Tranzitivní uzávěr acyklického grafu (Simon)

Nechť $G=(V,E)$, kde $V=\{1,2,\dots,n\}$, $E\subseteq V\times V$ je orientovaný graf.
Cyklus v grafu je cesta $v_0, v_1, \dots, v_s$, taková, že $s\ge 1$ a $v_0 = v_s$.
Graf, který neobsahuje cykly se nazývá acyklický.
Topologické uspořádání je zobrazení $ord: V \to \{1,2,\dots,|V|\}$ takové, že pro všechny hrany $(v,w)\in E$ platí $ord(v) < ord(w)$.
(tj. vede-li šipka z v do w, pak v musí předcházet w v tomto uspořádání.) Platí:

Věta: $G=(V,E)$ je acyklický, právě když má topologické uspořádání.
Topologické uspořádání acyklického grafu se dá spočítat v čase $O(|V|+|E|)$.
(bez důkazu)

Budeme nadále předpokládat, že máme acyklický graf $G=(V,E)$ topologicky uspořádaný, tj. $(v,w)\in E$ implikuje, že $v<w$, a že seznamy následníků $out(v)=\{w\in V;\ (v,w)\in E\}$ jsou uspořádány rostoucím způsobem. (Uspořádání se dá udělat v čase $O(|V|+|E|)$.
Označíme:

$out^*(v)$ reflexivní a tranzitivní uzávěr vrcholu v, tj.

$out^*(v)=\{w;$ existuje cesta v G z vrcholu v do $w\}\cup\{v\}$

$out^{red}(v)$ tranzitivní redukt vrcholu v, tj.

$out^{red}(v)=\{w\in out(v);$ neexistuje cesta v G délky ≥ 2 z v do $w\}$

$E^*=\{(v,w);\ w\in out^*(v)\}$, $G^*=(V,E^*)$ je tranzitivní uzávěr G

$E^{red}=\{(v,w),\ w\in out^{red}(v)\}$, $G^{red}=(V,E^{red})$ je tranzitivní redukt G

Topologické uspořádání grafu (Kahn 1962, Knuth 1968)
(Mehlhorn II)

Jednoduchý algoritmus:

$G_{current} \leftarrow G$, COUNT $\leftarrow 0$

while $G_{current}$ má aspoň 1 vrchol, do kterého nevede žádná hrana

do $v \leftarrow$ vrchol, do kt. nevede hrana

COUNT $\leftarrow$ COUNT + 1

ORD(v) $\leftarrow$ COUNT

$G_{current} \leftarrow G_{current} - v$

od

if $G_{current} \neq 0$ **then** G není acyklický

else G je acyklický

fi

čas: Nalezení vrcholu, do kterého nevede hrana ... n

počet průchodů cyklem ... n

celkem .. $\Omega(n^2)$

Dá se to udělat lépe v čase $O(|V|+|E|)$:

pro každý vrchol spočteme počet hran, které do něj vchází (při umazání vrcholu se tato čísla budou snižovat)

udržujeme seznam vrcholů, do kterých nevedou hrany

```
COUNT ← 0
ZEROINDEG ← ∅
for all i∈V do INDEG(i) ← 0 od
for all i∈V do
    for all j∈V with (i,j)∈E do INDEG(j) ← INDEG(j)+1 od
od
for all i∈V do
    if INDEG(i) = 0 then add i to ZEROINDEG fi
od
while ZEROINDEG ≠ ∅ do
    v ← libovolný vrchol z ZEROINDEG
    delete v z ZEROINDEG
    COUNT ← COUNT+1 ; ORD(v) ← COUNT
    for all w∈V with (v,w)∈E do
        INDEG(w) ← INDEG(w)-1
        if INDEG(w)=0 then add w to ZEROINDEG fi
    od
od
if COUNT < n then G cyklický
             else G acyklický fi
```

$e = |E|$, $e^* = |E^*|$, $e^{red} = |E^{red}|$, $n = |V|$,

$j(v) = |out(v)|$, $j^*(v) = |out^*(v)|$, $j^{red}(v) = |out^{red}(v)|$

Platí: když $out(v) = \{w_1 < w_2 < \dots < w_s\}$

pak $out^*(v) = \{v\} \cup out^*(w_1) \cup out^*(w_2) \cup \dots \cup out^*(w_s)$

$w_i \notin out^{red}(v) \Leftrightarrow w_i \in out^*(w_1) \cup \dots \cup out^*(w_{i-1})$

