

# Παράλληλη Επεξεργασία

## Κεφάλαιο 5

### Μοίρασμα Δεδομένων

Κωνσταντίνος Μαργαρίτης  
Καθηγητής

Τμήμα Εφαρμοσμένης Πληροφορικής

Πανεπιστήμιο Μακεδονίας

[kmarg@uom.gr](mailto:kmarg@uom.gr)

<http://eos.uom.gr/~kmarg>

Αρετή Καππάν

Υποψήφια Διδάκτορας

Τμήμα Εφαρμοσμένης Πληροφορικής

Πανεπιστήμιο Μακεδονίας

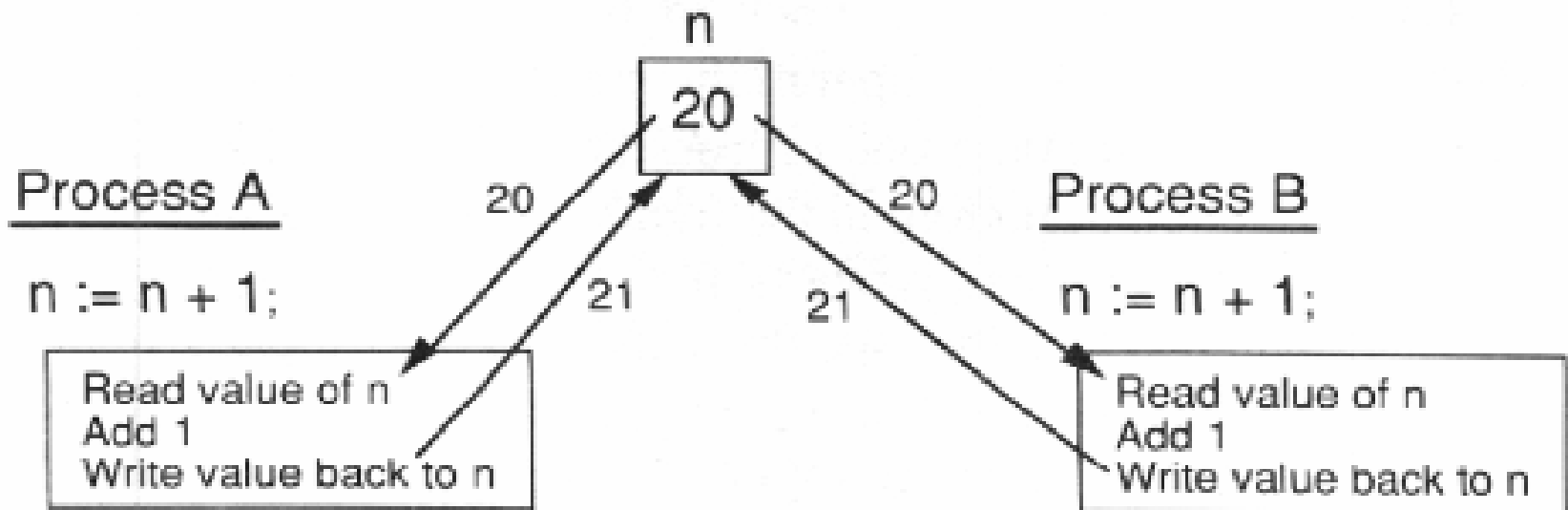
[areti@uom.gr](mailto:areti@uom.gr)

<http://eos.uom.gr/~areti>

# Εισαγωγή

- ⇒ Ορισμός ατομικής λειτουργίας
- ⇒ Τεχνική κλειδώματος
- ⇒ Αποκλειστική πρόσβαση στα διαμοιραζόμενα δεδομένα
- ⇒ Αναμονή των διεργασιών - Ανταγωνισμός για πρόσβαση στη μνήμη
- ⇒ Συμφόρηση
- ⇒ Αποκέντρωση του μηχανισμού πρόσβασης στην μνήμη

