Λειτουργικά Συστήματα (HY321)

Διάλεξη 9:

Πολιτικές Αντικατάστασης Σελίδων - Λυγισμός (Thrashing)

Χρήστος Δ. Αντωνόπουλος
6/4/2017

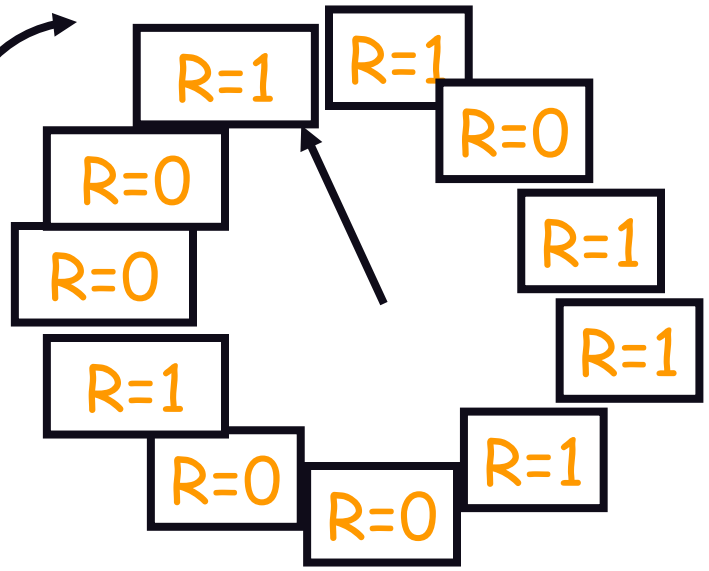
Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών, Πανεπιστήμιο Θεσσαλίας

Η Απόλυτη LRU

- Σε κάθε αναφορά στη μνήμη
  - «Χρονοσφράγισε» το πλαίσιο
- Την ώρα της έξωσης:
  - Ψάξε για την παλαιότερη χρονοσφραγίδα
- Προβλήματα:
  - Χώρος για λογιστικά
  - Στην πράξη οι χρονοσφραγίδες δεν υποστηρίζονται από το υλικό
- Ας βάλουμε λίγο νερό στο κρασί μας…
  - Θέλω κάτι απλό και γρήγορο που να βρίσκει «παλιές» σελίδες
  - Εξάλλου δεν έχει σημασία και το πολύ μακρυνό παρελθόν…

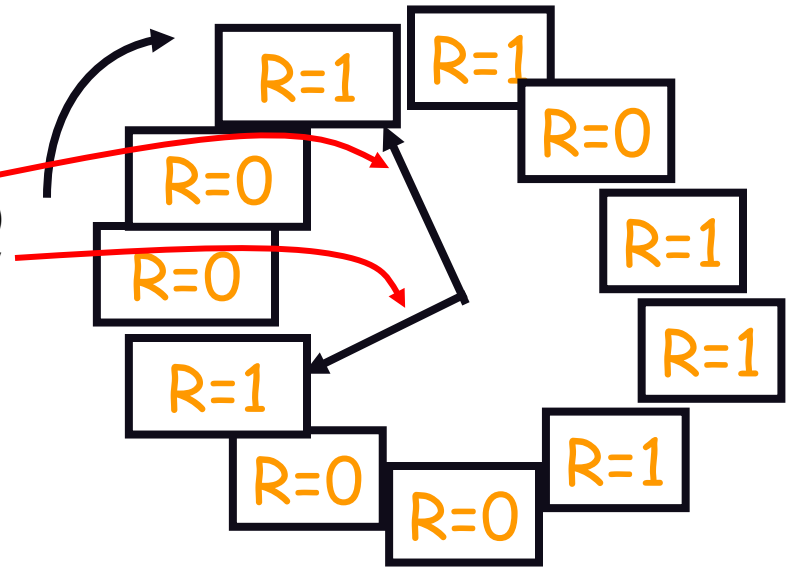
Ο Αλγόριθμος Ρολογιού (Δεύτερης Ευκαιρίας)