$\Leftrightarrow out^*(w_i) \subseteq out^*(w_1) \cup \dots out^*(w_{i-1})$

(Když w_i není v redukci, pak ex. cesta z v do w_i délky alespoň 2, a tudíž w_i musí být v tranzitivním uzávěru některého z následníků v, ale protože hrany vedou jenom z vrcholů s nižšími čísly do vyšších, musí to být jeden z vrcholů $w_1, \dots, w_{i-1}$.)

negací dostaneme: $out^{red}(v) = \{w_i \in out(v); w_i \notin out^*(w_1) \cup \dots \cup out^*(w_{i-1})\}$

dále platí $\bigcup_{w \in out(v)} out^*(w) = \bigcup_{x \in out^{red}(v)} out^*(x)$

a $G^* = G^*_{red}$

Algoritmus pro výpočet tranzitivního uzávěru a redukce:

pro každý vrchol $v \in V$ sestrojíme $out^*(v)$ a $out^{red}(v)$

Algoritmus: (Goralčíková, Koubek 1979)

vstup: $G=(V,E)$

výstup: $out^*(v)$ a $out^{red}(v)$ $\forall v \in V$

```
for v ← n downto 1 do
    out*(v) ← {v}
    out^red(v) ← ∅
    for ∀ w ∈ out(v) do        {v rostoucím pořadí}
        if w ∉ out*(v) then
            out*(v) ← out*(v) ∪ out*(w)
            out^red(v) ← out^red(v) ∪ {w}
        fi
    od
od  ←  převod char. vektorů na lin. seznamy
```

Všechny operace až na $out^*(v) \cup out^*(w)$ se dají udělat v konstantním čase (seznam $out^*(v)$ můžeme reprezentovat uvnitř cyklu jako charakteristický vektor). Operace $out^*(v) \cup out^*(w)$ potřebují čas $O(|out^*(w)|) = O(n)$. Provádí se v cyklu přes $w \in out(v)$ a zároveň $w \notin out^*(v)$, tedy přes hrany v redukci.

Proto čas je $O(n \cdot e^{red})$.

Počet hran v redukci je v nejhorším případě kvadratický, v průměrném případě $n \log n$ (to spočítáme později).

Průměrná časová složitost algoritmu tedy bude $O(n^2 \log n)$.

Nejdříve trochu jednodušší a hrubší analýza: (Mehlhorn, Simon 1983)

nechť p je pravděpodobnost hrany v G

def. $X_j(G) = \begin{cases} 1 & \text{když } (i,j) \in E_{red} \\ 0 & \text{jinak} \end{cases}$

pak $\sum_j X_j(G) = |out^{red}(i)|$

$$E\left(\sum_j X_j\right) = \sum_j EX_j = \sum_j \underbrace{P((i,j) \in E_{red})}$$

tuhle pravděpodobnost musíme spočítat

$$P((i,j) \in E_{red}) \leq P((i,j) \in E \cap \forall h: i < h < j \text{ buď } (i,h) \notin E \text{ nebo } (h,j) \notin E)$$

$$= \underbrace{P((i,j) \in E)}_{p} * \prod_h \Big[\underbrace{P((i,h) \notin E)}_{1-p} + \underbrace{P((h,j) \notin E)}_{1-p} - \underbrace{P((i,h) \notin E \cap (h,j) \notin E)}_{(1-p)^2}\Big] =$$

$$= p\,(1-p^2)^{j-i-1}$$

$$\sum_j p(1-p^2)^{j-i-1} = \sum_{j=i+1}^{n} p(1-p^2)^{j-i-1} \leq \sum_{j=0}^{n-2} p(1-p^2)^j =$$

$$= p \cdot \frac{1-(1-p^2)^{n-1}}{1-(1-p^2)} = \frac{1-(1-p^2)^{n-1}}{p}$$

platí: $\dfrac{1-(1-p^2)^{n-1}}{p} \leq \sqrt{n}$

dk: nechť $p > \frac{1}{\sqrt{n}}$

pak $\dfrac{1-(1-p^2)^{n-1}}{p} < \dfrac{1}{p} < \sqrt{n}$

nechť $p < \frac{1}{\sqrt{n}}$

$$1-(1-p^2)^{n-1} = 1-\sum_{k=0}^{n-1}\binom{n-1}{k}p^{2k}(-1)^k$$

$$= \underbrace{1-1}_{0} + \underbrace{(n-1)p^2}_{< np^2} \underbrace{- \binom{n-1}{2}p^4 + \binom{n-1}{3}p^6}_{\le 0} \underbrace{- \dots}_{< 0} \le$$

