Λειτουργικά Συστήματα

Ενότητα 5: ΛΣ Κατανεμημένα & Πραγματικού Χρόνου. Χρονοπρογραμματισμός. Πολιτικές/Μηχανισμοί.

Δρ. Μηνάς Δασυγένης
mdasyg@ieee.org
Εργαστήριο Ψηφιακών Συστημάτων και Αρχιτεκτονικής Υπολογιστών
http://arch.icte.uowm.gr/mdasyg

Τμήμα Μηχανικών Πληροφορικής και Τηλεπικοινωνιών

Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons.
- Για εκπαιδευτικό υλικό, όπως εικόνες, που υπόκειται σε άλλου τύπου άδειας χρήσης, η άδεια χρήσης αναφέρεται ρητώς.

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Ψηφιακά Μαθήματα στο Πανεπιστήμιο Δυτικής Μακεδονίας**» 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.

ΛΣ Πραγματικού Χρόνου Κατανεμημένα Δικτυακά

Συστήματα πραγματικού χρόνου

- Συχνά χρησιμοποιούνται ως μια συσκευή ελέγχου σε μια συγκεκριμένη εφαρμογή όπως ο έλεγχος επιστημονικών πειραμάτων, ο έλεγχος βιομηχανικών συστημάτων, σε συστήματα επεξεργασίας εικόνας ιατρικών εφαρμογών κλπ.
- Διαθέτουν καλά σχεδιασμένους περιορισμούς χρόνου.
- **Hard real-time system:**
  - Χαρακτηρίζονται από την περιορισμένη χρήση δευτερεύουσας μνήμης και τα δεδομένα αποθηκεύονται σε μνήμες βραχείας διάρκειας ή σε ROM.
- **Soft real-time system:**
  - Περιορισμένη χρησιμότητα σε βιομηχανικό έλεγχο και σε ρομποτική.
  - Χρήσιμα σε εφαρμογές (multimedia, virtual reality) που απαιτούν εξειδικευμένα χαρακτηριστικά Λ.Σ.

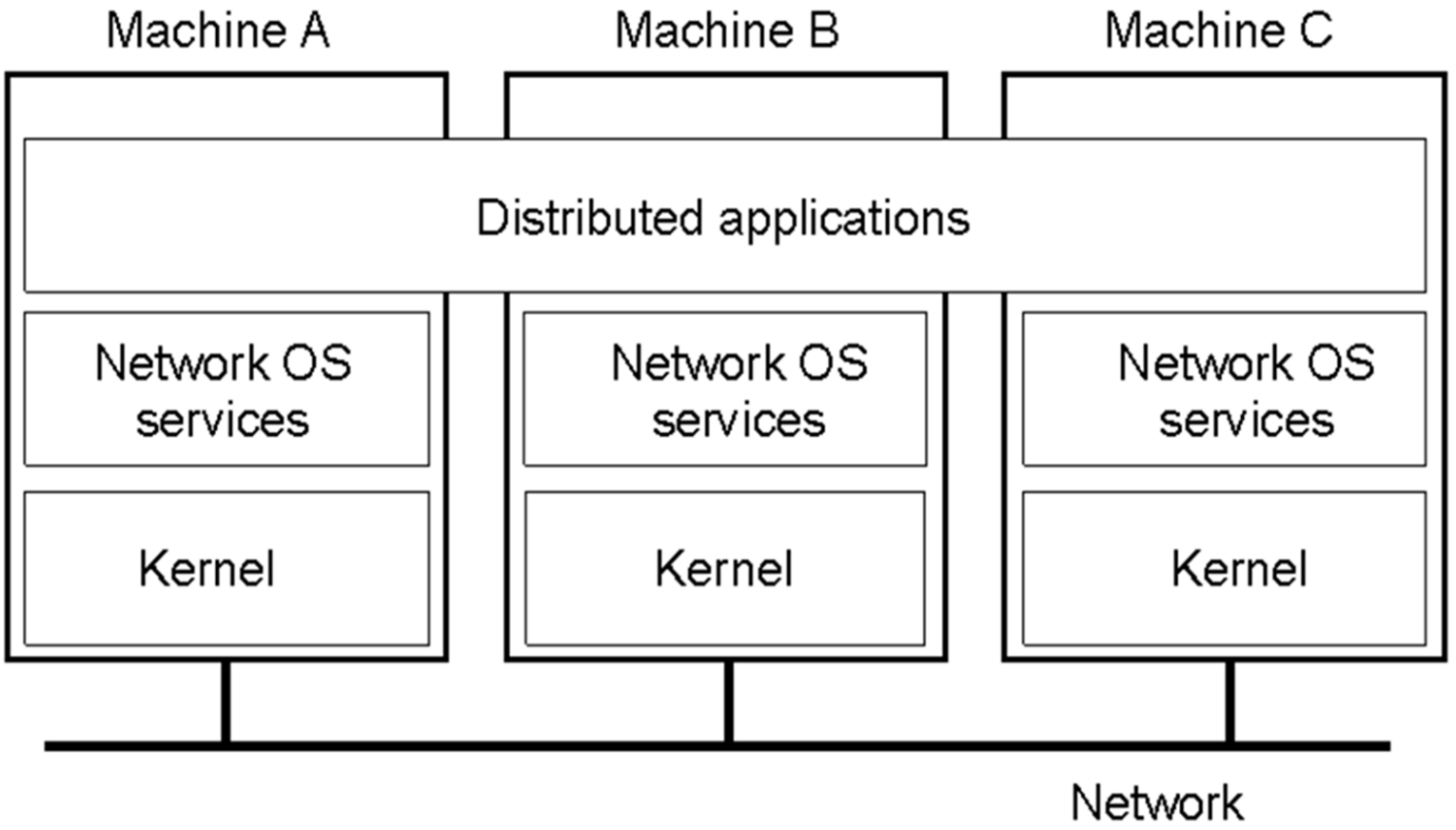
Κατανεμημένα Συστήματα

- Κατανέμουν τη διαδικασία υπολογισμών σε πολλούς φυσικούς επεξεργαστές.
- Παρέχουν την ψευδαίσθηση ενός μοναδικού χώρου για την κύρια μνήμη και ενός ξεχωριστού μοναδικού χώρου για τη δευτερεύουσα μνήμη, μαζί με άλλες ευκολίες όπως ένα κατανεμημένο σύστημα αρχείων (αφορά στην ουσία ένα πολυ-υπολογιστικό σύστημα δηλ. μια συλλογή από οντότητες (ΗΥ), που η κάθε μια έχει τη δική της κύρια, δευτερεύουσα μνήμη και άλλα στοιχεία I/O).

Χαρακτηριστικά των Κ.Σ.

- Πλεονεκτήματα:
  - Διαμοίραση πόρων .
  - Αύξηση της ταχύτητας υπολογισμού .
  - Αξιοπιστία.
  - Δυνατότητες επικοινωνίας.
  - Μειονεκτήματα.
  - Ασφάλεια και προστασία.
- Τα κατανεμημένα συστήματα αποκρύπτουν:
  - Τον τρόπο πρόσβασης σε έναν πόρο.
  - Το χώρο όπου βρίσκεται κάποιος πόρος.
  - Τη διαμοίραση πόρων από πολλούς χρήστες που ανταγωνίζονται για τη χρήση τους.
  - Τη μετακίνηση ενός πόρου σε άλλο μέρος ενώ είναι σε χρήση.
  - Τις διαφορές στην αναπαράσταση δεδομένων.

Network Operating System

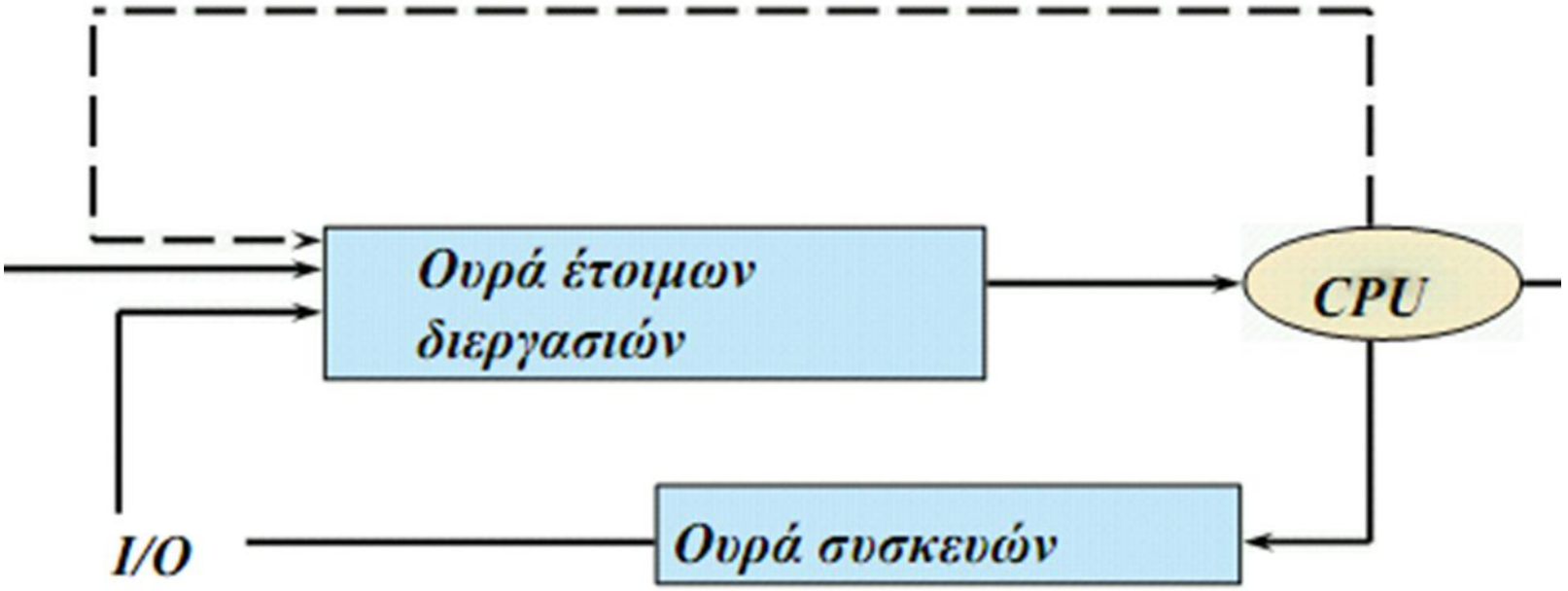


Ουρές Χρονοδρομολόγησης

Ποιες είναι οι βασικές ουρές που συνδέονται με τις διεργασίες;

- **Ουρά διεργασιών:**
  - Σύνολο όλων των διεργασιών του συστήματος.
- **Ουρά έτοιμων διεργασιών:**
  - Σύνολο όλων των διεργασιών που βρίσκονται στην Κύρια Μνήμη, έτοιμες για εκτέλεση.
- **Ουρά συσκευών:**
  - Σύνολο των διεργασιών που αναμένουν μία συγκεκριμένη I/O συσκευή.

Μετανάστευση διεργασιών μεταξύ των διαφόρων ουρών



Πρόβλημα Παραγωγού Καταναλωτή: Monitors (1/2)

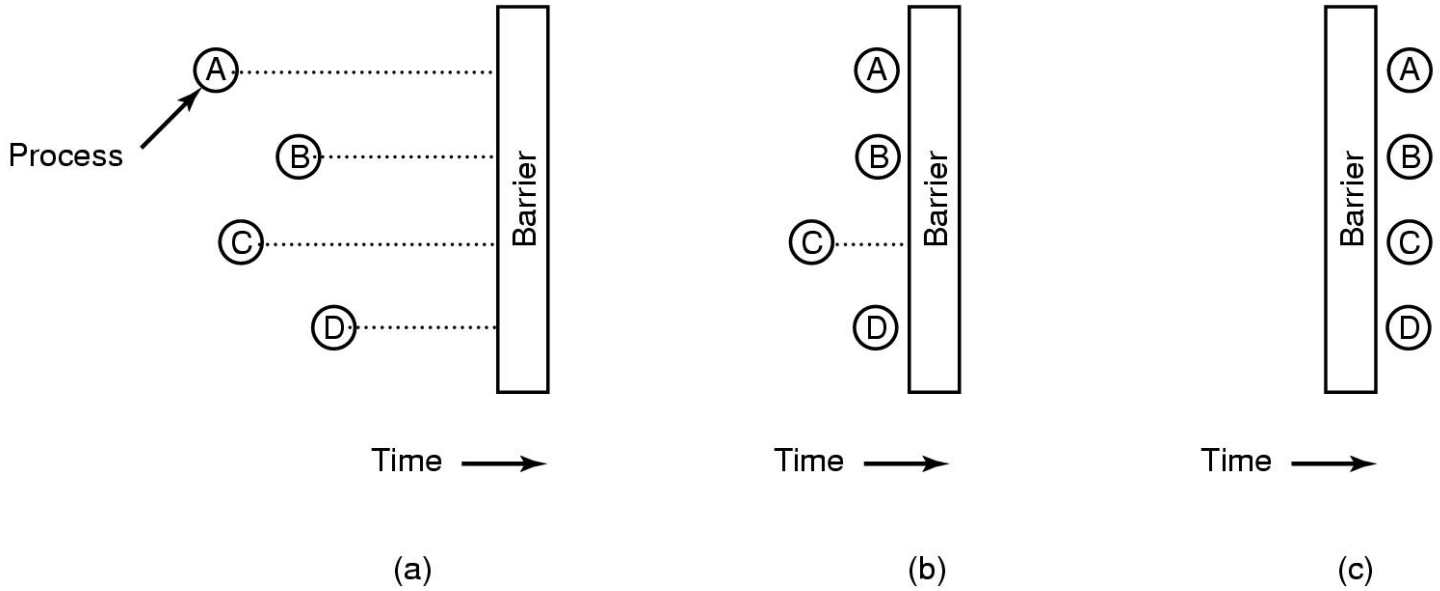
```
monitor example
        integer i;
        condition c;

        procedure producer( );
        .
        .
        .
        end;

        procedure consumer( );
        .       .       .
        end;
end monitor;
```

Monitors (2/2)

```
monitor ProducerConsumer
        condition full, empty;
        integer count;

        procedure insert(item: integer);
        begin
                if count = N then wait(full);
                insert_item(item);
                count := count + 1;
                if count = 1 then signal(empty)
        end;

        function remove: integer;
        begin
                if count = 0 then wait(empty);
                remove = remove_item;
                count := count - 1;
                if count = N - 1 then signal(full)
        end;

        count := 0;
end monitor;

procedure producer;
begin
        while true do
        begin
                item = produce_item;
                ProducerConsumer.insert(item)
        end
end;

procedure consumer;
begin
        while true do
        begin
                item = ProducerConsumer.remove;
                consume_item(item)
        end
end;
```

Πρόβλημα Παραγωγού Καταναλωτή: Barriers

- Use of a barrier. (a) Processes approaching a barrier. (b) All processes but one blocked at the barrier. (c) When the last process arrives at the barrier, all of them are let through.



Δομές Ελέγχου Λ.Σ.

- Το Λ.Σ. πρέπει να έχει πληροφορίες σχετικά με τη τρέχουσα κατάσταση κάθε διεργασίας και πόρου.
- Κατασκευάζονται και συντηρούνται Πίνακες για κάθε οντότητα που διαχειρίζεται το Λ.Σ.

Πίνακες Μνήμης (Memory Tables)

- **Περιέχουν στοιχεία για:**
  - Την Ανάθεση κύρια μνήμης στις διεργασίες.
  - Την Ανάθεση δευτερεύουσας μνήμης στις διεργασίες.
  - Χαρακτηριστικά προστασίας για πρόσβαση σε περιοχές διαμοιραζόμενες μνήμης.
  - Πληροφορίες που απαιτούνται για τη διαχείριση της ιδεατής μνήμης.

Πίνακες Ε/Ε

- Κάθε συσκευή Ε/Ε μπορεί να είναι είτε διαθέσιμη ή να έχει ανατεθεί σε μια διεργασία.
- Το Λ.Σ. πρέπει να γνωρίζει τη κατάσταση λειτουργίας της Ε/Ε καθώς (και).
- Τις θέσεις της κύριας μνήμης που χρησιμοποιούνται ως πηγή ή προορισμός της Ε/Ε μεταφοράς.

Πίνακες Αρχείων

- Περιέχουν στοιχεία για:
  - Την ύπαρξη Αρχείων.
  - Τη θέση τους στη δευτερεύουσα μνήμη.
  - Την τρέχουσα κατάστασή τους.
  - Άλλα χαρακτηριστικά τους.
- Κάποιες φορές αυτές οι πληροφορίες συντηρούνται από ένα σύστημα διαχείρισης αρχείων.

Πίνακες Διεργασίας Process Tables)

Περιέχουν στοιχεία για:

- Το που είναι τοποθετημένη η διεργασία.
- Χαρακτηριστικά της διεργασίας απαραίτητα για τη διαχείρισή της, όπως:
  - Ταυτότητα Διεργασίας (Process ID).
  - Κατάσταση διεργασίας.
  - Θέση στη μνήμη.

Ενέργειες που εκτελεί το ΛΣ κατά τη δημιουργία μας διεργασίας

- Ανάθεση ενός μοναδικού προσδιοριστή διεργασίας.
- Ανάθεση χώρου για τη διεργασία.
- Αρχικοποίηση του μπλοκ ελέγχου των διεργασιών.
- Ρύθμιση των κατάλληλων διασυνδέσεων.
  - Π.χ. προσθήκη διεργασίας στη έτοιμη ή αναστέλλουσα λίστα-ουρά.
- Δημιουργία ή επέκταση άλλων δομών δεδομένων.
  - Π.χ. δημιουργία αρχείου λογιστικής παρακολούθησης για λόγους τιμολόγησης.

Πότε γίνεται αλλαγή διεργασίας

- Διακοπή Ρολογιού (Clock interrupt).

  Η διεργασία εκτελέστηκε για χρόνο ίσο με το μέγιστο επιτρεπτό χρονικό της κομμάτι (slice).
- Διακοπή Ε/Ε (I/O interrupt).
- Σφάλμα Μνήμης (Memory fault).

  Η διεύθυνση μνήμης είναι στην ιδεατή μνήμη και έτσι πρέπει να μεταφερθεί στη κύρια μνήμη (η σελίδα).
- Παγίδα (Trap).
  - Συνέβη λάθος.
  - Μπορεί να οδηγήσει τη διεργασία στην κατάσταση «Σε έξοδο».
- Κλήση επόπτη (Supervisor call).
    - Π.χ. άνοιγμα αρχείου.

