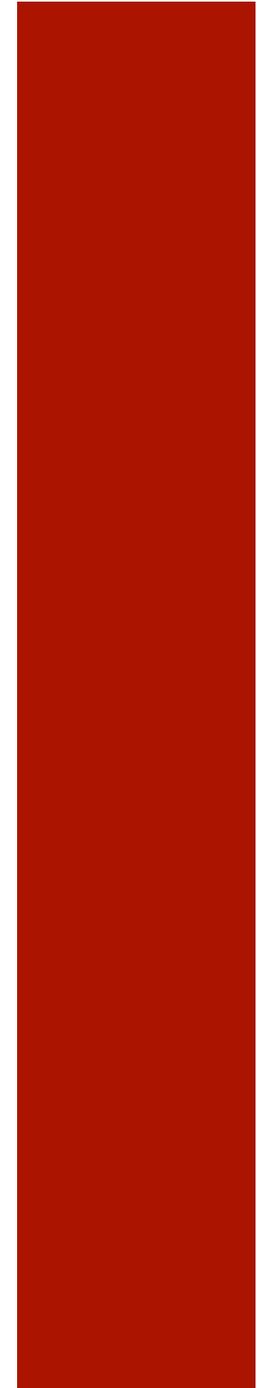


Programowanie współbieżne

Laboratorium-8

MPI



MPI. Zadania

- Uruchom program „Hello.f90”
 - na jednym procerze
 - Na wielu procesorach
- Napisz program z zastosowaniem procedur `send()` i `recv()` do przesłania danej jednego typu. Wydrukuj odebrane dane we wszystkich procesach.
- Sprawdź sytuacje impasu (deadlock) dyskutowane na wykładzie.
- Prześlij pojedynczym poleceniem `MPI_send` dane różnych typów (`integer::a(3)`, `real::b(4)`, `Real*8::c(4)`) między procesem głównym i resztą. Wydrukuj odebrane dane odpowiednio je formatując
- Opisz dokładnie procedury `MPI_scatter` i `MPI_gather`. Zastosuj je w przykładowym programie.



Zadanie

- Załóżmy, że każdy z pracujących w danym obszarze, czterech wątków, może modyfikować tablicę $A(1:100, 1:100)$. Porządek nie gra roli. Aby uniknąć niekorzystnej sytuacji wyścigu do danych można zastosować następujące rozwiązanie

```
!OMP PARALLEL SHARED  
  
!OMP CRITICAL  
  
...           ! Praca z A  
  
!OMP END CRITICAL  
  
!OMP END PARALLEL
```

Cała tablica A należy do jednego wątku. Lepsze rozwiązanie polega na podziale tablicy na cztery części $A1=A(1:50, 1:50)$, $A2=A(51:100, 1:50)$, ... i na wykorzystaniu pozostałych wątków. Każdy wątek może zmieniać wszystkie cztery bloki $A1, \dots, A4$ i najlepszą będzie sytuacja gdy będą to robić razem, jeden wątek – jeden blok. W ten sposób eliminujemy wyścig i zyskujemy na czasie. Taką synchronizację można otrzymać stosując zamki. Napisz fragment programu, który zrealizuje postawiony problem

