PODZIAŁ NA GRUPY

KOMUNIKATORY

To o czym tu bedzie mowa, jest małym wycinkiem zagadnienia podziału klastra. Podział który tu jest opisany odpowiada problemom, którymi zajmujemy się na zajęciach **WPR**.

Zajmowaliśmy się problemami z 'otoczenia' równania transportu, stosując aproksymację różnicową na siatkach przestrzenno-czasowych. W tym kontekście metoda **Time Splitting** pozwala na dość proste rozwiązania związane z podziałem na grupy, gdyż ważna część komunikacji między poszczególnymi partiami algorytmu odbywa się poprzez 'ślady' pozostawione na siatce.

Dla problemów tego typu w przypadku przestrzeni dwuwymiarowej, gdy mamy do czynienia z ruchem o składowych w dwóch kierunkach do siebie prostopadłych wygodnie będzie odwołać się do narzędzia istniejącego w ramach **MPI**, do tak zwanej dwuwymiarowej **Wirtualnej Topologii Kartezjańskiej**.

Poruszanie się w ramach utworzonych grup umożliwiaja **komunikatory**. Dotychczas mielismy do czynienia tylko z jednym **Komunikatorem:**

MPI_Comm_World,

który zapewnia komunikację w ramach grupy **wszystkich** wybranych przez nas procesorów.

Dla uproszczenia, przypuśćmy, że rozwiązujemy zwykłe równanie transportu na kwadracie i że wybraliśmy **16** procesorów.

0	**1**	**2**	**3**
4	**5**	**6**	**7**
8	**9**	**10**	**11**
12	**13**	**14**	**15**

Niech boki powyższego prostokąta będą równoległe do linii siatki, na której aproksymujemy równanie: poziomo oś $\boldsymbol{x}$, pionowo oś $\boldsymbol{y}$. Przypuśćmy, że przez każdy z małych prostokątów przechodzi jednakowa liczba linii poziomych i pionowych naszej siatki. W ramach tej **MPI Wirtualnej Topologii Kartezjańskiej** podzielmy nasze 16 procesorów na **4** grupy wierszowe i **4** grupy kolumnowe. Będziemy rozwiązywać (**Time-Splitting!**) równanie transportu metodą z macierzą Schura w kierunku

LEWO==>PRAWO i PRAWO==>LEWO

rownolegle w **4** grupach wierszowych, zaś w kierunku

GÓRA==> DÓŁ i DÓŁ==>GÓRA

równolegle w **4** grupach kolumnowych.

Aby to zrealizować trzeba utworzyć **nowe komunikatory**. Służy do tego poniżej podany **fragment programu CART uruchamiający cały system MPI i wszystkie potrzebne komunikatory.** Ten fragment należy włączyć do przygotowywanego programu, pamiętając o nowych deklaracjach wniesionych przez CART. Ponieważ wymiary podawane są tam przy pomocy zmiennych, należy we właściwy sposób wybrać miejsce włączenia tego fragnentu programu.[1]

CART (dla 16 procesorów 4×4.)

Podany tu spis deklaracji jest troche za bogaty dla przykładu, którym się zajmujemy. Nie warto jednak kasować zbędnych pozycji, gdyż mogą one przydać się, gdy będziemy chcieli zmodyfikować nasze zadanie.

```
c       DEKLARACJE
        integer size, source, dest, ierr, rcv_buff, snd_buff, tag, i, j
        integer sum, my_rank, left, right, top, bottom, row_rank
        integer col_rank, stat(MPI_Status_Size)
        integer requestr, requestc
        integer comm_old, comm_row, comm_col, comm_cart
        integer dims, ndims, max_dims
        parameter(max_dims=2)
        integer dims(max_dims), cart_coords(max_dims)
        logical reorder(max_dims), remain_dims(max_dims)
        logical periods(max_dims)

c       INICJALIZACJA MPI
        call MPI_Init(ierr)
```

[1] **Programy pisane w niektórych starszych wersjach Fortranu pozwalają na takie deklaracje jedynie w podprogramach. W związku z tym program główny tworzy się w postaci dwuczęsciowej: krótki 'właściwy' program główny, który podaje tylko pewne niezbędne parametry, oraz 'podprogram główny' w którym deklaracje (wymiaru) mogą zawierać zmienne.**

```
      call MPI_Comm_Rank(MPI_Comm_World, my_rank, ierr)
      call MPI_Comm_Size(MPI_Comm_World, size, ierr)

c     PIERWOTNY KOMUNIKATOR==>OLD
      comm_old=MPI_Comm_World

c     LICZBA WYMIARÓW CARTESIAN SPACE
      ndims=max_dims

c     LICZBA PROCESORÓW W KAŻDYM WYMIARZE
      data dims/4,4/

c     BEZ ZŁĄCZANIA BRZEGÓW W KAŻDYM WYMIARZE
      data periods/.false.,.false./

c     POZWALAM NA ZMIANĘ NUMERACJI
c     W NOWYM KOMUNIKATORZE
      reorder=.true.

c     UTWORZENIE NOWEGO KOMUNIKATORA
c     DLA CART 2D
      call MPI_Cart_Create
     $(comm_old, ndims, dims, periods, reorder, comm_cart, ierr)

      data remain_dims=(/.true.,.false./)

c     UTWORZENIE KOMUNIKATORA DLA WIERSZY
      call MPI_Cart_Sub
     $(comm_cart, remain_dims, comm_row, ierr)
      call MPI_Comm_Rank(comm_row, row_rank, ierr)

      data remain_dims=(/.false.,.true./)

c     UTWORZENIE KOMUNIKATORA DLA KOLUMN
      call MPI_Cart_Sub
     $(comm_cart, remain_dims, comm_col, ierr)
      call MPI_Comm_Rank(comm_col, col_rank, ierr)
```

Utworzone komunikatory pozwolą nam działać w poszczególnych grupach.

Działając na przykład w grupie wierszy, używamy komunikatora

MPI_comm_row

zamist komunikatora **MPI_Comm_World**.