



Παράλληλη Επεξεργασία
Κεφάλαιο 8^ο
Προγράμματα Περάσματος
Μηνυμάτων

Κωνσταντίνος Μαργαρίτης
Καθηγητής
Τμήμα Εφαρμοσμένης Πληροφορικής
Πανεπιστήμιο Μακεδονίας
kmarg@uom.gr
<http://eos.uom.gr/~kmarg>

Αρετή Καπτάν
Υποψήφια Διδάκτορας
Τμήμα Εφαρμοσμένης Πληροφορικής
Πανεπιστήμιο Μακεδονίας
areti@uom.gr
<http://eos.uom.gr/~areti>

Χαρακτηριστικά της Multi-Pascal για Συστήματα Κατανεμημένης Μνήμης (Μέρ.Ι)

- ⇒ Κάθε διεργασία έχει τοπικές μόνο μεταβλητές (τοπική μνήμη)
- ⇒ Πέρασμα μηνυμάτων μεταξύ διεργασιών μέσω των θυρών επικοινωνίας (μεταβλητές κανάλια)
- ⇒ Μόνο μια διεργασία μπορεί να λαμβάνει δεδομένα από μια θύρα επικοινωνίας

Χαρακτηριστικά της Multi-Pascal για Συστήματα Κατανεμημένης Μνήμης (Μέρ. II)

- ⇒ Οι μεταβλητές του κυρίως προγράμματος είναι τοπικές (δε διαμοιράζονται)
- ⇒ Μηνύματα που στέλνονται από μία αφετηρία προς το ίδιο προορισμό θα φτάσουν με την ίδια σειρά που στέλνονται (δεν ισχύει για διαφορετικές αφετηρίες)

Δυνατότητες της MPWin

- ⇒ Καθορισμός κατανεμημένης τοπολογίας
- ⇒ Ορισμός του πλήθους των επεξεργαστών
- ⇒ Παραμετροποίηση της βασικής καθυστέρησης επικοινωνίας (communication link delay)
- ⇒ Ενεργοποίηση παραμέτρου συμφόρησης του δικτύου επικοινωνίας (congestion)
- ⇒ Ανάθεση των διεργασιών σε επεξεργαστές (@)

