Antoni Pawlak

401480

Laboratorium 12: MPI

Cel ćwiczenia

- Opanowanie podstaw programowania z przesyłaniem komunikatów MPI.
- Poznanie składni MPI
- Kompilacja programu MPI

Wykonanie

Przygotowania

1. Utworzenie katalogu i pobranie plików ze strony
2. Instalacja MPI

Zadanie 1

1. Dla procesu zerowego ustawiamy ilosc wyrazow

```
if(rank == 0) {
  max_liczba_wyrazow = 100;
}
```

2. Przesyłamy ilość wyrazów do pozostałych wątków (rozgłaszamy)

```
MPI_Bcast(&max_liczba_wyrazow, 1, MPI_INT, 0 , MPI_COMM_WORLD);
```

3. Ustalamy wielkość obliczeń w każdym wątku

```
int N = ceil((float)max_liczba_wyrazow / size);
```

4. Ustalamy początek I koniec danych

```
int own_start = N * rank;
int own_end = N * (rank+1);

if(rank == size - 1){
own_end = max_liczba_wyrazow;
}
```

5. Wykonujemy obliczenia

```
for(i=own_start; i<own_end; i++)
```

6. Zliczamy cząstkowe wyniki do całościowej sumy

```
MPI_Reduce(&suma, &wynik_pi, 1, MPI_DOUBLE, MPI_SUM,0,MPI_COMM_WORLD);
```

7. Kończymy proces MPI

```
MPI_Finalize();
```

Zadanie 2

1. **Rozdzielamy tablicę pomiędzy wątki**

MPI_Scatter(a, n_wier*WYMIAR, MPI_DOUBLE, a_local, n_wier * WYMIAR, MPI_DOUBLE, 0, MPI_COMM_WORLD);

2. **Rozdzielamy tablicę wynikową pomiędzy wąrki**

MPI_Scatter(x, n_wier, MPI_DOUBLE, &x[n_wier*rank], n_wier, MPI_DOUBLE, 0, MPI_COMM_WORLD);

3. **Zbieramy tablice wynikowe**

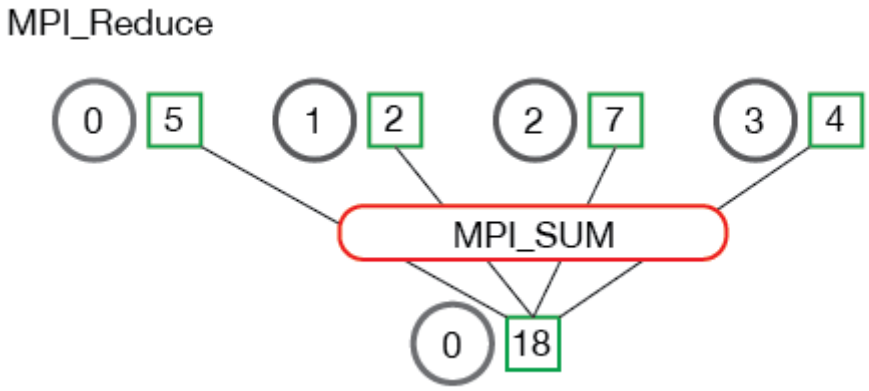
```
MPI_Allgather(&x[rank*n_wier], n_wier, MPI_DOUBLE, x, n_wier, MPI_DOUBLE,
MPI_COMM_WORLD );
```

```
if (rank == 0) {
MPI_Gather(MPI_IN_PLACE, n_wier, MPI_DOUBLE, z, n_wier, MPI_DOUBLE, 0,
MPI_COMM_WORLD);
}
else {
MPI_Gather(&z, n_wier, MPI_DOUBLE, od, n_wier, MPI_DOUBLE, 0, MPI_COMM_WORLD);
}
```

Wnioski

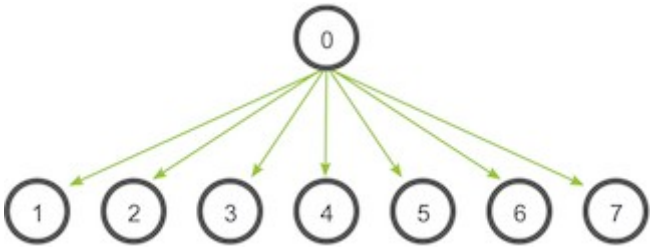
MPI_Reduce

Jedna z funkcji komunikacji grupowej, pozwala na agregowanie wartości odbieranych na komunikatorze. Możliwe jest ustawienie operacji agregującej oraz procesu który je wykonuje. W naszym przypadku została wykorzystana do zliczania sumy całki.



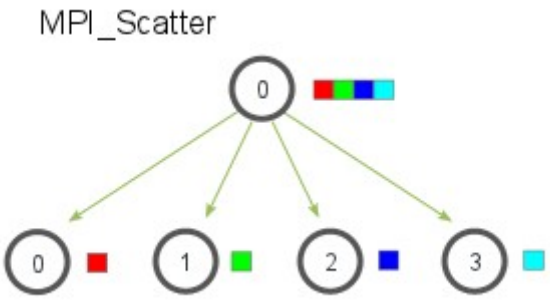
MPI_Bcast

Broadcast jest jedną ze standardowych technik komunikacji zbiorowej. Podczas transmisji jeden proces wysyła te same dane do wszystkich procesów w komunikatorze. Jednym z głównych zastosowań nadawania jest wysyłanie danych wejściowych użytkownika do programu równoległego lub wysyłanie parametrów konfiguracyjnych do wszystkich procesów.



MPI_Scatter

MPI_Scatter polega na wyznaczonym procesie roota wysyłającym dane do wszystkich procesów w komunikatorze. Podstawowa różnica między MPI_Bcast a MPI_Scatter jest niewielka, ale ważna. MPI_Bcast wysyła ten sam fragment danych do wszystkich procesów, podczas gdy MPI_Scatter wysyła fragmenty tablicy do różnych procesów.



MPI_Gather

Zamiast rozkładać elementy z jednego procesu na wiele procesów, MPI_Gather pobiera elementy z wielu procesów i gromadzi je w jednym procesie. Ta procedura jest bardzo przydatna dla wielu algorytmów równoległych, takich jak sortowanie równoległe i wyszukiwanie