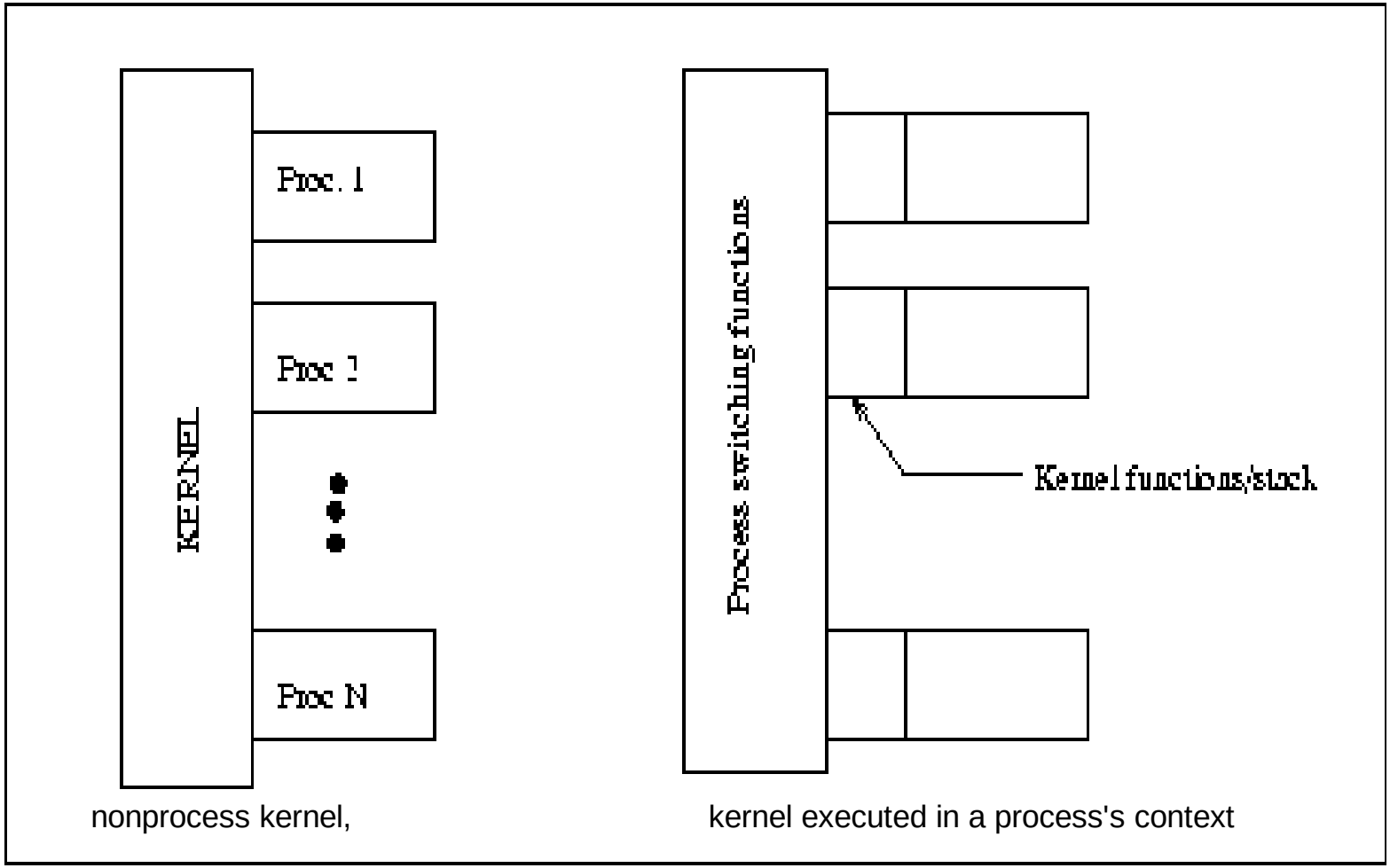
Τι γίνεται στην αλλαγή διεργασίας στο CPU

- Αποθήκευση του περιεχομένου του επεξεργαστή (μαζί με το μετρητή προγράμματος και άλλους καταχωρητές).
- Ενημέρωση του μπλοκ ελέγχου της διεργασίας που επί του παρόντος είναι στην «Εκτελούμενη» κατάσταση.
- Μετακίνηση του μπλοκ ελέγχου της διεργασίας στη κατάλληλη ουρά (ready, blocked…).
- Επιλογή μιας άλλης διεργασίας για εκτέλεση.
- Ενημέρωση του μπλοκ ελέγχου της επιλεγμένης διεργασίας.
- Ενημέρωση των δομών δεδομένων διαχείρισης της μνήμης.
- Επαναφορά του περιεχομένου του επεξεργαστή σύμφωνα με τα στοιχεία της επιλεγμένης διεργασίας.

Το ΛΣ μπορεί να εκτελεστεί με 2 τρόπους (1/2)

- **Μη- Διεργασιακός Πυρήνας** (Non-process Kernel):
  - Εκτέλεση πυρήνα έξω από κάθε διεργασία.
  - Ο κώδικας του Λ.Σ. εκτελείται ως ξεχωριστή οντότητα που λειτουργεί με τον προνομιούχο τρόπο λειτουργίας.
- **Εκτέλεση μέσα σε Διεργασίες Χρήστη:**
  - Το λογισμικό του Λ.Σ. εκτελείται στο πλαίσιο μιας διεργασίας χρήστη.
  - Η διεργασία εκτελείται σε προνοιακή κατάσταση όταν εκτελεί κώδικα του Λ.Σ.

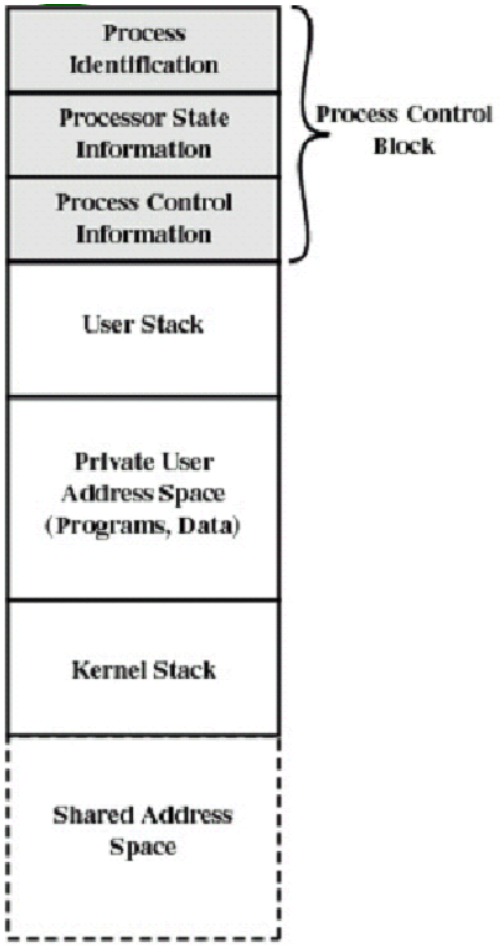
Το ΛΣ μπορεί να εκτελεστεί με 2 τρόπους (2/2)



Non process kernel

- Παραδοσιακή προσέγγιση.
- Δεν έχει τη δομή της διεργασίας.
- Λειτουργεί στη priviledge κατάσταση του επεξεργαστή.

ΛΣ ως διεργασία χρήστη



Χρονοπρογραμματισμός ή χρονοδρομολόγηση (scheduling)

Τι είναι χρονοδρομολόγηση;

- Χρονοδρομολόγηση είναι η στρατηγική χρονικής πολύπλεξης των διεργασιών στη CPU.

Τα σύγχρονα ΛΣ έχουν 3 χρονοδρομολογητές (1/2)

- **Long term scheduling**: which determines which programs are admitted to the system for execution and when, and which ones should be exited.
- **Medium term scheduling**: which determines when processes are to be suspended and resumed;
- **Short term scheduling** (or dispatching): which determines which of the ready processes can have CPU resources, and for how long.

Τα σύγχρονα ΛΣ έχουν 3 χρονοδρομολογητές (2/2)

- **dispatching** affects processes:
  - running;
  - ready;
  - blocked;
- **the medium term** scheduling affects processes:
  - ready-suspended;
  - blocked-suspended;
- **the long term** scheduling affects processes:
  - new;
  - Exite;

Τι είναι ο διεκπεραιωτής (Dispatcher);

- Είναι το τμήμα που έχει την ευθύνη για την ανάθεση CPU στην κάθε διεργασία και περιλαμβάνει :
- Εναλλαγή περιεχομένου διεργασίας (switching context).
- Εναλλαγή στην κατάσταση χρήστη.
- Μετακίνηση στην κατάλληλη θέση του προγράμματος χρήστη για την επανεκκίνηση του προγράμματος.
- Ο διεκπεραιωτής πρέπει να είναι όσο γρήγορος γίνεται, αφού καλείται σε κάθε εναλλαγή διεργασίας.
- Dispatcher latency : ο χρόνος για την παύση μιας διεργασίας και την εκκίνηση μίας άλλης διεργασίας.

Παράδειγμα dispatcher (1/2)

- Έστω 3 διεργασίες που καταλαμβάνουν συγκεκριμένες θέσεις μνήμης:

(a) Trace of Process A	(b) Trace of Process B	(c) Trace of Process C
5000	8000	12000
5001	8001	12001
5002	8002	12002
5003	8003	12003
5004		12004
5005		12005
5006		12006
5007		12007
5008		12008
5009		12009
5010		12010
5011		12011

5000 = Starting address of program of Process A
8000 = Starting address of program of Process B
12000 = Starting address of program of Process C

Παράδειγμα dispatcher (2/2)

1	5000	27	12004
2	5001	28	12005
3	5002	----------	Time out
4	5003	29	100
5	5004	30	101
6	5005	31	102
----------	Time out	32	103
7	100	33	104
8	101	34	105
9	102	35	5006
10	103	36	5007
11	104	37	5008
12	105	38	5009
13	8000	39	5010
14	8001	40	5011
15	8002	----------	Time out
16	8003	41	100
----------	I/O request	42	101
17	100	43	102
18	101	44	103
19	102	45	104
20	103	46	105
21	104	47	12006
22	105	48	12007
23	12000	49	12008
24	12001	50	12009
25	12002	51	12010
26	12003	52	12011
		----------	Time out

100 = Starting address of dispatcher program

shaded areas indicate execution of dispatcher process;
first and third columns count instruction cycles;
second and fourth columns show address of instruction being executed

Αιτίες για τερματισμό διεργασίας (1/2)

- Φυσιολογική ολοκλήρωση.
- Υπέρβαση χρονικού ορίου.
- Μη διαθέσιμη μνήμη.
- Υπέρβαση Ορίων.
- Σφάλμα προστασίας.