- Ένα bit αναφοράς (ref bit) ανά σελίδα στον πίνακα σελίδων
  - Το υλικό το θέτει όταν ακουμπά τη σελίδα, το ΛΣ το σβήνει περιοδικά
  - Οι σελίδες που έχουν το bit 1 έχουν χρησιμοποιηθεί πιο πρόσφατα από αυτές που το έχουν 0!
- Αλγόριθμος αντικατάστασης: FIFO, αλλά αγνόησε τις σελίδες με bit αναφοράς 1
  - Κράτησε τις σελίδες σε μια κυκλική λίστα
  - Αναζήτηση: Αν το ref bit = 1, κάντο 0 & άφησε τη σελίδα, διαφορετικά διώξτη.
- Τι συμβαίνει αν…
  - Γυρίζουμε γρήγορα;
  - Γυρίζουμε αργά;



Ρολόι με 2 Δείκτες

- Και αν έχουμε πολύ μεγάλη μνήμη;
  - Μήπως βρίσκουμε σχεδόν όλα τα ref bits 1;
  - Αν είναι όλα 1;
    - Πίσω στη FIFO
- Λύση:
  - Άλλος ένας δείκτης στο ρολόι
    - Ο πρώτος μηδενίζει τα ref bits
    - Ο δεύτερος (κάποιες σελίδες πίσω) απομακρύνει τις σελίδες που έχουν το ref bit ακόμα 0
  - Τι κάναμε ουσιαστικά;
  - Προβλήματα:
    - Αν η γωνία είναι πολύ μικρή;
    - Αν η γωνία είναι πολύ μεγάλη;



Μια Μικρή Βελτίωση

- Μην πετάξεις τίποτα!!!
  - Καμία πληροφορία από το bit αλλαγής (modified)
  - Αν δύο σελίδες έχουν το bit προσπέλασης 0, ποια να προτιμήσω για έξωση;
    - Αυτή που έχει το modified bit 0
      - Γιατί;

Άλλη μια Μικρή Βελτίωση;

- Χρησιμοποίησε n bits αντί για 1
  - Μεγαλύτερη ακρίβεια!
  - Πού να τα αποθηκεύσω;
    - Στο H/W ακριβό (δεν υποστηρίζεται)
    - Άρα στο S/W

```
use_count = (ref bit << n-1) | (use_count >> 1)
```

  - Γιατί shift δεξιά;
  - Ποια σελίδα διώχνω;
    - Αυτή με το μικρότερο use_count

Ακόμα μια Βελτίωση (ουφ…)

- Πότε ψάχνω για ελεύθερα πλαίσια;
  - Όταν χρειαστούν
    - Αργό (πρέπει και να γράψω τα περιεχόμενα του πλαισίου στο δίσκο, αν τα περιεχόμενά του έχουν αλλαχθεί)
  - Προληπτικά
    - Κράτησε μια «ρεζέρβα» από ελεύθερα πλαίσια
    - Χρησιμοποίησέ τα με σειρά FIFO
    - Μην ξεχάσεις όμως σε ποια σελίδα αντιστοιχεί κάθε ελεύθερο πλαίσιο
      - Αν ξαναχρειαστεί αυτή η σελίδα χρησιμοποίησε το αντίστοιχο πλαίσιο από τη ρεζέρβα
    - Πόσα πλαίσια στη ρεζέρβα;
      - Δείκτες (πάνω και κάτω)

Αντικατάσταση σε Επίπεδο Διεργασίας ή Συστήματος;

- Ως τώρα: Μία δεξαμενή πλαισίων για όλες τις διεργασίες (επίπεδο συστήματος)
  - Ποιο πλαίσιο να αντικαταστήσω;
    - Το παλαιότερο στο σύστημα
  - Καλό;
    - Ναι, οι διαμοιραζόμενοι πόροι αξιοποιούνται γενικά καλύτερα!
      - Π.χ. μία διεργασία θέλει το 70% της μνήμης, και μια άλλη το 20%. Κανένα πρόβλημα!
    - Και πώς προστατεύομαι από τους «κακούς»; (απομόνωση)
      - Αν μια διεργασία πάει να προσπελάσει έναν πίνακα στο μέγεθος της κύριας μνήμης;

Τοπική Αντικατάσταση (Επίπεδο Διεργασίας / Χρήστη)

