MPI CZ. 2	PRZETWARZANIE WSPÓŁBIEŻNE

1. ROZSYŁANIE/ZBIERANIE DANYCH

MPI_Bcast - Funkcja rozsyła komunikat do wszystkich procesów w obrębie komunikatora comm poczynając od procesu root. Pozostałe argumenty - podobnie jak w MPI_Send().

```
int MPI_Bcast( void * buffer, int count, MPI_Dataty
pe datatype, int root, MPI_Comm comm );
```

buffer	adres do bufora z danymi do wysłania/odebrania.
count	ilość elementów w buforze.
datatype	typ danych.
root	numer procesu rozsyłającego dane.
comm	komunikator.

MPI_Reduce - Obliczanie równoległe. Bardzo ważna funkcja - pozwala wykonać na przykład sumowanie wszystkich częściowych wyników otrzymanych w procesach i umieszczenie wyniku w zmiennej. Argument root wskazuje dla którego procesu wynik ma być umieszczony w zmiennej result. Oto przykład użycia tej funkcji:

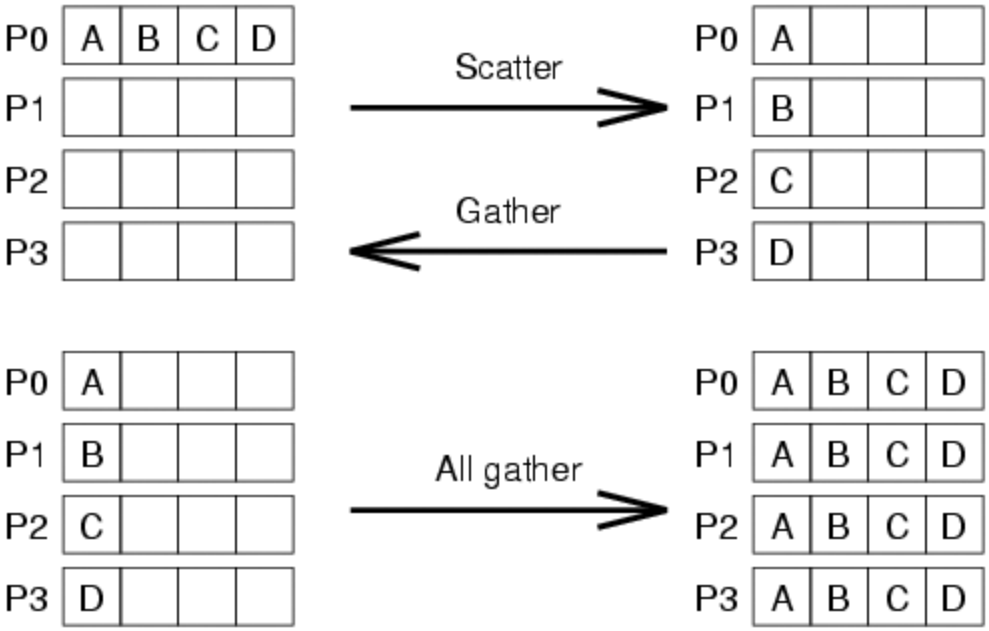
```
MPI_Reduce(&suma_cz,&suma,1,MPI_FLOAT,MPI_SUM,0,MPI_COMM_WORLD);
```

Przykładowe operatory to MPI_MAX, MPI_MIN, MPI_SUM - kompletna lista znajduje się w dokumentacji. Istnieje również możliwość definiowania własnych operatorów dla funkcji MPI_Reduce().

```
int MPI_Reduce( void * sendbuf, void * recvbuf, int
 count, MPI_Datatype datatype, MPI_Op
op, int root, MPI_Comm comm );
```

sendbuff	adres do bufora z danymi.
recvbuff	adres na bufor w którym mają znaleźć się wyniki obliczeń.
count	ilość elementów w buforze.
datatype	typ danych.
op	działanie do wykonania.
root	numer procesu rozsyłającego dane.
comm	komunikator.

MPI_Gather - zbieranie wszyscy do jednego,
MPI_Scatter - rozpraszanie jeden do wszystkich,



INSTALACJA SRODOWISKA MPI (UBUNTU):

```
sudo apt-get install libcr-dev mpich mpich-doc
```

KOMPILOWANIE PROGRAMU W ŚRODOWISKU MPI:

```
mpic++ program.cpp –o program
```

URUCHOMIENIE PROGRAMU:

```
mpirun –np 4 ./program
```

2. ZADANIE

Cel:

- Doskonalenie nabytych umiejętności programowania z wykorzystaniem kolektywnego przesyłania komunikatów MPI

Kroki:

1. Utworzenie katalogu roboczego.
2. Opracowanie programu obliczającego liczbę π z szeregu Leibniza:

$$1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \ldots = \frac{\pi}{4}$$

Proces o randze 0 powinien pobrać informację o liczbie sumowanych składników (podaną jako parametr przy uruchomieniu programu, z klawiatury itp.).

Liczba składników szeregu powinna zostać równo rozdzielona między procesy liczące sumy częściowe (należy rozwiązać problem w przypadku niepodzielności liczby składników przez liczbę procesów liczących).

3. Testowanie opracowanego programu (sprawdzenie poprawności otrzymanego wyniku).
4. Stałą gamma Eulera definiuje się jako granicę ciągu g_n gdzie:

$$g_n = \sum_{k-1}^{n} \frac{1}{n} - \ln(n)$$

Napisz program współbieżny w MPI, który oblicza g_n, przy następujących założeniach:

- n - liczba elementów sumy jest wczytywana z klawiatury
- p - liczba procesów, które będą liczyć sumę jest stałą programu
- każdy z p procesów liczy pewien fragment sumy tzn. przykładowo dla n=100, p=10, pierwszy proces liczy sumę od 1 do 10, drugi od 11 do 20, itd., na zakończenie sumy częściowe są dodawane i odejmowany jest logarytm, uwaga: n nie musi być podzielne przez p
- wynik jest wyświetlany na ekranie

Warunki zaliczenia:

1. Obecność na zajęciach i wykonanie wszystkich kroków.
2. Oddanie sprawozdania z opisem zadania, kodem źródłowym programów, wynikami i wnioskami.

Gamma Eulera. γ ≈ 0,577216