Π.χ. Απόπειρα εγγραφής σε read-only αρχείο.

- Αριθμητικό Λάθος.
- Υπερχείλιση χρόνου.
- Η διεργασία περίμενε περισσότερο από ένα καθορισμένο μέγιστο όριο για να προκύψει ένα γεγονός.

Αιτίες για τερματισμό διεργασίας (2/2)

- Σφάλμα Ε/Ε.
- Μη έγκυρη εντολή.

Π.χ. προσπάθεια εκτέλεσης δεδομένων.

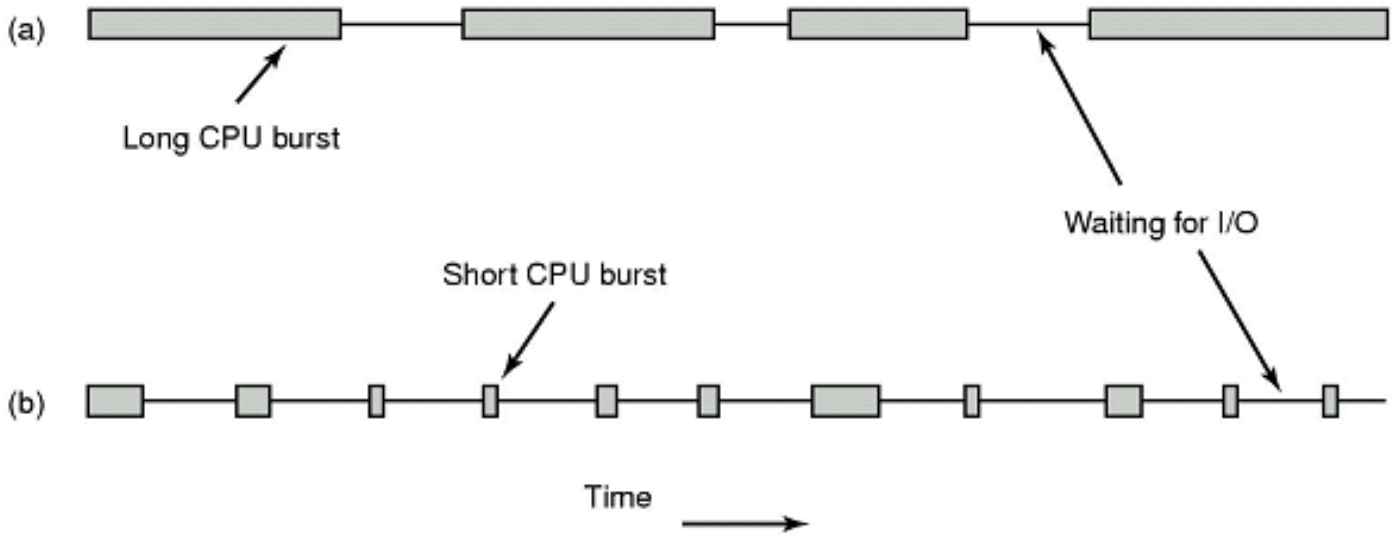
- Προνομιούχα εντολή.
- Λαθεμένη χρήση δεδομένων.
- Μεσολάβηση Λ.Σ.

Π.χ. όταν προκύψει αδιέξοδο (deadlock)

- Τερματισμός διεργασίας Γονέα, μπορεί να προκαλέσει τερματισμό όλων των απογόνων του.
- Αίτηση γονέα.

Βασική έννοια στα ΛΣ είναι ο χρονοπρογραμματισμός (ή χρονοδρομολόγηση)

- Χρήσεις της CPU εναλλάσσονται με περιόδους αναμονής Ε/Ε (I/O wait):
  - CPU-bound διεργασία.
  - I/O bound διεργασία.



Δυο είναι οι κατηγορίες του scheduling ως προς την αποδέσμευση του CPU

- **Μη προεκτοπιστικοί** (non-preemptive) αλγόριθμοι:

  => Κάθε διεργασία εκτελείται μέχρι να μπλοκαριστεί (αναμονή Ε/Ε ή άλλης διεργασίας).

- **Προεκτοπιστικοί** (preemptive) αλγόριθμοι:

  => Χρησιμοποιείται ρολόι διακοπών για να εναλλάσσονται ανοικειοθελώς.

Άλλη κατηγοριοποίηση αλγορίθμων

- **Συστήματα Δέσμης** (ΔΕΝ υπάρχουν αδημονούντες χρήστες!):

  ==> Non-preemptive ή preemptive με μεγάλο quantum.

- **Αλληλεπιδραστικά Συστήματα** (Προέχουν οι χρήστες):

  ==> Preemptive με σωστά επιλεγμένο quantum.

- **Συστήματα Πραγματικού Χρόνου:**

  ==> Παραδόξως, απαιτείται μικρότερος έλεγχος, διότι είναι κατάλληλα προγραμματισμένες οι διεργασίες.

Κριτήρια χρονοδρομολόγησης (1/2)

- **Χρησιμοποίηση CPU**: όσο γίνεται απασχολημένη.
- **Φόρτος εργασίας**: # διεργασιών που ολοκληρώνουν την εκτέλεση τους ανά μονάδα χρόνου.
- **Χρόνος επιστροφής** (turnaround time): χρονικό διάστημα από την υποβολή μιας διεργασίας έως και την ολοκλήρωση της (δηλ. άθροισμα χρόνων αναμονής για μνήμη, ουρά αναμονής, εκτέλεση στη CPU και I/O).
- **Χρόνος αναμονής**: ο χρόνος που περνά μια διεργασία στην ουρά έτοιμων διεργασιών.
- **Χρόνος απόκρισης** (response time): χρονικό διάστημα που μεσολαβεί από τη στιγμή του αιτήματος έως την έναρξη της πρώτης απόκρισης.

Κριτήρια χρονοδρομολόγησης (2/2)

- **Σεβασμός των deadlines** : αν υπάρχει ανάγκη να ολοκληρωθεί κάποια διεργασία μέσα σε αυστηρά χρονικά πλαίσια.
- **Προβλεψιμότητα**: η ιδιότητα του συστήματος να εγγυηθεί ότι κάποια διεργασία θα εκτελεστεί μέσα σε ένα λογικό χρονικό πλαίσιο, ανεξαρτήτου του φόρτου του συστήματος.
- **Δικαιοσύνη (fairness):** Όλες οι διεργασίες πρέπει κάποια στιγμή να εκτελεστούν στο CPU ή να τους δοθούν οι πόροι που χρειάζονται.
- **Επιβολή προτεραιοτήτων**: Να επιτρέπεται κάποιες διεργασίες να έχουν καλύτερη μεταχείριση, αλλά μέσα στα πλαίσια της δικαιοσύνης.

Βασικά θέματα για το χρονοδρομολογητή dispatcher

- Είναι το πιο σημαντικό κομμάτι, γιατί επηρεάζει άμεσα το σύστημα.
- Είναι γραμμένος σε assembly.
- Η αλλαγή διεργασίας στο CPU πρέπει να γίνει όσο το δυνατόν πιο γρήγορα (λίγα microsec). Πρέπει η πληροφορία που αποθηκεύεται και επαναφέρεται σε κάθε αλλαγή διεργασίας να είναι ελάχιστη. Μια υλοποίηση είναι να αλλάζει μόνο η διεύθυνση του PCB που διατηρείται στη μνήμη και όχι όλο το PCB.

Βελτιστοποίηση δρομολόγησης

- Μεγιστοποίηση Χρήσης CPU.
- Μεγιστοποίηση Φόρτου εργασίας.
- Ελαχιστοποίηση χρόνου επιστροφής.
- Ελαχιστοποίηση χρόνου αναμονής.
- Ελαχιστοποίηση χρόνου απόκρισης.

Στόχοι του scheduling

- **Για όλα τα συστήματα:**
  - Δικαιοσύνη (δίκαιο μερίδιο της CPU).
  - Επιβολή της πολιτικής (παρακολούθηση).
  - Ισορροπία (ενεργά όλα τα τμήματα του συστήματος)
- **Συστήματα Δέσμης:**
  - Διεκπεραιωτική ικανότητα (μέγιστος αριθμός εργασιών ανά ώρα).
  - Χρόνος διεκπεραίωσης (ελαχιστοποίηση χρόνου από την υποβολή μέχρι την περαίωση).
  - Χρήση CPU (διαρκώς ενεργή).
- **Αλληλεπιδραστικά Συστήματα:**
  - Χρόνος απόκρισης (ταχύτατη απόκριση).
  - Τήρηση αναλογιών (ικανοποίηση προσδοκιών χρηστών).
- **Συστήματα Πραγματικού Χρόνου:**
  - Τήρηση προθεσμιών (να αποφεύγεται απώλεια δεδομένων).
  - Προβλεψιμότητα (να αποφεύγεται ο υποβιβασμός της ποιότητας πολυμέσων).