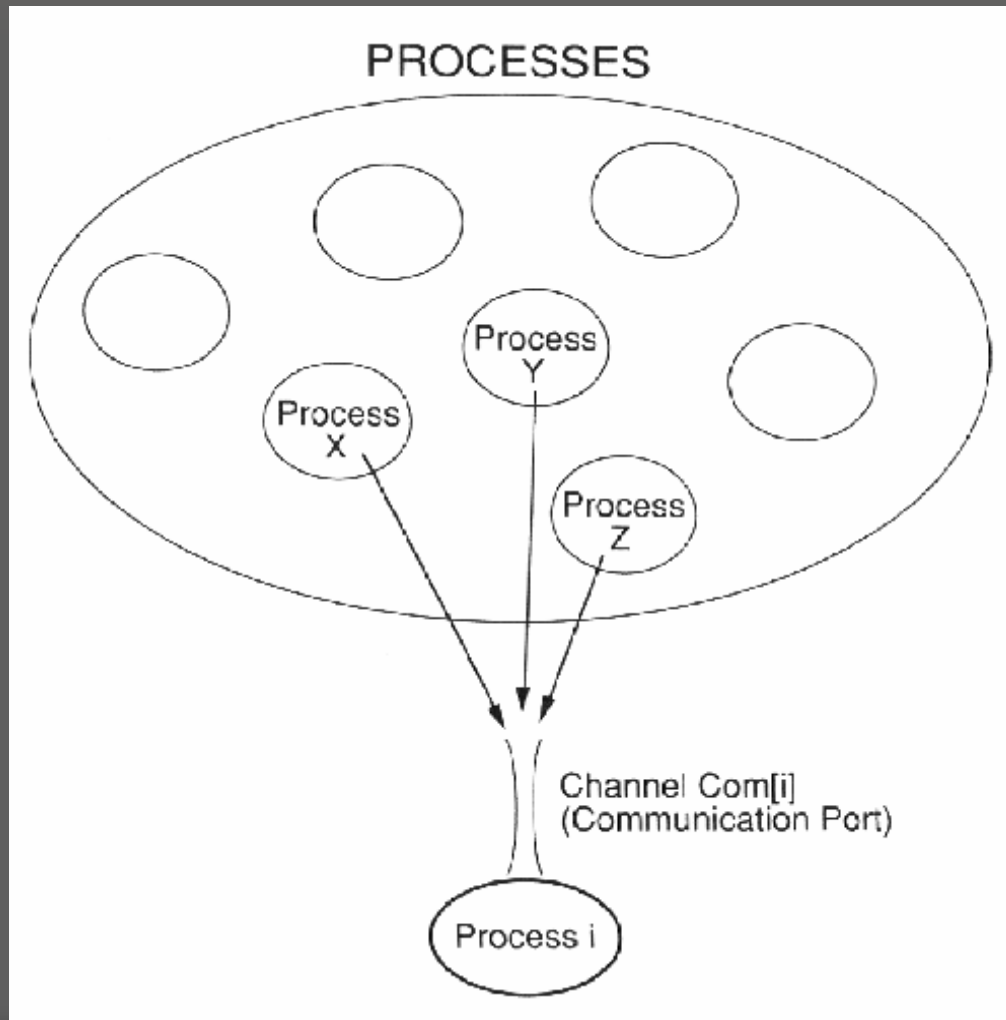
Θύρες Επικοινωνίας

ΥΛΙΚΟ	ΛΟΓΙΣΜΙΚΟ
Επεξεργαστής	Διεργασία
Φυσικός σύνδεσμος επικοινωνίας	Κανάλι

Κανάλια Επικοινωνίας

- ⇒ Το άκρο του καναλιού συνδέεται με μια συγκεκριμένη διεργασία
- ⇒ Η δρομολόγηση/προώθηση των μηνυμάτων κρύβονται από τον προγραμματιστή
- ⇒ Η καθυστέρηση επικοινωνίας εξαρτάται από:
 - την τοπολογία και
 - τα χαρακτηριστικά της απόδοσης του συστήματος
- ⇒ Τα μηνύματα που στέλνονται από την ίδια αφετηρία προς τον ίδιο προορισμό θα φτάσουν με την σειρά που στάλθηκαν. Δεν ισχύει για διαφορετικές αφετηρίες

Κανάλια επικοινωνίας στη Multi-Pascal



Δήλωση θύρας Επικοινωνίας Σε Πρόγραμμα

(*Κανάλια επικοινωνίας*)

VAR Com: ARRAY [1..30] OF CHANNEL OF INTEGER;

...

PROCEDURE Process(i: INTEGER);

VAR invalue, outvalue, k: INTEGER;

...

BEGIN

...

invalue:= Com[i]; (*Διαβάζει από τη δική μου θύρα επικοινωνίας*)

...

Com[k]:= outvalue; (*Στέλνει την "outvalue" στη διεργασία k*)

..

END;

BEGIN

...

FORALL i:= 1 to 30 DO

(PORT Com[i]) Process(i);

...

END.

Διαφορές Μεταξύ Καναλιού (ΠΣΔΜ) Και Θύρας Επικοινωνίας (ΠΣΚΜ)

Χαρακτηριστικά της θύρας επικοινωνίας:

- ⇒ μπορεί να έχει πολλούς συγγραφείς, αλλά μόνο ένα αναγνώστη
- ⇒ αν η διεργασία i εκτελείται στον επεξεργαστή i , τότε το κανάλι $Com[i]$ είναι αποθηκευμένο στην τοπική μνήμη του επεξεργαστή i
- ⇒ η λειτουργία εγγραφής δεν αναστέλλει τη διεργασία-συγγραφέα
- ⇒ η μετάδοση έχει πραγματική καθυστέρηση βάσει της τοπολογίας και της σχετικής θέσης του καναλιού προορισμού

Τεχνικές Επικοινωνίας ΠΣΚΜ

⇒ Σύγχρονη επικοινωνία:

- ανασταλτική για τη διεργασία συγγραφείας
- μη απομονωμένη (non-buffered)
- ανασταλτική για τη διεργασία αναγνώστης

⇒ Ασύγχρονη επικοινωνία:

- μη ανασταλτική για τη διεργασία συγγραφείας
- απομονωμένη (buffered)
- ανασταλτική για τη διεργασία αναγνώστης

Η Multi-Pascal υποστηρίζει ασύγχρονη επικοινωνία διεργασιών

Ασύγχρονη Επικοινωνία Σε Παράλληλες Γλώσσες Και Σε Multi- Pascal

Writer Process executes:

```
Send(processor_number, process_number, message_pointer);
```

Reader Process executes:

```
Receive(message_pointer);
```

Multi Pascal

Writer Process:

```
Com[i] := message;          (*Αποστολή στη διεργασία i*)
```

Reader Process:

```
message := Com[i];          (*Λήψη μηνύματος*)
```

Σύγχρονη Επικοινωνία Σε Παράλληλες Γλώσσες

Writer Process executes:

```
Send(link_number, message_pointer);
```

Reader Process executes:

```
Receive(link_number, message_pointer);
```

Γλωσσική Υποστήριξη Προγραμματισμού Περάσματος Μηνυμάτων

- ⇒ Η δήλωση Port
- ⇒ Παράμετροι Διαδικασίας (παράμετρος τιμής/διεύθυνσης)
- ⇒ Ο τελεστής @ για την ανάθεση διεργασίας σε επεξεργαστή

Η δήλωση Port

⇒ Σύνταξη της εντολής Port:

- (PORT <channel-list>)<statement>;

⇒ Παραδείγματα:

```
VAR C: CHANNEL OF CHAR;
```

```
archan: ARRAY [1..10] OF CHANNEL OF INTEGER;
```

- (PORT C)
- (PORT archan[2])
- (PORT archan)

```
VAR A: ARRAY [1..10, 1..20] OF CHANNEL OF REAL;
```

- (PORT A[3,4])
- (PORT A[1,1]; A[2,3])
- (PORT A)
- (PORT A[2])

Επικοινωνία Με Το Κυρίως Πρόγραμμα

- ⇒ Κατανομή αρχικών δεδομένων στις διεργασίες-παιδιά ⇒ παράμετροι τιμής σε διαδικασία (Value Parameters)
- ⇒ Συλλογή τελικών αποτελεσμάτων από τις διεργασίες-παιδιά ⇒ παράμετροι απομακρυσμένης διεύθυνσης σε διαδικασία (Remote Var Parameters)

