

Einführung in das Hochleistungsrechnen

Sommersemester 2019

Übung 10

Aufgabe 1 (6 + 2 = 8 Punkte).

Wir betrachten das parallele m -Farben-SOR-Verfahren (Algorithmus 4.8 der Vorlesung) zum Lösen der diskreten Laplace-Gleichung

$$-u_{i,j-1} - u_{i-1,j} + 4u_{i,j} - u_{i,j+1} - u_{i+1,j} = h^2 f_{i,j}, \quad i, j = 1, \dots, N$$

auf einem regelmäßigen Gitter der Größe $(N+2) \times (N+2)$, $N \in \mathbb{N}$ (d.h. mit Schrittweite $h := 1/(N+1)$) mit den Randbedingungen

$$\begin{aligned} u_{0,j} &= g_{0,j}, & u_{N+1,j} &= g_{N+1,j}, & j &= 1, \dots, N, \\ u_{i,0} &= g_{i,0}, & u_{i,N+1} &= g_{i,N+1}, & i &= 1, \dots, N. \end{aligned}$$

Weiterhin betrachten wir eine zweidimensionale Partition des Gitters in Quadrate (siehe Abbildung 1) und ordnen jedem der $p = P^2$ Prozessoren $P_{l,j}$, $l, j = 1, \dots, P$ ein Teilgebiet der Größe $N/P \times N/P$ zu. Wir gehen hier zur Vereinfachung davon aus, dass N durch P teilbar ist.

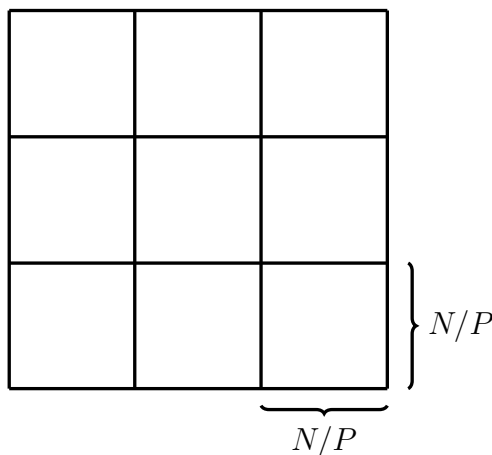


Abbildung 1: Partition des $N \times N$ -Gitters in Quadrate mit $p = P^2 = 9$.

In der Vorlesung haben wir folgende Beobachtung gemacht: Für die Parallelisierbarkeit des SOR-Verfahrens genügt es, wenn die Knoten einer Farbe entkoppelt sind von den Knoten derselben Farbe in **anderen Prozessoren**! Daher eignen sich sogenannte **Blockfärbungen**. Die Idee dabei ist die folgende:

- Verwende möglichst eine zeilenweise Anordnung für die inneren Gitterpunkte eines Prozessors,
 - entkopple Variablen am Rand durch geschicktes Einfärben und
 - stelle sicher, dass von jeder Farbe auf jedem Prozessor etwa gleich viele Knoten vorkommen, um eine gute Verteilung der Last bei der parallelen Ausführung zu erhalten.
- (a) Beschreiben Sie eine Blockfärbung mit m Farben für das Laplace-Problem. Überlegen Sie sich dazu insbesondere, wie viele Farben für die Partition in Quadrate höchstens notwendig sind, um alle oben erwähnten Anforderungen zu erfüllen.
- (b) Entwickeln Sie einen parallelen SOR-Algorithmus für Ihre Blockfärbung.

Installation PETSc (Notwendig für die PETSc-Programmiersession in der letzten Semesterwoche)

Die Installation von PETSc ist vergleichsweise einfach.

Installation von PETSc (MacOS oder Linux):

1. Laden Sie z. B. Version 3.9.x oder 3.10.x unter <https://www.mcs.anl.gov/petsc/download/index.html> herunter. **Nicht die *lite*-Version!**
2. Entpacken Sie das Archiv an einem Ort Ihrer Wahl.
3. Navigieren Sie im Terminal in den PETSc-Ordner.
4. Führen Sie den Befehl
`./configure --with-cc=mpicc --with-cxx=0 --with-fc=0 --download-f2cblaslapack`
 aus, um PETSc zu konfigurieren.
5. Achten Sie darauf, dass Ihr *mpicc*-Wrapper im PATH liegt (wie es auch sein muss, wenn Sie ein MPI Programm übersetzen). Alternativ können Sie auch den Pfad angeben mit: `--with-cc=/Pfad/zu/Ihrem/mpicc`.
6. Setzen Sie den PETSc-Pfad mit `export PETSC_DIR=/Pfad/zu/Ihrem/PETSc/Ordner`
7. Übersetzen Sie PETSc mit `make all`.

Unter Windows: Mit GNU-Compilern (Cygwin, MinGW) sollte es genauso funktionieren wie oben beschrieben. Alternativ empfiehlt sich eine VirtualBox mit einer Linux-Installation (zum Beispiel auf einem USB-Stick). Allerdings muss man dann zunächst erneut C-Compiler und MPI installieren. Eine Anleitung zur Installation von PETSc unter Windows gibt es auch unter <https://www.mcs.anl.gov/petsc/documentation/installation.html#windows>

Kompilieren eines PETSc Programms:

- Um ein PETSc-Programm übersetzen zu können, muss erst der Pfad zu den PETSc-Bibliotheken gesetzt werden durch: `export PETSC_DIR=/Pfad/zum/PETSc/Ordner` wobei der Pfad der bei der PETSc-Installation gewählte Ort ist.
- In dem Ordner, in dem sich das zu übersetzende Programm befindet, muss sich eine Datei namens `makefile` befinden. Eine PETSc Version des berühmten und typischen *Hello World* inklusive der zugehörigen makefile befindet sich auf der Homepage zum download.
- Übersetzen Sie dann einfach durch Eingabe von `make`.
- Führen Sie die nun erstellte und ausführbare Datei wie jedes andere MPI-parallele Programm bisher auch aus!

Abgabedatum: Donnerstag, 04. Juli 2019 bis 12:00 Uhr.
--