Κεφάλαιο

Διαχείριση Μνήμης στα Λειτουργικά Συστήματα

Σκοπός του κεφαλαίου αυτού είναι να σου γνωρίσει τον τρόπο με τον οποίο ένα λειτουργικό σύστημα διαχειρίζεται τη μνήμη, ώστε οι διεργασίες να χρησιμοποιούν τη μνήμη με απλό τρόπο, και η απόδοση του συστήματος να είναι βέλτιστη.

Όταν ολοκληρώσεις το κεφάλαιο αυτό, θα μπορείς:

- Να περιγράφεις τον τρόπο που το ΛΣ εκμεταλλεύεται τη δευτερεύουσα μνήμη για να ενισχύσει το μέγεθος της κύριας.
- Να εξηγείς τι είναι εικονική μνήμη και πώς αντιστοιχίζεται στη φυσική μνήμη.
- Να απαριθμείς τα προβλήματα που καλείται να λύσει η διαχείριση μνήμης του ΛΣ και να περιγράφεις τις λύσεις που δίνονται στα προβλήματα αυτά.

Μαθήματα

Μάθημα 9.1 Διαχείριση Μνήμης

Σκοπός του μαθήματος

Σκοπός του μαθήματος αυτού είναι να εξηγήσει τον τρόπο που ένα ΛΣ χειρίζεται τις ανάγκες των διεργασιών για μνήμη.

Τι θα μάθεις;

Όταν ολοκληρώσεις το μάθημα αυτό, θα μπορείς:

- *Να απαριθμείς τα είδη μνήμης σε ένα υπολογιστή*
- *Να εξηγείς πώς το ΛΣ εκμεταλλεύεται τη δευτερεύουσα μνήμη για να ενισχύσει το μέγεθος της κύριας*
- *Να ορίζεις την εικονική μνήμη*
- *Να περιγράφεις πώς διαχειριζόμαστε την εικονική μνήμη και πώς την αντιστοιχίζουμε στη φυσική μνήμη*

Η μνήμη ενός ανθρώπου τον βοηθά να συγκρατεί όλες τις χρήσιμες πληροφορίες που έχει συγκεντρώσει με τη βοήθεια των αισθήσεών του από τον κόσμο που τον περιβάλλει. Αντίστοιχα ένας υπολογιστής πρέπει να διαθέτει έναν τρόπο να καταγράφει διάφορα στοιχεία: είτε τα προγράμματα που εκτελεί, είτε τα δεδομένα των προγραμμάτων αυτών.

Κάθε μονάδα ενός υπολογιστή που χρησιμεύει για τη μόνιμη ή προσωρινή αποθήκευση δεδομένων ανήκει στη *μνήμη* (memory) του υπολογιστή.

Οι μονάδες μνήμης ανήκουν σε δυο αρκετά διαφορετικές κατηγορίες ανάλογα με το αν αποθηκεύουν τα δεδομένα προσωρινά ή μόνιμα:

➲ Στην *κύρια* ή *κεντρική μνήμη* (main memory) του υπολογιστή αποθηκεύονται τα προγράμματα που εκτελεί η ΚΜΕ και τα δεδομένα για τα προγράμματα αυτά. Η αποθήκευση αυτή είναι προσωρινή, και διαρκεί όσο και η λειτουργία του υπολογιστή. Όταν κλείσει ο υπολογιστής τα δεδομένα αυτά χάνονται. Επειδή η κύρια μνήμη των υπολογιστών είναι έτσι οργανωμένη ώστε να μπορεί να προσπελαστεί άμεσα οποιαδήποτε θέση της, αναφέρεται ως *μνήμη τυχαίας προσπέλασης* (Random Access Memory) και αποκαλείται RAM[1].

Συμπλήρωμα της κύριας μνήμης είναι η *μνήμη ROM* (Read Only Memory), στην οποία είναι καταγραμμένα μόνιμα από τον κατασκευαστή του υπολογιστή ορισμένα βασικά προγράμματα ή μικρά τμήματα του ΛΣ.

[1] Οι όροι «κύρια μνήμη» και «RAM» δεν είναι εξ ορισμού ταυτόσημοι, αλλά έχει καθιερωθεί η από κοινού χρήση τους για την περιγραφή της κύριας μνήμης.

➲ Στη *δευτερεύουσα* ή *περιφερειακή μνήμη* (secondary memory) τα δεδομένα αποθηκεύονται μόνιμα. Τα δεδομένα που θα καταγραφούν στη δευτερεύουσα μνήμη δε χάνονται όταν κλείσει ο υπολογιστής, και είναι διαθέσιμα την επόμενη φορά που θα λειτουργήσει. Μονάδες δευτερεύουσας μνήμης είναι οι σκληροί δίσκοι, οι εύκαμπτοι δίσκοι, τα CD ROM κλπ.

> Η φυσική μνήμη του ζαχαροπλάστη είναι η κύρια μνήμη του, ενώ δευτερεύουσα μνήμη για αυτόν είναι ένα σημειωματάριο με αριθμημένες σελίδες που περιέχει τις συνταγές του ζαχαροπλάστη και λευκές σελίδες για τις σημειώσεις του. Όταν ο ζαχαροπλάστης διαβάζει μια συνταγή και τη θυμάται για λίγη ώρα, την μεταφέρει στην κύρια μνήμη του από τη δευτερεύουσα. Επίσης θυμάται διάφορες πληροφορίες για την πρόοδο των συνταγών. Επειδή δεν μπορεί να θυμάται όλες τις συνταγές που εκτελεί ταυτόχρονα και όλες τις σχετικές πληροφορίες, όταν πρόκειται να διαβάσει μια νέα συνταγή καταγράφει στο σημειωματάριο τις πληροφορίες που χρειάζεται για κάποιες άλλες (όπως π.χ. σε ποιο σημείο της εκτέλεσής τους βρίσκονται) ώστε να μην τις ξεχάσει.
>
> Όταν ψάχνει κάποια πληροφορία στο σημειωματάριό του, ο ζαχαροπλάστης δε διαβάζει μία σελίδα για να διαπιστώσει αν είναι αυτή που θέλει, αλλά τη βρίσκει απευθείας από τον αριθμό της σελίδας της, το όνομα του πελάτη ή το όνομα της συνταγής.

Στην πράξη έχει καθιερωθεί η χρήση του όρου μνήμη για την κύρια μνήμη, ενώ οι μονάδες δευτερεύουσας μνήμης αναφέρονται με το όνομά τους (π.χ. δίσκος). Έτσι, στη συνέχεια όταν αναφερόμαστε στη μνήμη του υπολογιστή θα εννοούμε την κύρια μνήμη.

Στη συντριπτική πλειοψηφία των σύγχρονων υπολογιστών, τα προγράμματα που εκτελεί ο υπολογιστής και τα δεδομένα που χρειάζονται ή δημιουργούν τα προγράμματα αυτά είναι καταχωρημένα στη μνήμη με τον ίδιο ακριβώς τρόπο. Το ΛΣ κρατά πληροφορίες για τη μνήμη, ώστε να μπορεί να διακρίνει σε ποιες θέσεις της βρίσκονται προγράμματα και σε ποιες βρίσκονται δεδομένα.

Για να εκτελεστεί μια διεργασία πρέπει τόσο το πρόγραμμά της όσο και τα δεδομένα της να έχουν καταχωρηθεί σε κάποιες θέσεις της κύριας μνήμης. Για να εκτελεστεί η διεργασία, είναι απαραίτητο να χρησιμοποιηθούν το πρόγραμμα και τα δεδομένα. Οι αντίστοιχες θέσεις μνήμης διαβάζονται και γράφονται, γίνονται λοιπόν *προσπελάσεις* (accesses) σε αυτές.

Είναι φανερό ότι ο τρόπος με τον οποίο το ΛΣ διαχειρίζεται την κύρια μνήμη ενός υπολογιστή επηρεάζει πολύ τον τρόπο λειτουργίας του, αλλά και την απόδοσή του. Όταν σχεδιάζεται ένα ΛΣ, η διαχείριση της κύριας μνήμης είναι ένα πεδίο σοβαρών σχεδιαστικών αποφάσεων:

☒ Κάθε στιγμή θα χρησιμοποιούν τη μνήμη μια ή περισσότερες διεργασίες;

☒ Αν η μνήμη χρησιμοποιείται από πολλές διεργασίες ταυτόχρονα, θα είναι χωρισμένη σε τμήματα ίσου ή διαφορετικού μεγέθους;

☒ Ο χωρισμός της μνήμης σε τμήματα θα είναι μόνιμος ή θα αλλάζει ανάλογα με τις διεργασίες που εκτελούνται;

☒ Το πρόγραμμα και τα δεδομένα κάθε διεργασίας θα είναι αποθηκευμένα σε ένα συνεχόμενο τμήμα της μνήμης ή θα μπορούν να κατανεμηθούν ανάμεσα σε πολλά διάσπαρτα τμήματα;

Στατική και δυναμική κατανομή της κύριας μνήμης

Για να μπορούν πολλές διεργασίες να εκτελούνται ταυτόχρονα σε ένα υπολογιστή, πρέπει η κύρια μνήμη του να είναι διαιρεμένη σε τμήματα και σε καθένα από αυτά να βρίσκεται το πρόγραμμα ή τα δεδομένα μίας διεργασίας. Με τον τρόπο αυτό, η ΚΜΕ μπορεί να εναλλάσσεται μεταξύ των διεργασιών, διαβάζοντας από και γράφοντας σε διαφορετικές θέσεις της μνήμης κάθε φορά.

