6. Transitive Hülle

6.1 Min-Plus-Matrix-Produkt und Min-Plus-Transitive Hülle

Ring $\mathbb{Z}(+, \times)$ → Gruppe, Halbgruppe

Semiring $\mathbb{N}(+, \times)$ → Halbgruppe, Halbgruppe

Wir betrachten den (kommutativen) Semiring über $\mathbb{R} \cup \{\infty\}$ mit den Operationen $\min$ und $+$. Für jede der beiden Operationen haben wir ein Monoid. Es gilt das Distributivgesetz $a + \min\{b, c\} = \min\{a + b, a + c\}$.

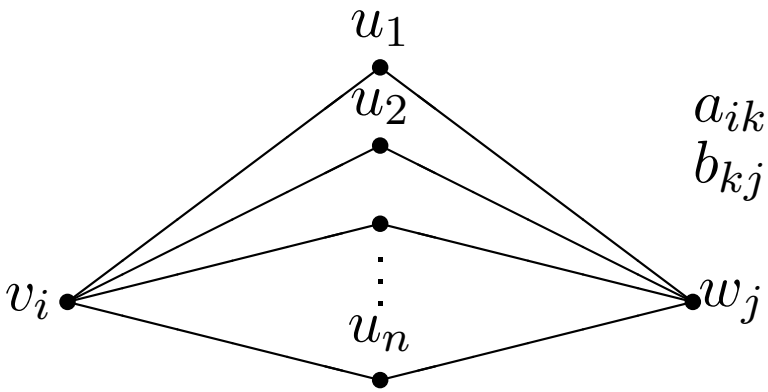
Normale Matrixmultiplikation:

$$A = (a_{ij})_{1 \le i,j \le n}, \quad B = (b_{ij})_{1 \le i,j \le n}, \quad I = (\delta_{ij})_{1 \le i,j \le n}$$

$$C = A \cdot B = (c_{ij})_{1 \le i,j \le n}, \quad c_{ij} = \sum_{k=1}^{n} a_{ik} \cdot b_{kj}$$

Entsprechend für Min-Plus:

$$c_{ij} = \min\{a_{ik} + b_{kj};\ 1 \leq k \leq n\}$$



$$a_{ik} = \mathrm{d}(v_i, u_k)$$
$$b_{kj} = \mathrm{d}(u_k, w_j)$$

Anwendung:
kürzeste Wege von v_i nach w_j in einem Graph ($A = B$); dabei ist

$$I_{\min,+} = \begin{pmatrix} 0 & & \infty \\ & \ddots & \\ \infty & & 0 \end{pmatrix}$$

Sei A Entfernungsmatrix, $A = (a_{ij})_{1 \le i,j \le n} = (\mathrm{d}(v_i, v_j))_{1 \le i,j \le n}$.
Setze $a_{ii} = 0$ für $i = 1, \ldots, n$.

Betrachte A^2 mit dem Min-Plus-Produkt, $A^2 =: (a_{ij}^{(2)})_{1 \le i,j \le n}$.

Dann ist $a_{ij}^{(2)}$ die Länge eines kürzesten Pfades von v_i nach v_j, der höchstens zwei Kanten enthält. Induktion ergibt: $a_{ij}^{(k)}$ ist die Länge eines kürzesten Pfades von v_i nach v_j mit höchstens k Kanten.
Falls die $\mathrm{d}(v_i, v_j)$ alle ≥ 0 sind, gibt es immer kürzeste Pfade, die höchstens $n - 1$ Kanten enthalten.

Damit ergibt sich folgende alternative Lösung des all-pairs-shortest-path-Problems:

Berechne A^{n-1} (Min-Plus)!

Es genügt auch, $A^{2^{\lceil \log(n-1) \rceil}}$ durch wiederholtes Quadrieren zu berechnen (nicht $A^2, A^3, A^4, \ldots$).

Definition 113
$A^* := \min_{i \geq 0}\{A^i\}$ heißt Min-Plus-Transitive Hülle.

Bemerkung: $\min$ wird komponentenweise gebildet. Wenn $\mathrm{d} \geq 0$, dann $A^* = A^{n-1}$.

6.2 Boolesche Matrixmultiplikation und Transitive Hülle

Wir ersetzen nun im vorhergehenden Abschnitt die Distanzmatrix durch die (boolesche) Adjazenzmatrix und $(\min, +)$ durch $(\vee, \wedge)$, d.h.:

$$C = A \cdot B; \quad c_{ij} = \bigvee_{k=1}^{n} a_{ik} \wedge b_{kj}$$

Wenn wir zudem $a_{ii} = 1$ für $1 \leq i \leq n$ setzen, dann gilt für A^k (boolesches Produkt, $A^0 = I$)

$$a_{ij} = \begin{cases} 1 & \text{falls es im Graphen einen Pfad von } v_i \text{ nach } v_j, \\ & \text{bestehend aus } \leq k \text{ Kanten gibt} \\ 0 & \text{falls es im Graphen keinen Pfad von } v_i \text{ nach } v_j, \\ & \text{bestehend aus } \leq k \text{ Kanten gibt} \end{cases}$$

Transitive Hülle:

$$A^* := \bigvee_{i \geq 0} A^i \qquad (= A^{n-1})$$

ist damit die Adjazenzmatrix der transitiven Hülle des zugrunde liegenden Digraphen.

Satz 114

Sei $M(n)$ die Zeitkomplexität für das boolesche Produkt zweier $n \times n$-Matrizen, $T(n)$ die Zeitkomplexität für die transitive Hülle einer $n \times n$ booleschen Matrix.

Falls $\underbrace{T(3n) \leq cT(n)}_{\substack{\text{sicher erfüllt, falls} \\ T \text{ polynomiell}}}$ *und* $\underbrace{M(2n) \geq 4M(n)}_{\substack{\text{sicher erfüllt, falls} \\ M(n) \geq n^2}}$*, dann gilt:*

$$T(n) = \Theta(M(n)) \,.$$

Beweis:

(1) Matrixmultiplikation $\prec$ transitive Hülle:

Seien boolesche Matrizen A, B gegeben und ihr boolesches Produkt $C = A \cdot B$ gesucht.

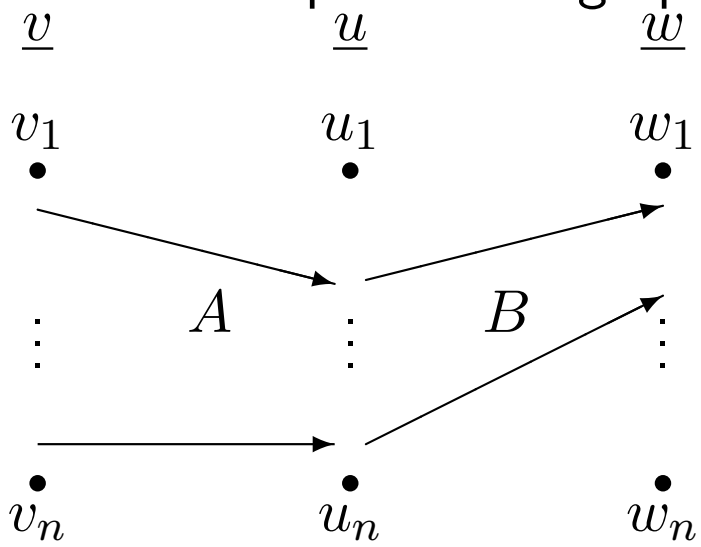
Setze:

$$L = \underbrace{\left.\begin{pmatrix} 0 & A & 0 \\ 0 & 0 & B \\ 0 & 0 & 0 \end{pmatrix}\right\} 3n}_{3n}$$

Beweis (Forts.):

L ist die Adjazenzmatrix eines tripartiten Digraphen, denn:

	$\underline{v}$	$\underline{u}$	$\underline{w}$
$\underline{v}$	0	A	0
$\underline{u}$	0	0	B
$\underline{w}$	0	0	0



Daher kann L^* leicht bestimmt werden:

$$L^* = \begin{pmatrix} I & A & AB \\ 0 & I & B \\ 0 & 0 & I \end{pmatrix} \qquad (= I \vee L \vee L^2)$$

Also gilt: $M(n) \leq T(3n) = \mathcal{O}(T(n))$.

Beweis (Forts.):

(2) Transitive Hülle $\prec$ Matrixmultiplikation:
Gegeben: $n \times n$ boolesche Matrix L; gesucht: L^*; Annahme: n ist Zweierpotenz. Teile auf:

$$L = \underbrace{\begin{pmatrix} A & B \\ C & D \end{pmatrix}}_{\frac{n}{2}\ \frac{n}{2}} \begin{matrix} \} \frac{n}{2} \\ \} \frac{n}{2} \end{matrix} ; \quad L^* = \begin{pmatrix} E & F \\ G & H \end{pmatrix}$$

Es gilt also:

$E =$	$(A \vee BD^*C)^*$	betrachte alle Pfade von der ersten Hälfte der Knoten zur ersten Hälfte der Knoten
$F =$	EBD^*	analog
$G =$	D^*CE	analog
$H =$	$D^* \vee GF$	analog

Beweis (Forts.):

Um L^* zu berechnen, benötigen wir zwei Transitive-Hülle-Berechnungen und sechs Matrixprodukte für Matrizen der Dimension $\frac{n}{2} \times \frac{n}{2}$ (nämlich $M_1 = D^*C$, $M_2 = BM_1$, $M_3 = EB$, $M_4 = M_3D^*$, $M_5 = M_1E$, $M_6 = GF$), plus den Aufwand für $\vee$, der $\leq c'n^2$ ist. Wir zeigen nun durch Induktion ($n = 1\surd$), dass $T(n) \leq cM(n)$:

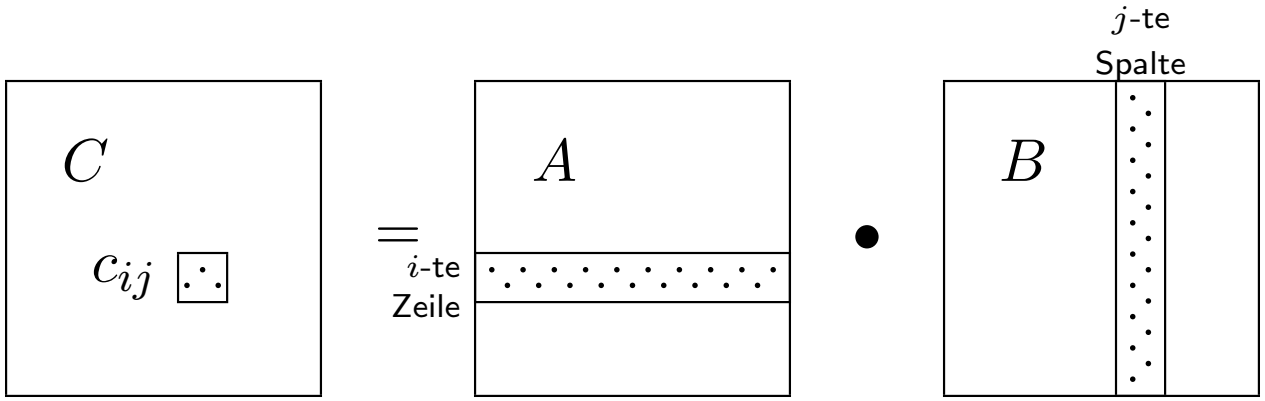
$$
\begin{array}{lll}
T(n) & \leq 2T(\frac{n}{2}) + 6M(\frac{n}{2}) + c'n^2 & \\
& \leq 2cM(\frac{n}{2}) + 6M(\frac{n}{2}) + c'n^2 & \left|\begin{array}{ll}\text{Vor.:} & M(2n) \geq 4M(n) \\ & \text{da } M(n) \geq n^2\end{array}\right. \\
& \leq \frac{1}{4}(2c + 6 + 4c')M(n) & \\
& \leq cM(n) &
\end{array}
$$

falls $c \geq \frac{1}{4}(2c + 6 + 4c')$, also falls $c \geq 3 + 2c'$.

Also $T(n) = \mathcal{O}(M(n))$. □

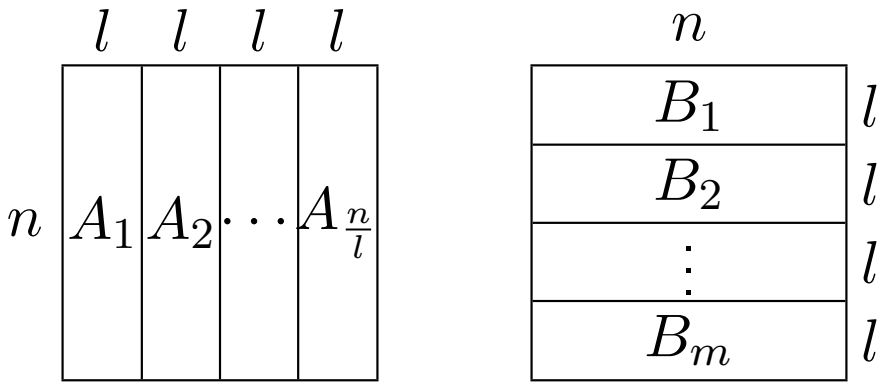
6.3 Der 4-Russen-Algorithmus für boolesche Matrixmultiplikation

Gegeben zwei boolesche $n \times n$ Matrizen A, B; gesucht $C = A \cdot B$.



Sei $l := \lfloor \log n \rfloor$, o.B.d.A. gelte $l | n$ (l teilt n).

Teile A auf (setze $m := \frac{n}{l}$):



Sei $A = A'_1 \vee A'_2 \vee \ldots \vee A'_m$, $B = B'_1 \vee B'_2 \vee \ldots \vee B'_m$,
$C_i := A'_i \cdot B'_i$ für $i = 1, \ldots, m$. Dann gilt

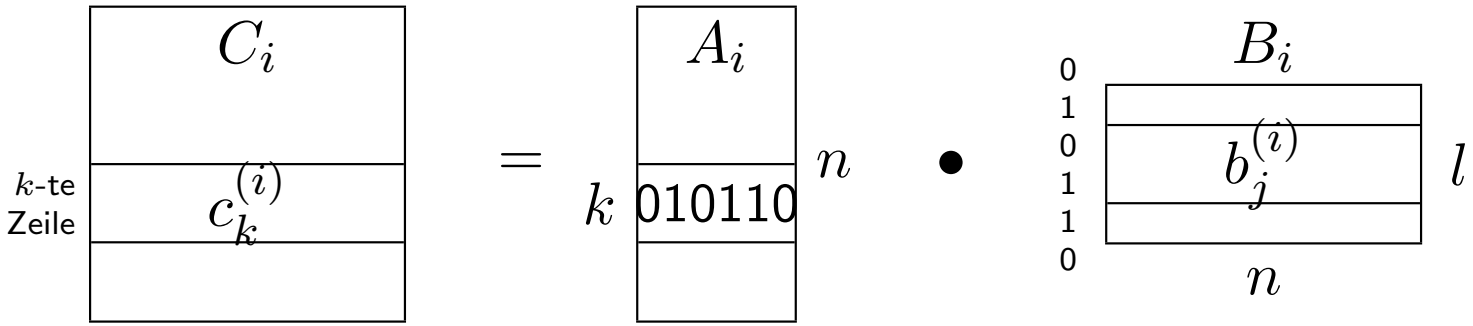
$$C = \bigvee_{i=1}^{m} C_i, \text{ da}$$

$$C = AB = \left(\bigvee_{i=1}^{m} A'_i\right)\left(\bigvee_{i=1}^{m} B'_i\right) = \bigvee_{\substack{1 \le i \le m \\ 1 \le j \le m}} A'_i B'_j = \bigvee_{i=1}^{m} A_i B_i,$$

da $A'_i B'_j = 0$ für $i \neq j$ (A'_i und B'_j sind ja $n \times n$ Matrizen mit 0 außerhalb des jeweiligen Streifens).

Gegeben die C_i's, benötigen wir Zeit $\mathcal{O}(mn^2)$.

Betrachte eine Zeile von C_i:



$$c_k^{(i)} = \bigvee_{j=1}^{l} a_k^{(i)} \cdot b_j^{(i)}$$

Der Algorithmus berechnet einfach zunächst alle booleschen Linearkombinationen der Zeilen von B_i (Prozedur bcomb) und damit $c_k^{(i)}$ für alle überhaupt möglichen $a_k^{(i)}$.
Betrachte A, B und C als Matrizen von Zeilenvektoren:

$$A = \begin{pmatrix} a_1 \\ \vdots \\ a_n \end{pmatrix}, \quad B = \begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix}, \quad C = \begin{pmatrix} c_1 \\ \vdots \\ c_n \end{pmatrix}$$

proc bcomb(**int** i) =
  comb$[0] := [0, \ldots, 0]$
  for $j := 1$ **to** $2^{\lfloor \log n \rfloor} - 1$ **do**
    $p := \lfloor \log j \rfloor$ **co** p Index der vordersten 1 von j **oc**
    comb$[j]$:=comb$[j - 2^p] \vee b_{(i-1)\lfloor \log n \rfloor + 1 + p}$
  od

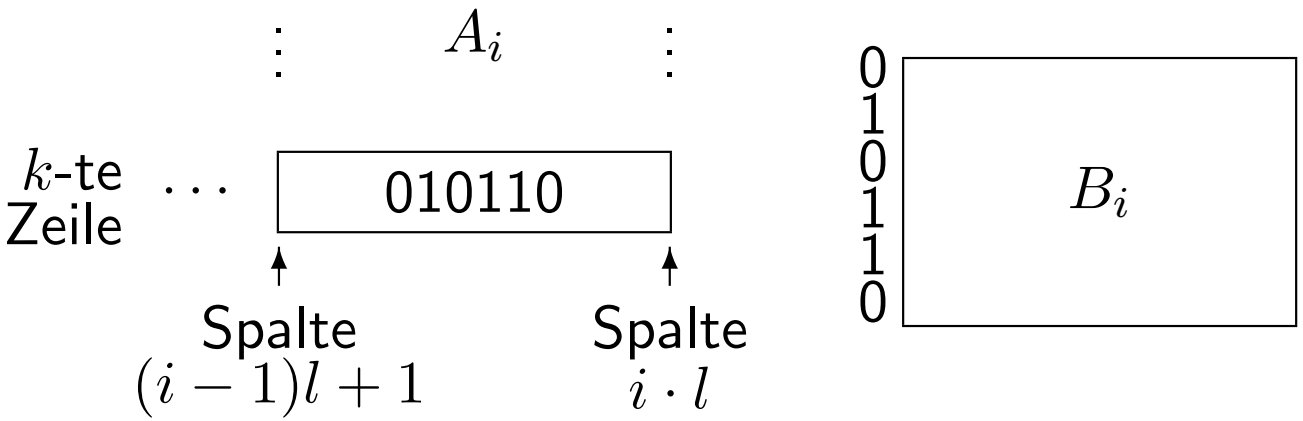
Zeitbedarf:

(a) sequentiell: $\boxed{\mathcal{O}(n^2)}$

(b) Vektoroperationen der Breite n: $\boxed{\mathcal{O}(n)}$

```
algorithm four-russians(array a, b, c) =
  co a, b, c als Vektoren von n Zeilenvektoren organisiert oc
  const l = ⌊log n⌋ co wir nehmen an l|n oc
  array comb[0..2^{l-1}] of boolean-vector; int nc
  for i := 1 to n do c[i] := [0, ..., 0] od
  for i := 1 to n/l do    co berechne die C_i's oc
      bcomb(i)
      for j := 1 to n do
          co Bitmuster in Binärzahl wandeln oc
          nc := 0
          for k := i · l downto (i − 1) · l + 1 do
              nc := nc + nc+ if a[j, k] then 1 else 0 fi
          od
          c[j] := c[j] ∨ comb[nc]
      od
  od
```

Beispiel 115



Zeitbedarf:

(a) sequentiell:

$$\mathcal{O}\left(\frac{n}{l}\cdot\left(n^2+n(l+n)\right)\right)=\mathcal{O}\left(\frac{n^3}{l}\right)=\boxed{\mathcal{O}\left(\frac{n^3}{\log n}\right)}$$

(b) Vektorrechner der Breite n (Interpretation eines Bitintervalls als Zahl in $\mathcal{O}(1)$ Zeit):

$$\mathcal{O}\left(\frac{n}{l}\cdot\left(n+n(1+1)\right)\right)=\boxed{\mathcal{O}\left(\frac{n^2}{\log n}\right)\text{ (Vektoroperationen)}}$$

Satz 116

Der 4-Russen-Algorithmus berechnet das Produkt zweier boolescher Matrizen sequentiell in Zeit $\mathcal{O}\left(\frac{n^3}{\log n}\right)$ *bzw. mit* $\mathcal{O}\left(\frac{n^2}{\log n}\right)$ *Bitvektoroperationen der Breite* n*.*

Beweis:

s.o. □


V.L. Arlazarov, E.A. Dinic, M.A. Kronrod, I.A. Faradzev:
On economical construction of the transitive closure of an oriented graph
Soviet Math. Dokl. **11**, pp. 1209–1210 (1970)