# Σφάλμα Συγχρονισμού



# Υλοποίηση Κλειδωμάτων

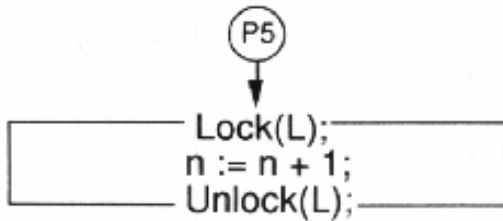
```
PROGRAM Search;
VAR  A : ARRAY [1..200] OF INTEGER;
     i, val, n : INTEGER;
     L : SPINLOCK;

BEGIN

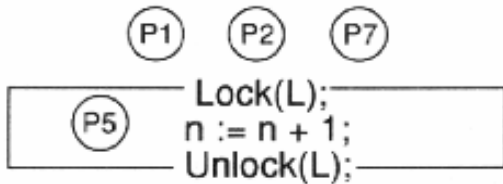
.....

n := 0;
Readln(val);
FORALLL i := 1 TO 200 GROUPING 10 DO
  IF A[i] = VAL THEN
  BEGIN
    Lock(L);  (*Εισέρχεται και ενεργοποιεί το κλείδωμα*)
    n := n + 1;
    Unlock(L);  (*Εξέρχεται και απενεργοποιεί το κλείδωμα*)
  END;
WRITELN('Total occurrences : ', n);

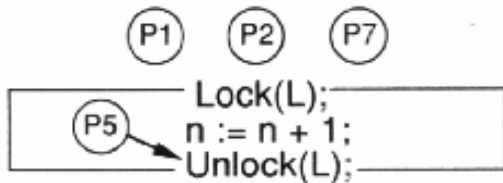
.....
```



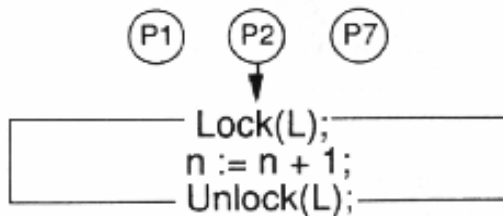
L initially unlocked  
P5 enters and locks L



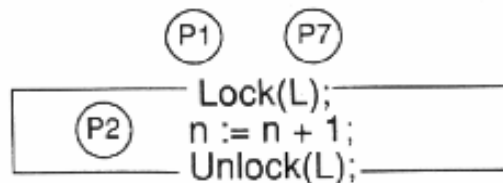
L is now locked  
P1, P2, P7 kept out



P5 unlocks L as it leaves



L is now unlocked  
P2 enters and locks L



L is now locked  
P1, P7 kept out

# Η επίδραση ΤΟΥ κλειδώματος

# Τεχνικές για την υλοποίηση των Ατομικών Λειτουργιών

- ⇒ Semaphores [Dijkstra, 1968]
- ⇒ Locks [Shaw, 1948]
- ⇒ Critical regions [Hansen, 1983]
- ⇒ Monitors [Hoare, 1974]

Αμοιβαίος αποκλεισμός των παράλληλων διεργασιών κατά την εκτέλεση των ατομικών λειτουργιών

# Κλειδώματα στη Multi-Pascal

- ⇒ Ενεργός αναμονή (Busy Waiting)
- ⇒ Τύπος δεδομένων Spinlock
- ⇒ Διαδικασία Lock()
- ⇒ Διαδικασία Unlock()
- ⇒ Τιμές μεταβλητής spinlock:
  - 0: κατάσταση ξεκλειδώματος
  - 1: κατάσταση κλειδώματος

# Διαδικασίες

## Κλειδώματος/ξεκλειδώματος

```
Unlock(L) : Write 0 into L;
```

```
Lock(L) :
```

```
Repeat
```

```
X:=TestandSet(L) ;
```

```
(*Διαβάζεται η παλιά κατάσταση και  
ενεργοποιείται το κλείδωμα*)
```

```
until X=0;
```

```
(*Σταματά η επανάληψη του βρόγχου αν το L είναι  
απενεργοποιημένο*)
```