Κατανομή Αρχικών Δεδομένων στις Διεργασίες

```
PROGRAM Message-Passing;
CONST n= ...; (*Αριθμός των διεργασιών*)
TYPE   datatype= ARRAY [1..m] OF REAL;
VAR    inchan: ARRAY [1..n] OF CHANNEL OF INTEGER;
        i: INTEGER;
        inputdata: ARRAY [1..n] OF datatype; (*Πρωταρχικός πίνακας δεδομένων*)
        ....

PROCEDURE Process(i: INTEGER; mydata: datatype);
VAR x: REAL;
... (*Άλλες τοπικές μεταβλητές για τη διεργασία*)
BEGIN
    ...
END;

BEGIN (*Κυρίως πρόγραμμα*)
    ... (*Ανάγνωση των αρχικών τιμών για τον πίνακα inputdata*)
    FORALL i:= 1 TO n DO (*Δημιουργία των διεργασιών*)
        (PORT inchan[i] Process(i, inputdata[i]));
    ...
END.
```


Επικοινωνία Διεργασιών Με Το Κυρίως Πρόγραμμα

```
PROGRAM Message-Passing;
CONST n= ...; (*Αριθμός των διεργασιών*)
TYPE   datatype= ARRAY [1..m] OF REAL;
VAR    inchan: ARRAY [1..n] OF CHANNEL OF INTEGER;
        i: INTEGER;
        inputdata: ARRAY [1..n] OF datatype; (*Πρωταρχικός πίνακας δεδομένων*)
        outputdata: ARRAY [1..n] OF REAL; (*Τελικά υπολογισμένα αποτελέσματα*)
        ....
PROCEDURE Process(i: INTEGER; in: datatype; VAR out: REAL);
VAR x: REAL;
... (*Άλλες τοπικές μεταβλητές για τη διεργασία*)
BEGIN
    ... (*Υπολογισμός του τελευταίου αποτελέσματος "x"*)
    out:= x; (*Αντιγραφή των τελικών αποτελεσμάτων πίσω στο κυρίως πρόγραμμα*)
END;

BEGIN (*Κυρίως πρόγραμμα*)
    ... (*Ανάγνωση των αρχικών τιμών για τον πίνακα inputdata*)
    FORALL i:= 1 TO n DO (*Δημιουργία των διεργασιών*)
        (PORT inchan[i]) Process(i, inputdata[i], outputdata[i]);
    ...
END.
```

Ο τελεστής @ για την ανάθεση διεργασίας σε επεξεργαστή

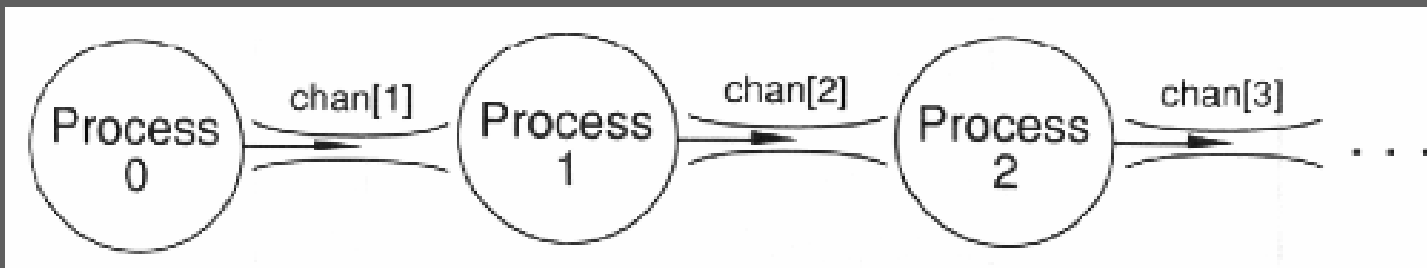
⇒ Σύνταξη του τελεστή @:

- (@ <expression>)<statement>;
- (@ <expression> PORT <channel_list>) <statement>;

⇒ Παραδείγματα:

- FORK (@3) Expand;
- FORK (@i) Filter(outchan);
- FORK (@i*10+j) Compute(i);
- FORALL i:= 1 TO 50 DO (@ i-1) Compute(i);
- FORALL j:= 1 TO 10 DO (@ j) Process(j, A[j]);
- FORK (@ %Self+1) Filter(outchan);
- FORK (@ %Self+10) Point(i, j);

Διασώληνωση Διεργασιών



Παράλληλο Πρόγραμμα Ταξινόμησης με Εισαγωγή (I)

```
PROGRAM InsertionSort;
ARCHITECTURE FULLCONNECT(101); (*Πλήρως Συνδεδεμένη τοπολογία*)
CONST n=100;
VAR list, sorted: ARRAY [1..n] OF INTEGER;
    pipechan: ARRAY [1..n] OF CHANNEL OF INTEGER;
    j, k: INTEGER;

PROCEDURE Pipeprocess(me: INTEGER; VAR sorteditem: INTEGER);
VAR internal, newitem, l: INTEGER;
BEGIN
    internal := pipechan[me];
    FOR i:= 1 to n-me DO
        BEGIN
            newitem:= pipechan[me];
            IF newitem < internal THEN
                BEGIN
                    pipechan[me+1]:= internal;
                    internal:= newitem;
                END
            ELSE pipechan[me+1]:= newitem;
        END;
        sorteditem:= internal;
    END;

BEGIN
...
(* initialization of List *)
FOR j:= 1 TO n DO
    FORK ( PORT pipechan[j] ) Pipeprocess(j, sorted[j]);
FOR k:= 1 TO n DO
    pipechan[1]:= list[k];
...
END.
```