- Ίδια λογική
  - Κάθε διεργασία / χρήστης έχει μια δεξαμενή από πλαίσια
  - Ουκ επιθυμήσεις το πλαίσιον του πλησίον σου…
    - Κάθε διεργασία / χρήστης μπορεί να χρησιμοποιήσει μόνο τα πλαίσια στη δεξαμενή του.
  - Καλό;
    - Απομόνωση (δε μας ενοχλούν οι «κακοί»)
    - Τι γίνεται με τη συνολική αξιοποίηση των πόρων; (πλαισίων)
      - Πόσα πλαίσια σε κάθε διεργασία / χρήστη;
      - Και αν χρειαστεί περισσότερα / λιγότερα;
  - Μηχανισμός για την «αργή» μεταφορά πλαισίων μεταξύ διεργασιών / χρηστών
    - Πόσο αργή;
    - Πότε ενεργοποιείται;

Διαφορές VM και Cache;

- Χρονοβελτίωση από cache:

p: ποσοστό ευστοχίας στην cache

μέσος χρόνος προσπέλασης = p * (χρόνος προσπέλασης στην cache) +
(1-p) * (χρόνος προσπέλασης στο χαμηλότερο επίπεδο +
επιβάρυνση αστοχίας στην cache)

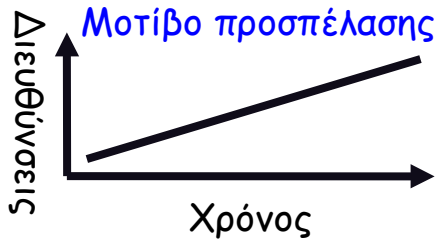
- TLB και cache σε ταχύτητα κοντά στου επεξεργαστή
  - Όχι πολύς χρόνος για κολπάκια
- Κόστος αστοχίας στην ιδεατή μνήμη
  - Κοντά στο κόστος προσπέλασης στο δίσκο (τεράστιο)
    - Συμφέρει να «πληρώσουμε» για πιο πολύπλοκες τεχνικές

Λυγισμός (Thrashing): Το Ξεμπρόστιασμα της Ιδεατής Μνήμης

- Λυγισμός: Οι διεργασίες χρειάζονται (πραγματικά) περισσότερη μνήμη από τη διαθέσιμη
  - Κάθε φορά που μια εκτελείται φέρνει σελίδες της στην κύρια μνήμη
    - Για να το κάνει αυτό διώχνει άλλες σελίδες που θα χρειαστούν σύντομα στο μέλλον
    - Οι διεργασίες τρώνε το χρόνο τους σε μεταφορές σελίδων
      - Τρέχει λίγο, σφάλμα σελίδας, μπλοκάρει για πολύ κ.ο.κ.
    - Χρησιμοποίηση I/O 100%
      - Όλα τα άλλα όμως κάαααθονται ☹
- Τι θέλαμε;
  - Ιδεατή μνήμη μεγάλη σαν το δίσκο και γρήγορη σαν τη φυσική
- Τι πετυχαίνουμε;
  - Μνήμη με χρόνο προσπέλασης ίσο με το χρόνο προσπέλασης του δίσκου ☹

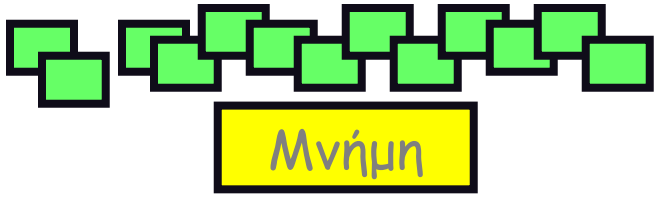
Πότε Συμβαίνει;

- Οι διεργασίες ζητούν «συχνά» σελίδες που δεν είναι στην κύρια μνήμη
  - Τρώνε περισσότερο χρόνο περιμένοντας για I/O παρά κάνοντας δουλειά
- Τρεις λόγοι
  - Κακό μοτίβο προσπέλασης. Μη επαναχρησιμοποίηση. Παρελθόν != Μέλλον



  - Επαναχρησιμοποίηση αλλά η διεργασία δε «χωράει»

  - Επαναχρησιμοποίηση, κάθε διεργασία χωράει, αλλά όλες μαζί δε χωράνε



Με Αριθμούς…

h: Ποσοστό ευστοχίας στη φυσική μνήμη
aat: Μέσος χρόνος προσπέλασης
mat: Χρόνος προσπέλασης στη μνήμη (~100 nsec)
dat: Χρόνος προσπέλασης στο δίσκο (~10 msec)
mo: Επιβάρυνση αστοχίας (έστω μικρή)

aat = h*mat + (1-h)*(dat+mo)

Αν h = 0.99 (μία στις 100 αναφορές πάει στο δίσκο)

aat = 0.99*100nsec + 0.01*10.000.000nsec ~ 100μsec

~ 1000 φορές πιο αργό από την προσπέλαση στη μνήμη !!!