# Σειριακό Πρόγραμμα Για Τη Δημιουργία Ιστογράμματος

```
PROGRAM Histogram;
CONST n=20;      (*Μέγεθος της εικόνας*)
      max=10;   (*Μέγιστη ένταση του εικονοστοιχείου*)

VAR Image: Array [1..n,1..n] of integer;
    hist: Array [0..max] of integer;
    i,j,intensity: integer;

BEGIN
  FOR i := 1 to n do begin(*Εισαγωγή του πίνακα της εικόνας*)
    For j := 1 to n do
      Read((image[i,j]));
    Readln;
  END;
  FOR i := 1 to max do (*Αρχικοποίηση του ιστογράμματος*)
    hist[i] := 0;
  FOR i := 1 to n do
    FOR j := 1 to n do BEGIN
      intensity := image[i,j];
      hist[intensity] := hist[intensity]+1;
    END;
  END.
END.
```

# Σενάριο για την τεχνική κλειδώματος

```
begin
```

```
    intensity := image[i,j];
```

```
    Lock(L);
```

```
    hist[intensity] := hist[intensity]+1;
```

```
    Unlock(L);
```

```
End;
```

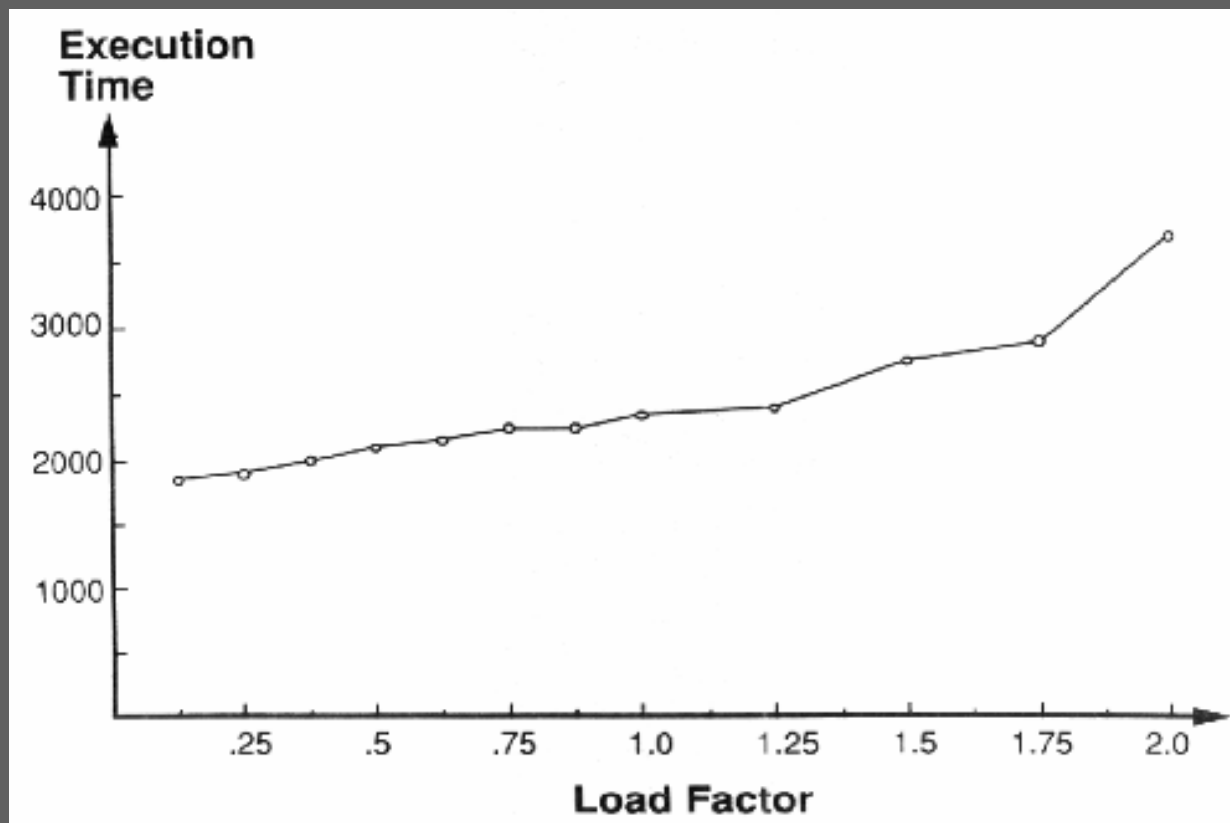
# Παράλληλο Πρόγραμμα Για Τη Δημιουργία Ιστογράμματος

```
PROGRAM ParallelHistogram;
CONST n=20; (*Μέγεθος της εικόνας*)
      max=10; (*Μέγιστη ένταση του εικονοστοιχείου*)

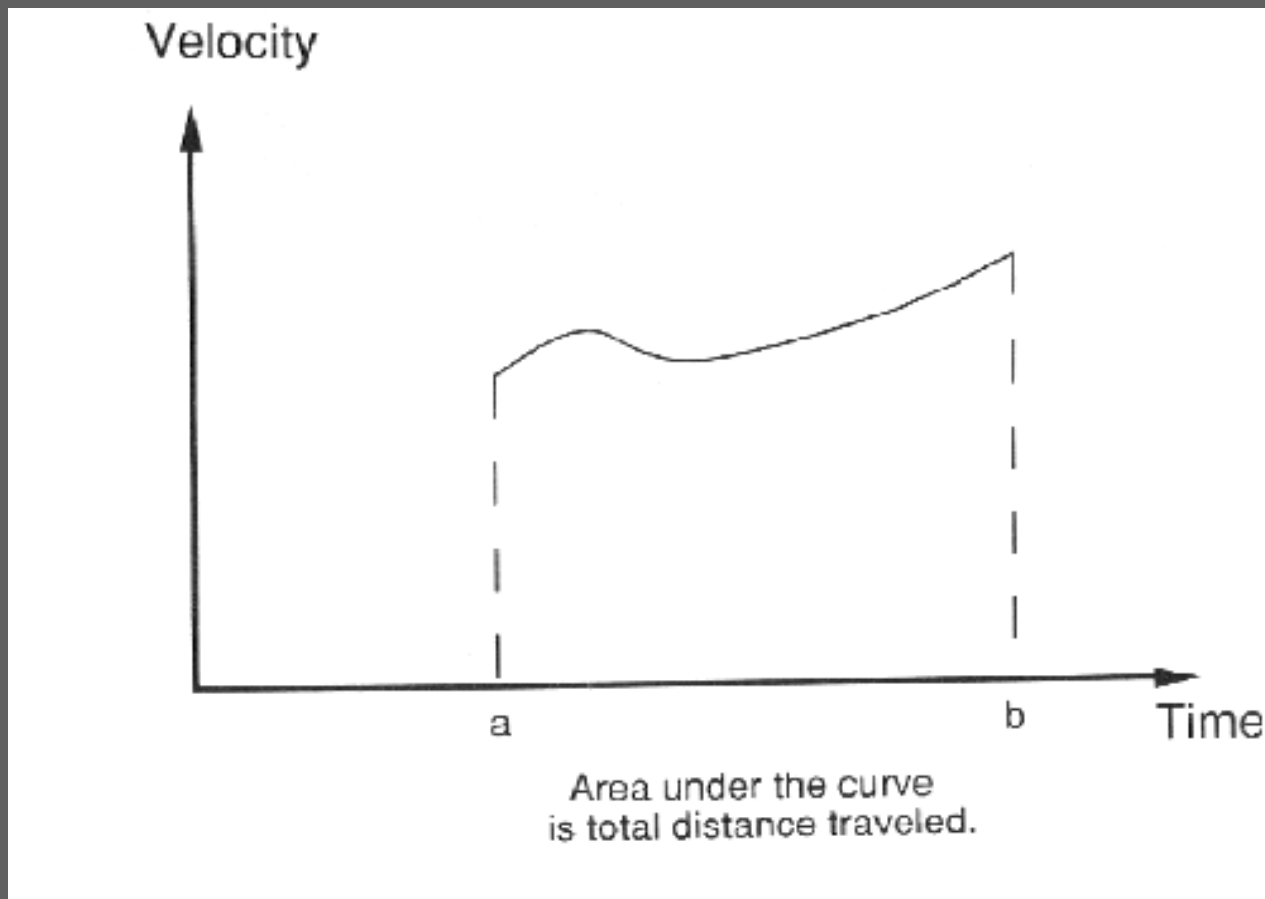
VAR Image: Array [1..n,1..n] of integer;
    hist: Array [0..max] of integer;
    L: Array [0..max] of spinlock;
    i: integer;

BEGIN
  ... (*Αρχικοποίηση του πίνακα εικόνας*)
  For I := 0 to max do hist[i] := 0;
  Forall i := 1 to n do
    VAR j, intensity : Integer;
    BEGIN
      For j := 1 to n do BEGIN
        intensity := image[i,j];
        Lock(L[intensity]);
        hist[intensity] := hist[intensity]+1;
        Unlock(L[intensity]);
      END;
    END;
  END;
END.
```

