Parallele und verteilte Systeme
WS 2020/2021
Deadline: 07.01.2021 12:00 Uhr

Bauhaus-Universität Weimar

Übung 3

Teil 1. Das erste Programm zum verteilten Rechnen

```
1  #include<stdio.h>
2  #include<mpi.h>                                    // ???
3
4  int main(int argc, char** argv){
5      int nodeID, numNodes;
6
7      /* ??? */
8      MPI_Init(&argc, &argv);                        // ???
9      MPI_Comm_size(MPI_COMM_WORLD, &numNodes);      // ???
10     MPI_Comm_rank(MPI_COMM_WORLD, &nodeID);        // ???
11
12     /* ??? */
13     printf("Hello world from process %d of %d\n", nodeID, numNodes);
14
15     /* ??? */
16     MPI_Finalize();
17
18     return 0;
19 }
```

1. Kommentieren Sie den Quelltext aussagekräftig an den dafür vorgesehenen Stellen.
2. Richten Sie Ihre Entwicklungsumgebung für die Benutzung von OpenMPI ein. ***Hinweise zur Einrichtung unter Linux und Windows sind auf den Übungsfolien zu finden.***
3. Kompilieren Sie das Programm und führen Sie es aus.

Teil 2. Matrix-Multiplikation

Versuchen Sie die **Matrixmultiplikation zu beschleunigen**, indem Sie die **3** unten beschriebenen **Varianten** verteilter Programmierung mit Hilfe von OpenMPI (bzw. MS-MPI) implementieren. Benutzen Sie die Datei `matmult.cpp` als Vorlage für die serielle Multiplikation zweier Matrizen.

1. (***Variante 1***) Verteilen Sie die Aufgabe mit Hilfe von MPI auf mehrere Arbeiter. ***Hinweis***: Ein Master-Task soll die Matrix A zeilenweise in Blöcke aufteilen und auf die Arbeiter-Tasks verteilen. Alle Matrizen sollen die feste Größe 1000 x 1000 haben. Benutzen Sie dabei blockierende Funktionen wie `MPI_Send, MPI_Recv`.

Computer Vision in Engineering – Prof. Dr. Rodehorst
M.Sc. Mariya Kaisheva
mariya.kaisheva@uni-weimar.de

2. (***Variante 2***) Ersetzen Sie etwaige blockierende Kommunikationsfunktionen durch entsprechende nicht-blockierende Funktionen wie `MPI_Isend, MPI_Irecv`. [optional]

3. (***Variante 3***) Verwenden Sie kollektive Kommunikation mit Funktionen wie `MPI_Bcast, MPI_Scatter, MPI_Gather`. ***Hinweis***: Relevante Information für die kollektive Kommunikation finden Sie in der **7. Vorlesung**. Die Hauptarbeitsschritten für diese 3. Variante können so zusammengefasst werden:
    - verteilen Sie Matrix B auf die Arbeiter
    - verteilen Sie Matrix A zeilenweise in Blöcke an die Arbeiter
    - sammeln Sie die Teilergebnisse von C ein

4. Bestimmen Sie die Laufzeiten
    - der blockierenden Variante 1
    - der nicht-blockierenden Variante 2 [optional]
    - der Broadcast-basierten Variante 3

    Messen Sie insbesondere die zeitlichen Anteile:
    - zur Verteilung der Daten
    - der eigentlichen Berechnung
    - zum Einsammeln der Daten

    jeweils für 1, 2 und 4 Arbeiter.

5. Weisen Sie nach, dass Ihre verteilte Version korrekte Ergebnisse liefert wie eine nicht-verteilte Referenzimplementierung.

6. Schreiben Sie das Programm so um, dass die Matrixmultiplikation mit Matrizen beliebiger Größe durchgeführt werden kann. [optional]

Hinweis: OpenMPI kann auch auf Multi-Core Rechnern (anstatt auf Rechencluster) eingesetzt werden. Für Nutzung- und Einrichtungsunterstützung beziehen Sie sich bitte auf die Folien von Übung 3.

Alle Gruppen, die **bis spätestens 4. Januar** 2021 eine korrekt funktionierende Lösung von Aufgabe 2.1 / 2.3 und Aufgabe 2.4 implementieret haben, können diese Lösungen per E-Mail an mariya.kaisheva@uni-weimar.de schicken. Diese Lösungen werden für euch dann auf einen Rechner-Cluster verteilt ausgeführt und die gemessenen Laufzeiten werden dann während der Übung am 8. Januar präsentiert.

Computer Vision in Engineering – Prof. Dr. Rodehorst
M.Sc. Mariya Kaisheva
mariya.kaisheva@uni-weimar.de