Και τι Κάνω αν Συμβεί σε Εμένα;

- Οφείλεται σε 1 διεργασία;
  - Περιόρισε τη ζημιά (απομόνωση διεργασίας, φωτιά και τσεκούρι…)
- Οφείλεται γενικώς στο φορτίο του συστήματος;
  - Προσαρμόσου
    - Πόση μνήμη χρειάζεται κάθε διεργασία;
    - Μπορούμε να παίξουμε με τη χρονοδρομολόγηση;
    - Αν εμφανιστούν νέες διεργασίες «ρίξε πόρτα»…
  - Εναλλακτικές (κοινωνικές λύσεις):
    - Αγόρασε κι άλλη μνήμη
    - Βρες τον «αγενή» και βάλε τις φωνές

Πώς να Λύσω το Πρόβλημα;

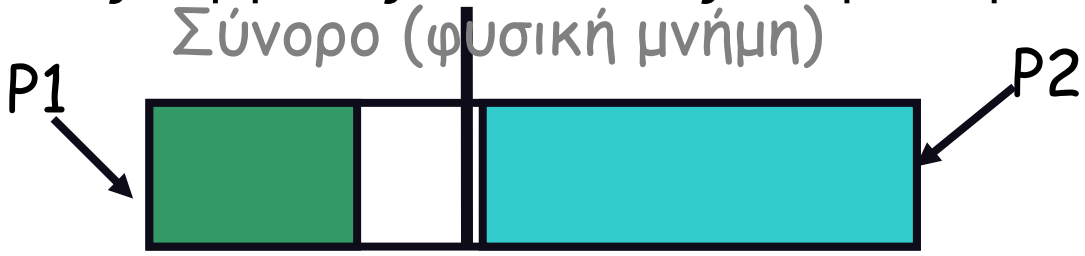
- Ανάλογα πώς το βλέπεις…
  - Δε με φτάνει η «cache» (κύρια μνήμη) – Η προσέγγιση του συνόλου εργασίας (λειτουργικού συνόλου / working set)
    - Πόσο μεγάλη «cache» χρειάζεται η διεργασία για να προχωρήσει;
    - … ή πόση μνήμη χρειάζεται η διεργασία για να προχωρήσει με «λογικό» ρυθμό; (σύνολο εργασίας / λειτουργικό σύνολο)
    - Τρέξε λοιπόν μόνο διεργασίες των οποίων οι απαιτήσεις είναι ικανοποιήσιμες!
  - Δεν αξιοποιώ τη CPU – Η προσέγγιση των σφαλμάτων σελίδας
    - Κακός λόγος εργασίας προς διαχείριση
    - Συχνότητα σφαλμάτων σελίδας (PFF): Σφάλματα σελίδας ανά εντολή
      - Αν το PFF ανέβει πάνω από κάποιο όριο η διεργασία χρειάζεται παραπάνω μνήμη
        - Αν δεν υπάρχει μνήμη, βγάλτη στο δίσκο
      - Αν το PFF πέσει κάτω από κάποιο όριο, μπορείς να πάρεις μνήμη από τη διεργασία (δεν την αξιοποιεί)

Το Σύνολο Εργασίας (Working Set / Λειτουργικό Σύνολο)

- Μπορώ να ξέρω…
  - … πόσες σελίδες θα χρειαστεί η διεργασία για να μη φτάσει σε λυγισμό;
    - Και ποιες αν είναι δυνατό…
    - Τη γυάλινη σφαίρα!!!
- Τρυκ:
  - Το σύνολο εργασίας: Οι σελίδες τις οποίες «άγγιξε» η διεργασία τις τελευταίες Τ μονάδες χρόνου
    - Τ: Παράμετρος
- Χρήσεις:
  - Αντικατάσταση σελίδων: Αντικατάστησε σελίδες εκτός του συνόλου εργασίας
  - Χρονοδρομολόγηση: Τρέξε μια διεργασία μόνο αν μεγάλο μέρος του συνόλου εργασία της είναι στη μνήμη
  - Διαμοίραση cache: Δώσε σε κάθε διεργασία αρκετή cache για να βάλει το σύνολο εργασίας της

