HPC, UE04 – Matrix Multiply

Zum Testen dieser Aufgabe wurde zuerst mit einer kleineren Matrix gestartet:

```
2
4
2

1 2 3 4
4 5 6 5

2 3
4 5
6 7
8 9
```

Diese wurde mit 1, 2 und 3 Knoten mit MPI getestet. Ersteres muss immer funktionieren, 2 ist das Maximum an Knoten, das sowohl n und m teilt, und 3 sollte einen Fehler werfen. Das ist auch der Fall:

```
C:\Data\git\Development\FH\CPP\HPC3\04_MPI_MatrixMultiply\x64\Debug>mpiexec -n 1 04_MPI_MatrixMultiply.exe
[RA] Rank 0 of 1
0 has the result C:
 60     70
104     124


C:\Data\git\Development\FH\CPP\HPC3\04_MPI_MatrixMultiply\x64\Debug>mpiexec -n 2 04_MPI_MatrixMultiply.exe
[RA] Rank 1 of 2
[RA] Rank 0 of 2
0 has the result C:
 60     70
104     124


C:\Data\git\Development\FH\CPP\HPC3\04_MPI_MatrixMultiply\x64\Debug>mpiexec -n 3 04_MPI_MatrixMultiply.exe
[RA] Rank 0 of 3
[RA] Rank 1 of 3
[RA] Rank 2 of 3
Either m or n is not dividable by the node count without remainder. Terminating.

C:\Data\git\Development\FH\CPP\HPC3\04_MPI_MatrixMultiply\x64\Debug>
```

Laut Wolfram Alpha ist auch der Output korrekt:

Input

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 5 & 6 & 5 \end{pmatrix} \cdot \begin{pmatrix} 2 & 3 \\ 4 & 5 \\ 6 & 7 \\ 8 & 9 \end{pmatrix}$$

Result

$$\begin{pmatrix} 60 & 70 \\ 104 & 124 \end{pmatrix}$$

Nachfolgend wurde eine größere Matrix, verwendet, die (selbst transponiert) verschiedene Dimensionen aufweisen. Hierfür wurde das zur Verfügung gestellte Sample-Beispiel um eine Zeile bei Matrix A erweitert. Ansonsten wäre nur die Knotenanzahl = 1 möglich gewesen (5 und 6 hat keinen größeren gemeinsamen Teiler):

```
6
6
4

5 4 2 8 6 5
5 8 7 9 2 0
2 4 0 1 4 4
8 7 9 0 0 3
3 0 1 2 3 1
1 2 3 4 5 6

5 7 3 2
2 1 2 4
5 3 4 2
8 3 6 9
3 5 4 7
6 6 7 4
```

Die Tests sind wieder gleich aufgebaut: 1 Knoten, mehrere valide Varianten, und invalide Variante.

```
C:\Data\git\Development\FH\CPP\HPC3\04_MPI_MatrixMultiply\x64\Debug>mpiexec -n 1 04_MPI_MatrixMultiply.exe
[RA] Rank 0 of 1
0 has the result C:
 155    129     138     164
154     101     121     151
62      65      64      73
117     108     95      74
51      51      44      51
107     91      105     111
```

```
C:\Data\git\Development\FH\CPP\HPC3\04_MPI_MatrixMultiply\x64\Debug>mpiexec -n 2 04_MPI_MatrixMultiply.exe
[RA] Rank 1 of 2
[RA] Rank 0 of 2
0 has the result C:
 155    129     138     164
154     101     121     151
62      65      64      73
117     108     95      74
51      51      44      51
107     91      105     111


C:\Data\git\Development\FH\CPP\HPC3\04_MPI_MatrixMultiply\x64\Debug>mpiexec -n 3 04_MPI_MatrixMultiply.exe
[RA] Rank 1 of 3
[RA] Rank 0 of 3
[RA] Rank 2 of 3
0 has the result C:
 155    129     138     164
154     101     121     151
62      65      64      73
117     108     95      74
51      51      44      51
107     91      105     111


C:\Data\git\Development\FH\CPP\HPC3\04_MPI_MatrixMultiply\x64\Debug>mpiexec -n 6 04_MPI_MatrixMultiply.exe
[RA] Rank 0 of 6
[RA] Rank 4 of 6
[RA] Rank 2 of 6
[RA] Rank 3 of 6
[RA] Rank 1 of 6
[RA] Rank 5 of 6
0 has the result C:
 155    129     138     164
154     101     121     151
62      65      64      73
117     108     95      74
51      51      44      51
107     91      105     111
```

```
C:\Data\git\Development\FH\CPP\HPC3\04_MPI_MatrixMultiply\x64\Debug>mpiexec -n 4 04_MPI_MatrixMultiply.exe
[RA] Rank 3 of 4
[RA] Rank 2 of 4
[RA] Rank 0 of 4
[RA] Rank 1 of 4
Either m or n is not dividable by the node count without remainder. Terminating.

C:\Data\git\Development\FH\CPP\HPC3\04_MPI_MatrixMultiply\x64\Debug>mpiexec -n 5 04_MPI_MatrixMultiply.exe
[RA] Rank 4 of 5
[RA] Rank 0 of 5
[RA] Rank 2 of 5
[RA] Rank 1 of 5
[RA] Rank 3 of 5
Either m or n is not dividable by the node count without remainder. Terminating.
```

Und wiederum der Goldstandard mit Wolfram Alpha

Input

$$\begin{pmatrix} 5 & 4 & 2 & 8 & 6 & 5 \\ 5 & 8 & 7 & 9 & 2 & 0 \\ 2 & 4 & 0 & 1 & 4 & 4 \\ 8 & 7 & 9 & 0 & 0 & 3 \\ 3 & 0 & 1 & 2 & 3 & 1 \\ 1 & 2 & 3 & 4 & 5 & 6 \end{pmatrix} \cdot \begin{pmatrix} 5 & 7 & 3 & 2 \\ 2 & 1 & 2 & 4 \\ 5 & 3 & 4 & 2 \\ 8 & 3 & 6 & 9 \\ 3 & 5 & 4 & 7 \\ 6 & 6 & 7 & 4 \end{pmatrix}$$

Result

$$\begin{pmatrix} 155 & 129 & 138 & 164 \\ 154 & 101 & 121 & 151 \\ 62 & 65 & 64 & 73 \\ 117 & 108 & 95 & 74 \\ 51 & 51 & 44 & 51 \\ 107 & 91 & 105 & 111 \end{pmatrix}$$