

(-1)

av ena fru

0

Frames: 10

Page requests: 50

Page faults: 26

Diskwrites: 3.

$\psi_{sim}$  Erv 30 12 0 200

Frames: 30

Page requests: 200

Page 90/10: 82

Piskovites: 16

./psim br 10 15 0 200

Frames: 10

Payerequest: 200.

Page farts: 103

Disk writes: 24.

lipsum aus 25 20 10 300

Frames: 25

Payerequestfs:300.

Page 15: 11.

Disk writes: 47.

/psim WS 100 25 15 500

Frames: 100    Page requests 500    Page faults: 102

Disk writes: 18.

(-4).

Αν η σελίδα δεν είναι στην μνήμη. Καταργείσεται  
Αγ.ταλίστ.

Αν υπάρχει διαθεσιμο frame ανά την Βολή  
στον IPT (και συνεχώς στην μνήμη).

Αν δεν υπάρχει διαθεσιμο frame.

πρόσφατος πρέπει να δημιουργηθεί ένα ke  
την Πύλη του IPT πρόσφατος αφορούν  
στο IPT της σελίδα που λειτουργεί πιο αλκα  
και ελεγχώμενο το frame της

→ GetVictim

→ Getframe

→ Fillframe // with the new Page.

αν ο indicator της ήταν W core  
diskwriteit;

### Now working Set:

Παράγει ένα Page αν στο το Page.  
αυτή σε διαδοχικά διαγράφα στα ποτήρια που  
τώρα τώρα θα πρέπει να ελεγχώ και  
να επαναφέρω το working set της στο  
main Memory (SetBack to Mem)  
→ κάνω update το ws.

### Αν το Page είναι στο Main Memory

αλλά κανονικά ένα update στα στοιχεία του. (w.e)

αλλά και στο δέντρο του στο recent list - working set

Αν το Page δεν είναι στο Main Memory (Αγ.ταλίστ)

→ Αν δεν είναι: Προσέχει και ελεγχώμενο

ένα frame ~~στο~~ στο στο Προσέχει

σελίδα που δεν υπάρχει στο working set.

Τέλος τονίζω ότι η σελίδα σε στο το  
frame.

## InvPgT.h + InvPg.cpp

- Η δομή Pentry αναπαριστά τα addresses (pid, PageNum, dirty (b, w))
- Η δομή InvPgT: το Inverted PageTable.

getFrame: given a page returns the frame in which this page is

getFreeframe: returns a free frame.

FillFrame: Puts a Page in a frame.

## WS.h, WS.cpp:

- Η δομή workingSet:

Διατηρεί δύο λίστες από διεύθυνσεις:

recentList: Διατηρεί διεύθυνσεις που όσο πιο πρόσφατη είναι τόσο πιο πρόσφατη γίνεται.

workingSet: Διατηρεί τις διεύθυνσεις που ανήκουν στο workingSet.

- Η δομή WS-Handler διαχειρίζεται τις workingSet διαχειρίζεται τα workingSet καθώς έχουμε περισσότερες από μία διεργασίες.

Working Set:   
 ↳ An δεν υπάρχει χώρος   
 ↳ Wsupdate: αφαιρεί την least-recently σελίδα από το working set αν δεν έχει βρεθεί πιο πρόσφατο φυσικό   
 ↳ Αν υπάρχει χώρος και η σελίδα δεν υπάρχει στο working set → Βρίσκει.

WS-Handler: Give.Pr-TakeSet

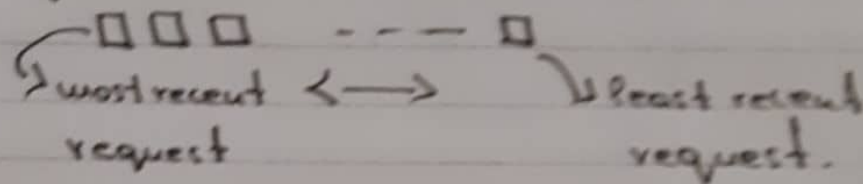
↳ Given a process id returns its working set.



(3)

### LRU.h + LRU.cpp.

Η δομή LRU διατηρεί μια λίστα των σελίδων αναπαριστών την "χρονική ροή" ή την σειρά πρόσβασης οι σελίδες



→ GetVictim: Προσφέρει "Σελίδα θύρα" είναι αυτή στο τέλος της λίστας αφού βίβλη για LRU.

→ MakeItUL <sup>Use it most recently used</sup> αναφορά σε σελίδα της λίστας από όπου να πάει στην αρχή της.

### MemSim.h + MemSim.h.

{ MemSim: Η δομή ~~αποτελεί~~ Υπερδομή για το Simulation της λίστας → έχει τις features αναμενόμενες → τον IDT.  
→ όλα και τις δομές που χρειάζονται για τους αλγόριθμους LRU και WS.

→ Παιχνάει τις αναφορές getRef καινοτομία οπότε χρειάζεται μια σύλληψη λίστας των αρχικών και συνεπώς των βαν αναφορών, και τρέφει είτε τον LRU είτε τον WS.

LRU algo: → Αν η σελίδα υπάρχει ήδη στην λίστα από αναμνήσε λα δεδομένα → ολικά θεωρείται την <sup>πιο</sup> πρόσφατη και αλλάζει τον indicator (Q<sub>ref</sub>) αν είναι διαφορετικός

→