Επίσης, για να λειτουργεί ο υπολογιστής ομαλά, κάθε διεργασία πρέπει να έχει πρόσβαση μόνο σε εκείνα τα τμήματα της μνήμης που της αντιστοιχούν και όχι στα τμήματα άλλων διεργασιών. Ιδιαίτερα προστατευμένα πρέπει να είναι τα τμήματα της μνήμης όπου είναι αποθηκευμένος ο κώδικας και τα δεδομένα του ίδιου του ΛΣ.

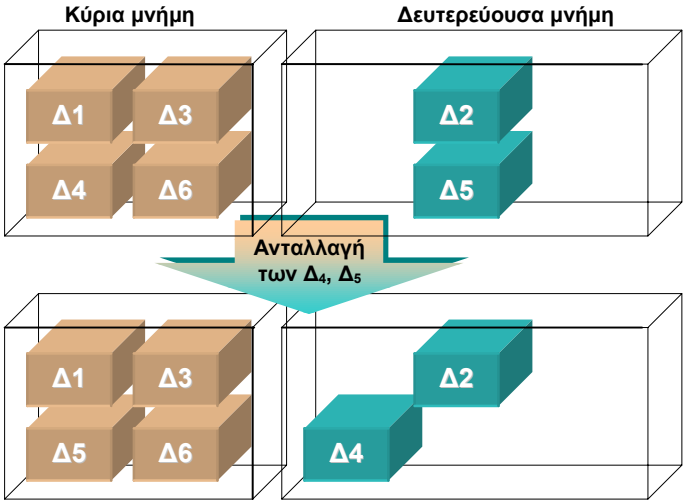
Η κατανομή της κύριας μνήμης μπορεί να είναι *στατική* (static allocation) ή *δυναμική* (dynamic allocation).

Στη **στατική κατανομή** η μνήμη είναι χωρισμένη από το ΛΣ σε προκαθορισμένα τμήματα πριν από την εκτέλεση των διεργασιών. Όταν ένα πρόγραμμα πρόκειται να εκτελεστεί, το ΛΣ επιλέγει ένα τμήμα μνήμης που καλύπτει τις ανάγκες της νέας διεργασίας σε μνήμη και της το παραχωρεί. Η στατική κατανομή μνήμης είναι απλή και λύνει το πρόβλημα της προστασίας της μνήμης. Όμως, τα τμήματα μνήμης δε χρησιμοποιούνται πλήρως από τις διεργασίες και έτσι μένουν αναξιοποίητα. Επιπλέον οι διεργασίες δεν μπορούν να χρησιμοποιήσουν μνήμη από κοινού, κάτι που είναι χρήσιμο όταν πολλές διεργασίες εκτελούν το ίδιο πρόγραμμα.

Στη **δυναμική κατανομή** το ΛΣ κάνει τη διαίρεση της μνήμης κάθε φορά που μία διεργασία αρχίζει να εκτελείται και της διαθέτει ακριβώς όση μνήμη χρειάζεται. Η δυναμική κατανομή της μνήμης είναι ευέλικτη και μπορεί να αξιοποιήσει μικρές περιοχές της μνήμης. Με τον τρόπο αυτό βελτιώνεται το πρόβλημα της αναξιοποίητης μνήμης, χωρίς όμως να λύνεται τελείως. Το βασικό μειονέκτημα της λύσης αυτής είναι η αυξημένη πολυπλοκότητα των μηχανισμών προστασίας της μνήμης.

Ανταλλαγή

Η δευτερεύουσα μνήμη είναι πολύ πιο αργή από την κύρια (π.χ. ένας σκληρός δίσκος είναι εκατό χιλιάδες φορές πιο αργός από την κύρια μνήμη)· συγχρόνως όμως είναι και αρκετά πιο φθηνή. Έτσι, οι περισσότεροι υπολογιστές διαθέτουν πολύ περισσότερο χώρο αποθήκευσης σε δευτερεύουσα μνήμη παρά σε κύρια (π.χ. εκατό φορές περισσότερο).



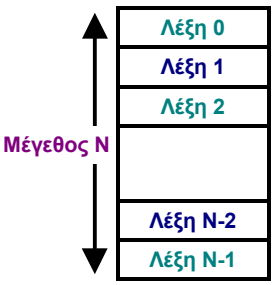
Όταν ένας υπολογιστής εξυπηρετεί ένα μεγάλο αριθμό διεργασιών, τα προγράμματα και τα δεδομένα όλων δε χωρούν στην κύρια μνήμη. Τότε επιστρατεύεται η δευτερεύουσα μνήμη για να βοηθήσει: τα προγράμματα και τα δεδομένα ορισμένων διεργασιών κρατούνται στη δευτερεύουσα μνήμη, και κάθε φορά που είναι η σειρά μιας τέτοιας διεργασίας να εκτελεστεί, τότε μόνο φορτώνονται στην κύρια μνήμη. Για να απελευθερωθεί όμως χώρος στην κύρια μνήμη για αυτά, πρέπει κάποια άλλη διεργασία να μεταφερθεί με τη σειρά της στη δευτερεύουσα μνήμη.

Η διαδικασία αυτή ονομάζεται *ανταλλαγή* (swapping) και δημιουργεί σοβαρές χρονικές καθυστερήσεις στην εκτέλεση των διεργασιών, εξαιτίας της συνεχούς λειτουργίας της δευτερεύουσας μνήμης. Ο χρόνος που απαιτεί μια διεργασία

για να εκτελεστεί είναι το άθροισμα του πραγματικού χρόνου εκτέλεσής της στην ΚΜΕ, του χρόνου που απαιτήθηκε για τις μεταφορές της μεταξύ κύριας και βοηθητικής μνήμης και του χρόνου για Ε/Ε. Για να είναι η λειτουργία του υπολογιστή αποδοτική απαιτείται πιο πολύπλοκη οργάνωση της ανταλλαγής δεδομένων μεταξύ κύριας και δευτερεύουσας μνήμης.

Εικονική μνήμη

Η κεντρική μνήμη του υπολογιστή αποτελείται από διαδοχικές θέσεις, με μέγεθος η κάθε μια π.χ. 16 ή 32 bits, που ονομάζονται *λέξεις* (words). Κάθε λέξη της μνήμης έχει τη δική της διεύθυνση, με την οποία αναφέρονται τα προγράμματα σε αυτή. Αν η μνήμη έχει μέγεθος Ν λέξεων, τότε η πρώτη λέξη έχει διεύθυνση 0, η δεύτερη έχει διεύθυνση 1 κ.ο.κ., ενώ η τελευταία λέξη έχει διεύθυνση Ν-1. Το σύνολο αυτών των Ν διευθύνσεων είναι σταθερό και ονομάζεται *χώρος φυσικών διευθύνσεων* (physical address space) ή *χώρος απολύτων διευθύνσεων* (absolute address space) ή *χώρος πραγματικών διευθύνσεων* (real address space).



Κάθε πρόγραμμα πρέπει να χρησιμοποιήσει διευθύνσεις για να αναφερθεί στα δεδομένα του, τα οποία βρίσκονται στη μνήμη. Τι γίνεται όμως αν οι διευθύνσεις των δεδομένων για δυο διεργασίες συμπίπτουν;

> Μια διεργασία δ_1 υπολογίζει ένα άθροισμα, το οποίο τοποθετεί στη θέση μνήμης με διεύθυνση 156. Μια άλλη διεργασία δ_2 ζητά διάφορους αριθμούς από το χρήστη, τους οποίους τοποθετεί επίσης στη θέση μνήμης με διεύθυνση 156. Οι δυο διευθύνσεις συμπίπτουν, γιατί τα δυο προγράμματα γράφτηκαν ανεξάρτητα το ένα από το άλλο. Αν και οι δυο διεργασίες εκτελούνται ταυτόχρονα, η θέση μνήμης 156 θα χρησιμοποιείται πότε από τη μια διεργασία και πότε από την άλλη. Έτσι όμως καμία από τις δυο δε θα χρησιμοποιεί τις σωστές τιμές, αφού σε τυχαίες χρονικές στιγμές επεμβαίνει η άλλη και τις αλλάζει, και τελικά καμία δε θα δώσει τα σωστά αποτελέσματα.

Τι γίνεται επίσης όταν η διαθέσιμη κύρια μνήμη δε χωρά όλες τις διεργασίες;

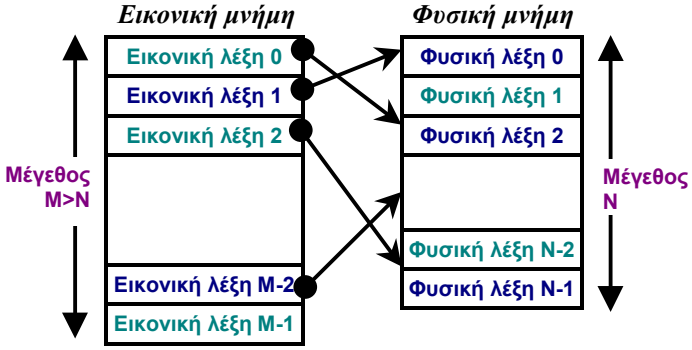
Εικονικές διευθύνσεις

Η πρώτη και πιο απλή σκέψη που μπορεί να γίνει για να λυθεί το πρόβλημα της επικάλυψης των διευθύνσεων είναι να γράφονται έτσι τα προγράμματα ώστε καθένα να χρησιμοποιεί διαφορετικές διευθύνσεις μνήμης. Με τον τρόπο αυτό δεν υπάρχει περίπτωση η μια διεργασία να επηρεάσει την άλλη. Αυτή η λύση όμως δεν είναι στην πράξη και πολύ αποτελεσματική για δυο λόγους:

- Τα προγράμματα γράφονται συνήθως ανεξάρτητα από διάφορους ανθρώπους, και δεν είναι δυνατό να βρεθεί ένας τρόπος συνεννόησης και συντονισμού τους.
- Το πλήθος των διαθέσιμων διευθύνσεων κύριας μνήμης είναι πεπερασμένο και μάλλον μικρό, οπότε αρκετά γρήγορα θα εξαντληθούν.

Θέλουμε λοιπόν μια λύση που να επιτρέπει στα προγράμματα να χρησιμοποιούν ελεύθερα οποιεσδήποτε διευθύνσεις, και να εξασφαλίζεται από το ΛΣ ότι δε θα υπάρχει επικάλυψη μεταξύ τους. Η λύση αυτή είναι εκείνη των *εικονικών διευθύνσεων* (virtual addresses): κάθε πρόγραμμα μεταφράζεται σε γλώσσα μηχανής σαν να έχει όλο το χώρο διευθύνσεων στη διάθεσή του, και αναφέρεται στα δεδομένα του χρησιμοποιώντας εικονικές -δηλαδή όχι πραγματικές- διευθύνσεις· όταν το πρόγραμμα φορτώνεται για εκτέλεση, το ΛΣ επιλέγει θέσεις μνήμης που είναι ελεύθερες και τις αντιστοιχίζει στις εικονικές διευθύνσεις.

Κατά την εκτέλεση του προγράμματος, κάθε αναφορά σε μια εικονική διεύθυνση μεταφράζεται από το ΛΣ στην αντίστοιχη φυσική διεύθυνση, η οποία τελικά προσπελάζεται. Η διαδικασία αυτή της μετάφρασης εικονικής διεύθυνσης σε φυσική συνήθως γίνεται πολύ γρήγορα, γιατί οι ΚΜΕ περιέχουν ειδικά κυκλώματα για αυτή.



Ο *χώρος εικονικών διευθύνσεων* (virtual address space), που πολλές φορές αναφέρεται και ως *εικονική μνήμη* (virtual memory), έχει συνήθως μεγαλύτερο μέγεθος από το χώρο φυσικών διευθύνσεων για να έχουν οι διεργασίες περισσότερο «χώρο μνήμης» στη διάθεσή τους. Οι εικονικές διευθύνσεις που δεν έχουν αντίστοιχες στη φυσική μνήμη, συνήθως αντιστοιχίζονται σε κάποια διεύθυνση της δευτερεύουσας μνήμης. Με τον τρόπο αυτό η δευτερεύουσα μνήμη συμβάλλει στην αύξηση της διαθέσιμης κύριας μνήμης του συστήματος.

Ο τρόπος αντιστοίχισης των διευθύνσεων για διαφορετικά προγράμματα μπορεί να γίνει με διάφορους τρόπους, δηλαδή με διάφορες στρατηγικές διαχείρισης εικονικής μνήμης. Οι βασικές στρατηγικές είναι δυο, η *σελιδοποίηση* (paging) και η *κατάτμηση* (segmentation), και υπάρχει ένας συνδυασμός των δύο, η *κατατμημένη σελιδοποίηση* (segmented paging). Αυτές οι στρατηγικές έχουν την ιδιότητα να επιτρέπουν πολλές διεργασίες να χρησιμοποιούν από κοινού μνήμη, όπως π.χ. διεργασίες που εκτελούν ταυτόχρονα το ίδιο πρόγραμμα χρησιμοποιούν από κοινού τη μνήμη στην οποία είναι αυτό αποθηκευμένο.

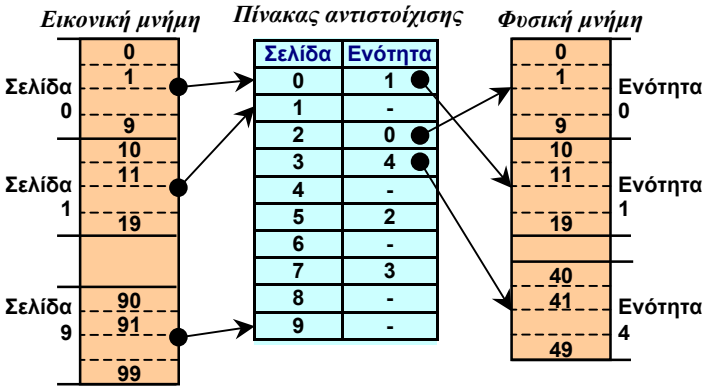
Διαχείριση εικονικής μνήμης με σελιδοποίηση

Στη μέθοδο της σελιδοποίησης, η εικονική μνήμη διαιρείται σε ίσα και συνεχόμενα μέρη, τα οποία ονομάζονται *σελίδες* (pages). Με τον ίδιο τρόπο διαιρείται και η φυσική μνήμη σε *ενότητες* (blocks, frames). Το μέγεθος της ενότητας είναι ίδιο με αυτό της σελίδας· έτσι μια σελίδα της εικονικής μνήμης και όλες οι διευθύνσεις που αυτή περιέχει αντιστοιχούν ακριβώς σε μια ενότητα της φυσικής μνήμης και τις διευθύνσεις της.

Το ΛΣ κρατά ένα πίνακα αντιστοίχισης σελίδων και ενοτήτων. Για κάθε σελίδα της εικονικής μνήμης φαίνεται στον πίνακα αυτό η ενότητα της φυσικής μνήμης στην οποία αντιστοιχεί ή, αν δεν αντιστοιχεί σε κάποια ενότητα, σημειώνεται ότι η σελίδα βρίσκεται στη δευτερεύουσα μνήμη.

Για να μεταφραστεί μια εικονική διεύθυνση (ΕΔ), πρώτα βρίσκεται η σελίδα (Σ) στην οποία ανήκει. Το ΛΣ αναζητά τη σελίδα αυτή στον πίνακα αντιστοίχισης, και βρίσκει ότι αντιστοιχεί στην ενότητα Ε. Η φυσική διεύθυνση (ΦΔ) θα ανήκει στην ενότητα αυτή, και θα βρίσκεται σε αντίστοιχη θέση με αυτή της ΕΔ μέσα στη σελίδα Σ.

Σε έναν υπολογιστή η εικονική μνήμη έχει μέγεθος 100 (εικονικές διευθύνσεις από 0 ως 99) και χωρίζεται σε 10 σελίδες ενώ η φυσική μνήμη έχει μέγεθος 50 (φυσικές διευθύνσεις από 0 ως 49) και διαιρείται σε 5 ενότητες. Το μέγεθος σελίδων και ενοτήτων είναι 10. Έτσι π.χ. οι εικονικές διευθύνσεις από 0 ως 9 ανήκουν στη σελίδα με αριθμό 0, ενώ οι φυσικές διευθύνσεις από 30 ως 39



ανήκουν στην ενότητα με αριθμό 3.