Scheduling σε συστήματα δέσμης

- **Non-preemptive First Come First Served:**

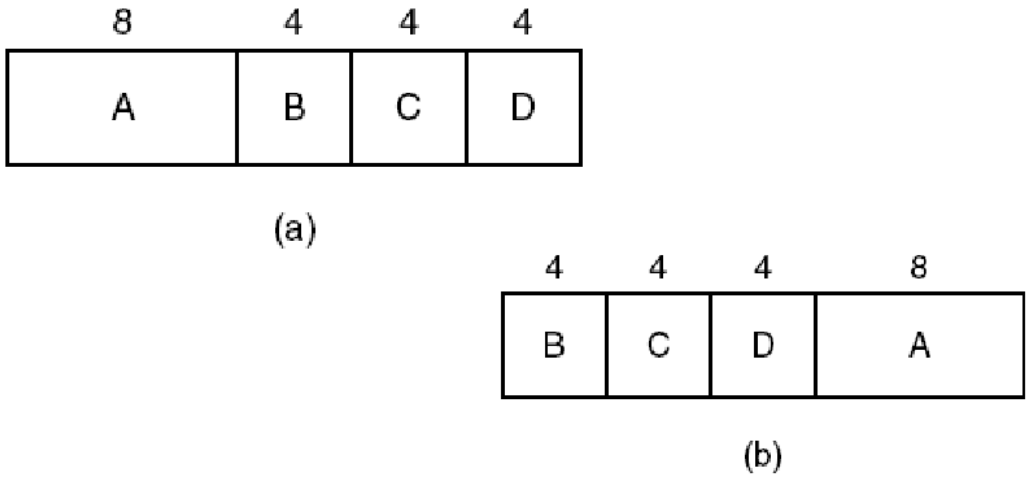
  Η απλούστερη λύση αλλά και αφελέστερη.
- **Non-preemptive Shortest Job First:**

  Βέλτιστος μέσος χρόνος διεκπεραίωσης.
- **Shortest Remaining Time First:**

  Εκτελεί preemption, αλλά όχι με βάση quantum.
- **Scheduling Τριών Επιπέδων.**

Χρονοδρομολόγηση SFJ (shortest job first)

- (a) First Come First Served (non-preemptive).
- (b) Shortest Job First (non-preemptive).



Παράδειγμα FCFS (1/2)

- Έστω σε ένα σύστημα δημιουργούνται οι διεργασίες P1,P2,P3 με αυτή τη σειρά και απαιτούμενους χρόνους ολοκλήρωσης 24,3,3 αντίστοιχα.
- Να υπολογιστεί ο μέσος χρόνος αναμονής.

- Χρόνος Αναμονής P1=0, P2=24,P3=27.
- Μέσος χρόνος αναμονής (0+24+27)/3=17.

Παράδειγμα FCFS (2/2)

- Έστω σε ένα σύστημα δημιουργούνται οι διεργασίες P2,P3,P1 με αυτή τη σειρά και απαιτούμενους χρόνους ολοκλήρωσης 3,3,24 αντίστοιχα.
- Να υπολογιστεί ο μέσος χρόνος αναμονής.

- Χρόνος Αναμονής P2=0,P3=3,P1=6.
- Μέσος χρόνος αναμονής (6+0+3)/3=3.

Προτιμότερο: Οι σύντομες διεργασίες πριν από τις χρονοβόρες.

Κατηγορίες SJF (=SPN)

- **Μη-προεκχωρημένη (nonpreemptive):**

Από τη στιγμή που η CPU ανατίθεται σε μια διεργασία, η CPU δεν μπορεί να προ εκχωρηθεί μέχρι να ολοκληρωθεί η διεργασία.

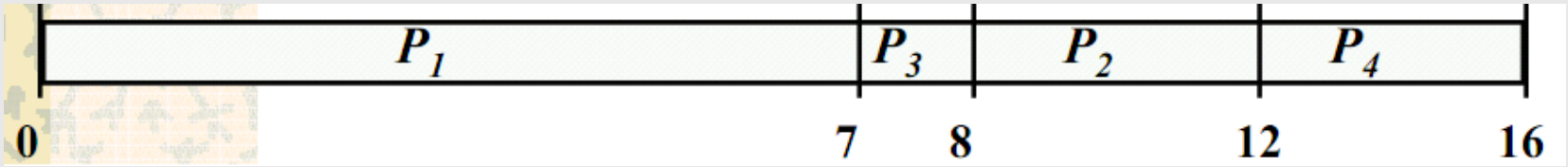
- **Προεκχωρημένη (preemptive):**

Εάν φθάσει μια διεργασία με απαιτούμενο χρόνο ολοκλήρωσης CPU μικρότερου μήκους από τον χρόνο που απομένει για την τρέχουσα εκτελούμενη διεργασία έχουμε προεκχώρηση. Αυτό ονομάζεται: { Shortest Remaining Time First (SRTF)}.

Ο SJF είναι βέλτιστος: καταλήγει σε ελάχιστο μέσο χρόνο αναμονής για δεδομένο σύνολο διεργασιών.

Παράδειγμα με SJF

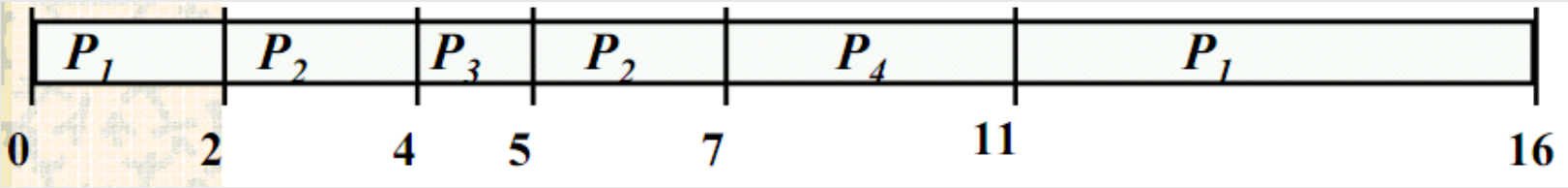
- Έστω καταφτάνουν οι διεργασίες P1,P2,P3,P4 τους χρόνους 0,2,4,5 με απαιτούμενο χρόνο εκτέλεσης 7,4,1,4 αντίστοιχα.
- Να υπολογιστεί ο μέσος χρόνος αναμονής:



- Μέσος Χρόνος αναμονής : (0+6+3+7)/4 = 4.

Παράδειγμα με Shortest Remaining Time First (SRTF)

- Να υπολογιστεί ο μέσος χρόνος αναμονής για τις διεργασίες του προηγούμενου παραδείγματος και τον αλγόριθμο SRTF.



- Μέσος Χρόνος αναμονής : (9+1+0+2)/4 = 3.

Υπολογισμός απαιτουμένου χρόνου ολοκλήρωσης

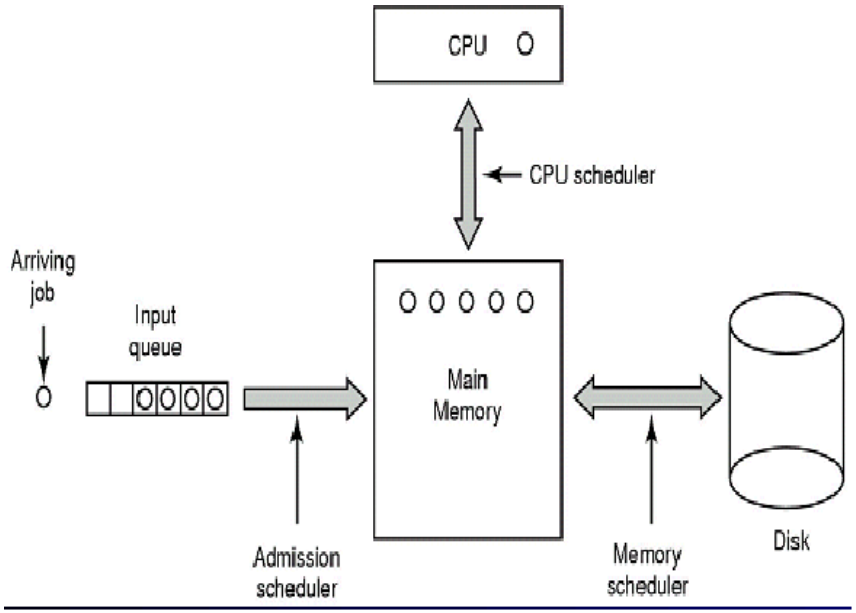
- Δε μπορούμε να γνωρίζουμε ακριβώς τον απαιτούμενο χρόνο ολοκλήρωσης μιας διεργασίας.
- Μπορούμε να τον εκτιμήσουμε από την παρελθοντική της συμπεριφορά.
- Ορίζουμε:
  - T_n = πραγματικό μήκος του n-οστού CPU χρόνου.
  - ψ_n = προβλεπόμενη τιμή του n-οστού CPU χρόνου.
  - $0 \le W \le 1$, $\psi_{n+1} = W*T_n + (1-W)\,\psi_n$.

    Αν W=0, τότε η πιο πρόσφατη συμπεριφορά αγνοείται.

    Αν W=1, τότε μόνο η πρόσφατη συμπεριφορά χρησιμ.

