Algorithmen für die Speicherhierarchie

Lineare Algebra: untere Schranken

Riko Jacob

Lehrstuhl für Effiziente Algorithmen
Fakultät für Informatik
Technische Universität München

Vorlesung Sommersemester 2009

Gliederung

1. Modellbildung: Semiring I/O Maschine
2. Volumenschranken
   - Algorithmen: Vollbesetzte Matrix mal Vektor
   - Gradschranke
   - Algorithmus: Vollbesetzte Matrix mal Matrix
   - Untere Schranke: Vollbesetzte Matrix mal Matrix

Zusätzliche Überlegungen

Erinnerung Permutieren

Elemente sind Atome (nur bewegen, kopieren, löschen~~kopieren, löschen~~)

Multiplikation erfordert mehr Modellbildung:

- Zwischenergebnisse nötig
- Struktur der Zahlen kann Aufgabe trivialisieren (0 oder $\mathbb{R}$)

$\Rightarrow$ Semiring I/O-Maschine:

- Nur Addieren und Multiplizieren, kein Minus oder geteilt (inverse Elemente)
- Rechnen nur im Hauptspeicher, wird nicht gezählt
- Speicherplatz in Anzahl von Zahlen
- Indizes und Adressen für

Diskussion Semiring I/O-Maschine

- Alle Zwischenergebnisse haben Form

Matrix-Vektor	x_i,	a_{ij},	$x_j \cdot a_{ij}$,	$\sum_{j \in S} x_j \cdot a_{ij}$
Matrix-Matrix	a_{ij},	b_{ij},	$a_{ik} \cdot b_{kj}$,	$\sum_{k \in S} a_{ik} \cdot b_{kj}$
Skalarprodukt	x_i, y_j,	a_{ij},	$x_i \cdot a_{ij}, a_{ij} \cdot y_j$,	$\sum x_i a_{ij} y_j$

  sind so klassifizierbar
- Verzweigungen im Programm helfen nicht
- Indizes sind Teil des Programms
- Wie “I/O-Pebble Game / independent evaluation” [Hong, Kung 1981]
- Fokus auf Datenfluss (statt Algebra)

Vollbesetzte Matrix mal Vektor

Problem

$$y = A \cdot x\,, \quad y_i = \sum_{k=1..N} a_{ik} \cdot x_k$$

Direkter Algorithmus

Total $O(N^2/B)$ I/Os

Annahme

a_{ik} ohne I/O verfügbar

Algorithm 2: Blockweiser Algorithmus

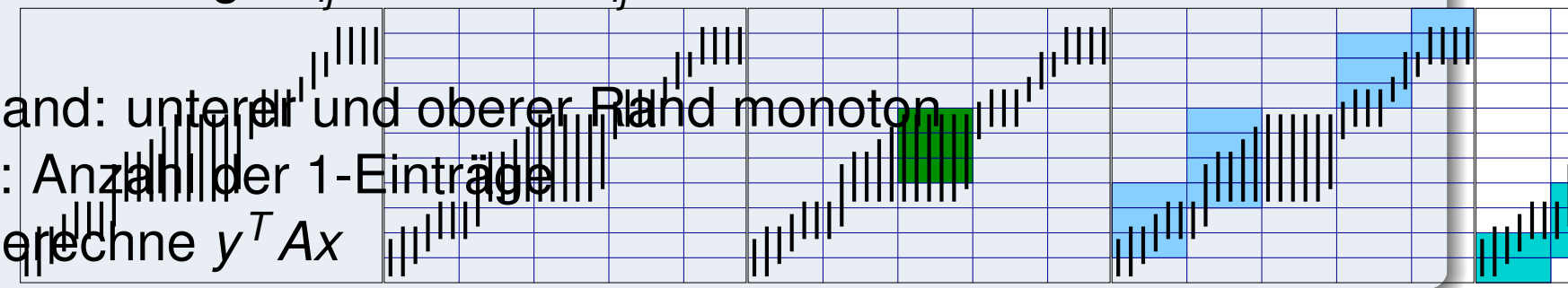
- Teile y in Blöcke der Größe $s = M - B$
- $\frac{N}{s}$ mal x scannen

$$O\left(\frac{N}{M}\frac{N}{B}\right) = O\left(\frac{N^2}{MB}\right) \text{ I/Os}$$