Παράλληλο Πρόγραμμα Ταξινόμησης με Εισαγωγή (II)

```
PROGRAM InsertionSort;
ARCHITECTURE LINE(101); (*Τοπολογία Γραμμής*)
CONST n=100;
VAR list, sorted: ARRAY [1..n] OF INTEGER;
    pipechan: ARRAY [1..n] OF CHANNEL OF INTEGER;
    j, k: INTEGER;

PROCEDURE Pipeprocess(me: INTEGER; VAR sorteditem: INTEGER);
VAR internal, newitem, l: INTEGER;
BEGIN
    internal := pipechan[me];
    FOR i:= 1 to n-me DO
        BEGIN
            newitem:= pipechan[me];
            IF newitem < internal THEN
                BEGIN
                    pipechan[me+1]:= internal;
                    internal:= newitem;
                END
            ELSE pipechan[me+1]:= newitem;
        END;
        sorteditem:= internal;
    END;

BEGIN
...
(* initialization of List *)
FOR j:= 1 TO n DO
    FORK ( @j PORT pipechan[j] ) Pipeprocess(j, sorted[j]);
FOR k:= 1 TO n DO
    pipechan[1]:= list[k];
...
END.
```

Συμβάντα που προκαλούν την Αποστολή Μηνύματος

- ⇒ Εγγραφή σε ένα κανάλι επικοινωνίας
- ⇒ Δημιουργία μίας νέας διεργασίας
- ⇒ Εγγραφή σε μια παράμετρο απομακρυσμένης διεύθυνσης
- ⇒ Τερματισμός διεργασίας

Παράγοντες δημιουργίας της Καθυστέρησης Επικοινωνίας

- ⇒ Τοπολογία συστήματος κατανεμημένης μνήμης
- ⇒ φυσικά χαρακτηριστικά δικτύου επικοινωνίας
- ⇒ τρέχουσα κυκλοφορία μηνυμάτων στο δίκτυο

Υπολογισμός της Καθυστέρησης Επικοινωνίας

⇒ P αμελητέο:

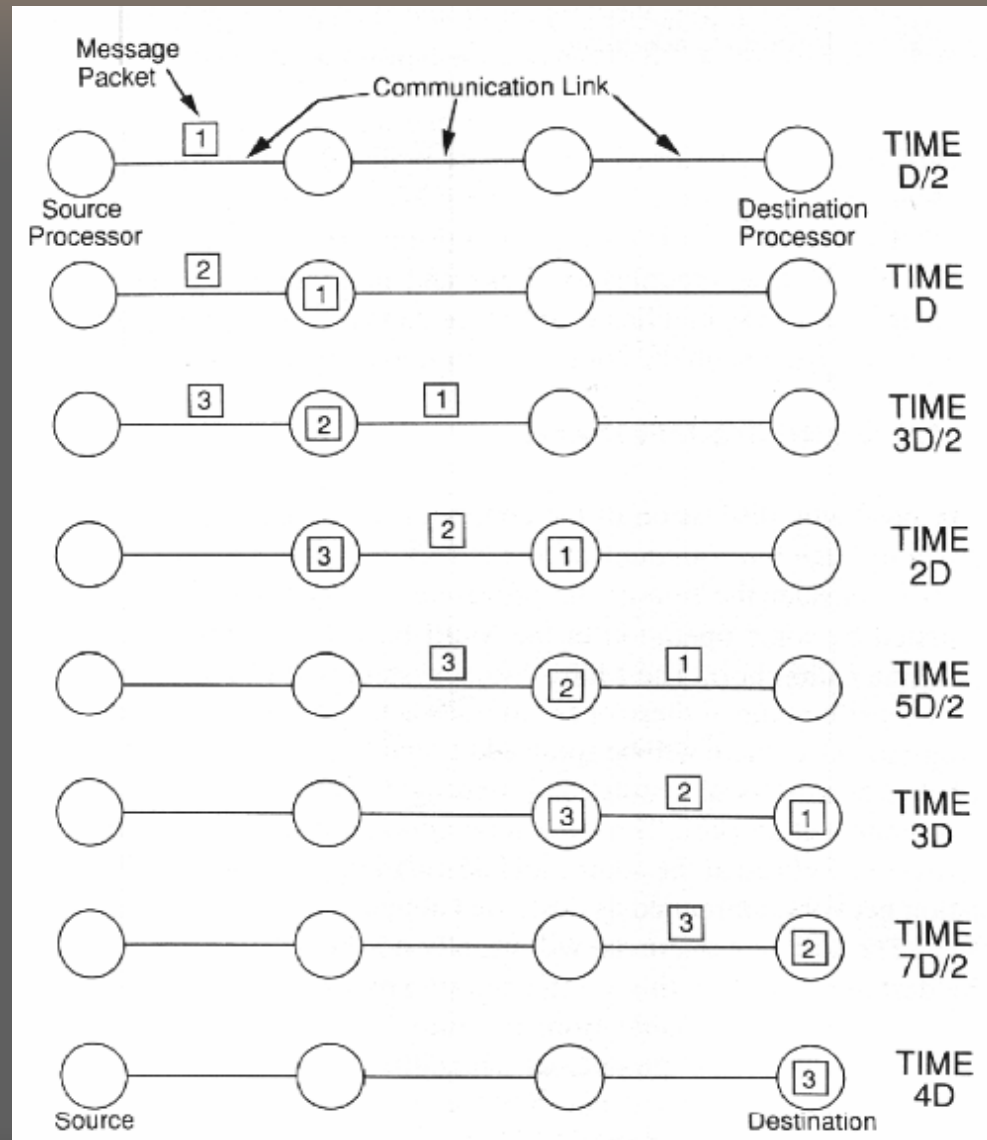
$$(m + k - 1) T$$

⇒ P υπολογίσιμο:

$$m (T + P) + (k - 1) \max(T, P)$$

$$\left(m + \frac{k - 1}{2} \right) D$$

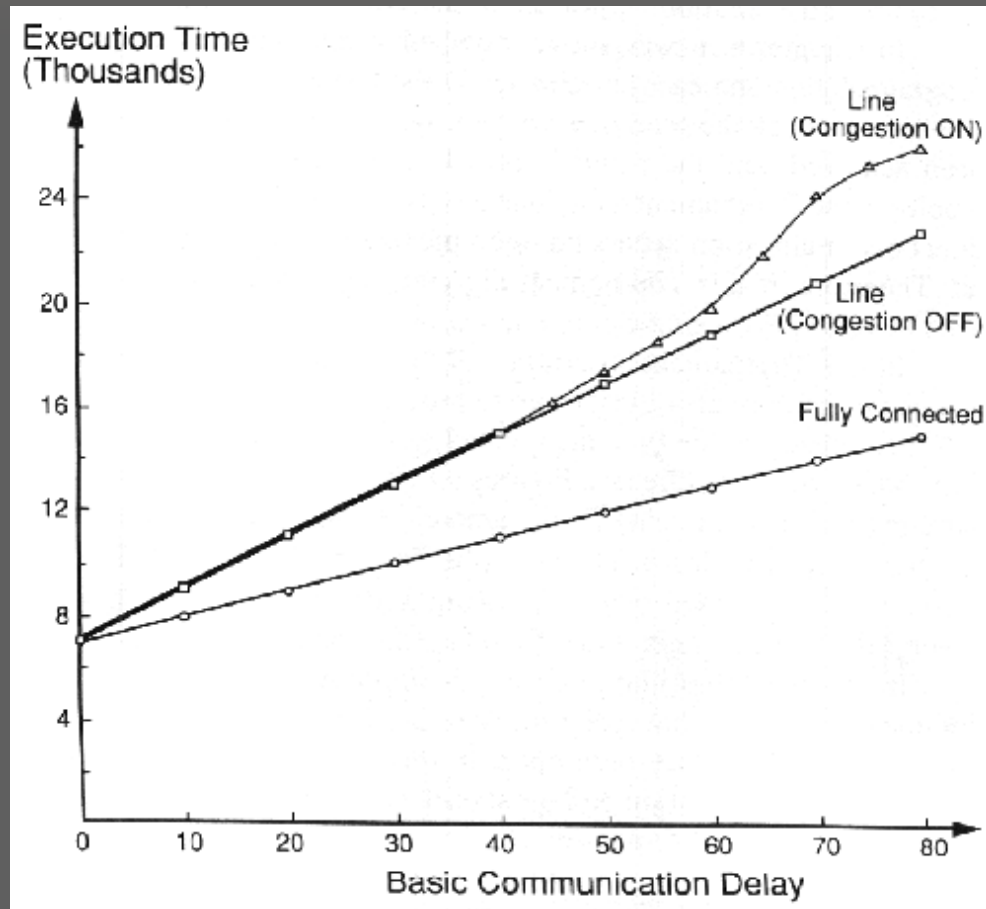
Αποστολή μηνύματος με πολλά πακέτα

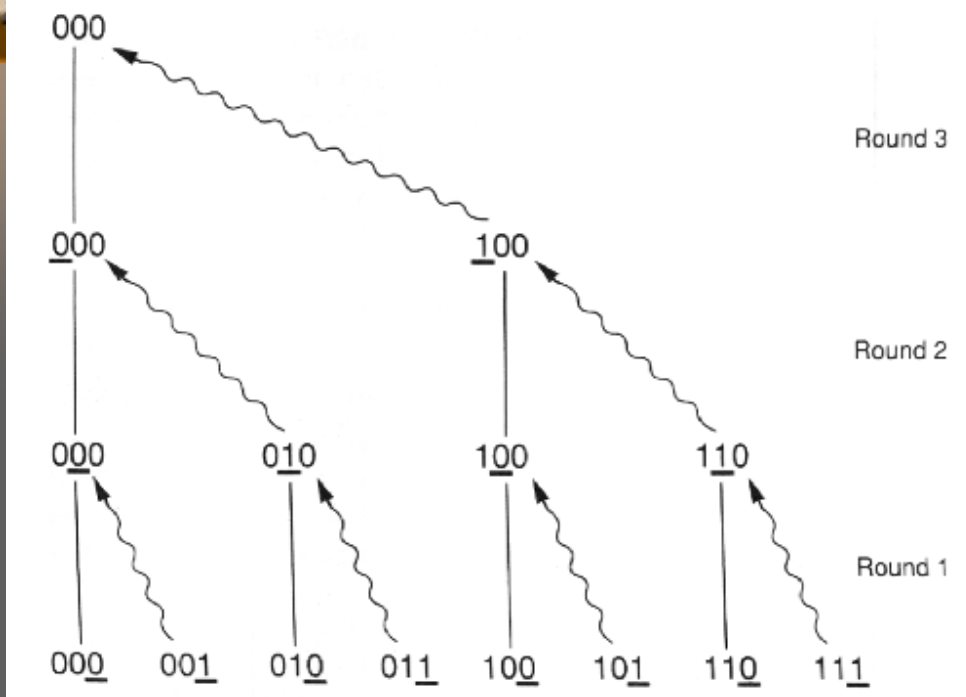


Επιβάρυνση Λογισμικού

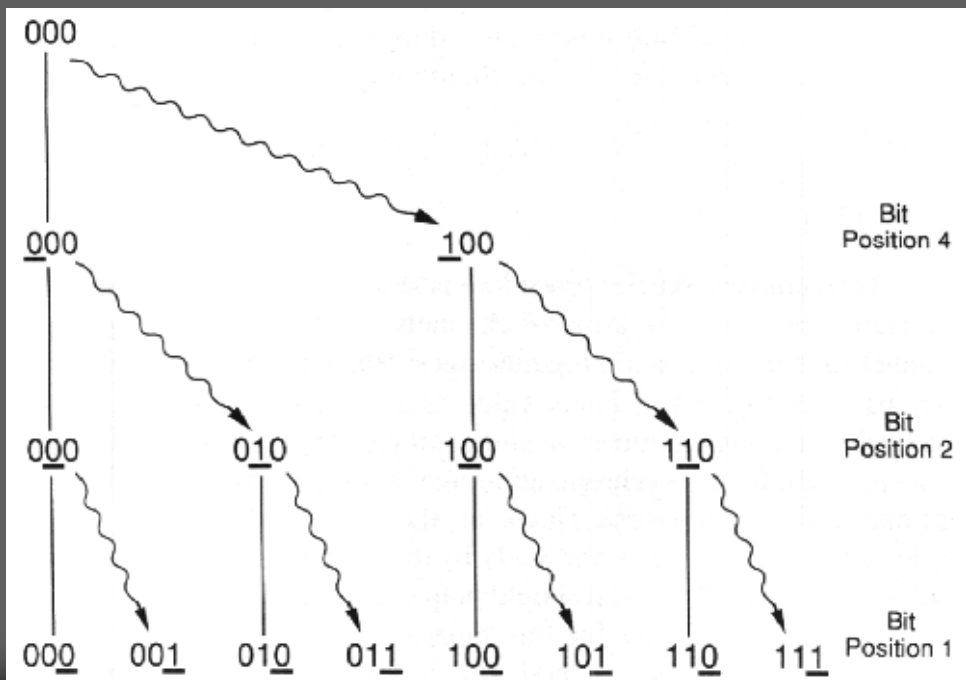
- ⇒ στον επεξεργαστή αφετηρίας
- ⇒ στον επεξεργαστή προορισμού
- ⇒ στο δίκτυο επικοινωνίας