Συλλογή Πλαισίων ανά Διεργασία (Μνήμες από το Παρελθόν…)

- Κάθε διεργασία έχει ένα σύνολο πλαισίων
- Για την ικανοποίηση ενός σφάλματος σελίδας μπορεί να χρησιμοποιηθεί μόνο ένα πλαίσιο από αυτά που ανήκουν στη διεργασία
- Απομονώνει τις διεργασίες / Μειώνει τις αλληλεπιδράσεις μεταξύ τους



- Πόσο μεγάλο σύνολο πλαισίων;
  - Περίπου όσο και το σύνολο εργασίας της
- Αποτέλεσμα:
  - Απομόνωση
  - Καλή αξιοποίηση πόρων

Να και η Χρονοδρομολόγηση: Το Σύνολο «Ισορροπίας» (1/2)

- Χωράει το άθροισμα των συνόλων εργασίας όλων των διεργασιών στη μνήμη;
  - Ναι: OK, όλα όπως παλιά…
  - Όχι: Μην τις τρέχεις όλες μαζί, χώρισέ τις σε 2 ομάδες:
    - Ενεργές: Το σύνολο εργασίας στην κύρια μνήμη
    - Ανενεργές: Το σύνολο εργασίας συνειδητά εκτός μνήμης
  - Σύνολο ισορροπίας:
    - Ένωση (άθροισμα) των συνόλων εργασίας όλων των ενεργών διεργασιών

Να και η Χρονοδρομολόγηση: Το Σύνολο «Ισορροπίας» (2/2)

- Χρονοδρομολογητής 1ου επιπέδου:
  - Μετακίνησε διεργασίες από το ενεργό στο ανενεργό σύνολο όσο σύνολο ισορροπίας <= κύρια μνήμη
  - Πρέπει όμως και οι ανενεργές που και που να γίνονται ενεργές (και αντίστροφα)
    - Αν γίνεται πολύ συχνά;
    - Αν γίνεται πολύ σπάνια;
- Χρονοδρομολογητής 2ου επιπέδου:
  - Επιλέγει, όπως πριν, εργασίες για εκτέλεση από το ενεργό μόνο σύνολο
- Το σύνολο ισορροπίας πρέπει να ανανεώνεται διαρκώς
  - Το σύνολο εργασίας κάθε διεργασίας ανανεώνεται διαρκώς

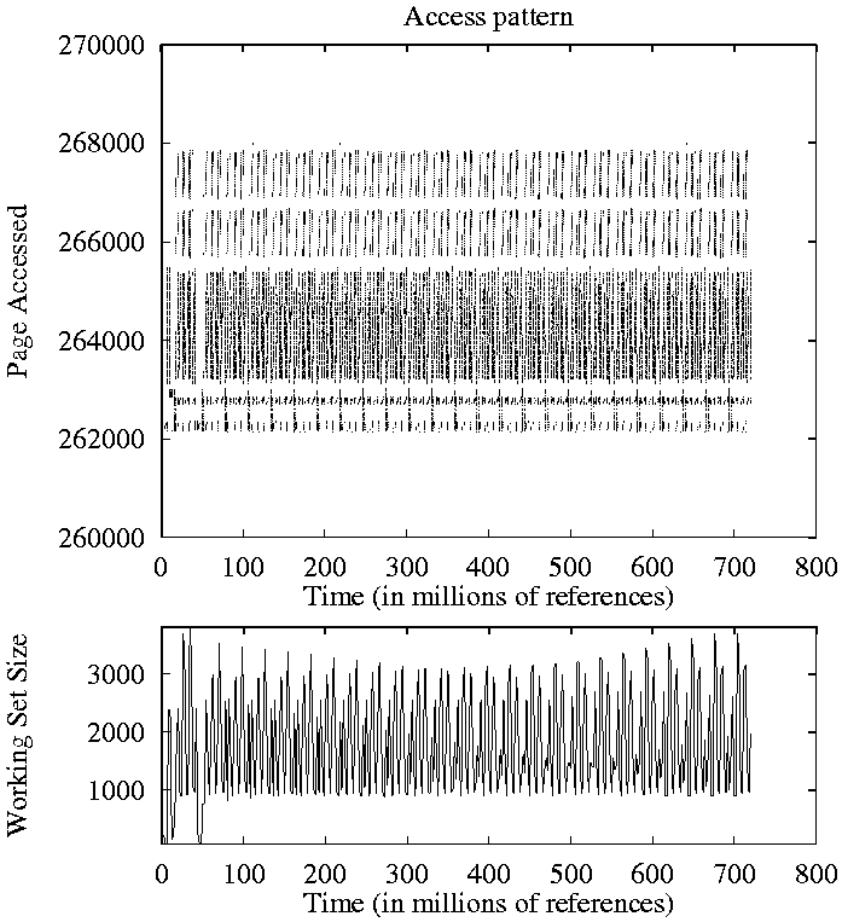
Και πώς Υπολογίζεται το Σύνολο Εργασίας;