Skalarprodukt mit Bandmatrix

Problem

- Alle Einträge $a_{ij} = 1$ oder $a_{ij} = 0$
- Band: unterer und oberer Rand monoton
- k: Anzahl der 1-Einträge
- Berechne $y^T Ax$



Anwendung

Suche gemessene in synthetisierten Massenspektren

[Roos, Jacob, Grossmann, Fischer, Buhmann, Gruissem, Baginsky, Widmayer, 2007]

Algorithmus

Arbeite in Streifen der Breite $M - B$

$$O\left(\frac{N_x+N_y}{B} + \frac{k}{BM}\right) \text{ I/Os}$$

N_x ist Dimension von x, N_y von y

Gradschranke

Annahme

Alle elementaren Produkte haben Grad $\leq D$

Ein elementares Produkt ist D-Tupel von Eingabevariablen.
Nach einem Input sind B neue Variablen im Speicher, diese erlauben höchstens $M^{D-1}B$ neue Tupel zu bilden.

Beispiel (Bandmatrix Skalarprodukt / Matrix–Vektor)

Alle elementaren Produkte haben Grad 2
Nach einem I/O höchstens MB neue $\Rightarrow \Omega\left(\frac{k}{MB}\right)$ I/Os

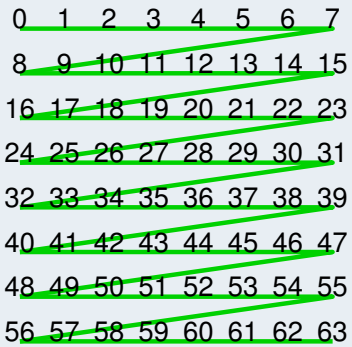
Zusammen mit Scanning-Bound:
vorgestellte Algorithmen sind optimal

Matrix Multiplikation

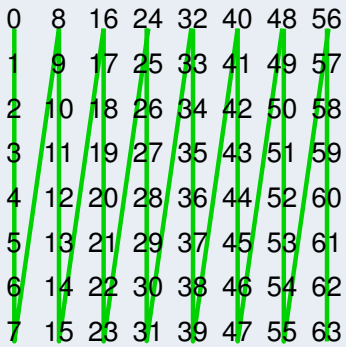
Problem

$$C = A \cdot B\,, \quad c_{ij} = \sum_{k=1..N} a_{ik} \cdot b_{kj}$$

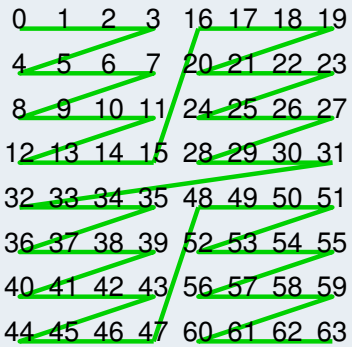
Layout von Matrizen



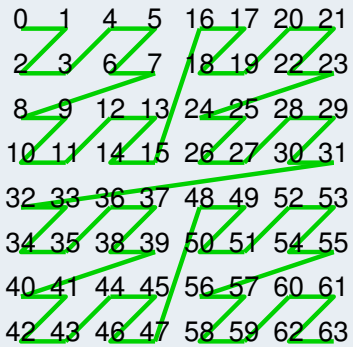
Zeilen



Spalten



4×4-Blöcke



Bit interleaved

Vollbesetzte Matrix mal Matrix

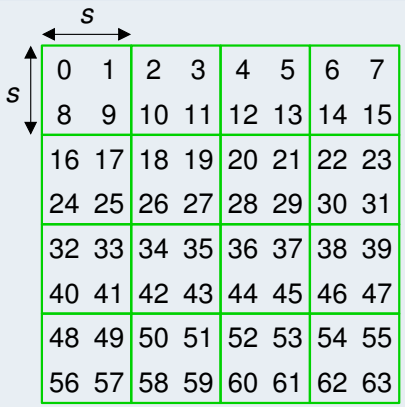
Algorithmus 1: Nested loops

- Zeilen Layout
- Lesen einer Spalte von B
  benötigt N I/Os
- Total $O(N^3)$ I/Os

```
for i = 1 to N
  for j = 1 to N
    c_ij = 0
    for k = 1 to N
      c_ij = c_ij + a_ik · b_kj
```