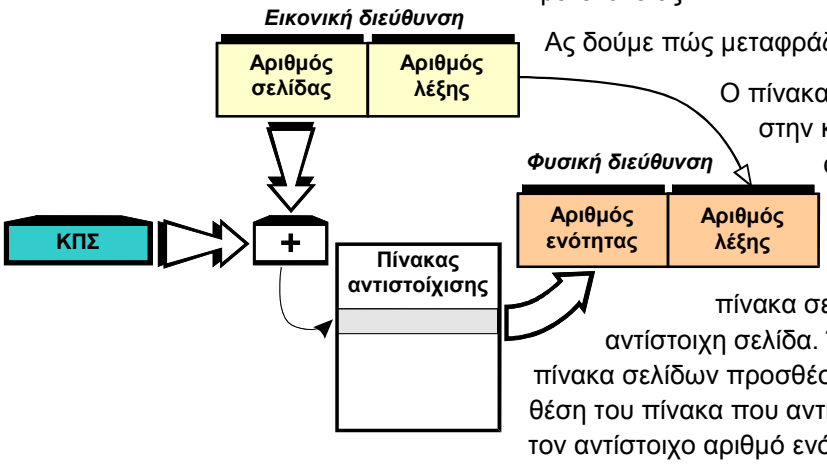
Στον πίνακα αντιστοίχισης του σχήματος, η σελίδα 0 αντιστοιχεί στην ενότητα 1, δηλαδή οι εικονικές διευθύνσεις 0-9 αντιστοιχούν στις φυσικές διευθύνσεις 10-19. Η εικονική διεύθυνση 2 τελικά θα μεταφραστεί στη φυσική διεύθυνση 12.

Παρατηρούμε ότι κάθε διεύθυνση μπορεί να χωριστεί σε δυο τμήματα: οι δεκάδες της διεύθυνσης δείχνουν τη σελίδα ή την ενότητα στην οποία ανήκει η διεύθυνση, και οι μονάδες δείχνουν τη θέση της διεύθυνσης μέσα στη σελίδα ή την ενότητα. Έτσι για να υπολογίσουμε τη φυσική διεύθυνση ΦΔ από την εικονική διεύθυνση ΕΔ, ακολουθούμε τα εξής βήματα:

1. Βρίσκουμε τον αριθμό σελίδας Σ, που είναι οι δεκάδες της ΕΔ.
2. Βρίσκουμε τη θέση Θ της διεύθυνσης μέσα στη σελίδα, που είναι οι μονάδες της ΕΔ.
3. Από τον πίνακα αντιστοίχισης, βρίσκουμε τον αριθμό ενότητας Ε για τη φυσική μνήμη.
4. Η ΦΔ έχει για δεκάδες τον αριθμό ενότητας Ε και για μονάδες τη θέση Θ μέσα στην ενότητα.

Για την εικονική διεύθυνση ΕΔ = 5, ο αριθμός Σ της σελίδας είναι 0 και η θέση Θ είναι 5. Από τον πίνακα βρίσκουμε ότι η αντίστοιχη ενότητα Ε είναι η 1, έτσι η φυσική διεύθυνση θα είναι 15.

Κάθε εικονική διεύθυνση σε έναν υπολογιστή χωρίζεται σε δύο τμήματα: το πρώτο είναι ο αριθμός της σελίδας και το δεύτερο ο αριθμός της λέξης μέσα στη σελίδα. Στο παράδειγμά μας, κάθε τέτοιο τμήμα ήταν ένα δεκαδικό ψηφίο· στους υπολογιστές κάθε τμήμα της εικονικής διεύθυνσης αποτελείται από πολλά ψηφία, αφενός γιατί χρησιμοποιείται το δυαδικό σύστημα, αφετέρου γιατί το μέγεθος της εικονικής μνήμης είναι μεγάλο. Αντίστοιχα και κάθε φυσική διεύθυνση χωρίζεται στον αριθμό ενότητας και τον αριθμό λέξης. Όταν μεταφράζεται μια εικονική διεύθυνση σε φυσική, ο αριθμός λέξης μένει ο ίδιος.



Ας δούμε πώς μεταφράζεται μια εικονική διεύθυνση σε φυσική:

Ο πίνακας αντιστοίχισης της διεργασίας βρίσκεται στην κύρια μνήμη, και η διεύθυνση όπου αρχίζει κρατείται σε μια μονάδα αποθήκευσης της ΚΜΕ που ονομάζεται *καταχωρητής πίνακα σελίδων* (ΚΠΣ, page table register). Κάθε «θέση» του πίνακα σελίδων κρατά τον αριθμό ενότητας για την αντίστοιχη σελίδα. Έτσι, αν στην αρχική διεύθυνση του πίνακα σελίδων προσθέσουμε ένα αριθμό σελίδας, θα πάρουμε τη θέση του πίνακα που αντιστοιχεί στη σελίδα αυτή, και θα βρούμε τον αντίστοιχο αριθμό ενότητας για τη σελίδα.

Αν ο πίνακας σελίδων αρχίζει στη διεύθυνση 10 της μνήμης, ο αριθμός ενότητας για τη σελίδα 0 βρίσκεται στη διεύθυνση 10, ο αριθμός ενότητας για τη σελίδα 1 βρίσκεται στη διεύθυνση 11, κλπ. Γενικά ο αριθμός ενότητας για τη σελίδα κ βρίσκεται στη διεύθυνση 10+κ.

Για να βρούμε λοιπόν τον αριθμό ενότητας για μια εικονική διεύθυνση προσθέτουμε τον αριθμό σελίδας της με το περιεχόμενο του καταχωρητή πίνακα σελίδων. Το αποτέλεσμα είναι μια διεύθυνση μνήμης, και στη διεύθυνση αυτή θα βρούμε τον αριθμό ενότητας στη φυσική μνήμη. Αυτός ο αριθμός ενότητας συνδυάζεται με τον αριθμό λέξης για να δώσουν την τελική φυσική διεύθυνση.

Αν για μια σελίδα δεν υπάρχει αντίστοιχη ενότητα στην κύρια μνήμη, τότε πρέπει να ενεργοποιηθεί μια διαδικασία μεταφοράς της ενότητας αυτής από τη δευτερεύουσα στην κύρια μνήμη, ενώ συγχρόνως κάποια άλλη σελίδα μεταφέρεται από την κύρια στη δευτερεύουσα μνήμη για να απελευθερώσει την ενότητα που χρειάζεται. Η θέση στη δευτερεύουσα μνήμη όπου βρίσκονται όσες σελίδες είναι εκτός κύριας μνήμης μπορεί να καταγράφεται στον πίνακα σελίδων ή σε κάποιον άλλο πίνακα.

Ο πίνακας σελίδων είναι και αυτός αποθηκευμένος στη μνήμη. Έτσι κάθε μετάφραση εικονικής διεύθυνσης σε φυσική απαιτεί μια επιπλέον προσπέλαση μνήμης για την ανάγνωση του πίνακα[2]. Αυτή η προσπέλαση εισάγει μια ανεπιθύμητη καθυστέρηση στην εκτέλεση των εντολών. Μια λύση στο πρόβλημα αυτό είναι η χρήση ειδικής, γρήγορης μνήμης, για την αποθήκευση του πίνακα σελίδων.

Το κυριότερο μειονέκτημα της σελιδοποίησης είναι ότι κάθε διεργασία καταλαμβάνει περισσότερο χώρο από ό,τι χρειάζεται. Συνήθως η μνήμη που απαιτεί μια διεργασία δεν είναι πολλαπλάσιο του μεγέθους της σελίδας, αλλά χρειάζεται π.χ. 10 σελίδες ολόκληρες και 20 λέξεις επιπλέον. Επειδή ο χώρος εικονικών διευθύνσεων που της δίνεται έχει μέγεθος που είναι πολλαπλάσιο της σελίδας, θα της δοθούν 11 σελίδες· από την $11^{η}$ θα χρησιμοποιηθούν μόνο οι 20 πρώτες λέξεις. Η τελευταία σελίδα κάθε διεργασίας έτσι έχει ένα τμήμα, μικρότερο ή μεγαλύτερο, το οποίο μένει αχρησιμοποίητο. Αυτό το φαινόμενο ονομάζεται *εσωτερικός κατακερματισμός* (internal fragmentation).

Διαχείριση εικονικής μνήμης με κατάτμηση και κατατμημένη σελιδοποίηση

Η σελιδοποίηση είναι η πιο κοινά χρησιμοποιούμενη μέθοδος για τη διαχείριση της εικονικής μνήμης, γιατί συνήθως υποστηρίζεται από το υλικό των υπολογιστών. Δυο άλλες μέθοδοι που χρησιμοποιούνται, αλλά όχι τόσο ευρέως, είναι η κατάτμηση και η κατατμημένη σελιδοποίηση· η δεύτερη είναι συνδυασμός κατάτμησης και σελιδοποίησης.

