Operacje grupowego przesyłania komunikatów MPI

Operacje grupowego przesyłania komunikatów

- Operacje, w ramach których ten sam komunikat lub zbiór komunikatów przesyłany jest pomiędzy więcej niż dwoma procesami nazywane są operacjami komunikacji grupowej (*collective communication*) lub operacjami globalnymi
- Przykładami takich operacji są m.in. rozgłaszanie (*broadcast*), rozpraszanie (*scatter*), zbieranie (*gather)*, redukcja (*reduction*) czy wymiana (*exchange*)
- Wydajność realizacji takich operacji może być bardzo różna, zależnie od przyjętej strategii
- Optymalna strategia realizacji wymaga często uwzględnienia architektury systemu komputerowego i sposobu przesyłania komunikatów w sieci połączeń

Operacje grupowego przesyłania komunikatów

- Schematy grupowego przesyłania komunikatów:
  - rozgłaszanie (*broadcast*) jeden do wszystkich
  - rozpraszanie (*scatter*) jeden do wszystkich
  - zbieranie (*gather)* wszyscy do jednego
  - redukcja (gromadzenie, *reduction*) wszyscy do jednego
  - rozgłaszanie wszyscy do wszystkich (równoważne zbieraniu wszyscy do wszystkich)
  - gromadzenie (redukcja) wszyscy do wszystkich
  - wymiana wszyscy do wszystkich (równoważna rozpraszaniu wszyscy do wszystkich)
- Nieoptymalna i optymalna realizacja rozgłaszania dla *p* procesorów i różnych technologii i topologii sieciowych

Procedury komunikacji grupowej MPI

- Realizacja procedur komunikacja grupowej w MPI polega na wywołaniu odpowiedniej procedury przez wszystkie procesy w grupie
- Argumentami procedur komunikacji grupowej są i bufory z danymi wysyłanymi i (jeśli trzeba) bufory na dane odbierane
  - w przypadku, gdy bufory danych wysyłanych i odbieranych pokrywają się (np. proces dokonujący rozproszenia danych pozostawia swoją część danych w tym samym miejscu) MPI może wymagać użycia argumentu MPI_IN_PLACE, zamiast jednego z buforów
- Wszystkie procedury komunikacji grupowej są blokujące (ale zakończenie operacji przez proces nie oznacza koniecznie, że inne procesy także ją zakończyły)

Procedury komunikacji grupowej MPI

- bariera

```
int MPI_Barrier( MPI_Comm comm )
```

- rozgłaszanie jeden do wszystkich

```
int MPI_Bcast( void *buff,   int count,   MPI_Datatype datatype,   int root,
                    MPI_Comm comm )
```

- zbieranie wszyscy do jednego

```
int MPI_Gather(void *sbuf, int scount, MPI_Datatype sdtype,  void *rbuf,
        int rcount, MPI_Datatype rdtype, int root, MPI_Comm comm)
```

- zbieranie wszyscy do wszystkich – równoważne rozgłaszaniu wszyscy do wszystkich (brak wyróżnionego procesu root)

```
int MPI_Allgather( void *sbuf,   int scount,   MPI_Datatype sdtype,
        void *rbuf, int rcount, MPI_Datatype rdtype, MPI_Comm comm )
```

Procedury komunikacji grupowej MPI

- rozpraszanie jeden do wszystkich

```
int MPI_Scatter(void *sbuf, int scount, MPI_Datatype sdtype,  void *rbuf,
        int rcount, MPI_Datatype rdtype, int root, MPI_Comm comm)
```

- rozpraszanie wszyscy do wszystkich równoważne wymianie wszyscy do wszystkich (brak wyróżnionego procesu root)

```
int MPI_Alltoall( void *sbuf,   int scount,   MPI_Datatype sdtype,
    void *rbuf,   int rcount,   MPI_Datatype rdtype,   MPI_Comm comm )
```

