Fakultät für Informatik und Mathematik
Lehrstuhl für Programmierung
Prof. C. Lengauer, Ph.D.

WS 2010/11

Übungen zur Vorlesung: Praktische Parallelprogrammierung
Blatt 2 (Bildverarbeitung: Grauwertskalierung)

Aufgabe 2 (Grauwertskalierung) [2+2+6 Punkte]

Schreiben Sie ein Programm, das ein Graustufenbild im PGM (Portable Graymap) Format einliest, den Grauwert jedes Bildpunkts auf ein gegebenes Intervall skaliert und das Bild wieder abspeichert (in einer neuen Datei).
Die Skalierung der Grauwerte soll dabei folgendermaßen erfolgen:

- Der Benutzer gibt das gewünschte Intervall $[n_{\min}, n_{\max}]$ der neuen Grauwerte vor, z. B. über Kommandozeilenparameter.
- Nach dem Laden des Bildes werden der minimale Grauwert $a_{\min}$ und der maximale Grauwert $a_{\max}$ des Bildes bestimmt. (Nehmen Sie nicht einfach 0 und `maxval`.)
- Der neue Grauwert n eines Bildpunktes mit dem alten Grauwert a bestimmt sich nach folgender Formel:

$$n = \frac{(a - a_{\min}) \cdot (n_{\max} - n_{\min}) + \frac{a_{\max} - a_{\min}}{2}}{a_{\max} - a_{\min}} + n_{\min}$$

  Die Division ist dabei die ganzzahlige Division. (Der Term $+\frac{a_{\max} - a_{\min}}{2}$ sorgt dafür, dass korrekt gerundet wird.)

(a) Implementieren Sie eine sequienzielle Variante der Grauwertskalierung.

(b) Reichern Sie Ihr Programm mit OpenMP-Direktiven für Shared Memory Parallelität an. Führen Sie die Bestimmung der minimalen und maximalen Grauwerte, sowie die Skalierung parallel durch.

(c) Implementieren Sie die Grauwertskalierung mit Hilfe von MPI. Verteilen Sie dazu die Bilddaten auf die Prozessoren (indem Sie auf jedem Knoten nur einen Teil des Bildes laden), bestimmen Sie $a_{\min}$ und $a_{\max}$ mit Hilfe einer Reduktion (`MPI_Allreduce`), und fügen das Bild nach dem Skalieren wieder zusammen (beispielsweise mit `MPI_Gather`). Sie können zusätzlich OpenMP-Direktiven für mehr Parallelität verwenden. Rufen Sie MPI-Funktionen nur vom Master Thread außerhalb von `parallel` Konstrukten auf.

Sie können alle drei Varianten in einer Quelldatei implementieren. Messen Sie die Laufzeiten der einzelnen Programmteile (Laden, Verteilen, Reduktion, Skalierung, Gather) auf 1,2,4 Knoten (hermes) bzw. 1,2,4,8,16,32 Cores (`hydra`). Testen Sie Ihr Programm mit dem Bild `passau.pgm` (in `/home/hpcuser/ppp2010-data/images`) für MPI. Für Benchmarks auf `hydra` verwenden Sie bitte das 10000×10000-Bild `big.pgm` in `/home/hpcuser/ppp2010-data/images`; die Ausgabe können Sie nach `/dev/null` leiten.

Schicken Sie den Quelltext Ihrer Programme per E-Mail an Armin Größlinger (`groessli@fim.uni-passau.de`) und geben Sie einen Ausdruck in der Vorlesung am 16.11.2010 ab. Bitte beachten Sie die Deadline! Aus prüfungsrechtlichen Gründen können verspätete Abgaben nicht akzeptiert werden!

Abgabe der Lösung: E-Mail bis Dienstag, 16.11.2010, 12.00 Uhr, Ausdruck in der Vorlesung

Hinweise zum Laden/Speichern von Bildern

Im Verzeichnis `/home/hpcuser/ppp2010-data/packages/ppp_pnm-32` bzw. `/home/hpcuser/ppp2010-data/packages/ppp_pnm-64` ist eine kleine Bibliothek zum Laden von PGM-Bildern installiert (als 32-bit Version für hermes bzw. 64-bit Version für hydra).

In Stud.IP finden Sie ein Beispielprogramm zur Benutzung der Bibliothek. Sie können in Ihrem C Programm die Bibliothek benutzen, indem Sie

```
#include "ppp_pnm.h"
```

in Ihre Quelldatei aufnehmen und den Compiler z. B. mit

```
mpigcc4.mpich2 -Wall -O3 -o datei datei.c
    -I/home/hpcuser/ppp2010-data/packages/ppp_pnm-32
    -L/home/hpcuser/ppp2010-data/packages/ppp_pnm-32
    -lppp_pnm
```

aufrufen um Include-Pfad (`-I`), Linker-Pfad (`-L`) und Linker-Optionen (`-l`) zu setzen. Die Option `-O3` schaltet eine hohe Optimierungsstufe ein.

Eine Beschreibung der Funktionen von `ppp_pnm` finden Sie auch in der Datei `ppp_pnm.h` in `/home/hpcuser/ppp2010-data/packages/ppp_pnm-32`.

Bilder können Sie auf dem Frontend mit dem Kommando `display` anzeigen.

Hinweise zu OpenMP

Um OpenMP benutzen zu können, müssen Sie GCC Version 4 benutzen (`gcc4 -fopenmp` auf `hermes` bzw. `gcc -fopenmp` auf `hydra`). Um MPI und OpenMP gleichzeitig benutzen zu können, verwenden Sie

```
mpigcc4.mpich2 -Wall -fopenmp -O3 -o datei datei.c
```

auf `hermes` bzw. `mpicc ...-fopenmp ...` auf `hydra`. Die `ppp_pnm` Bibliothek binden Sie mit den gleichen Optionen ein wie oben.

Um die Anzahl der Threads bei der Ausführung zu bestimmen, benutzen Sie `omp_wrapper`:

```
omp_wrapper -t T ./datei
```

wobei T die Anzahl der Threads ist. `omp_wrapper` kann auch mit `mpiexec` kombiniert werden für Programme, die MPI und OpenMP benutzen:

```
mpiexec -n N omp_wrapper -t T ./datei ...
```