Übungen zur Vorlesung
Parallele numerische Algorithmen
Blatt 5

Aufgabe 18

Wir wollen uns in dieser Aufgabe mit unterschiedlichen Datenstrukturen zur Speicherung von von Matrizen und einigen darauf realisierten Operationen beschäftigen. Dazu betrachten wir die folgenden Matrixstrukturen für Matrizen der Dimension $n \times n$:

- Dicht-besetzte Matrizen, in denen alle Elemente $\neq 0$ sind
- Matrizen, in denen nur die Haupt- und Nebendiagonalelemente $\neq 0$ sind (d.h. $A(i,j) \neq 0 \Rightarrow |i-j| \leq 1$)
- Matrizen, in denen nur die Hauptdiagonale und bestimmte Nebendiagonalen Element $\neq 0$ enthalten (d.h. $A(i,j) \neq 0 \Rightarrow |i-j| \in K$ mit $K \subset \{1, \ldots, n\}$)
- Spärlich-besetzte Matrizen, in denen die meisten Elemente 0 sind, die Nichtnullelemente aber zufällig verteilt sind.

1. Überlegen Sie sich, in welchen Anwendungsgebieten welche Matrixform vorkommen könnte.
2. Entwerfen Sie verschiedene Datenstrukturen zur Speicherung der Matrizen und geben Sie die zugehörige Deklaration in C an. Vergleichen den Speicherplatzbedarf der unterschiedlichen Datenstrukturen für die angegebenen Matrixtypen.
3. Realisieren sie die folgenden Operationen für die einzelnen Matrixtypen auf Basis der in der vorherigen Teilaufgabe entwickelten Datenstrukturen. Analysieren Sie für jede Operation den Aufwand.
   (a) Transponieren der Matrix.
   (b) Multiplikation einer Matrix mit einem Vektor.
   (c) Multiplikation zweier Matrizen vom gleichen Typ.

Aufgabe 19

Die in der vorherigen Aufgabe entwickelten Datenstrukturen sollen nun für eine parallele Implementierung verwendet werden. Wir nehmen an, dass ein iteratives Problem der Form $b^{k+1} = A \cdot b^k$ berechnet werden soll, wobei A eine $n \times n$ Matrix und b ein n-dimensionaler Vektor ist. Zur Analyse stehen p homogene Prozessoren zur Verfügung, die über ein Bussystem verbunden sind und einen verteilten Speicher haben. Beschreiben Sie möglichst effiziente Realisierungen der Iteration für die vorgestellten Matrixtypen (der Fall dicht-besetzter Strukturen wurde bereits in der Vorlesung abgearbeitet).

Wenn wir annehmen, dass die Prozessoren als Hyperwürfel strukturiert wären, wie würden sie die Datenverteilung vornehmen, um eine möglichst effiziente Kommunikation zu ermöglichen?