Χρονοδρομολόγηση τριών επιπέδων

- Κριτήρια Memory Scheduler:
  - Πόσος χρόνος πέρασε από τη στιγμή που η διεργασία μεταφέρθηκε στη μνήμη ή στο δίσκο;
  - Πόσο χρόνο CPU είχε πρόσφατα στη διάθεσής της η διεργασία;
  - Ποιο είναι το μέγεθος της διεργασίας; (οι μικρές δεν ενοχλούν).
  - Πόσο σημαντική είναι η διεργασία;



Χρονοδρομολόγηση σε αλληλεπιδραστικά συστήματα

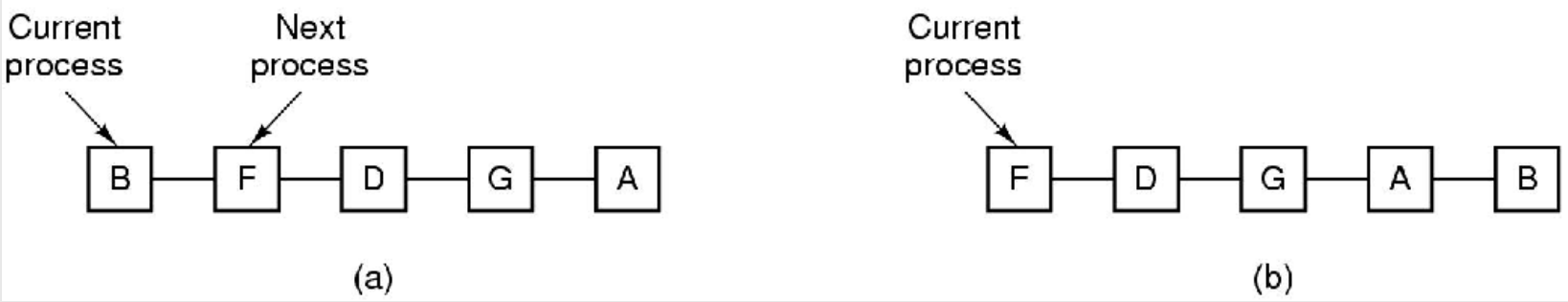
- Round Robin:
  - Η preemptive εκδοχή του FCFS.
- Priority Scheduling:
  - Θα πρέπει να δίνεται η δυνατότητα και στις διεργασίες χαμηλότερων προτεραιοτήτων.
- Πολλαπλές Ουρές.
- Εξυπηρέτηση με βάση τη μικρότερη διάρκεια:
  - Η preemptive εκδοχή του SJF.
- Εγγυημένος Χρονοπρογραμματισμός.
- Lottery Scheduling.
- Χρονοπρογραμματισμός Δίκαιης Κατανομής.

Χρονοδρομολόγηση RoundRobin (RR) (1/3)

- Κάθε διεργασία παίρνει μια μικρή ενότητα του CPU χρόνου (quantum) περίπου 10-100 millisecs. Μετά την πάροδο αυτού του χρόνου, η διεργασία προεκχωρείται και προστίθεται στο τέλος της ουράς έτοιμων διεργασιών.
- Εάν υπάρχουν n διεργασίες στην ουρά έτοιμων διεργασιών και η ενότητα χρόνου (quantum) είναι q κάθε διεργασία παίρνει 1/ n του CPU χρόνου σε κομμάτια το πολύ έως q ενοτήτων χρόνου κάθε φορά.
- Καμία διεργασία δεν περιμένει περισσότερο από (n-1) q ενότητες χρόνου.

Χρονοδρομολόγηση RoundRobin (RR) (2/3)

- Λογικό quantum χρόνου: 10msec – 100 msec.



(a) Η τρέχουσα λίστα των READY διεργασιών.

(b) Η λίστα των READY διεργασιών, ύστερα από τη χρησιμοποίηση του quantum της Β.

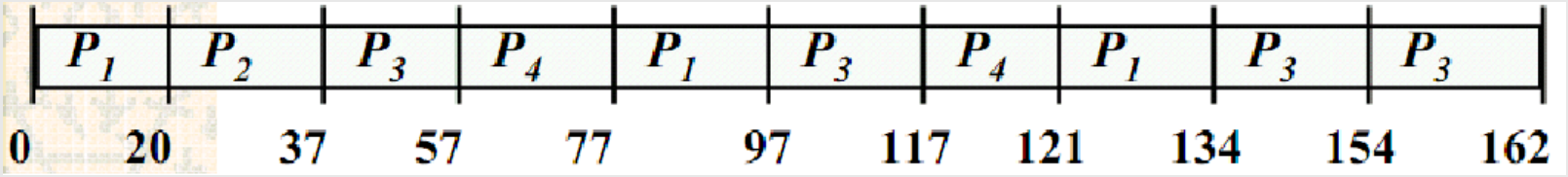
Χρονοδρομολόγηση RoundRobin (RR) (3/3)

Επιδόσεις:

- Μεγάλο q ==> FIFO.
- Μικρό q ==> το q πρέπει να είναι μεγάλο σχετικά με την εναλλαγή περιεχομένου, διαφορετικά το κόστος (overhead) είναι μεγάλο.

Παράδειγμα (RR)

- Έστω ενότητα χρόνου quantum 20.
- Διεργασίες P1,P2,P3,P4 με χρόνους CPU 53,17,68,24.

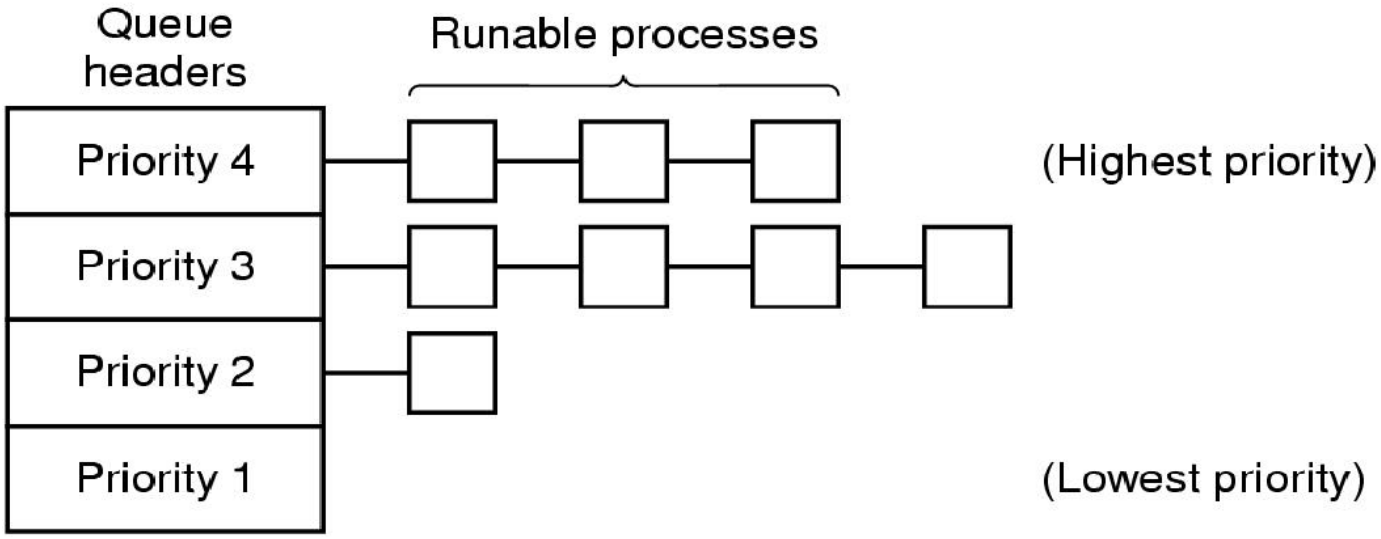


Συνήθως, μεγαλύτερο μέσο όρο χρόνου διεκπαιρέωσης(turnaround) από το SRT, αλλά καλύτερη αποκρισιμότητα.

Χρονοδρομολόγηση προτεραιοτήτων (1/3)

- Ένας ακέραιος αριθμός συνδέεται με κάθε διεργασία.
- Η CPU κατανέμεται στη διεργασία σύμφωνα με την υψηλότερη προτεραιότητα. Συνήθως ο μικρότερος ακέραιος έχει τη μέγιστη προτεραιότητα.
- Υπάρχουν δύο τύποι:
  - Μη- προεκχώρηση (nonpreemptive).
  - Προεκχώρηση (preemptive).

Χρονοδρομολόγηση προτεραιοτήτων (2/3)



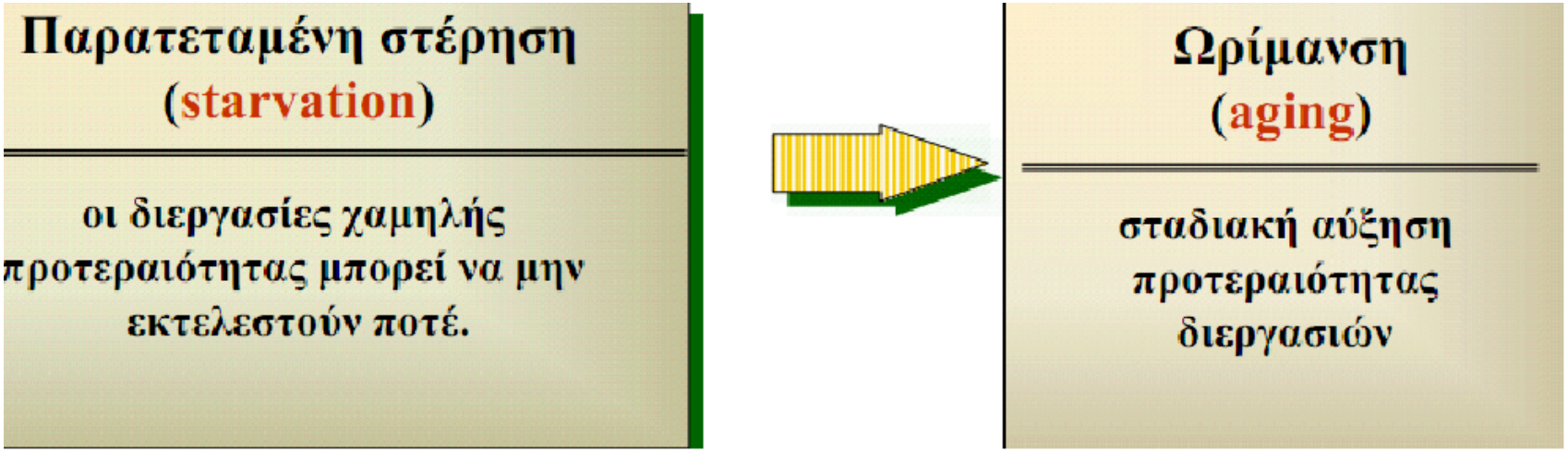
Ένας αλγόριθμος χρονοδρομολόγησης με 4 προτεραιότητες.