$$\le np^2$$

$$\binom{n-1}{3}p^6 - \binom{n-1}{2}p^4 = p^4\left(\frac{(n-1)(n-2)(n-3)}{3\cdot 2}p^2 - \frac{(n-1)(n-2)}{2}\right) =$$

$$= p^4\binom{n-1}{2}\left(\frac{n-3}{3}p^2 - 1\right) < \frac{1}{n^2}\binom{n-1}{2}\left(\frac{n-3}{3n} - 1\right) =$$

$$= \frac{1}{n^2}\binom{n-1}{2}\frac{-2n-3}{3n} < 0$$

podobně pro další dvojice po ~~souk~~ sobě jdoucích členů

$$\frac{1-(1-p^2)^{n-1}}{p} \le np < \frac{n}{\sqrt{n}} = \sqrt{n}$$

$E\,|out^{red}(i)| \le \sqrt{n}$ pro vš. i

$E\,|E_{red}| \le n\sqrt{n} = n^{3/2}$

očekávaný čas celého algoritmu: $n \cdot n^{3/2} = n^{5/2}$

%

$$-\binom{n-1}{k}p^{2k}+\binom{n-1}{k+1}p^{2(k+1)}=$$

$$=p^{2k}\left(\frac{(n-1)(n-2)\cdots(n-k-1)}{(k+1)!}p^2-\frac{(n-1)(n-2)\cdots(n-k)}{k!}\right)=$$

$$=p^{2k}\binom{n-1}{k}\left(\frac{n-k-1}{k+1}p^2-1\right)<\frac{1}{n^k}\binom{n-1}{k}\left(\frac{n-k-1}{(k+1)n}-1\right)=$$

$$=\frac{1}{n^k}\binom{n-1}{k}\frac{-kn-k-1}{(k+1)n}<0$$

12649 Svépomoc II/91-100 – TZ 13-322-91-G

K analýze průměrného případu použijeme model náhodného acyklického grafu s vrcholy $\{1, 2, \dots, n\}$, kde možné hrany se vyskytují nezávisle s pravděpodobností p, tj. pro každé $1 \le i < j \le n$ platí

$$P\{(i,j) \in E\} = p, \quad P\{(i,j) \notin E\} = 1-p.$$

Postup: (Simon 1988)

1) vypočteme střední hodnotu veličiny $\gamma_n^* = |\,out^*(1)(G_{n,p})|$ (resp. její odhad)

2) odvodíme a použijeme vztah

$$E(\gamma_n^{red}) = \frac{p}{1-p}\left(n - E(\gamma_n^*)\right)$$

kde $\gamma_n^{red} = |\,out^{red}(1)\,(G_{n,p})|$

3) použijeme 2) k odhadu střední hodnoty e^{red}

$$E e^{red} \le n\, E\gamma_n^{red}$$

ad 1) nechť $\xi_n^* = |out^*(1)|$

Pro každé $n = 1, 2, \dots$ ξ_n^* tvoří Markovský řetězec s diskrétním časem, tzv. čistý proces množení, protože přidáním dalšího vrcholu grafu se může pouze zvětšit nebo zůstat na stejné hodnotě.

Markovský řetězec s diskrét. časem je posloupnost celočíselných náh. veličin $X_1, X_2, \dots$, kde

$$P(X_n = j \mid X_{n-1} = i, X_{n-2} = i_{n-2}, \dots, X_1 = i_1) =$$
$$= P(X_n = j \mid X_{n-1} = i) = p_{ij}$$

je pravděpodobnost přechodu ze stavu i do stavu j.

Pro nás:

$$P(\xi_n^* = \ell \mid \xi_{n-1}^* = \ell-1) = P(\text{ex. } w \in out_{n-1}^*(1) : (w, n) \in E)$$
$$= 1 - P(\text{pro vš. } w \in out_{n-1}^*(1) : (w, n) \notin E) =$$
$$= 1 - (1-p)^{\ell-1}$$

$$P(\xi_n^* = \ell-1 \mid \xi_{n-1}^* = \ell-1) = (1-p)^{\ell-1}$$

ostatní přechody mají nulovou pravděpodobnost

pro absolutní (nepodmíněné) pravděpodobnosti platí

$$P(X_n = j) = \sum_i P(X_{n-1} = i)\, p_{ij}$$

pro nás

$$P(\xi_n^* = \ell) = P(\xi_n^* = \ell \mid \xi_{n-1}^* = \ell-1)\, P(\xi_{n-1}^* = \ell-1) +$$
$$+ P(\xi_n^* = \ell \mid \xi_{n-1}^* = \ell)\, P(\xi_{n-1}^* = \ell)$$

$$P(\xi_n^* = \ell) = \left(1-(1-p)^{\ell-1}\right) P(\xi_{n-1}^* = \ell-1) +$$
$$+ (1-p)^{\ell} P(\xi_{n-1}^* = \ell) \qquad \text{pro } n > 1$$