# Ανταγωνισμός στο πρόγραμμα Ιστογράμματος Εικόνας



# Εφαρμογή της Αριθμητικής Ολοκλήρωσης

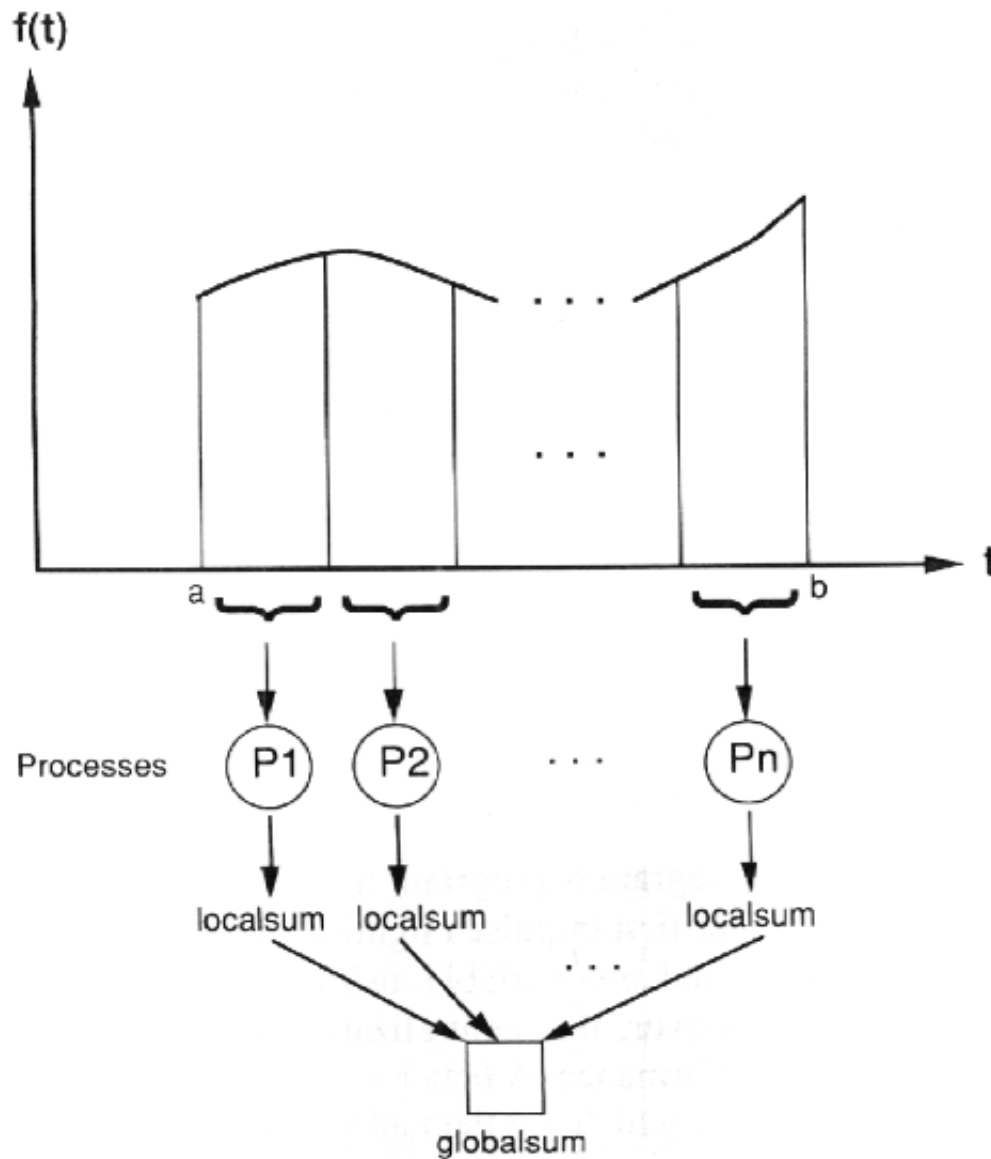


# Ακολουθιακό πρόγραμμα για την αριθμητική ολοκλήρωση

Sequential Numerical Integration:

```
sum := 0;
t := a;
For i := 1 to n do
  BEGIN
    t:=t+w;          (*Προχωρά στο επόμενο σημείο *)
    sum:=sum+f(t);  (*Προστίθεται το σημείο δείγματος στο άθροισμα*)
  END;
sum := sum + (f(a) + f(b)) / 2;
answer := w * sum;
```

# Παράλληλη μορφή αριθμητικής ολοκλήρωσης



```

PROGRAM Numerical Integration;
CONST numproc = 40;      (*πλήθος των διεργασιών*)
      numpoints = 30;   (*πλήθος των στοιχείων ανά διεργασία*)

VAR a, b, w, globalsum, answer: real;
    i: integer;
    L: spinlock;

Function f(t: real): Real; (*Η συνάρτηση που θα ολοκληρωθεί*)
BEGIN
..... (*Υπολογισμός της τιμής της f(t)*)

END;

Procedure Integrate(myindex: Integer);
VAR localsum, t: Real;
    j: Integer;

BEGIN
t:= a + myindex * (b-a)/numproc; (*Η αρχική θέση*)
For j:= 1 to numpoints do
    BEGIN