Χρονοδρομολόγηση προτεραιοτήτων (3/3)

- Σε κάθε χρονική στιγμή εκτελείται η έτοιμη διεργασία με την υψηλότερη προτεραιότητα.
- Ενδεχόμενη αλλαγή προτεραιοτήτων για την αποφυγή λιμοκτονίας.
- Για την εξυπηρέτηση I/O Bound processes, μπορεί να οριστεί προτεραιότητα: 1/f, όπου f το κλάσμα του τελευταίου κβάντου, που χρησιμοποίησε η διεργασία.
- Μπορούμε να ομαδοποιούμε διεργασίες ίδιας προτεραιότητας και να εφαρμόζουμε άλλον αλγόριθμο scheduling εντός της ίδιας ομάδας.

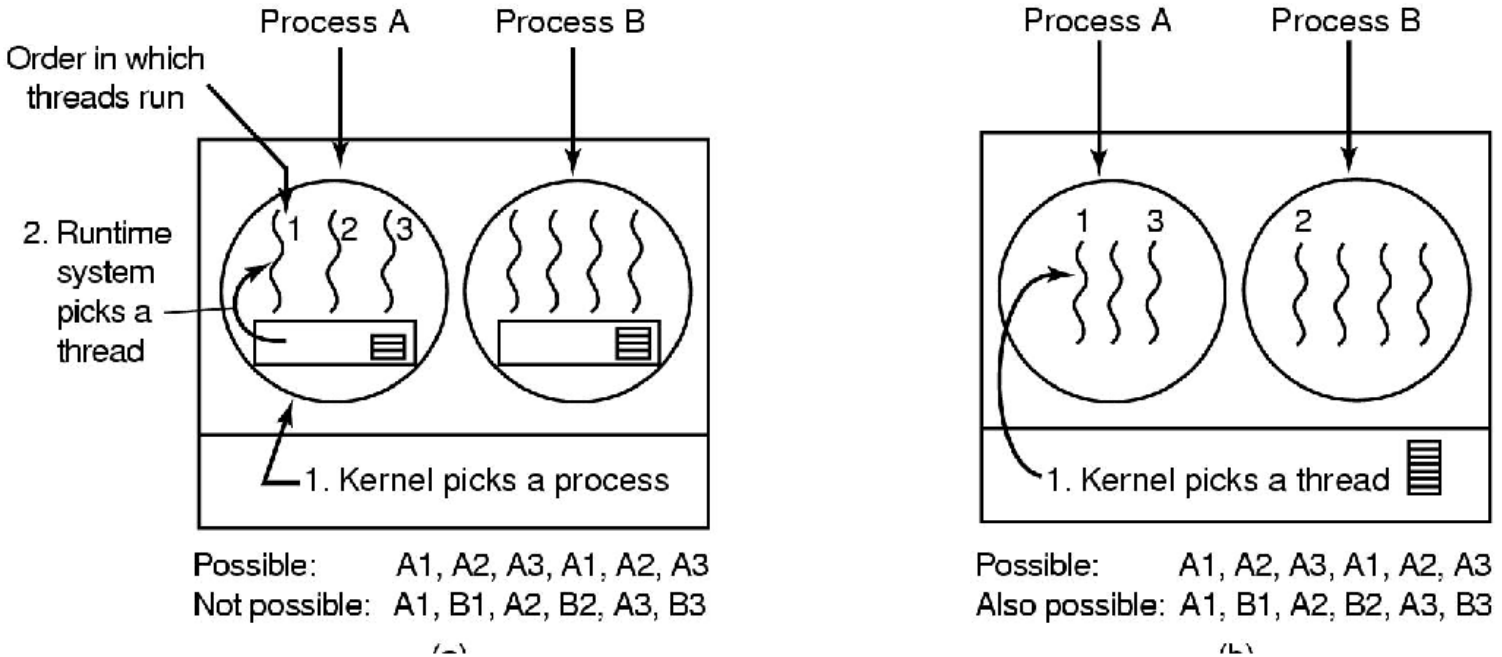
Πρόβλημα λιμοκτονίας..

Αν δεν τροποποιούνται οι προτεραιότητες τότε κάποιες διεργασίες μπορεί να μην εκτελεστούν ποτέ.



Χρονοδρομολόγηση νημάτων

- (a) Χρονοδρομολόγηση νημάτων χρήστη.
- (b) Χρονοδρομολόγηση νημάτων πυρήνα.



Πολλαπλές ουρές (1/3)

- Στον Η/Υ CTSS ο αριθμός των κβάντων που εκχωρούνταν σε μια διεργασία διπλασιαζόταν, με ταυτόχρονη μείωση της προτεραιότητας της (τάξεις προτεραιοτήτων).
- Εμφανίστηκαν πολλές εκδοχές τάξεων προτεραιοτήτων (π.χ. τερματικού, Ε/Ε, μικρού κβάντου, μεγάλου κβάντου).

Πολλαπλές ουρές (2/3)

- Η ουρά έτοιμων διεργασιών χωρίζεται σε 2 ξεχωριστές ουρές. Παράδειγμα :
  - Προσκήνιο (διαλογική, interactive).
  - Παρασκήνιο (ομαδική επεξεργασία, batch).
- Κάθε ουρά έχει τον δικό της αλγόριθμο δρομολόγησης.
- Παράδειγμα :
  - Προσκήνιο (RR).
  - Παρασκήνιο (FCFS).

Πολλαπλές ουρές (3/3)

- Δρομολόγηση πρέπει να επιβληθεί και μεταξύ των ουρών.
- **Δρομολόγηση σταθερής προτεραιότητας:**

  Παράδειγμα: εξυπηρέτηση όλου του προσκήνιου και στη συνέχεια από το παρασκήνιο.

  ==> Πιθανότητα παρατεταμένης στέρησης.
- **Δρομολόγηση κομματιών χρόνου:**

  κάθε ουρά έχει ένα συγκεκριμένο χρόνο CPU που μπορεί να δρομολογήσει μεταξύ των διεργασιών της.

  Παράδειγμα: 80% στο προσκήνιο με RR.

  20% στο παρασκήνιο με FCFS.

Ουρά Πολλαπλής Ανάδρασης

- Μία διεργασία μπορεί να μετακινηθεί μεταξύ των διαφόρων ουρών. Με αυτό τον τρόπο μπορεί να υλοποιηθεί η ωρίμανση.
- Η ουρά Πολλαπλής Ανάδρασης καθορίζεται από τις εξής παραμέτρους
  - Αριθμό Ουρών.
  - Αλγόριθμος δρομολόγησης για κάθε ουρά.
  - Μέθοδος που καθορίζει πότε θα γίνει η αναβάθμιση μιας διεργασίας.
  - Μέθοδος που καθορίζει πότε θα γίνει η υποβάθμιση μιας διεργασίας.
  - Μέθοδος που καθορίζει την ουρά στην οποία θα εισαχθεί μια διεργασία που απαιτεί εξυπηρέτηση.

Μικρότερη Διάρκεια (1/2)

- Είναι δύσκολο να γνωρίζουμε τον ακριβή χρόνο εκτέλεσης μιας αλληλεπιδραστικής διεργασίας.

- Βασιζόμαστε στην «προηγούμενη» συμπεριφορά της: σταθμισμένο άθροισμα χρόνων εκτέλεσης.

Μικρότερη Διάρκεια (2/2)

- Αν T_0 η αρχική πρόβλεψη της διεργασίας και T_1 ο πραγματικός χρόνος εκτέλεσης, της αποδίδεται εκτιμώμενος χρόνος:

$$\alpha T_0 + (1-\alpha)T_1$$

- Αν $\alpha=1/2$ οι διαδοχικές εκτιμήσεις θα είναι:T_0, $T_0/2+T_1/2$, $T_0/4+T_1/4+T_2/2$, $T_0/8+T_1/8+T_2/4+T_3/2$.
- Η επιλογή $\alpha=1/2$ βολεύει, διότι προσθέτουμε μια νέα τιμή και κάνουμε μια ολίσθηση προς τα δεξιά.

Εγγυημένος Χρονοπρογραμματισμός

- Εγγυόμαστε μια συνθήκη: π.χ. Οι N διεργασίες χρηστών θα λάβουν 1/N κλάσμα χρόνου της CPU.
- Είναι εύκολο να βρίσκουμε ποια διεργασία έχει λόγο μικρότερο του 1 (δε χρησιμοποίησε το μερίδιο που της αναλογεί) και να την χρονοπρογραμματίζουμε.

Lottery Scheduling

- Νέα ιδέα! (1994).
- Είναι δύσκολο να υλοποιούμε προτεραιότητες,οπότε καλύτερα να βασιστούμε στις πιθανότητες.
- Εκτελούνται τακτικές κληρώσεις διεργασιών.
- Στις σημαντικές διεργασίες δίνονται περισσότεροι λαχνοί.