$P(\xi_1^* = 1) = 1$ je počáteční podmínka

označme $\pi_\ell = 1-(1-p)^\ell$

a $P_{n,\ell} = P(\xi_n^* = \ell)$

pak máme

$$P_{1,1} = 1$$
$$P_{n,\ell} = (1-\pi_\ell) P_{n-1,\ell} + \pi_{\ell-1} P_{n-1,\ell-1}$$

Platí: Nechť φ je reálná funkce definovaná jako $\varphi(\ell) = \sum_{j=1}^{\ell-1} \frac{1}{\pi_j}$. Pak pro všechna n je $E(\varphi(\xi_n^*)) = n-1$.

Důkaz indukcí:

$$E(\varphi(\xi_1^*)) = \sum_{\ell=1}^{1} \varphi(\ell) P_{1,\ell} = 0 \cdot 1 = 0$$

$$E(\varphi(\xi_n^*)) = \sum_{\ell=1}^{n} \varphi(\ell) P_{n\ell} =$$

$$= \sum_{\ell=1}^{n} \varphi(\ell)\left((1-\pi_\ell) P_{n-1,\ell} + \pi_{\ell-1} P_{n-1,\ell-1}\right) =$$

$$= \sum_{\ell=1}^{n} \varphi(\ell) P_{n-1,\ell} - \sum_{\ell=1}^{n} \varphi(\ell) \pi_\ell P_{n-1,\ell} + \sum_{\ell=1}^{n} \varphi(\ell) \pi_{\ell-1} P_{n-1,\ell-1}$$

$$= \underbrace{E\left(\varphi(\xi^*_{n-1})\right)}_{\substack{n-2 \\ \text{podle indukč. předp.} \\ (P_{n-1,n}=0)}} - \underbrace{\sum_{\ell=1}^{n} \varphi(\ell)\pi_\ell P_{n-1,\ell}}_{\substack{\text{da' se sčítat} \\ \text{do } n-1 \\ \text{protože } P_{n-1,n}=0}} + \underbrace{\sum_{\ell=0}^{n-1} \varphi(\ell+1)\pi_\ell P_{n-1,\ell}}_{\substack{\text{da' se sčítat od 1} \\ \text{protože } P_{n-1,0}=0}}$$

ještě dosadíme $\varphi(\ell+1) = \varphi(\ell) + \frac{1}{\pi_\ell}$

máme $E(\varphi(\xi^*_n)) = n-2 + \sum_{\ell=1}^{n-1} \underbrace{\left(\varphi(\ell)\pi_\ell P_{n-1,\ell} - \varphi(\ell)\pi_\ell P_{n-1,\ell}\right)}_{0} +$

$$+ \underbrace{\sum_{\ell=1}^{n-1} \frac{1}{\pi_\ell} \cdot \pi_\ell P_{n-1,\ell}}_{1} = n-2+1 = n-1$$

dále platí:

$$\varphi(\ell) = \sum_{j=1}^{\ell-1} \frac{1}{\pi_j} = \frac{1}{p} + \sum_{j=2}^{\ell-1} \frac{1}{1-(1-p)^j} \leq$$

→ klesající (neroստoucí) funkce j

$$\leq \frac{1}{p} + \sum_{j=2}^{\ell-1} \int_{j-1}^{j} \frac{dx}{1-(1-p)^x} = \frac{1}{p} + \int_1^{\ell-1} \frac{dx}{1-(1-p)^x} =$$

$$= \frac{1}{p} + (\ell-1) - \underbrace{\frac{\log(1-(1-p)^{\ell-1})}{\log(1-p)}}_{\text{zřejmě } \geq 0} - 1 + \frac{\log(1-(1-p))}{\log(1-p)} \leq$$

$$\leq \ell - 2 + \frac{\log(1-p) + p\log p}{p\log(1-p)} \leq \ell - 2 + \underbrace{\frac{1+|\log p|}{p}}_{\Phi(\ell)}$$

%

%.