- redukcja wszyscy do jednego

```
int MPI_Reduce(void *sbuf, void *rbuf, int count, MPI_Datatype datatype,
MPI_Op op, int root, MPI_Comm comm)
```

- redukcja połączona z rozgłaszaniem – MPI_Allreduce

Procedury komunikacji grupowej MPI

- Operacje stosowane przy realizacji redukcji:
  - predefiniowane operacje – uchwyty do obiektów typu MPI_Op (każda z operacji ma swoje dozwolone typy argumentów):
    - MPI_MAX – maksimum
    - MPI_MIN – minimum
    - MPI_SUM – suma
    - MPI_PROD – iloczyn
    - operacje maksimum i minimum ze zwróceniem indeksów
    - operacje logiczne i bitowe
  - operacje definiowane przez użytkownika za pomocą procedury:

```
int MPI_Op_create(MPI_User_function *func, int commute,
  MPI_Op *pop);
```

Procedury komunikacji grupowej MPI – przykład 1

- Całkowanie w przedziale (przypomnienie)
  - warianty dekompozycji:
    - dekompozycja w dziedzinie w problemu
      - podział przedziału na podprzedziały
    - zrównoleglenie pętli
      - algorytm sekwencyjny
      - narzędzia zrównoleglenia pętli:
        - » automatyczne (OpenMP)
        - » ręczne (Pthreads,MPI)
    - wyniki w obu przypadkach mogą być różne
      - różne liczby podprzedziałów i ich długości

Procedury komunikacji grupowej MPI – przykład 1

```
MPI_Init( &argc, &argv );
MPI_Comm_rank( MPI_COMM_WORLD, &rank );
MPI_Comm_size( MPI_COMM_WORLD, &size );
if( rank == 0 ) scanf("%lf %lf %d", &a, &b, &N);
root=0;
MPI_Bcast( &N, 1, MPI_INT, root, MPI_COMM_WORLD );
MPI_Bcast( &a, 1, MPI_DOUBLE, root, MPI_COMM_WORLD );
MPI_Bcast( &b, 1, MPI_DOUBLE, root, MPI_COMM_WORLD );
n_loc = ceil(N/size);   dx = (b-a)/N;   x1 = a + rank*n_loc*dx;
if(rank==size-1)   n_loc = N – n_loc*(size-1);  // N>>size
c=0;
for(i=0;i<n_loc;i++) {
    x2 = x1+dx;   c += 0.5*(f(x1)+f(x2))*dx;   x1=x2;
}
MPI_Reduce( &c, &calka, 1, MPI_DOUBLE, MPI_SUM, root, MCW);
```

Procedury komunikacji grupowej MPI – przykład 2

- Mnożenie macierz-wektor
  - algorytm sekwencyjny, naiwny
  - dekompozycja danych
    - warianty:
      - wierszowy
      - kolumnowy
      - blokowy
    - algorytmy dla rozmaitych wariantów dekompozycji
  - implementacja
    - procedury MPI wykorzystywane w implementacji

Procedury komunikacji grupowej MPI – przykład 2

```
double x[WYMIAR], y[WYMIAR], a[WYMIAR*WYMIAR];
// inicjowanie a, x, y
MPI_Init( &argc, &argv );
MPI_Comm_rank( MPI_COMM_WORLD, &rank );
MPI_Comm_size( MPI_COMM_WORLD, &size );
n_wier = ceil(WYMIAR / size);
// dodatkowy kod dla WYMIAR niepodzielnego przez size

MPI_Allgather(&x[rank*n_wier], n_wier, MPI_DOUBLE,
              x, n_wier,  MPI_DOUBLE, MPI_COMM_WORLD );
// zamiast &x[rank*n_wier] można użyć MPI_IN_PLACE – obszar
// danych od tego adresu pozostaje bez zmian w każdym procesie

for(i=0;i<n_wier;i++){
    int ni = n*i;
    for(j=0;j<n;j++){
        y[i] += a[ni+j] * x[j];
    }
}
```