- Άεργος χρόνος (idle time) σε κάθε πλαίσιο σελίδας
  - Χρόνος υπολογισμού από την τελευταία προσπέλαση στη σελίδα
    - Γιατί όχι πραγματικός χρόνος από την τελευταία προσπέλαση;
  - Χρόνος > Τ: Σελίδα εκτός του συνόλου εργασίας
- Υπολογισμός:
  - Διάτρεξη όλων των σελίδων της διεργασίας στην κύρια μνήμη
    - Bit χρήσης 1; Κάνε το 0 και κάνε 0 τον άεργο χρόνο της σελίδας
    - Bit χρήσης 0; Πρόσθεσε το διάστημα από την προηγούμενη διάτρεξη στον άεργο χρόνο της σελίδας
  - Διάτρεξη τυπικά ανά κάποια δευτερόλεπτα
    - Το Τ είναι της τάξης του/ων λεπτού/ών

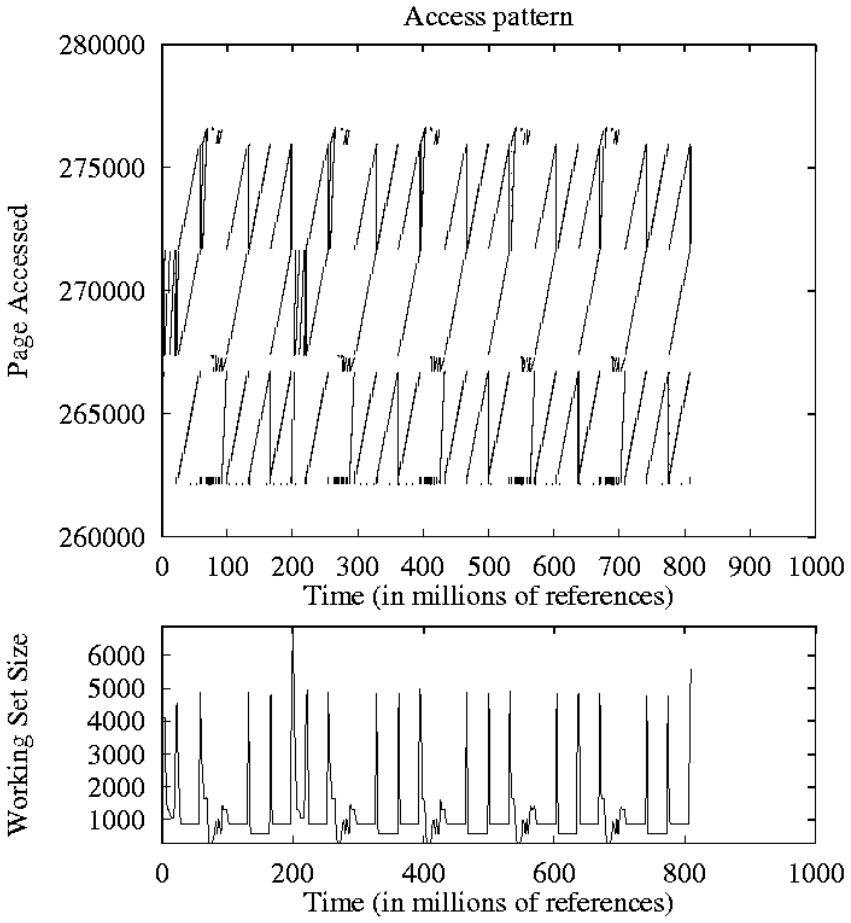
Προβλήματα (Κλασσικά…)