Απόδοση της Ταξινόμησης με Εισαγωγή





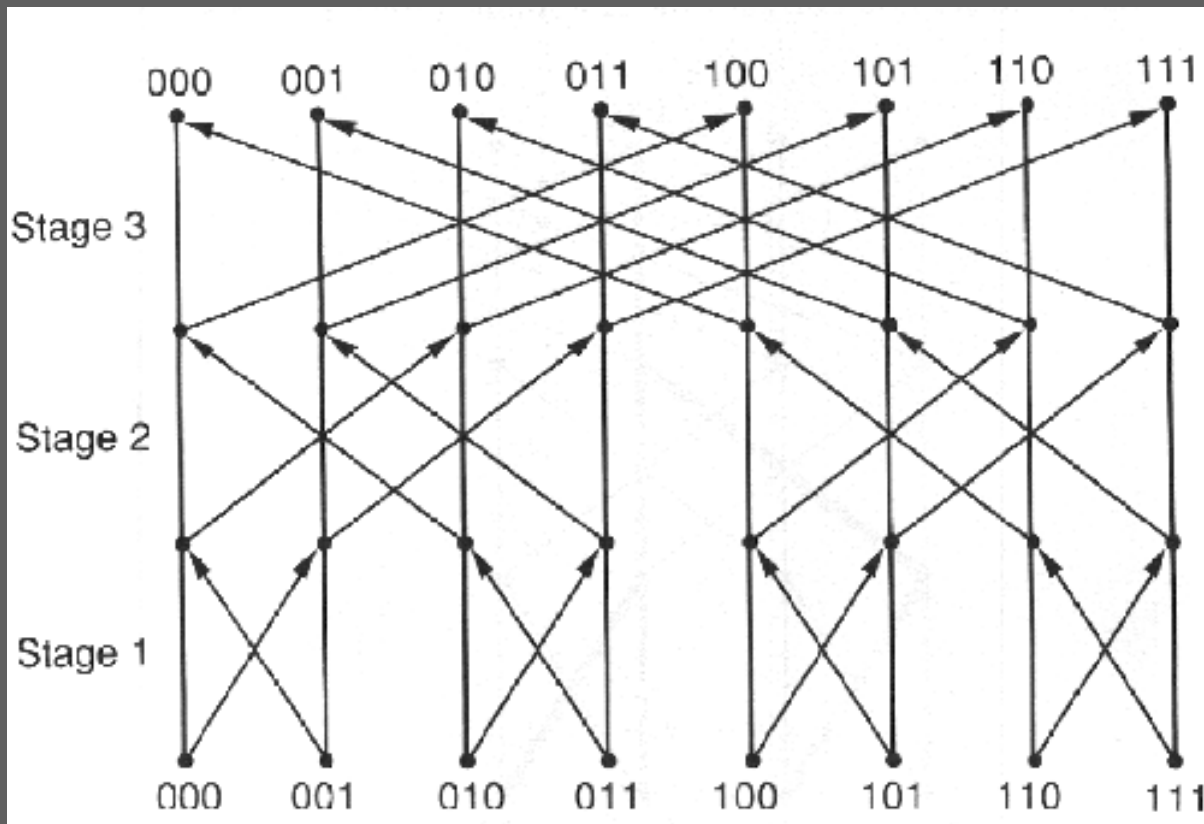
Τεχνική του Τουρνουά



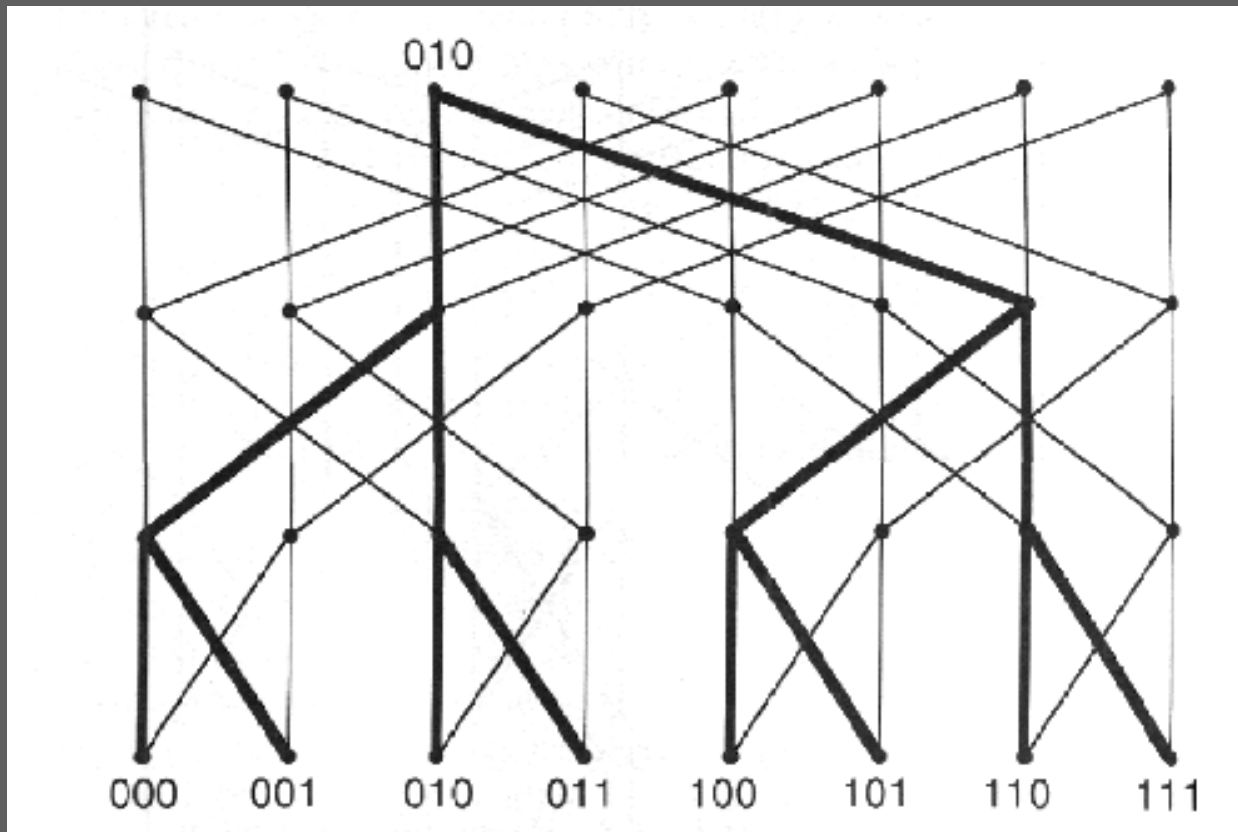
Αλγόριθμος Πολλαπλής Συλλογής

```
FOR i:=1 to d DO
  BEGIN
    Compute “partner” by reversing ith bit of my number;
    Send “myboolean” to partner;
    Receive “hisboolean” from partner;
    myboolean:= myboolean AND hisboolean;
  END;
final result is in “myboolean”;
```

