Συστήματα Παράλληλης & Κατανεμημένης Επεξεργασίας

Ενότητα 3: MPI_Get_count,
non blocking send/recv, εμφάνιση και αποφυγή αδιεξόδων

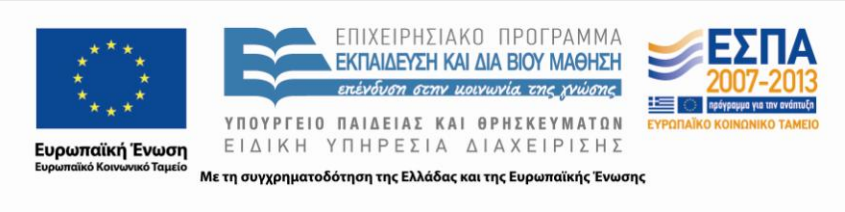
Δρ. Μηνάς Δασυγένης
mdasyg@ieee.org
Εργαστήριο Ψηφιακών Συστημάτων και Αρχιτεκτονικής Υπολογιστών
http://arch.icte.uowm.gr/mdasyg
Τμήμα Μηχανικών Πληροφορικής και Τηλεπικοινωνιών

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.



Σκοπός της Ενότητας

- Η χρήση των μη παρεμποδιστικών συναρτήσεων για την αποφυγή αδιεξόδων.

Πρότυπα συναρτήσεων MPI_Get_count

- Το πρωτότυπό έχει την μορφή:
- **Int MPI_Get_count (MPI_Status * status, MPI_Datatype * datatype, int * count);**
  - status: το τελευταίο όρισμα της MPI_Recv .
  - datatype: τύπος δεδομένων που έχουν παραληφθεί από την MPI_Recv.
  - count: πλήθος στοιχείων που έχουν παραληφθεί.
- Χρησιμοποιείται σε συνδυασμό με την MPI_Recv για να ανακτήσουμε το μέγεθος του μηνύματος που έχει παραληφθεί.
- Παράδειγμα:
  - **MPI_Get_count (&status, MPI_INT, &count);**
- Απαιτείται δήλωση της μορφής:
  - **int count;**

Οι συναρτήσεις Send και Receive

- Οι **MPI_Send** και **MPI_Recv** είναι blocking συναρτήσεις:
  - Η εκτέλεση μετά από μία MPI_Send εντολή μπορεί να προχωρήσει μόνο εφόσον έχουν σταλεί τα δεδομένα και ο χώρος τους στη μνήμη είναι προσβάσιμος.
  - Η εκτέλεση μετά από μία MPI_Recv εντολή μπορεί να προχωρήσει μόνο εφόσον έχουν ληφθεί τα δεδομένα και ο χώρος τους στη μνήμη είναι προσβάσιμος.
- Υπάρχουν διαθέσιμες οι αντίστοιχες non-blocking ρουτίνες με την ίδια σύνταξη:
  - **MPI_Isend** (buffer , count, data_type, destination, tag, communicator , request).
  - **MPI_Irecv** (buffer , count, data_type, destination, tag, communicator , request).

Αποφυγή αδιεξόδων (deadlock) στο MPI

- Αδιέξοδο (deadlock): Είναι η κατάσταση στην οποία 2 ή περισσότερες διεργασίες εμπλέκονται σε μια κατάσταση αμοιβαίας αναμονής ο τερματισμός της οποίας απαιτεί την χρονική εξέλιξη των υπολοίπων διεργασιών.
- Εμφανίζεται σε περιπτώσεις κακής χρήσης των παρεμποδιστικών συναρτήσεων αποστολής (MPI_Send) και λήψης δεδομένων (MPI_Recv).

Προσοχή μη δημιουργηθεί deadlock

- Ο παρακάτω κώδικας δημιουργεί deadlock:

```
If( rank == 0 ) Then
     Call MPI_send( buffer1, 1, MPI_integer, 1, 10, &
                    MPI_comm_world, error )
     Call MPI_recv( buffer2, 1, MPI_integer, 1, 20, &
                    MPI_comm_world, status, error )
  Else If( rank == 1 ) Then
     Call MPI_send( buffer2, 1, MPI_integer, 0, 20, &
                    MPI_comm_world, error )
     Call MPI_recv( buffer1, 1, MPI_integer, 0, 10, &
                    MPI_comm_world, status, error )
  End If
```

Προσοχή μη δημιουργηθεί deadlock: Λύση Α

```
If( rank == 0 ) Then
     Call MPI_Isend( buffer1, 1, MPI_integer, 1, 10, &
                     MPI_comm_world, REQUEST, error )
     Call MPI_recv( buffer2, 1, MPI_integer, 1, 20, &
                     MPI_comm_world, status, error )
  Else If( rank == 1 ) Then
     Call MPI_Isend( buffer2, 1, MPI_integer, 0, 20, &
                     MPI_comm_world, REQUEST, error )
     Call MPI_recv( buffer1, 1, MPI_integer, 0, 10, &
                     MPI_comm_world, status, error )
  End If
  Call MPI_wait( REQUEST, status ) ! Wait until send is complete
```

Προσοχή μη δημιουργηθεί deadlock: Λύση Β

```
If( rank == 0 ) Then
     Call MPI_send( buffer1, 1, MPI_integer, 1, 10, &
                    MPI_comm_world, error )
     Call MPI_recv( buffer2, 1, MPI_integer, 1, 20, &
                    MPI_comm_world, status, error )
  Else If( rank == 1 ) Then
     Call MPI_recv( buffer1, 1, MPI_integer, 0, 10, &
                    MPI_comm_world, status, error )
     Call MPI_send( buffer2, 1, MPI_integer, 0, 20, &
                    MPI_comm_world, error )
  End If
```

Τέλος Ενότητας

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης