Übung 1

Max Wisniewski, Alexander Steen

Aufgabe 1.

Im Folgenden werden zwei Vorhergehensweisen angeben, wie man den Algorithmus von Strassen zur Multiplikation zweier $n \times n$ Matrizen verwenden kann, falls n nicht unbedingt eine Zweierpotenz ist.
Bestimmen Sie die jeweilige Laufzeit einschließlich der Konstante im signifikantesten Term genau und berechnen Sie, für welche n diese Algorithmen weniger Operationen als die klassische Methode benötigen.

(primitive Methode)

Es bezeichne $M_p(n)$ die Anzahl der Operationen bei Multiplikation zweier $n \times n$-Matrizen mit der primitiven Methode. Pro Eintrag in der Ergebnismatrix werden n Multiplikationen und n Additionen benötigt. Es gilt also $M_p(n) = n^2 \cdot 2n = 2n^3$.

(a)

Die Matrizen werden bis zur nächsten Zweierpotenz geeignet aufgefüllt.

Lösung:
Sei $M_1(n)$ die Kosten dieser Methode bei Eingabe einer $n \times n$-Matrix. $2^n \leq m \leq 2^{n+1}$.
$M(n) = 7 * M(n/2) + 18\frac{n}{2} = \frac{23}{5}n^{\log 7} - \frac{18}{5}n$
–> $\frac{23}{5}m^{\log 7} - \frac{18}{5}m = \frac{23 \cdot 2^{\log 7}}{5}n^{\log 7} - \frac{39}{5}n$ BENENNUNG MACHT NOCH KEINEN SINN.
Dieses Verfahren ist bei einer Eingabegröße von $n > 16023$ besser als der naive Ansatz.

(b)

Ist n gerade so führt man einen Rekursionsschritt nach Strassen aus. Andernfalls zerlegt man

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}, \quad B = \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix},$$

wobei A_{11} und B_{11} $(n-1) \times (n-1)$-, A_{12} und B_{12} $(n-1) \times 1$-, A_{21} und B_{21} $1 \times (n-1)$- und A_{22} und B_{22} 1×1- Matrizen sind.
Dann berechnet man AB in der Aufteilung, wie bei der klassischen Multiplikation von 2×2 Matrizen, wobei $A_{11}B_{11}$ rekursiv, die übrigen Produkte klassisch berechnet werden.

Lösung:

tbd

Aufgabe 2.

(a)

Zeigen Sie, dass die Multiplikation von $n \times n$- Matrizen mit $O(I(n))$ Operationen durchführbar ist, falls man mit $I(n)$ Operationen Matrizen invertieren kann.

Lösung:
Zur Multiplikation den Matrizen A, B betrachten wir das folgende Inverse einer $3n \times 3n$-Matrix M:

$$M^{-1} = \begin{pmatrix} 1_n & A & 0_n \\ 0_n & 1_n & B \\ 0_n & 0_n & 1_n \end{pmatrix}^{-1} = \begin{pmatrix} 1_n & -A & AB \\ 0_n & 1_n & -B \\ 0_n & 0_n & 1_n \end{pmatrix} \tag{1}$$

Wir sehen also in Gleichung (1), dass durch Invertierung der Matrix M das Produkt AB in der oberen rechte Ecke enthalten ist. Damit gilt $M(n) = I(3n)$, wobei $M(n)$ die Kosten der Multiplikation darstellt. Eine Abschätzung erhalten wir durch Nutzung der Abschätzung $(*)$: $I(n) = O(n^3)$. Dann gilt:

$$M(n) = I(3n) \overset{(*)}{\leq} 9c \cdot I(n) = O(I(n)) \tag{2}$$

(b)

Zeigen Sie, dass die Multiplikation von $n \times n$- Matrizen mit $O(S(n))$ Operationen durchführtbar ist, falls man mit $S(n)$ Operationen Matrizen quadrieren kann.

Lösung:
Zur Multiplikation den Matrizen A, B betrachten wir die gleiche $3n \times 3n$-Matrix M wie in a):

$$M^{2} = \begin{pmatrix} 1_n & A & 0_n \\ 0_n & 1_n & B \\ 0_n & 0_n & 1_n \end{pmatrix}^{2} = \begin{pmatrix} 1_n & 2A & AB \\ 0_n & 1_n & 2B \\ 0_n & 0_n & 1_n \end{pmatrix} \tag{3}$$

Wieder steht das Produkt AB oben rechts. Es gilt also wieder $M(n) = S(3n)$. Auch für Quadrieren gilt die Abschätzung $S(n) = O(n^3)$, also:

$$M(n) = S(3n) \leq 9c \cdot S(n) = O(S(n)) \tag{4}$$

Aufgabe 3.

Bei der Multiplikation Boolescher Matrizen wird $+$ durch $\vee$ und $\cdot$ durch $\wedge$ ersetzt. Strassens Algorithmus ist nicht direkt anwendbar, da $(\{0,1\}, \vee, \wedge)$ kein Ring ist.
Zeigen Sie, dass die Boolsche Matrizenmultiplikation mit $O(n^{\omega+\varepsilon})$ Operationen aus $\{\vee, \wedge, \neg\}$ für jedes $\varepsilon > 0$ möglich ist, wenn die Matrizenmultiplikation für ganze Zahlen mit $O(n^{\omega})$ arithmetischen Operationen möglich ist.

Lösung:

Eine Boolesche Matrix ist eine Matrix $A = (a_{ij})_{1 \leq i,j \leq n}$ mit $a_{ij} \in \{0,1\}$. Das Boolesche Matrixprodukt zweier Boolescher Matrizen AB ist eine Boolesche Matrix $C = (c_{ij})_{1 \leq i,j \leq n}$ mit $c_{ij} = \bigvee_{k=1}^{n} a_{ik} \wedge b_{kj}$. Um die Multiplikation zweier Boolescher Matrizen A, B zu lösen, definieren wir eine $n \times n$-Hilfsmatrix $H = (h_{ij})_{1 \leq i,j \leq n}$ durch $h_{ij} = \sum_{k=1}^{n} a_{ik} \cdot b_{kj}$. Es gilt nun $0 \leq h_{ij} \leq n$, damit können wir diese Berechnungen in

$\mathbb{Z}_{n+1}$ durchführen. Da $\mathbb{Z}_{n+1}$ ein Ring ist, können wir den Algorithmus für Matrizenmultiplikation für ganze Zahlen (mit $O(n^\omega)$ Operationen) nutzen.
Nun definieren wir die Ergebnismatrix $C = (c_{ij})_{1 \leq i,j \leq n}$ durch

$$c_{ij} = \begin{cases} 0 & \text{, if } h_{ij} = 0 \\ 1 & \text{otherwise} \end{cases}$$

Die hierdurch erhaltene Produktmatrix C ist offensichtlich richtig $\ddot\smile$

Laufzeit: Wie bekannt ist, können wir die Multiplikation von zwei m-stelligen Binärzahlen durch $O(m^2)$ Operationen berechnen. Addition benötigt $O(m)$ Operationen.
Da jede Zahl aus $\mathbb{Z}_{n+1}$ mit maximal $O(\log n)$ Bits dargestellt werden kann, benötigen wir also pro arithmetischer Operation im Algorithmus $O(\log^2 n)$ Operationen. Damit ist die Gesamtlaufzeit $O(n^\omega \cdot \log^2 n) = O(n^{\omega+\varepsilon})$, da $\log^2 n = O(n^\varepsilon)$, für jedes $\varepsilon > 0$.