- Πόσο είναι το Τ;
  - Μαύρη μαγεία…
  - Αν είναι πολύ μικρό;
  - Πολύ μεγάλο;
  - Πώς το διαλέγουμε;
    - Δοκιμή και λάθος…
    - Ευτυχώς συνήθως η συμπεριφορά δεν είναι ευαίσθητη στην τιμή του Τ
- Ποιες διεργασίες πρέπει να συνεισφέρουν περισσότερο στο σύνολο ισορροπίας;
  - «Μεγάλες», ώστε να τελειώνουν γρήγορα;
  - «Μικρές» ώστε να τρέχουμε πολλές «μαζί»;
- Τι γίνεται με τις κοινόχρηστες σελίδες;
  - Πού χρεώνονται;

Και Ολίγη Τέχνη… : Σύνολα Εργασίας Πραγματικών Προγραμμάτων

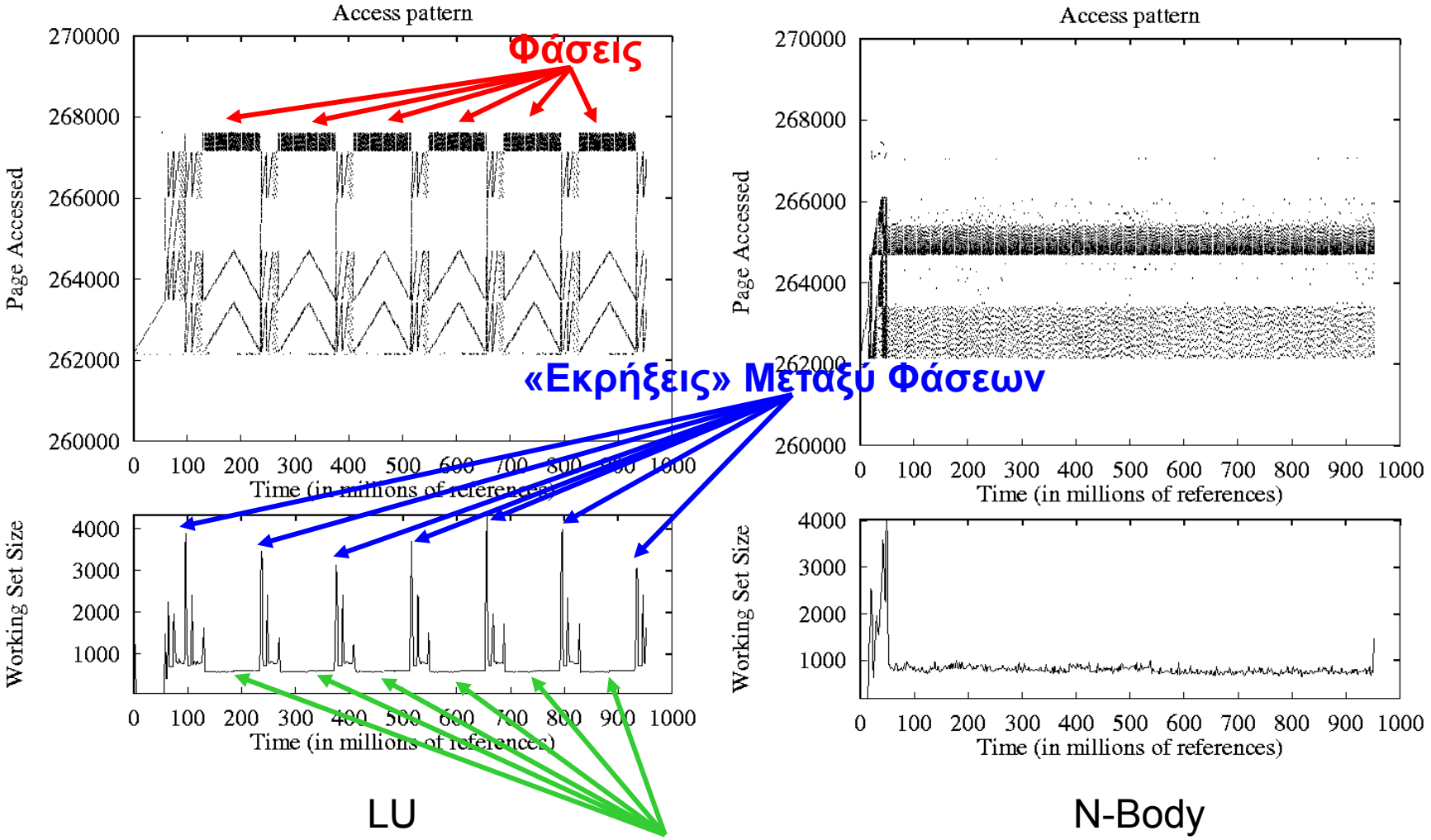


BT



MG

Σύνολα Εργασίας Πραγματικών Προγραμμάτων

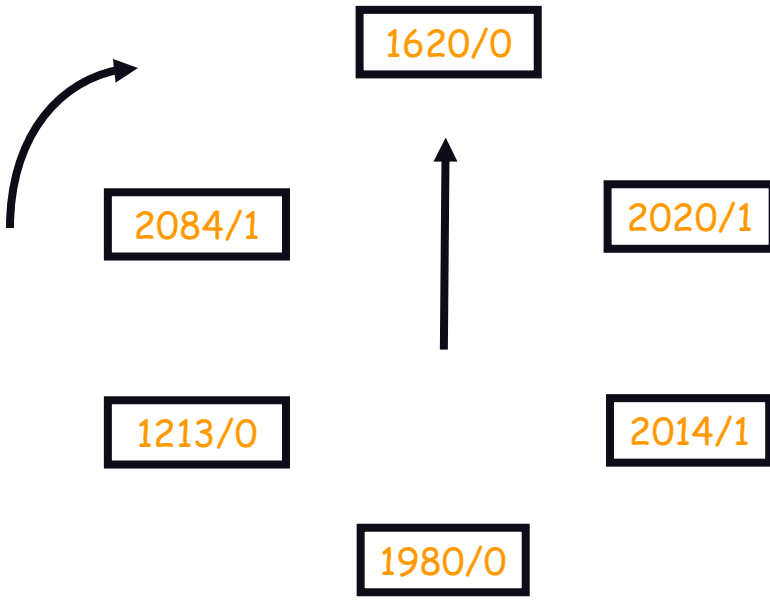


Πίσω στη Σελιδοποίηση (Πολιτικές Αντικατάστασης)

- Αλγόριθμος λειτουργικού συνόλου:
  - Αντικατέστησε κατά προτίμηση σελίδες που δεν είναι στο λειτουργικό σύνολο
    - Από αυτές διάλεξε αυτή που έχει να χρησιμοποιηθεί για το μεγαλύτερο διάστημα (μεγάλος άεργος χρόνος)
  - Πρόβλημα;
    - Πρέπει να σαρωθούν κάθε φορά όλες οι σελίδες…
  - Λύση;
    - Θυμάστε τον αλγόριθμο ρολογιού;
    - Τι έκανε;

Αλγόριθμος WSClock για Αντικατάσταση Σελίδας

