MATHEMATISCHES INSTITUT
DER UNIVERSITÄT ZU KÖLN

Dr. Stephanie Friedhoff, Dr. Martin Lanser

Einführung in HPC
Sommersemester 2016

Übung 8

Hinweis: Schreiben Sie bitte jede Aufgabe auf ein neues Blatt und auf **jedes Blatt Ihren Namen**. Auf die erste Seite Ihrer Übung schreiben Sie bitte zusätzlich zu Ihrem Namen Ihre Matrikelnummer.

Aufgabe 1 (2 Punkte).
Zeigen Sie für das SOR-Verfahren zur Lösung von $Ax = b$ die Äquivalenz der Iterationsvorschriften in Matrix-Form und der komponentenweisen Darstellung. D. h. zeigen Sie, dass die Iterationsvorschriften in den Gleichungen (1) und (2) für einen gegebenen Startwert $x^{(0)} \in \mathbb{R}^n$ dasselbe Iterationsverfahren definieren:

$$x^{(k+1)} := Bx^{(k)} + c \tag{1a}$$

mit

$$B = \left(\frac{1}{\omega}D - L\right)^{-1}\left(\frac{1-\omega}{\omega}D + U\right), \; c = \left(\frac{1}{\omega}D - L\right)^{-1} b. \tag{1b}$$

$$x_i^{(k+1)} = \frac{\omega}{a_{ii}}\left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k)}\right) + (1-\omega)\, x_i^{(k)} \tag{2}$$

Aufgabe 2 (1 + 1 + 1 Punkte).
Es sei $G = (V, E)$ ein Graph. Eine Partition

$$V = \bigcup_{k=1}^{m} C_k$$

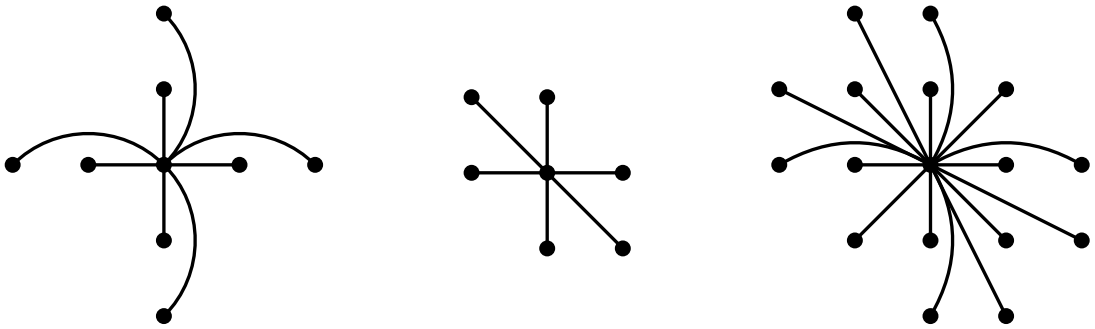
heißt eine **m-farbige Knotenfärbung** von G, falls gilt:

$$i, j \in C_k \text{ und } i \neq j \quad \Rightarrow \quad \{i, j\} \notin E, \quad k = 1, \ldots, m,$$

d. h. Anfangs- und Endknoten aller Kanten sind mit unterschiedlichen Farben eingefärbt.

Bestimmen Sie für die Differenzensterne (a), (b) und (c) auf einem regelmäßigen Gitter und zeilenweiser Nummerierung der Unbekannten eine m-farbige Knotenfärbung, d. h.

(i) skizzieren Sie jeweils für die durch den Differenzenstern definierte Matrix A den zugehörigen Graphen $G(A)$ an den inneren Gitterpunkten für ein Gitter mit 6×6 inneren Gitterpunkten und



(a) (b) (c)

(ii) definieren Sie die entsprechenden Mengen C_k, $k = 1, \ldots, m$.

Bestimmen Sie zudem für die Differenzensterne (a) und (b) die minimiale Anzahl von Farben m und zeigen Sie, dass für den Differenzenstern (c) sechs Farben ausreichen.

Aufgabe 3 (0 Punkte).
In der nächsten Übung werden wir mit Hilfe einiger Folien **OpenMP** (**Open M**ulti **P**rocessing) vorstellen, eine Schnittstelle zum Einsatz auf **shared memory** Systemen. Die Idee der Parallelisierung in OpenMP beruht auf parallelen Threads, die gemeinsam an einer Aufgabe arbeiten und dabei auf **gemeinsamen** Speicher zugreifen können. Das geschieht hauptsächlich durch die Parallelisierungg von langen *for*-Schleifen. Die Threads werden automatisch auf die verfügbaren Prozessoren verteilt. Jedes mit OpenMP parallelisierte Programm startet seriell mit einem **master**-Thread und bei Bedarf werden in vom Programmierer markierten parallelen Regionen, nach dem sog. **Fork-Prinzip** mehrere parallele Threads erstellt. Parallele Regionen werden mit sog. **pragmas** markiert, die der Compiler erkennt.

Um ein OpenMP paralleles Programm zu übersetzen, muss man nur die Bibliothek *omp.h* einbinden und zudem mit der zusätzlichen Compiler-Option *-fopenmp* kompilieren. Betrachten wir ein einfaches Beispiel:

- Übersetzen Sie als Test folgendes hello-world-Programm *openmp_hello.c* mit *gcc -fopenmp openmp_hello.c* und starten es wie gewohnt mit *./a.out*
- Ändern Sie die Laufzeitvariable *OMP_NUM_THREADS* beispielsweise mit *export OMP_NUM_THREADS=5* um die Anzahl der parallelen Threads auf 5 zu setzen und starten Sie das Programm erneut mit *./a.out*

Listing 1: Hello World mit OpenMP

```c
#include <stdio.h>

/* Einzubindende Bibliothek fuer OpenMP*/
#include <omp.h>

int main(int argc, char **argv)
{
  /* Einfache parallele OpenMP Umgebung mit
     OMP_NUM_THREADS vielen paallelen threads*/
    #pragma omp parallel
    {
        printf("\n Hello world!! \n\n");
    }
}
```

Programmieraufgabe 6 (6 Punkte).
Implementieren Sie das parallele Rot-Schwarz-SOR-Verfahren (Algorithmus 4.7 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 $(N+2) \times (N+2)$, $N \in \mathbb{N}$ **gerade** (d. h. mit Schrittweite $h := 1/(N+1)$) mit den Randbedingungen

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

Verwenden Sie eine ein- oder zweidimensionale Partition des Gitters (vgl. Partition in Streifen oder Quadrate in der Programmieraufgabe 5) und ordnen Sie jedem der $p = P^2$ Prozesse P_l, $l = 1,\dots,p$ (eindimensionale Partition) bzw. $P_{l,m}$, $l,j = 1,\dots,P$ (zweidimensionale Partition) ein Teilgebiet zu. Sie dürfen dabei annehmen, dass die Anzahl der Gitterpunkte in jedem Teilgebiet gerade ist, d. h. der linke untere Eckpunkt ist in jedem Teilgebiet mit der gleichen Farbe gefärbt.

Berechnen Sie die Iteration ab, wenn sich zwei aufeinanderfolgende Iterierte in der diskreten L^2-Norm um weniger als eine gegebene Toleranz $\varepsilon > 0$ unterscheiden.

Testen Sie Ihr Programm für $N = 24, 48, 120$, $p = 16$, $\omega = 1$, $f \equiv 0$,

$$g(x,y) = \begin{cases} 1-x & \text{für } y = 0 \\ x & \text{für } y = 1 \\ 1-y & \text{für } x = 0 \\ y & \text{für } x = 1 \end{cases}$$

und den Startvektor $u^{(0)} \equiv 0$. Die exakte Lösung ist die Funktion

$$u(x, y) = 2xy - x - y + 1.$$

Geben Sie am Ende die diskrete L^2-Norm des Fehlervektors

$$e := (u_{i,j})_{i,j=1:N} - (u(ih, jh))_{i,j=1:N} \in \mathbb{R}^{N^2}$$

aus.

Hinweise:

- Die diskrete L^2-Norm für einen Vektor $v \in \mathbb{R}^n$ auf einem regelmäßigen d-dimensionalen Gitter mit Schrittweite h ist definiert durch

$$||v||_2 = \left(h^d \sum_{i=1}^{n} v_i^2 \right)^{1/2}.$$

- Betrachten Sie wie in der Programmieraufgabe 5 für jeden Prozess lokal das zugewiesene Teilgebiet und dessen Rand der Breite 1 („ghost layer“).

Abgabedatum: 27. Juni 2016 bis 12:00 Uhr im entsprechenden Kasten in Raum 3.01 des Mathematischen Instituts oder am Ende der Vorlesung.