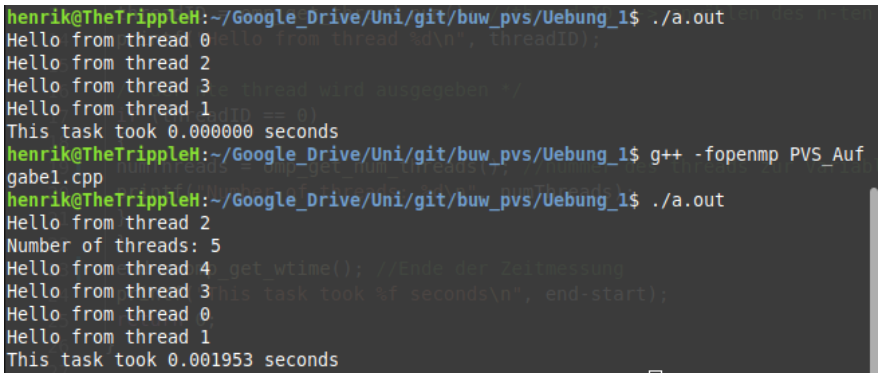
Software Engineering Übung 01

Lennard Gabriel (Matrikelnr. 118147)
Vanessa Retz (Matrikelnr. 117380)
Henrik Leisdon (Matrikelnr. 118334)

November 1, 2018

Aufgabe 1

Wenn man die variable num_threads verändert tritt folgendes auf:
(erster Durchlauf verändern der Variable 4, zweiter Durchlauf auf 5)



Aufgabe 2

1.

3 ineinander verschachtelte for Schleifen sind quasi prädestiniert dazu, diese durch ein parallel for zu beschleunigen, da genau an dieser Stelle die Matrixmultiplikation durchgefüht wird.

2.

Die parallelisierte Version liefert korrekte Ergebnisse, da sowohl die nicht beschleunigte Version, als auch die beschleunigte Version identische Matrizen zurück geben:

```
Matrix sizes C[3][3] = A[3][3] x B[3][3]
Perform parallel matrix multiplication...
Benoetigte Zeit ohne Parallel: 0.000002 Sekunden
Ausgabe der matrix ohne BeschleunigungMatrix A:
   3.0      6.0      7.0
   5.0      3.0      5.0
   6.0      2.0      9.0
Matrix B:
   1.0      2.0      7.0
   0.0      9.0      3.0
   6.0      0.0      6.0
Matrix C:
  45.0     60.0     81.0
  35.0     37.0     74.0
  60.0     30.0    102.0
Benoetigte Zeit mit Parallel: 0.000573 Sekunden
Ausgabe der matrix mit BeschleunigungMatrix A:
   3.0      6.0      7.0
   5.0      3.0      5.0
   6.0      2.0      9.0
Matrix B:
   1.0      2.0      7.0
   0.0      9.0      3.0
   6.0      0.0      6.0
Matrix C:
  90.0    120.0    162.0
  70.0     74.0    148.0
 120.0     60.0    204.0

Done.
```

3.

Ein Speedup ist bei größeren Matrizen zu sehen, da bei kleinen das Verteilen der Schleife auf die einzelnen Threads länger dauert, als das eigentliche rechnen.

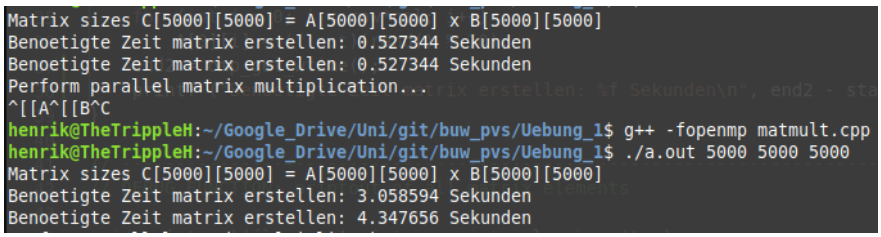
```
henrik@TheTrippleH:~/Google_Drive/Uni/git/buw_pvs/Uebung_1$ ./a.out 3 3 3
Matrix sizes C[3][3] = A[3][3] x B[3][3]
Perform parallel matrix multiplication...
Benoetigte Zeit ohne Parallel: 0.000001 Sekunden
Benoetigte Zeit mit Parallel: 0.000236 Sekunden

Done.
```

Jedoch gibt es eine beschleunigung bei größeren Matritzen z.B. 1000x1000:
Im Beispiel liegt der speedup bei ca 62%.

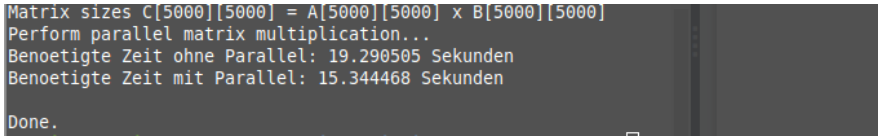
```
henrik@TheTrippleH:~/Google_Drive/Uni/git/buw_pvs/Uebung_1$ ./a.out 1000 1000 10
00
Matrix sizes C[1000][1000] = A[1000][1000] x B[1000][1000]
Perform parallel matrix multiplication...
Benoetigte Zeit ohne Parallel: 23.834703 Sekunden
Benoetigte Zeit mit Parallel: 8.935673 Sekunden

Done.
```

4.

Das parallelisieren der For Schleife zum generieren der Matrix hat einen negativen Effekt auf die Geschwindigkeit des programmes. Bei Erstellen einer 10000x10000 Matrix brauch das Programm ohne parallelisieren nur 2 Sekunden, während die Zeit mit parallelisierung zwischen 10 und 15 Sekunden liegt. Das Aufteilen auf die threads dauert also deutlich länger, als nur das reine Berechnen der Zufallszahlen.



Die zweite Beschleunigung haben wir bei der Berechnung der Matrixmultiplikation als omp atomic eingesetzt. Die hat bei einer 5000x5000 Matrix einen speedup von 3 sekunden erbracht:



Durch das Verteilen der Berechnung der Matrixmultiplikation werden mehrere Berechnungen gleichzeitig durchgeführt, wodurch es zu einem Speedup kommt.