- Δεδομένα
  - Άεργος χρόνος
  - Bit χρήσης
  - Bit μεταβολής (δε φαίνεται)
- Αλγόριθμος
  - Ψάξε για σελίδες με bit χρήσης 0
    - Όταν περνάς από κάποια με bit χρήσης 1, κάνε το 0 και προχώρα στην επόμενη
  - Από αυτές βρες κάποια με άεργο χρόνο > T
    - Αν bit μεταβολής = 1, προγραμμάτισέ τη για εγγραφή στο δίσκο και πήγαινε στην επόμενη
      - Βάλε και κάποιο όριο ανά κύκλο για να μη συμφορηθεί ο δίσκος
  - Αν φτάσεις να σαρώσεις όλο το ρολόι και ακόμα ψάχνεις…
    - Αν έχει προγραμματιστεί έστω μία για εγγραφή, keep spinning…
    - Αν όχι, διάλεξε μία του λειτουργικού συνόλου στην «τύχη» με bit μεταβολής 0
    - Αν δεν υπάρχει, μία με bit μεταβολής 1



Πόσο Σημαντικό το Λειτουργικό Σύνολο;

- Όχι τόσο πλέον…
  - Έγιναν οι σχεδιαστές ΛΣ καλύτεροι;
    - Όχι, sorry…
  - Μην ειν' το υλικό;
    - Προφανές:
      - Περισσότερη, φθηνότερη μνήμη
    - Λιγότερο προφανές:
      - Γρηγορότερη CPU, διεργασίες τελειώνουν γρηγορότερα…
      - … άρα επιστρέφουν και τη μνήμη γρηγορότερα!
- Πάντα υπάρχουν αδηφάγες εφαρμογές πάντως…