Η μέθοδος της κατάτμησης προσπαθεί να αποφύγει τον εσωτερικό κατακερματισμό δίνοντας σε κάθε διεργασία ακριβώς όση μνήμη της χρειάζεται. Χωρίζει τη μνήμη σε τμήματα διαφορετικών μεγεθών, και για κάθε ένα από αυτά κρατά την αρχική διεύθυνση και το μέγεθός του. Για να μεταφράσει μια εικονική διεύθυνση σε φυσική, αναζητεί τις πληροφορίες που έχει κρατήσει για το αντίστοιχο τμήμα, ελέγχει αν η δεδομένη διεύθυνση είναι μέσα στα όρια του τμήματος, βρίσκει την αρχική του διεύθυνση στη φυσική μνήμη, και τέλος τη συνδυάζει με τη θέση μνήμης μέσα στο τμήμα για να υπολογίσει την τελική φυσική διεύθυνση.

Όταν δημιουργείται μια νέα διεργασία, το ΛΣ αναζητά ένα τμήμα εικονικών διευθύνσεων αρκετά μεγάλο για να χωρέσει τη μνήμη που χρειάζεται η διεργασία, και της αποδίδει τις διευθύνσεις αυτές.

---

[2] Εδώ έχουμε κάνει την απλουστευτική παραδοχή ότι μία γραμμή του πίνακα αντιστοίχισης χωρά ακριβώς σε μία λέξη της μνήμης. Αν μία γραμμή του πίνακα είναι μεγάλη και απαιτεί περισσότερες από μία λέξεις, τότε χρειάζονται οι αντίστοιχες επιπλέον προσπελάσεις στη μνήμη.

Η κατατμημένη σελιδοποίηση συνδυάζει τις δυο τεχνικές, εργαζόμενη πάλι με τμήματα, τα οποία όμως αποτελούνται από σελίδες σταθερού μεγέθους. Η κατάτμηση έτσι είναι η ειδική περίπτωση της κατατμημένης σελιδοποίησης όπου κάθε σελίδα αποτελείται μόνο από μία λέξη.

Ανακεφαλαίωση

Η μνήμη ενός υπολογιστή αποτελείται από τις μονάδες του που χρησιμεύουν στην αποθήκευση προγραμμάτων και δεδομένων. Η κύρια μνήμη (ή απλώς μνήμη) κρατά τα προγράμματα και τα δεδομένα για όσο χρόνο ο υπολογιστής λειτουργεί και αναφέρεται επίσης και ως RAM. Η δευτερεύουσα μνήμη (δίσκοι, CD-ROM κλπ.) τα κρατά και όταν ο υπολογιστής δε λειτουργεί. Το πρόβλημα οργάνωσης της μνήμης είναι πολύ σημαντικό κατά το σχεδιασμό ενός ΛΣ, ιδιαίτερα αν ο υπολογιστής εκτελεί πολλές διεργασίες και πρέπει να κάνει ανταλλαγή διεργασιών μεταξύ κύριας και δευτερεύουσας μνήμης, επειδή δε χωρούν όλα τα απαραίτητα προγράμματα και δεδομένα στην κύρια μνήμη. Οι εικονικές διευθύνσεις βοηθούν στην παράλληλη εκτέλεση πολλών διεργασιών χωρίς η μια να παρεμβαίνει στις θέσεις μνήμης που χρησιμοποιεί η άλλη. Για τη μετάφραση εικονικών διευθύνσεων σε φυσικές, πραγματικές διευθύνσεις μνήμης, υπάρχουν διάφορες τεχνικές που χρησιμοποιούν έναν πίνακα αντιστοίχισης. Η πιο δημοφιλής από αυτές είναι η σελιδοποίηση, που αντιστοιχίζει σελίδες μνήμης σταθερού μεγέθους.

Γλωσσάριο όρων

Ανταλλαγή	*Swapping*
Δευτερεύουσα / Περιφερειακή Μνήμη	*Secondary Memory*
Εικονική Διεύθυνση	*Virtual Address*
Εικονική Μνήμη	*Virtual Memory*
Ενότητα	*Block – Frame*
Εσωτερικός Κατακερματισμός	*Internal Fragmentation*
Δυναμική Κατανομή	*Dynamic Allocation*
Κατατμημένη Σελιδοποίηση	*Segmented Paging*
Κατάτμηση	*Segmentation*
Καταχωρητής Πίνακα Σελίδων	*Page Table Register*
Κύρια / Κεντρική Μνήμη	*Main Memory*
Λέξη	*Word*
Μνήμη ROM	*Read Only Memory*
Μνήμη Τυχαίας Προσπέλασης	*Random Access Memory - RAM*
Προσπέλαση Μνήμης	*Memory Access*
Σελίδα	*Page*
Σελιδοποίηση	*Paging*
Στατική Κατανομή	*Static Allocation*
Τμήμα	*Segment*
Χώρος Απολύτων Διευθύνσεων	*Absolute Address Space*
Χώρος Εικονικών Διευθύνσεων	*Virtual Address Space*
Χώρος Πραγματικών Διευθύνσεων	*Real Address Space*
Χώρος Φυσικών Διευθύνσεων	*Physical Address Space*

Ερωτήσεις

? Ποια είναι η βασική διαφορά της κύριας από τη δευτερεύουσα μνήμη; Είναι και οι δύο απολύτως απαραίτητες για τη λειτουργία του υπολογιστή;

? Ποια είναι τα μειονεκτήματα της ανταλλαγής;

? Η εικονική μνήμη έχει φυσική υπόσταση; Αν όχι, τι είναι ο εικονικός χώρος διευθύνσεων;

? Πώς μεταφράζεται μία εικονική διεύθυνση σε φυσική όταν το ΛΣ χρησιμοποιεί σελιδοποίηση για τη διαχείριση της εικονικής μνήμης;

Μάθημα 9.2 Τεχνικές Διαχείρισης Εικονικής Μνήμης

Σκοπός του μαθήματος

Σκοπός του μαθήματος αυτού είναι να περιγράψει τις διάφορες τεχνικές που χρησιμοποιούν τα ΛΣ για να διαχειριστούν τη μνήμη.

Τι θα μάθεις;

Όταν ολοκληρώσεις το μάθημα αυτό, θα μπορείς:

- *Να αναφέρεις τα προβλήματα που λύνει η διαχείριση της εικονικής μνήμης*
- *Να εξηγείς πότε γίνεται μεταφορά σελίδων προς την κύρια μνήμη*
- *Να εξηγείς ποια δεδομένα αντικαθίστανται κατά τη μεταφορά αυτή*
- *Να περιγράφεις πώς επιλέγονται οι διευθύνσεις της φυσικής μνήμης όπου τοποθετούνται νέα δεδομένα*

Όπως είδαμε στα προηγούμενα μαθήματα, ο χώρος εικονικών διευθύνσεων μνήμης που χρησιμοποιεί κάθε διεργασία, είναι αρκετά μεγαλύτερος από το χώρο των φυσικών διευθύνσεων. Έτσι, διάφορες εικονικές διευθύνσεις για μια διεργασία δεν αντιστοιχούν στην κύρια, αλλά στη δευτερεύουσα μνήμη. Για να χρησιμοποιηθούν όμως από τη διεργασία, πρέπει τα αντίστοιχα δεδομένα να μεταφερθούν στην κύρια μνήμη, αφού κάποια άλλα δεδομένα μεταφερθούν από την κύρια στη δευτερεύουσα μνήμη και να αλλάξουν οι αντιστοιχίσεις εικονικών και φυσικών διευθύνσεων ώστε να απεικονίζουν σωστά τη νέα κατάσταση.

Μια διεύθυνση δε μεταφέρεται μόνη της από τη δευτερεύουσα στην κύρια μνήμη, αλλά μαζί με τις γειτονικές της, γιατί αυτές έχουν πολύ αυξημένη πιθανότητα να χρησιμοποιηθούν στη συνέχεια από τη διεργασία, εξαιτίας της τοπικότητας αναφοράς των προγραμμάτων. Έτσι μεταφέρεται ολόκληρη η σελίδα μνήμης στην οποία περιέχεται η διεύθυνση αυτή. Η τοπικότητα της αναφοράς είναι μία ιδιότητα των προγραμμάτων που χρησιμοποιείται γενικά στο σύστημα της μνήμης ενός υπολογιστή, όπως είδαμε και στο μάθημα για τη λανθάνουσα μνήμη.

Η διαχείριση της εικονικής μνήμης από το ΛΣ πρέπει να δίνει απάντηση σε τρεις ουσιαστικές ερωτήσεις:

- ✔ Πότε πρέπει να γίνεται **μεταφορά** δεδομένων από τη δευτερεύουσα στην κύρια μνήμη;
- ✔ Ποια δεδομένα θα **αντικατασταθούν** και θα επιστρέψουν στη δευτερεύουσα μνήμη;
- ✔ Σε ποιες διευθύνσεις της φυσικής μνήμης θα **τοποθετηθούν** νέα δεδομένα;

Τα δυο πρώτα ερωτήματα αφορούν τα συστήματα σελιδοποίησης και τα συστήματα κατάτμησης, ενώ το τρίτο αφορά κυρίως τα συστήματα κατάτμησης όπου τα τμήματα των δεδομένων έχουν διαφορετικά μεγέθη και πρέπει να βρεθεί μια συνεχόμενη περιοχή φυσικής μνήμης όπου να χωρούν.