Χρονοπρογραμματισμός Δίκαιης Κατανομής

- Ιδέα παρόμοια με τον εγγυημένο χρονοπρογραμματισμό, αλλά σε επίπεδο χρήστη.
- Αν ένας χρήστης ξεκινά 4 διεργασίες (Α,Β,Γ,Δ) και ένας άλλος μια (Ε), τότε:
  - Αν ο κάθε χρήστης δικαιούται 50% του χρόνου CPU η σειρά είναι: Α Ε Β Ε Γ Ε Δ Ε Α Ε Β Ε Γ Ε Δ Ε …
  - Αν ο πρώτος χρήστης δικαιούται 2/3 του χρόνου CPU η σειρά είναι: Α Β Ε Γ Δ Ε Α Β Ε Γ Δ Ε …

Scheduling Συστημάτων Πραγματικού Χρόνου (1/2)

- Υπάρχουν περιοδικά και απεριοδικά γεγονότα.
- Δίνονται:
  - m περιοδικά γεγονότα.
  - Το γεγονός i έχει περίοδο Pi και απαιτεί Ci sec.
- Το σύστημα είναι χρονοπρογραμματίσιμο (schedulable) αν:

$$\sum_{i=1}^{m} \frac{C_i}{P_i} \leq 1$$

Scheduling Συστημάτων Πραγματικού Χρόνου (2/2)

- Παράδειγμα: 3 γεγονότα με περίοδο 100, 200, 500 msec απαιτούν 50, 40 και 100 msec χρόνο CPU, αντίστοιχα.
- Από τον τύπο:

  $50/100 + 40/200 + 100/500 = 0,5 + 0,2 + 0,2 = 0,9 < 1.$

- Το σύστημα είναι χρονοπρογραμματίσιμο και αφήνει και κάποιο μικρό ακόμη περιθώριο (0,1).

Πολιτική vs Μηχανισμό

- Διάκριση του τι επιτρέπεται να γίνει από το πώς γίνεται:
  - Μία διεργασία γνωρίζει ποια από τα παιδιά της είναι σημαντικά και χρειάζονται προτεραιότητα.
- Πραγματοποιήσιμος αλγόριθμος χρονοπρογραμματισμού:
  - Μηχανισμός εντός του πυρήνα.
- Οι παράμετροι καθορίζονται από τις διεργασίες των χρηστών:
  - Η πολιτική καθορίζεται από τις διεργασίες χρήστη.

Δρομολόγηση Πολλαπλών Επεξεργαστών

- Η δρομολόγηση της CPU είναι πιο περίπλοκη όταν πολλαπλές CPU είναι διαθέσιμες.
- Ομογενείς επεξεργαστές με πολυ-επεξεργαστή.
- Διαμοίραση φορτίου.
- Ασύμμετρη πολυ-επεξεργασία : μόνο ένας επεξεργαστής έχει πρόσβαση στις δομές δεδομένων του συστήματος.

Δρομολόγηση Πραγματικού Χρόνου

- **“Αυστηρά“ συστήματα Πραγματικού Χρόνου:**

  απαιτείται να ολοκληρώνουν μια κρίσιμη ενέργεια σε συγκεκριμένο χρόνο.

- **“Χαλαρά“ συστήματα Πραγματικού Χρόνου :**

  απαιτείται να δίνεται προτεραιότητα στις κρίσιμες διεργασίες.

Χρονοδρομολόγηση στο 4.2BSD Unix

- Η χρονοδρομολόγηση γίνεται με το timeout, το οποίο λέει στο clock interrupt driver ποια συνάρτηση να εκτελέσει ύστερα από κάποιο χρονικό διάστημα.

- Η ρουτίνα είναι συνήθως η ρουτίνα χρονοδρομολόγησης, η οποία αποφασίζει τι θα εκτελέσει και αμέσως θέτει πάλι το timeout με συνάρτηση εκτέλεσης τον εαυτό της.

Άσκηση

- Θεωρούμε το παρακάτω σύνολο διεργασιών:

(διεργασία,άφιξη,χρόνος CPU):

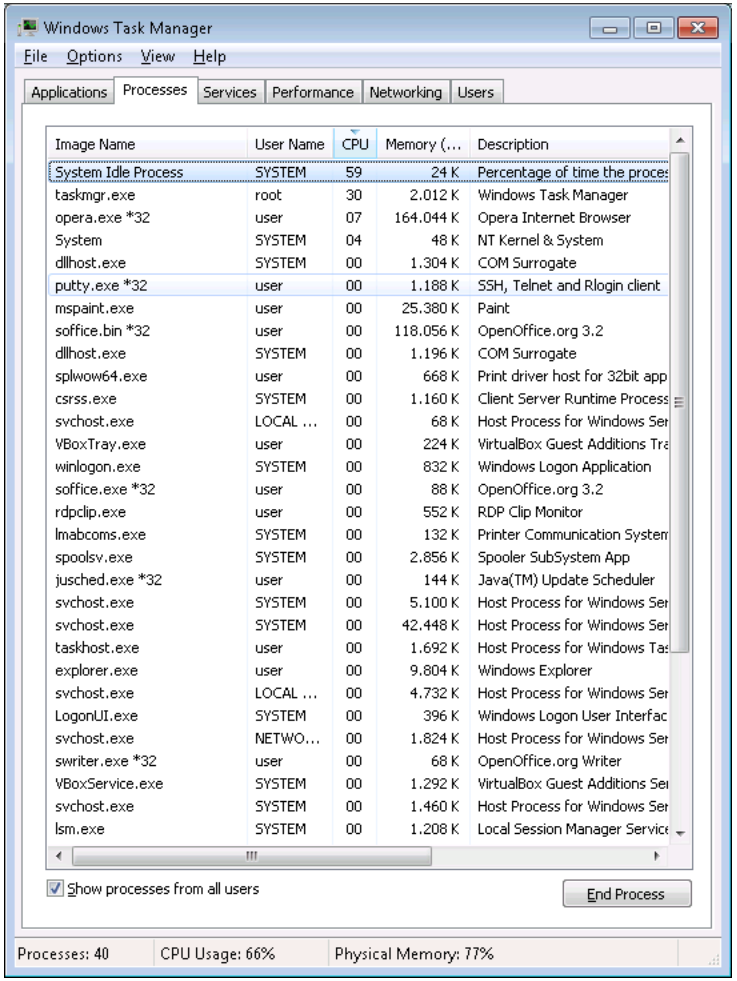
(P1,10,3) , (P2,1,1) , (P3,2,3), (P4,1,4), (P5,5,2).

- Έστω ότι οι διεργασίες φθάνουν με τη σειρά P1 ,P2 ,P3 ,P4, P5 κατά τη χρονική στιγμή 0. Να σχεδιασθεί το διάγραμμα Gantt για δρομολόγηση FCFS, SJF, μη-προεκχωρήσιμη προτεραιότητα, RR(quantum=1).
- Να υπολογισθεί ο χρόνος αναμονής για κάθε μια διεργασία ανά διαφορετικό αλγόριθμο δρομολόγησης.
- Ποιος αλγόριθμος καταλήγει στον ελάχιστο μέσο χρόνο αναμονής;

Διεργασίες στο FreeBSD 8

```
root            0  0.1  0.0     0   800  ??  DLs  19Jan11  66:36.37 [kernel]
root            1  0.0  0.0  3204   120  ??  ILs  19Jan11   0:01.01 /sbin/init --
root            2  0.0  0.0     0    16  ??  DL   19Jan11   3:18.59 [g_event]
root            3  0.0  0.0     0    16  ??  DL   19Jan11  91:30.90 [g_up]
root            4  0.0  0.0     0    16  ??  DL   19Jan11 312:03.65 [g_down]
root            5  0.0  0.0     0    16  ??  DL   19Jan11   0:00.00 [xpt_thrd]
root            6  0.0  0.0     0    16  ??  DL   19Jan11  32:23.77 [pagedaemon]
root            7  0.0  0.0     0    16  ??  DL   19Jan11   0:00.04 [vmdaemon]
root            8  0.0  0.0     0    16  ??  DL   19Jan11   0:00.05 [pagezero]
root            9  0.0  0.0     0    16  ??  DL   19Jan11   0:17.40 [bufdaemon]
root           10  0.0  0.0     0    16  ??  DL   19Jan11   0:00.00 [audit]
root           11  0.2  0.0     0    32  ??  RL   19Jan11 127702:59.63 [idle]
root           12  0.0  0.0     0   272  ??  WL   19Jan11 882:10.55 [intr]
root           13  0.0  0.0     0    16  ??  DL   19Jan11  16:13.48 [yarrow]
root           14  0.0  0.0     0   256  ??  DL   19Jan11   2:57.19 [usb]
root           15  0.0  0.0     0    16  ??  DL   19Jan11  27:44.29 [TIMER]
root           16  0.3  0.0     0    16  ??  DL   19Jan11  76:23.28 [syncer]
root           17  0.0  0.0     0    16  ??  DL   19Jan11   0:39.17 [vnlru]
root           18  0.0  0.0     0    16  ??  DL   19Jan11   0:54.33 [softdepflush]
root           19  0.0  0.0     0    16  ??  DL   19Jan11   1:29.41 [flowcleaner]
root           38  0.0  0.0     0    60  ??  DL   19Jan11  15:44.77 [zfskern]
root           91  0.0  0.0     0    16  ??  DL   19Jan11   0:01.18 [md0]
root           96  0.0  0.0     0    16  ??  DL   19Jan11   0:02.22 [md1]
root          112  0.0  0.0     0    32  ??  DL   19Jan11   0:00.00 [ng_queue]
root          139  0.0  0.0  1712     0  ??  IWs  -         0:00.00 adjkerntz -i
root          661  0.0  0.0  3200    16  ??  Is   19Jan11   0:00.32 /sbin/devd
```

Διεργασίες σε Windows



Τι είναι η διεργασία idle;

- Είναι νήματα πυρήνα για κάθε επεξεργαστή τα οποία εκτελούνται όταν δεν υπάρχει καμία άλλη διεργασία να εκτελεστεί (η ουρά READY είναι άδεια).
- Χρησιμοποιείται για να εξαλείψει την ειδική περίπτωση που δεν υπάρχει κάποια διεργασία για δρομολόγηση, κάτι που προκαλεί πρόβλημα στον αλγόριθμο χρονοδρομολόγησης.
- Ο χρονοδρομολογητής ποτέ δεν επιλέγει την idle, αν υπάρχει άλλη διεργασία στο READY.
- Ο χρόνος CPU που αναφέρεται στην idle είναι ο χρόνος που δε θέλει καμία άλλη διεργασία.

Τέλος Ενότητας

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης