μονάδες εκτέλεσης μεταξύ 2 (ή 4 ή 8..) διεργασιών
    - Κρατώντας ξεχωριστή κατάσταση (καταχωρητές) ανά διεργασία
    - Στο λειτουργικό σύστημα φαίνονται ως ανεξάρτητοι «λογικοί» πυρήνες

Παρεχόμενη παραλληλία: multicore

- Περισσότεροι πυρήνες (cores) στον επεξεργαστή
  - Η «κούρσα των GHz» έχει τελειώσει οριστικά
  - Παραλληλισμός σε επίπεδο thread (TLP)
  - Αυξάνεται η πίεση στη (μία και μοναδική) σύνδεση με τη μνήμη – προσθήκη μεγαλύτερης ιεραρχίας κρυφών μνημών



Ο ρόλος του λογισμικού

- Το λογισμικό επωμίζεται το βάρος της αποδοτικής χρήσης της προσφερόμενης παραλληλίας
  - Η αποδοτική παράλληλη επεξεργασία βασίζεται στη συνεργασία
    - Λειτουργικού συστήματος
    - Μεταγλωττιστή
    - Αλγορίθμων και Δομών δεδομένων
    - Και του κώδικά μας ☺
  - Και τη γνώση των χαρακτηριστικών του υλικού (hardware)
    - Εγκαταλείπουμε την αρχή «αποσύνδεσης του προγράμματός μας από το υλικό εκτέλεσης»;
    - Θα χρειαστούμε νέα frameworks που θα κρύβουν τις λεπτομέρειες του παραλληλισμού;

Γνωρίζοντας το σύστημα εκτέλεσης

```
$ cat /proc/cpuinfo


$ more /sys/devices/system/cpu/
cpu<i>/cache/index<j>/*
```

Απόδοση παράλληλων προγραμμάτων

$$\text{Speedup } S_P = \frac{\text{χρόνος σειριακής εκτέλεσης}}{\text{χρόνος παράλληλης εκτέλεσης}}$$

- Επιτάχυνση (speedup) με p επεξεργαστικούς κόμβους
  - Στην καλύτερη περίπτωση $S_p = p$
  - Μερικές φορές, για ανεξάρτητους λόγους προκύπτει $S_p > p$ (superlinear speedup)
  - Επίσης: αποδοτικότητα (efficiency) $E_p = S_p / p$

Παράγοντες περιορισμού του speedup

$$\text{Speedup } S_P = \frac{t_s}{f \times t_s + (1-f)t_s/p} = \frac{1}{f + (1-f)/p}$$

- Κάθε αλγόριθμος περιέχει ένα ποσοστό εργασίας που πρέπει να εκτελεστεί σειριακά (f)
  - Επιβάρυνση παραλληλισμού (overhead)
  - Επικοινωνία και συγχρονισμός επεξεργαστικών κόμβων για την ανταλλαγή δεδομένων
  - $S_P \rightarrow 1/f$ όταν $p \rightarrow \infty$ (νόμος του Amdahl)

Μια πιο αισιόδοξη εικόνα

$$\text{Speedup } S_P = \frac{f + p(1-f)}{f + (1-f)} = f + p(1-f)$$

- Ο «νόμος» του Gustafson
  - Scaled speedup
  - Με περισσότερους επεξεργαστικούς κόμβους, τα δεδομένα εισόδου μπορούν να έχουν μεγαλύτερο μέγεθος
  - Στην περίπτωση αυτή δεν μας περιορίζει το ποσοστό του σειριακού μέρους

Εμπόδια στη διαδικασία παραλληλισμού

- Μπορούμε πάντα να μετατρέψουμε αποδοτικά ένα σειριακό πρόγραμμα στο αντίστοιχο παράλληλο;
  - Αλληλεξαρτήσεις δεδομένων
    - Τα διάφορα στάδια εξαρτώνται από τιμές προηγούμενου υπολογισμού
    - Αναμονή για υπολογισμό εισόδων
  - Προσπέλαση μνήμης
    - Πολλά προγράμματα (και αλγόριθμοι) έχουν απόδοση που εξαρτάται από την επικοινωνία με τη μνήμη
    - Ο χρόνος μεταφοράς δεδομένων επισκιάζει κάθε όφελος παραλληλισμού
    - Ο τρόπος προσπέλασης μνήμης επηρεάζει τον χρόνο μεταφοράς
  - (σε επόμενα..)

Βιβλιογραφία

- Michael McCool, James Reinders, and Arch Robison. 2012. *Structured Parallel Programming: Patterns for Efficient Computation* (1st ed.). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Εμπόδια στη διαδικασία παραλληλισμού

- Μπορούμε πάντα να μετατρέψουμε αποδοτικά ένα σειριακό πρόγραμμα στο αντίστοιχο παράλληλο;
  - Αλληλεξαρτήσεις δεδομένων
    - Τα διάφορα στάδια εξαρτώνται από τιμές προηγούμενου υπολογισμού
    - Αναμονή για υπολογισμό εισόδων
  - Προσπέλαση μνήμης
    - Πολλά προγράμματα (και αλγόριθμοι) έχουν απόδοση που εξαρτάται από την επικοινωνία με τη μνήμη
    - Ο χρόνος μεταφοράς δεδομένων επισκιάζει κάθε όφελος παραλληλισμού
    - Ο τρόπος προσπέλασης μνήμης επηρεάζει τον χρόνο μεταφοράς
  - (σε επόμενα..)

Εμπόδια στη διαδικασία παραλληλισμού

- Μπορούμε πάντα να μετατρέψουμε αποδοτικά ένα σειριακό πρόγραμμα στο αντίστοιχο παράλληλο;
  - Αλληλεξαρτήσεις δεδομένων
    - Τα διάφορα στάδια εξαρτώνται από τιμές προηγούμενου υπολογισμού
    - Αναμονή για υπολογισμό εισόδων
  - Προσπέλαση μνήμης
    - Πολλά προγράμματα (και αλγόριθμοι) έχουν απόδοση που εξαρτάται από την επικοινωνία με τη μνήμη
    - Ο χρόνος μεταφοράς δεδομένων επισκιάζει κάθε όφελος παραλληλισμού
    - Ο τρόπος προσπέλασης μνήμης επηρεάζει τον χρόνο μεταφοράς
  - (σε επόμενα..)

Εμπόδια στη διαδικασία παραλληλισμού

- Μπορούμε πάντα να μετατρέψουμε αποδοτικά ένα σειριακό πρόγραμμα στο αντίστοιχο παράλληλο;
  - Αλληλεξαρτήσεις δεδομένων
    - Τα διάφορα στάδια εξαρτώνται από τιμές προηγούμενου υπολογισμού
    - Αναμονή για υπολογισμό εισόδων
  - Προσπέλαση μνήμης
    - Πολλά προγράμματα (και αλγόριθμοι) έχουν απόδοση που εξαρτάται από την επικοινωνία με τη μνήμη
    - Ο χρόνος μεταφοράς δεδομένων επισκιάζει κάθε όφελος παραλληλισμού
    - Ο τρόπος προσπέλασης μνήμης επηρεάζει τον χρόνο μεταφοράς
  - (σε επόμενα..)