Μεταφορά σελίδων

Όπως γνωρίζουμε, το ΛΣ επιδιώκει να εξυπηρετεί πολλές διεργασίες ταυτόχρονα, αλλά έτσι δεν είναι δυνατόν να βρίσκονται συγχρόνως φορτωμένες στην κύρια μνήμη όλες οι σελίδες για κάθε διεργασία. Ανάλογα με τις ανάγκες της κάθε διεργασίας γίνεται μεταφορά σελίδων από και προς την κύρια μνήμη· η στρατηγική μεταφοράς που εφαρμόζει το ΛΣ θα έχει σαν σκοπό να αποφασίσει πότε θα γίνεται μεταφορά σελίδων ώστε να εξοικονομείται χρόνος. Δυο είναι οι κύριες στρατηγικές μεταφοράς:

1. Η πιο συνηθισμένη στρατηγική είναι η *σελιδοποίηση με αίτηση* (demand paging), σύμφωνα με την οποία μια σελίδα μεταφέρεται στην κύρια μνήμη μόνο όταν ζητηθεί κάποια διεύθυνση που περιέχεται σε αυτή και διαπιστωθεί ότι η σελίδα βρίσκεται στη δευτερεύουσα μνήμη. Η στρατηγική αυτή εξασφαλίζει ότι μια σελίδα θα μεταφερθεί από τη δευτερεύουσα στην κύρια μνήμη μόνο όταν αυτό είναι απαραίτητο. Από την άλλη όμως, μια διεργασία πρέπει να περιμένει κάθε φορά που θα ζητήσει την προσπέλαση μιας εικονικής διεύθυνσης που αντιστοιχεί στη δευτερεύουσα μνήμη. Μάλιστα, όσο περισσότερες διεργασίες εκτελούνται ταυτόχρονα από το ΛΣ, τόσο πιο συχνά πρέπει να περιμένουν για τη μεταφορά των σελίδων τους στην κύρια μνήμη.
2. Αν μπορούν να προβλεφθούν σε ικανοποιητικό βαθμό οι σελίδες που θα προσπελάσει μια διεργασία πριν να τις προσπελάσει, τότε εφαρμόζεται η στρατηγική της *σελιδοποίησης με πρόβλεψη* (anticipatory paging). Η στρατηγική αυτή προβλέπει ποιες σελίδες θα προσπελάσει βραχυπρόθεσμα μια διεργασία και τις μεταφέρει εκ των προτέρων στην κύρια μνήμη· έτσι η διεργασία δε χρειάζεται να περιμένει για να προσπελάσει οποιαδήποτε διεύθυνση. Είναι όμως πιθανόν ότι θα μεταφερθούν στην κύρια μνήμη σελίδες που δε θα χρειαστούν και θα καταλαμβάνουν χώρο χωρίς λόγο. Επίσης, ο υπολογισμός των σελίδων που θα μεταφερθούν στην κύρια μνήμη απαιτεί κάποιο χρόνο και επιβαρύνει το ΛΣ.

Από τις δυο στρατηγικές συχνότερα χρησιμοποιείται η σελιδοποίηση με αίτηση, γιατί η υλοποίησή της είναι απλούστερη και επιπλέον διότι η πρόβλεψη των σελίδων που θα χρειαστεί στο άμεσο μέλλον μια διεργασία είναι στην πλειοψηφία των περιπτώσεων δύσκολη.

Αντικατάσταση σελίδων

Όταν το ΛΣ αποφασίσει ότι θα μεταφέρει μια σελίδα από τη δευτερεύουσα στην κύρια μνήμη, πρέπει να επιλέξει μια σελίδα στην κύρια μνήμη που θα αντικατασταθεί και θα μεταφερθεί στη δευτερεύουσα μνήμη. Στόχος του ΛΣ είναι να επιλέξει για μεταφορά στη δευτερεύουσα μνήμη μια σελίδα που έχει μικρή πιθανότητα να χρησιμοποιηθεί σύντομα.

> Σε έναν υπολογιστή με κύρια μνήμη μεγέθους 4 ενοτήτων εκτελείται η διεργασία δ_α, η οποία απαιτεί 7 σελίδες μνήμης (α_0, α_1, α_2, α_3, α_4, α_5, και α_6). Όταν ξεκινά η διεργασία, στην κύρια μνήμη δεν είναι αποθηκευμένη καμιά σελίδα της. Στη συνέχεια η ζήτηση σελίδων από τη διεργασία είναι: α_0, α_2, α_6, α_5, α_2, α_1, α_0, α_3, α_4, α_0, α_6. Υποθέτουμε ότι οι σελίδες ζητούνται σε διαδοχικές χρονικές στιγμές, δηλαδή τη χρονική στιγμή 0 ζητείται η σελίδα α_0, τη χρονική στιγμή 1 ζητείται η σελίδα α_2 κ.ο.κ.

Στη μία στρατηγική, αυτή της *αντικατάστασης με βάση το χρόνο παραμονής* (First In First Out, FIFO), επιλέγεται η σελίδα που βρίσκεται για το μεγαλύτερο χρονικό διάστημα στην κύρια μνήμη και αντικαθίσταται με τη νέα. Για να εντοπίζεται η παλαιότερη σελίδα, κάθε φορά που μια σελίδα μεταφέρεται στην κύρια μνήμη σημειώνεται με τον τρέχοντα χρόνο (από το ρολόι του υπολογιστή). **Η σελίδα που είναι σημειωμένη με το μικρότερο χρόνο είναι η παλαιότερη.**

Τις τέσσερις πρώτες χρονικές στιγμές (0, 1, 2, 3), που ζητούνται οι σελίδες α_0, α_2, α_6 και α_5 αντίστοιχα, οι τέσσερις σελίδες της μνήμη είναι κενές, οπότε οι σελίδες τοποθετούνται απευθείας. Στη συνέχεια, οι αντικαταστάσεις των σελίδων φαίνονται στο ακόλουθο σχήμα που παριστάνει τον πίνακα σελίδων. Στη στήλη 📖 φαίνονται οι σελίδες της εικονικής μνήμης με την αντίστοιχη ενότητα κύριας μνήμης (ή «-» αν η σελίδα δε βρίσκεται στην κύρια μνήμη), και στη στήλη 🕑 φαίνεται ο χρόνος με τον οποίο έχουν σημειωθεί όσες σελίδες βρίσκονται στην κύρια μνήμη. Για κάθε χρονική στιγμή, δίνεται η σελίδα που ζητά η διεργασία και εξηγείται η αντικατάσταση που θα γίνει, αν είναι απαραίτητη. Η σελίδα που θα αντικατασταθεί κάθε φορά είναι σκιασμένη.

Τη χρονική στιγμή 4 προσπελάζεται η σελίδα α_2, η οποία είναι ήδη φορτωμένη στη μνήμη, οπότε δε γίνεται καμία αντικατάσταση.

	📖	🕑
α_0	0	0
α_1	-	-
α_2	1	1
α_3	-	-
α_4	-	-
α_5	3	3
α_6	2	2

⇓⇓ **προσπέλαση της α_2** ⇓⇓

Τη χρονική στιγμή 5 η διεργασία ζητά τη σελίδα α_1. Η παλαιότερη σελίδα είναι η α_0, η οποία θα αντικατασταθεί από την α_1 και θα σημειωθεί με το χρόνο 5.

	📖	🕑
α_0	0	0
α_1	-	-
α_2	1	1
α_3	-	-
α_4	-	-
α_5	3	3
α_6	2	2

⇓⇓ **προσπέλαση της α_1** ⇓⇓

Τη χρονική στιγμή 6 προσπελάζεται πάλι η α_0, που δε βρίσκεται πλέον στη μνήμη. Μεταφέρεται έτσι η α_2, που ήταν η παλαιότερη, στη δευτερεύουσα μνήμη, και η α_0 παίρνει τη θέση της.

	📖	🕑
α_0	-	-
α_1	0	5
α_2	1	1
α_3	-	-
α_4	-	-
α_5	3	3
α_6	2	2

⇓⇓ **προσπέλαση της α_0** ⇓⇓

Τη χρονική στιγμή 7 ζητείται η σελίδα α_3, η οποία θα αντικατα-στήσει τη α_6 που ήταν η παλαιότερη. Η α_3 σημειώνεται με το χρόνο 7.

	📖	🕑
α_0	1	6
α_1	0	5
α_2	-	-
α_3	-	-
α_4	-	-
α_5	3	3
α_6	2	2

⇓⇒⇒⇒ **προσπέλαση της α_3** ⇒⇒⇒⇑

Συνολικά για τις 11 προσπελάσεις σελίδων έγιναν 5 αντικαταστάσεις.

	📖	🕑
α_0	1	6
α_1	-	-
α_2	-	-
α_3	2	7
α_4	3	8
α_5	-	-
α_6	0	10

⇑⇑ **προσπέλαση της α_6** ⇑⇑

Τη χρονική στιγμή 10 ζητείται η σελίδα α_6 · η παλαιότερη σελίδα είναι η α_1 με χρόνο 5.

	📖	🕑
α_0	1	6
α_1	0	5
α_2	-	-
α_3	2	7
α_4	3	8
α_5	-	-
α_6	-	-

⇑⇑ **προσπέλαση της α_0** ⇑⇑

Τη χρονική στιγμή 9, προσπελάζεται πάλι η σελίδα α_0 που ήδη βρίσκεται στην κύρια μνήμη, και έτσι δε γίνεται καμία αντικατάσταση.

	📖	🕑
α_0	1	6
α_1	0	5
α_2	-	-
α_3	2	7
α_4	3	8
α_5	-	-
α_6	-	-

⇑⇑ **προσπέλαση της α_4** ⇑⇑

Τη χρονική στιγμή 8, η διεργασία ζητά τη σελίδα α_4. Αυτή θα τοποθετηθεί αντί για τη α_5 που είχε το μικρότερο χρόνο εισόδου στην κύρια μνήμη.

	📖	🕑
α_0	1	6
α_1	0	5
α_2	-	-
α_3	2	7
α_4	-	-
α_5	3	3
α_6	-	-

Η δεύτερη στρατηγική είναι αυτή της *αντικατάστασης με βάση το χρόνο της τελευταίας προσπέλασης* (Least Recently Used, LRU), οπότε επιλέγεται για αντικατάσταση εκείνη η σελίδα που έχει τον περισσότερο χρόνο να προσπελαστεί. Για να είναι αυτό δυνατόν, πρέπει **κάθε φορά** που μια σελίδα προσπελάζεται, είτε μεταφέρεται στην κύρια μνήμη είτε ήδη βρίσκεται εκεί, να σημειώνεται με τον τρέχοντα χρόνο του υπολογιστή*. Όταν πρέπει να αντικατασταθεί μια σελίδα, το ΛΣ διαλέγει εκείνη που είναι σημειωμένη με το μικρότερο (δηλαδή τον παλαιότερο) χρόνο.

Και με την τεχνική της αντικατάστασης με βάση το χρόνο προσπέλασης, οι τέσσερις πρώτες προσπελάσεις σελίδων, οπότε ζητούνται οι σελίδες α_0, α_2, α_6 και α_5, έχουν σαν αποτέλεσμα την απευθείας τοποθέτηση των σελίδων στην κύρια μνήμη. Στη συνέχεια, οι αντικαταστάσεις των σελίδων γίνονται ως εξής:

Τη χρονική στιγμή 4 προσπελάζεται η σελίδα α_2, η οποία είναι ήδη φορτωμένη στη μνήμη, οπότε δε γίνεται καμία αντικατάσταση, αλλά μόνο σημείωσή της με τον τρέχοντα χρόνο.

	📖	🕑
α_0	0	0
α_1	-	-
α_2	1	1
α_3	-	-
α_4	-	-
α_5	3	3
α_6	2	2

⇓⇓ **προσπέλαση της α_2** ⇓⇓

Τη χρονική στιγμή 5 η διεργασία α ζητά τη σελίδα α_1. Η παλαιότερα χρησιμοποιημένη σελίδα είναι η α_0, η οποία θα αντικατασταθεί από την α_1 και θα σημειωθεί με το χρόνο 5.

	📖	🕑
α_0	0	0
α_1	-	-
α_2	1	4
α_3	-	-
α_4	-	-
α_5	3	3
α_6	2	2

⇓⇓ **προσπέλαση της α_1** ⇓⇓

Τη χρονική στιγμή 6 προσπελάζεται πάλι η α_0, που δε βρίσκεται πλέον στη μνήμη. Μεταφέρεται έτσι η α_6, που είχε το μικρότερο χρόνο προσπέλασης, στη δευτερεύουσα μνήμη.

	📖	🕑
α_0	-	-
α_1	0	5
α_2	1	4
α_3	-	-
α_4	-	-
α_5	3	3
α_6	2	2

⇓⇓ **προσπέλαση της α_0** ⇓⇓

Τη χρονική στιγμή 7 ζητείται η σελίδα α_3, η οποία θα αντικατα-στήσει τη α_5 που έχει τον παλαιότερο χρόνο προσπέλασης. Η α_3 σημειώνεται με το χρόνο 7.

	📖	🕑
α_0	2	6
α_1	0	5
α_2	1	4
α_3	-	-
α_4	-	-
α_5	3	3
α_6	-	-

⇓⇒⇒⇒ **προσπέλαση της α_3** ⇒⇒⇒⇑

Συνολικά για τις 11 προσπελάσεις σελίδων έγιναν και με την τεχνική αυτή 5 αντικαταστάσεις, αλλά διαφορετικές από ό,τι με την τεχνική του χρόνου παραμονής.

	📖	🕑
α_0	2	9
α_1	-	-
α_2	-	-
α_3	3	7
α_4	1	8
α_5	-	-
α_6	0	10

⇑⇑ **προσπέλαση της α_6** ⇑⇑

Τη χρονική στιγμή 10 ζητείται η σελίδα α_6 · η λιγότερο πρόσφατα χρησιμοποιημένη σελίδα είναι η α_1 με χρόνο προσπέλασης 5.

	📖	🕑
α_0	2	9
α_1	0	5
α_2	-	-
α_3	3	7
α_4	1	8
α_5	-	-
α_6	-	-

⇑⇑ **προσπέλαση της α_0** ⇑⇑

Τη χρονική στιγμή 9, προσπελάζεται πάλι η σελίδα α_0 που ήδη βρίσκεται στην κύρια μνήμη, και έτσι απλά η σελίδα σημειώνεται με το νέο χρόνο του υπολογιστή.

	📖	🕑
α_0	2	6
α_1	0	5
α_2	-	-
α_3	3	7
α_4	1	8
α_5	-	-
α_6	-	-

⇑⇑ **προσπέλαση της α_4** ⇑⇑

Τη χρονική στιγμή 8, η διεργασία ζητά τη σελίδα α_4. Αυτή θα τοποθετηθεί αντί για την α_2 που είχε το μικρότερο χρόνο προσπέλασης.

	📖	🕑
α_0	2	6
α_1	0	5
α_2	1	4
α_3	3	7
α_4	-	-
α_5	-	-
α_6	-	-

* Ενώ στην τεχνική FIFO μια σελίδα σημειώνεται *μόνο* όταν μεταφέρεται στην κύρια μνήμη.

Τοποθέτηση σελίδων

Σε ένα σύστημα σελιδοποίησης είναι πολύ εύκολο να επιλεγεί η ενότητα της κύριας μνήμης όπου θα τοποθετηθεί μια σελίδα εικονικών διευθύνσεων, γιατί όλες έχουν το ίδιο μέγεθος. Στα συστήματα κατάτμησης όμως, όπου τα τμήματα έχουν διαφορετικά μεγέθη, κάθε φορά που μεταφέρεται ένα νέο τμήμα στην κύρια μνήμη, πρέπει να βρεθεί μια κατάλληλη συνεχόμενη περιοχή της κύριας μνήμης η οποία να το χωρά. Η επιλογή της περιοχής αυτής γίνεται με βάση δυο κριτήρια:

- Η αναζήτηση της κατάλληλης ελεύθερης περιοχής πρέπει να γίνεται γρήγορα, για να μην προκαλούνται μεγάλες χρονικές επιβαρύνσεις.
- Η τοποθέτηση του τμήματος στην κύρια μνήμη πρέπει να παρεμποδίζει όσο το δυνατόν λιγότερο την τοποθέτηση επομένων τμημάτων.

Οι στρατηγικές τοποθέτησης σελίδων που χρησιμοποιούνται περισσότερο είναι η στρατηγική του *πρώτου ταιριάσματος* (first fit), η στρατηγική του *καλύτερου ταιριάσματος* (best fit) και η στρατηγική του *χειρότερου ταιριάσματος* (worst fit).

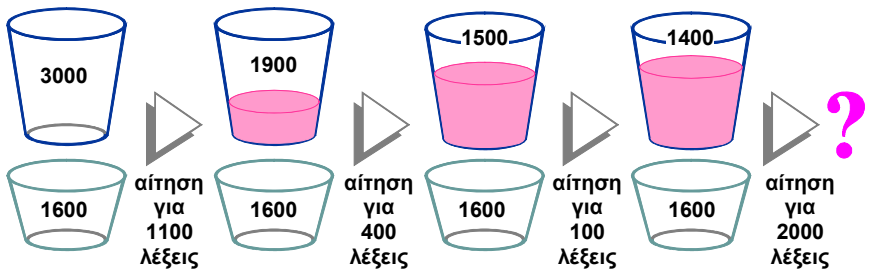
Όλες αυτές οι στρατηγικές προϋποθέτουν ότι το ΛΣ κρατά ένα κατάλογο με τα ελεύθερα τμήματα φυσικής μνήμης και το μέγεθος του καθενός από αυτά. Κάθε φορά που επιλέγεται ένα από τα ελεύθερα τμήματα για να τοποθετηθεί σε αυτό ένα τμήμα εικονικής μνήμης, αφαιρείται από τον κατάλογο. Αν τυχόν το τμήμα της εικονικής μνήμης είναι μικρότερο από αυτό της κύριας μνήμης, η μνήμη που περισσεύει επιστρέφεται στον κατάλογο των ελευθέρων τμημάτων.

Στρατηγική του πρώτου ταιριάσματος: Σαρώνεται ο κατάλογος ελεύθερων τμημάτων μέχρι να βρεθεί το πρώτο τμήμα που είναι αρκετά μεγάλο για να χωρά το νέο τμήμα εικονικής μνήμης. Ό,τι περισσεύει επιστρέφεται στον κατάλογο σαν ένα μικρότερο ελεύθερο τμήμα. Η στρατηγική αυτή είναι η πιο γρήγορη, γιατί δε σαρώνει ολόκληρο τον κατάλογο ελευθέρων τμημάτων, αλλά μπορεί να δημιουργήσει πολλά μικρά τμήματα ελεύθερης μνήμης που δεν μπορούν να χρησιμοποιηθούν.

Στρατηγική του καλύτερου ταιριάσματος: Η στρατηγική αυτή εξετάζει όλα τα ελεύθερα τμήματα, και επιλέγει το μικρότερο ελεύθερο κομμάτι φυσικής μνήμης που χωρά το τμήμα εικονικής μνήμης. Επειδή πρέπει να εξετάσει όλα τα ελεύθερα τμήματα είναι μάλλον αργή, και επιπλέον αφήνει πολύ μικρά κομμάτια ελεύθερης μνήμης τα οποία δεν μπορούν να χρησιμοποιηθούν λόγω του μικρού τους μεγέθους.

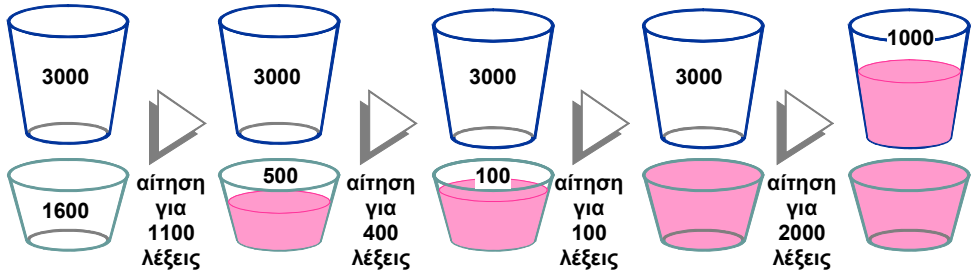
Στρατηγική του χειρότερου ταιριάσματος: Επιλέγει κάθε φορά το μεγαλύτερο κομμάτι μνήμης (εφόσον υπάρχει κάποιο αρκετά μεγάλο ώστε να χωρά το τμήμα που θέλουμε να τοποθετήσουμε). Και αυτή η μέθοδος πρέπει να εξετάσει ολόκληρο τον κατάλογο, εκτός και αν αυτός είναι ταξινομημένος κατά μέγεθος σε φθίνουσα σειρά.

> Αρχικά ο κατάλογος ελεύθερων τμημάτων περιέχει δυο τμήματα φυσικής μνήμης, ένα με μέγεθος 3000 και ένα με μέγεθος 1600 λέξεις. Στη συνέχεια ζητούνται διαδοχικά τέσσερα τμήματα μνήμης, με μεγέθη 1100, 400, 100 και 2000 λέξεις, αντίστοιχα.

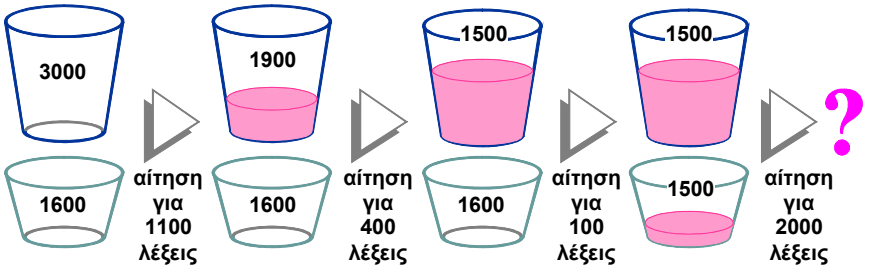


Στρατηγική του πρώτου ταιριάσματος

Στρατηγική του καλύτερου ταιριάσματος



Στρατηγική του χειρότερου ταιριάσματος



Στο παράδειγμα αυτό η στρατηγική του καλύτερου ταιριάσματος είναι καλύτερη από τις άλλες δύο, οι οποίες δεν μπόρεσαν να ικανοποιήσουν την τελευταία αίτηση μνήμης. Σε άλλα παραδείγματα όμως μπορεί αυτή να μην είναι τόσο καλή, ενώ κάποια άλλη (π.χ. η η στρατηγική του πρώτου ταιριάσματος) μπορεί να αποδειχθεί αποδοτικότερη.

Ανακεφαλαίωση

Στην αντιστοίχιση εικονικής με φυσική μνήμη, τρία είναι τα ζητήματα που πρέπει να αντιμετωπιστούν: αν η μεταφορά της μνήμης θα γίνεται με αίτηση ή με πρόβλεψη, πώς θα επιλέγεται η σελίδα που θα αντικατασταθεί και πού θα τοποθετηθεί ένα νέο τμήμα μνήμης. Για την αντικατάσταση υπάρχουν οι τεχνικές αντικατάστασης με βάση το χρόνο προσπέλασης ή το χρόνο πρόσβασης, και για την τοποθέτηση χρησιμοποιούνται οι στρατηγικές του πρώτου, του καλύτερου ή του χειρότερου ταιριάσματος.

Γλωσσάριο όρων

Αντικατάσταση με Βάση το Χρόνο Παραμονής	*First In First Out - FIFO*
Αντικατάσταση με Βάση το Χρόνο της Τελευταίας Προσπέλασης	*Least Recently Used - LRU*
Σελιδοποίηση με Αίτηση	*Demand Paging*
Σελιδοποίηση με Πρόβλεψη	*Anticipatory Paging*
Στρατηγική του Καλύτερου Ταιριάσματος	*Best Fit*
Στρατηγική του Πρώτου Ταιριάσματος	*First Fit*
Στρατηγική του Χειρότερου Ταιριάσματος	*Worst Fit*

Ερωτήσεις

? Ποια είναι τα πλεονεκτήματα και τα μειονεκτήματα της μεταφοράς σελίδων με πρόβλεψη;

? Περίγραψε τη μέθοδο αντικατάστασης σελίδων με βάση το χρόνο τελευταίας πρόσβασης.

? Ποια νομίζεις ότι είναι η βασική ιδέα της στρατηγικής του χειρότερου ταιριάσματος;

Τι μάθαμε σε αυτό το κεφάλαιο

- Οι εικονικές διευθύνσεις βοηθούν στην παράλληλη εκτέλεση πολλών διεργασιών χωρίς η μια να παρεμβαίνει στις θέσεις μνήμης που χρησιμοποιεί η άλλη.
- Η πιο δημοφιλής τεχνική αντιστοίχισης εικονικών διευθύνσεων σε φυσικές είναι η σελιδοποίηση.
- Στην αντιστοίχιση εικονικής με φυσική μνήμη, τρία είναι τα ζητήματα που πρέπει να αντιμετωπιστούν: αν η μεταφορά της μνήμης θα γίνεται με αίτηση ή με πρόβλεψη, πώς θα επιλέγεται η σελίδα που θα αντικατασταθεί και πού θα τοποθετηθεί ένα νέο τμήμα μνήμης.

Βιβλιογραφία – Πηγές

- Παπακωσταντίνου Γ., Ν. Μπιλάλη, Π. Τσανάκα, Λειτουργικά Συστήματα: Αρχές Λειτουργίας, Συμμετρία, 1989.
- Silberschatz A., Peterson J., Galvin P., Operating System Concepts, Addison - Wesley, 1991.