# Paralleles Rechnen – Übung 03

Benjamin Reißaus – 749897

Felix Kubicek – 749062

3.1

2.)

Insgesamt werden wesentlich weniger Lines of Code gebraucht, da nicht explizit Threads erstellt werden müssen und viel von OpenMP übernommen wird. Zusätzlich muss der sequentielle Code nicht großartig umgeschrieben werden. Es müssen lediglich die zusätzlichen OpenMP-Pragmas an die richtigen Stellen hinzugefügt werden.

3.)

Eine Möglichkeit besteht darin die globale *state* Variable „thread-safe“ zu machen. Dies ist mit dem folgenden Pragma möglich: #pragma omp threadprivate(var)“. Somit bekommt jeder thread eine eigene Kopie dieser Variable, auf die auch nur er Zugriff hat.

3.2

1.) MPI\_SCATTER hat die folgende Signatur:

int MPI\_Scatter(const void \*sendbuf, int sendcount, MPI\_Datatype sendtype, void \*recvbuf, int recvcount, MPI\_Datatype recvtype, int root, MPI\_Comm comm)

sendbuf – bekommt die Adresse des Sendepuffers

sendcount – die Anzahl der Elemente, die aus dem Sendepuffer pro Empfängerprozess, verschickt werden

sendtype – der Datentyp der Elemente im Sendepuffer

recvcount – die Anzahl der Elemente, die im Empfängerpuffer empfangen werden

recvtype – der Datentypen der Elemente im Empfängerpuffer

root – Rang des Senderprozesses

comm – die Gruppe, in der der Empfängerprozess vorhanden ist

MPI\_Scatter teilt den Sendepuffer in Chunks der Größe sendcount und verteilt die chunks an die Prozesse der Zielgruppe. Dabei bekommt jeder Prozess in der Zielgruppe seinen eigenen chunk. Wenn sendbuf die größe x hat und z Prozesse in der Zielgruppe existieren, dann sollte sendcount = x/z sein, damit alle Prozesse in der Zielgruppe ähnlich große Chunks bekommen.

2.)

int myRank, tag = 42;

char buf[20];

MPI\_Status status;

MPI\_Init(&argc, &argv);

MPI\_Comm\_Rank(MPI\_COMM\_WORLD, &myRank);

chunk\_size = size(sendbuf) / MPI\_Comm\_size

rest = size(sendbuf) % MPI\_Comm\_size

if (myRank == 0){

for(i = 1; i < MPI\_Comm\_size; i++){

if(i == MPI\_Comm\_size - 1){

chunk\_size =+ rest

}

MPI\_Send(sendbuf + i \* chunksize, chunk\_size, MPI\_CHAR, i, tag, MPI\_COMM\_WORLD)

}

}

else

MPI\_Recv(sendbuf, chunk\_size, MPI\_CHAR, 0, tag, MPI\_COMM\_WORLD, &status);

MPI\_Finalize();