MATHEMATISCHES INSTITUT
DER UNIVERSITÄT ZU KÖLN

Dr. Stephanie Friedhoff, Dr. Martin Lanser

Einführung in HPC
Sommersemester 2016

Übung 5

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 (4 Punkte).
Betrachten Sie das lineare Gleichungssystem $Ax = b$, wobei $A \in \mathbb{R}^{n\times n}$ eine Block-Tridiagonal-Matrix der Form

$$A = \begin{pmatrix} D_1^{(0)} & E_1^{(0)} & & & \\ C_2^{(0)} & D_2^{(0)} & E_2^{(0)} & & 0 \\ & \ddots & \ddots & \ddots & \\ & & C_{p-1}^{(0)} & D_{p-1}^{(0)} & E_{p-1}^{(0)} \\ 0 & & & C_p^{(0)} & D_p^{(0)} \end{pmatrix}$$

mit $C_i^{(0)}, D_i^{(0)}, E_i^{(0)} \in \mathbb{R}^{q\times q}$ für alle $i = 1, \ldots, p$ und $n = p \cdot q$, $p, q \in \mathbb{N}$ sei. Ferner sei die rechte Seite $b \in \mathbb{R}^n$ ebenfalls in p Blöcke $B_i^{(0)}$, $i = 1, \ldots, p$ der Länge q zerlegt.

Zeigen Sie, dass Gleichung (3.7) des Abschnitts 3.2.2 der Vorlesung für die Blöcke in dem Verfahren des rekursiven Verdoppelns nach Hockney und Golub gilt, d. h. zeigen Sie folgende Gleichung:

$$\begin{cases} C_i^{(k)} = 0 & \text{falls } i \notin \{1 + 2^k, \ldots, p\} \\ D_i^{(k)} = I & \text{falls } i \notin \{1, \ldots, p\} \\ E_i^{(k)} = 0 & \text{falls } i \notin \{1, \ldots, p - 2^k\} \\ B_i^{(k)} = 0 & \text{falls } i \notin \{1, \ldots, p\}. \end{cases}$$

Aufgabe 2 (6 Punkte).
Es sei $n = p \cdot q$, $p, q \in \mathbb{N}$ und $L \in \mathbb{R}^{n\times n}$ sei eine Block-Bidiagonal-Matrix der Form

$$L = \begin{pmatrix} D_1^{(0)} & & & 0 \\ C_2^{(0)} & D_2^{(0)} & & \\ & \ddots & \ddots & \\ 0 & & C_p^{(0)} & D_p^{(0)} \end{pmatrix}$$

mit $C_i^{(0)}, D_i^{(0)} \in \mathbb{R}^{q \times q}$ für alle $i = 1, \ldots, p$ und $C_1^{(0)} := 0$. Ferner seien $X, B \in \mathbb{R}^n$ ebenfalls in Blöcke $X_i, B_i^{(0)} \in \mathbb{R}^n$, $i = 1, \ldots, p$ zerlegt.

Entwerfen Sie einen parallelen Algorithmus basierend auf der Idee des rekursiven Verdoppelns nach Hockney und Golub zur Lösung des Gleichungssystems $LX = B$, wobei die X_i's direkt berechnet werden sollen sobald alle dazu benötigten Informationen verfügbar sind. Wie viele der X_i werden nach k Schritten ($k = 0, 1, \ldots$) berechnet? **Hinweis:** Sofern die Datenabhängigkeiten in Ihrem Algorithmus den Datenabhängigkeiten in Algorithmus 3.10 der Vorlesung entsprechen, müssen Sie die Kommunikationsoperationen in Ihrem Algorithmus nicht explizit angeben.

Programmieraufgabe 4 (10 + 4 Punkte).
Implementieren Sie Algorithmus 3.8 der Vorlesung mit Zeilen-Pivotsuche sowie impliziter Spaltenvertauschung, d. h. implementieren Sie die parallele Gauß-Elimination mit Zeilen-Pivotsuche zur Berechnung von $A = LU$ in der kij-Form mit **send ahead**. Nehmen Sie an, dass die Matrix zyklisch nach Zeilen verteilt ist und die entsprechenden Zeilen auf jedem Prozess in einem 2D-double-Array abgespeichert sind.

Implementieren Sie die Matrix-Größe n als Übergabeparameter der Hauptfunktion, d. h. der Aufruf Ihrer übersetzten C-Datei (hier: `gauss.o`) sollte z. B. für $n = 8$ die folgende Form haben:

```
mpirun -n 3 ./gauss.o 8
```

Testen Sie Ihre Implementierung mit $p = 3$ MPI-Prozessen mit der Vandermonde-Matrix für den Vektor $v = (1, 2, 3, 4, 5, 6, 7, 8)^T$. Die Vandermonde-Matrix A ist definiert als

$$A = (a_{ij}) \in \mathbb{R}^{n \times n} \quad \text{mit } a_{ij} = v_i^{n-j}.$$

Vergleichen Sie Ihre Ergebnisse mit den Ergebnissen von

```
[L, U, P] = lu(vander(1:8))
```

in MATLAB.

Hinweise:

- Führen Sie die Zeilen-Pivotsuche und anschließende implizite Spaltenvertauschung auf allen Prozessen durch. Nach Erhalt der Zeile $(\hat{a}_{k,k}, ..., \hat{a}_{k,n})$ kann jeder Prozess mit diesen Werten die (auf allen Prozessen identische!) Zeilen-Pivotsuche durchführen und für alle lokalen Zeilen die nötigen Spaltenvertauschungen durchführen. Zusätzliche Kommunikation ist dazu nicht erforderlich.
- Implizite Spaltenvertauschung heißt, die innere Schleife in Algorithmus 3.8 läuft nicht mehr von $k+1$ bis n, sondern von $ind(k+1)$ bis $ind(n)$. Im Indexvektor ind werden dabei die Vertauschungen durchgeführt.

- Beachten Sie, dass die send-ahead-Strategie **nicht-blockierende** Kommunikation erfordert.

Zusatz (Bonus): Implementieren Sie zusätzlich den Algorithmus **ohne** send ahead, d. h. implementieren Sie Algorithmus 3.7 der Vorlesung mit Zeilen-Pivotsuche sowie impliziter Spaltenvertauschung. Messen Sie für beide Fälle (mit und ohne send ahead) die Laufzeit (mittels **MPI_Wtime**) für verschiedene Anzahlen von MPI-Prozesse und verschiedene n. Stellen Sie Ihre Ergebnisse tabellarisch dar. Was fällt Ihnen auf?

Abgabedatum: 06. Juni 2016 bis 12:00 Uhr im entsprechenden Kasten in Raum 3.01 des Mathematischen Instituts oder am Ende der Vorlesung.