Algorithm 2: Blockweiser Algorithmus

- Teile A und B in $s \times s$ Blöcke
  mit $s = \Theta(\sqrt{M})$
- Algorithm 1 für die $\frac{N}{s} \times \frac{N}{s}$ Matrizen
  Elemente sind $s \times s$ Matrizen
- $s \times s$-Blöcke oder $\left(\text{Zeilen und } M = \Omega(B^2)\right)$



$$O\left(\left(\frac{N}{s}\right)^3 \cdot \frac{s^2}{B}\right) = O\left(\frac{N^3}{s \cdot B}\right) = O\left(\frac{N^3}{B\sqrt{M}}\right) \text{ I/Os}$$

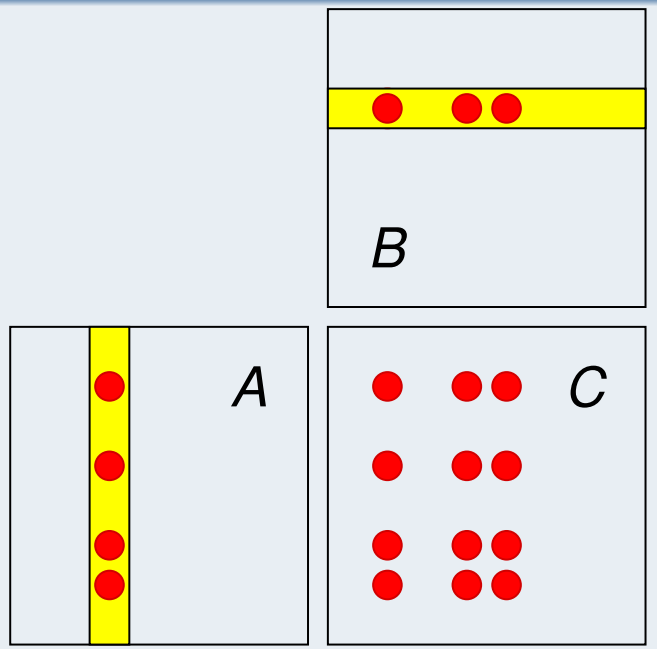
Vollbesetzte Matrix–Matrix

Berechnung in Runden [Hong, Kung 1981]

Für jede (M, B)-I/O Berechnung mit ℓ I/Os gibt es eine $(2M, B)$-I/O Berechnung mit höchstens 3ℓ I/Os, der Gestalt (Speicher leer, laden, speichern, Speicher leer)*

untere Schranke: elementare Produkte pro Runde

- Sei $c :=$ #Ergebnisse (C, Output)
- $a_k :=$ #Elemente in Spalte k von A
- $b_k :=$ #Elemente in Zeile k von B
- Speicher: $c, \sum_k a_k + b_k \leq M$
- Produkte: $\sum_i \min\{a_k \cdot b_k, c\}$
- Maximal für $a_k = b_k = \sqrt{M}, c = M$
  #Produkte $\leq M(1 + \sqrt{M})$



untere Schranke: vollbesetzte Matrix–Matrix

Theorem [Hong, Kung 1981]

Ein Semiring I/O Programm, dass eine vollbesetzte $N_1 \times N_2$ mit einer $N_2 \times N_3$ Matrix multipliziert benötigt

$$\Omega\left(\frac{N_1 N_2 N_3}{B\sqrt{M}}\right)$$

Beweis:

Es gibt $N_1 N_2 N_3$ elementare Produkte, also (vorige Folie) $\Omega\left(\frac{N_1 N_2 N_3}{M\sqrt{M}}\right)$ Runden.
Eine Runde hat M/B I/Os.