        localsum:= localsum + f(t); (*Πρόσθεση του επομένου στοιχείου
                                    της δειγματοληψίας*)
        t:= t + w;
    END;
localsum:= w*localsum;
Lock(L);
globalsum:= globalsum + localsum; (*Ατομική αλλαγή*)
Unlock(L);
END;

BEGIN
.... (*Αρχικοποίηση των τιμών των ακραίων σημείων a και b*)

w:= (b-a) / (numproc*numpoints); (*Η απόσταση μεταξύ των σημείων*)

Forall I:= 0 to numproc-1 do (*Δημιουργία διεργασιών*)
Integrate(i);
answer:=globalsum+w/2*(f(b)-f(a)); (*Πρόσθεση των ακραίων σημείων*)
END.

```

## Παράλληλο πρόγραμμα αριθμητικής ολοκλήρωσης



# Αποκέντρωση των Ατομικών Λειτουργιών – Ελαχιστοποίηση του Ανταγωνισμού

- ⇒ Πρόβλημα Ιστογράμματος:  
χρήση ξεχωριστών κλειδωμάτων
- ⇒ Πρόβλημα Αριθμητικής Ολοκλήρωσης:  
χρήση τοπικών μεταβλητών αθροίσματος

# Locks vs. Channels

- ⇒ Προσομοίωση κλειδώματος από κανάλι
- ⇒ Προσομοίωση καναλιού από κλείδωμα
- ⇒ Απαίτηση σε χρόνο
- ⇒ Κανάλια: ταυτόχρονα ανάγνωση κι εγγραφή
- ⇒ Spinning vs. Blocked
- ⇒ Επεξεργαστής: Busy Waiting vs. Blocked