Πολλαπλή Συλλογή σε Υπερκύβο



Διαδικό Δέντρο σε Πρότυπο Πεταλούδας



Πρόγραμμα Πολλαπλής Συλλογής

PROGRAM MultipleAggregate;

ARCHITECTURE HYPERCUBE(6); (*Τοπολογία Υπερκύβου διάστασης 6*)

CONST d=6; (*Διάσταση του Υπερκύβου*)

n=63;

VAR inchan: ARRAY [0..n, 1..d] OF CHANNEL OF BOOLEAN; (*Θύρες επικοινωνίας*)

i: INTEGER;

FUNCTION Aggregate(mydone: BOOLEAN): BOOLEAN;

VAR mynum, partner, bitvalue, stage: INTEGER;

hisdone: BOOLEAN;

BEGIN

mynum:=%SELF; (*Λήψη του αριθμού επεξεργαστή*)

bitvalue:= 1;

FOR stage:= 1 TO d DOBEGIN

IF mynum DIV bitvalue MOD 2 = 0 (*Υπολογισμός του ζεύγους*) THEN partner:= mynum+bitvalue

ELSE partner:= mynum-bitvalue;

inchan[partner, stage]:= mydone; (*Αποστολή του mydone στο ταίρι*)

hisdone:= inchan[mynum, stage]; (*Λήψη του hisdone από το ταίρι*)

mydone:= mydone AND hisdone;

bitvalue:= 2*bitvalue (*Μετατόπιση προς την επόμενη θέση bit*)

END;

Aggregate:= mydone;

END;

PROCEDURE Process(i: INTEGER);

VAR mydone, done: BOOLEAN;

BEGIN

REPEAT

.... (*Πραγματοποίηση επανάληψης και υπολογισμός της mydone*)

done:= Aggregate(mydone);

UNTIL done;

END;

BEGIN (*Κυρίως πρόγραμμα*)

....

FORALL i:= 0 TO n DO

(@i PORT inchan[i]) Process(i);

(*Δημιουργία των διεργασιών*)

END.

Πρόγραμμα Πολλαπλής Διάδοσης

```
PROGRAM MultipleBroadcast;
ARCHITECTURE HYPERCUBE(6); (*Τοπολογία Υπερκύβου με διάσταση 6*)
CONST d= 6;
      n= 63;
TYPE listtype= ARRAY [0..n] OF INTEGER;
VAR      inchan: ARRAY [0..n, 1..d] OF CHANNEL OF INTEGER; (*Θύρες επικοινωνίας*)
        i: INTEGER;

PROCEDURE Broadcast(myvalue: INTEGER; VAR mylist: listtype);
VAR mynum, bitvalue, stage, j: INTEGER;
    hisdone: BOOLEAN;

BEGIN
  mynum:= %SELF;
  mylist[0]:= myvalue;
  bitvalue:= 1;
  FOR stage:= 1 TO d DO BEGIN
    IF mynum DIV bitvalue MOD 2 = 0 (*Υπολογισμός του ταιριού*) THEN partner:= mynum+bitvalue
    ELSE partner:= mynum-bitvalue;
    FOR j:= 0 TO bitvalue-1 DO (*Αποστολή του mylist στο ταίρι*)
      inchan[partner, stage]:= mylist[j];
    FOR j:= bitvalue TO 2*bitvalue-1 DO
      mylist[j]:=inchan[mynum,stage](*Λήψη της λίστας από το ταίρι*)
    bitvalue:= 2*bitvalue; (*Μετατόπιση στην επόμενη θέση bit*)
  END;
END;
```

```
PROCEDURE Process(i: INTEGER);
VAR values: listtype;
    myvalue: INTEGER;
BEGIN
  ...
  Broadcast(myvalue, values);
  (*Αποστολή της myvalue σε όλες τις
  διεργασίες*)
  (*Λήψη στον πίνακα values*)
END;

BEGIN (*Κυρίως πρόγραμμα*)
  ...

FORALL i:= 0 TO n DO
  (@i PORT inchan[i]) Process(i);
  (*Δημιουργία των διεργασιών*)
END.
```

Άμεση μέθοδος Πολλαπλής Διάδοσης

```
PROGRAM DirectBroadcast;
ARCHITECTURE HYPERCUBE(6);
CONST d= 6;
        n= 63;
TYPE listtype= ARRAY [0..n] OF INTEGER;
VAR      inchan: ARRAY [0..n, 1..d] OF CHANNEL OF INTEGER;
        ...

PROCEDURE Broadcast(myvalue: INTEGER; VAR values:listtype);
VAR      mynum, bitvalue, stage, i: INTEGER;
        hisdone: BOOLEAN;

BEGIN
        mynum:= %SELF;
        FOR i:= 0 TO n DO (*Αποστολή ενός αντιγραφου της myvalue σε όλες τις διεργασίες*)
                inchan[i]:= myvalue;
        FOR i:= 0 TO n DO (*Λήψη των τιμών που έχουν στείλει οι άλλες διεργασίες*)
                values[i]:= inchan[mynum];
END;

        ...
```

Σύγκριση Μεθόδων Πολλαπλών Μεταδόσεων