$$\frac{1}{1-(1-p)^j} \quad \text{je nerostoucí podle } j$$

$$\int_{j-1}^{j} \frac{1}{1-(1-p)^x}\,dx \geq \frac{1}{1-(1-p)^j}\,\underbrace{\big(j-(j-1)\big)}_{1}$$

$$\log(1-p) = -p - \frac{p^2}{2} - \frac{p^3}{3} - \dots \qquad \text{Taylorův rozvoj}$$

$$\frac{p}{\log(1-p)} = -\frac{p}{p+\frac{p^2}{2}+\frac{p^3}{3}+\dots} = -\frac{1}{\underbrace{1+\frac{p}{2}+\frac{p^3}{2}+\dots}_{\geq 1}}$$

$$\int \frac{dx}{1-(1-p)^x} = x - \frac{\lg\left(1-(1-p)^x\right)}{\lg(1-p)}$$

(důkaz zderivováním)

nakonec:

$$n-1 = E\left(\varphi(\xi_n^*)\right) = \sum_{l=1}^{n} \varphi(l) P_{n,l} \leq \sum_{l=1}^{n} \phi(l) P_{nl} = E\phi(\xi_n^*) =$$

$$= E\xi_n^* - 2 + \frac{1+|\log p|}{p}$$

a tedy $E\xi_n^* \geqslant n+1 - \dfrac{1+|\log p|}{p}$

ad 2) Lemma: $E(\mathcal{J}_n^{red}) = \frac{p}{1-p}\left(n - E(\mathcal{J}_n^{*})\right)$

Dk: $P\{(1,n)\in E^*\} = P\{\text{ex. } w \in out^*_{n-1}(1) \text{ tak, že } (w,n)\in E\}$

dále:

$$P\{(1,n)\in E^*\} = \sum_{\ell=1}^{n-1} P\{(1,n)\in E^* \mid \mathcal{J}^*_{n-1} = \ell\} P\{\mathcal{J}^*_{n-1} = \ell\} =$$

(věta o celkové psti)

$$= \sum_{\ell=1}^{n-1} P\{\mathcal{J}^*_n = \ell+1 \mid \mathcal{J}^*_{n-1} = \ell\} P(\mathcal{J}^*_{n-1} = \ell) =$$

$$= \sum_{\ell=1}^{n-1} \left[1-(1-p)^{\ell}\right] P(\mathcal{J}^*_{n-1} = \ell)$$

nechť $n \geq 2$:

$$P\{(1,n)\in E_{red}\} = P\{(1,n)\in E\} P\{\forall w \in out^*_{n-1}(1),\ w\neq 1 : (w,n)\notin E\} =$$

(nezávislé jevy)

$$= p \sum_{\ell=1}^{n-1} \underbrace{(1-p)^{\ell-1}}_{P\{\mathcal{J}^*_n = \ell \mid \mathcal{J}^*_{n-1} = \ell\}} P(\mathcal{J}^*_{n-1} = \ell) \quad \text{(podle věty o celkové psti)}$$

$$= \frac{p}{1-p} \sum_{\ell=1}^{n-1} \left[1 - 1 + (1-p)^{\ell}\right] P(\mathcal{J}^*_{n-1} = \ell) =$$

$$= \frac{p}{1-p}\left[1 - \underbrace{\sum_{\ell=1}^{n-1}\left[1-(1-p)^{\ell}\right] P(\mathcal{J}^*_{n-1} = \ell)}_{P\{(1,n)\in E^*\}}\right]$$

takže

$$P\{(1,n)\in E_{red}\} = \frac{p}{1-p}\left[1 - P\{(1,n)\in E^*\}\right]$$

to platí nejen pro n, ale i pro všechna $j = 2,\dots,n$

def. náh. veličiny $X_j = \begin{cases} 1 & \text{když } (1,j) \in E_{red} \\ 0 & \text{když } (1,j) \notin E_{red} \end{cases}$

pak $j_n^{red} = \sum_{j=2}^{n} X_j$

$$E(j_n^{red}) = \sum_{j=2}^{n} EX_j = \sum_{j=2}^{n} 1 \cdot P\{(1,j) \in E_{red}\} =$$

$$\sum_{j=2}^{n} \frac{p}{1-p}\left[1 - P\{(1,j) \in E^*\}\right] = \frac{p}{1-p}\left[n-1 - \sum_{j=2}^{n} P\{(1,j) \in E^*\}\right] =$$

$$= \frac{p}{1-p}\Big[n - \underbrace{\sum_{j=1}^{n} 1 \cdot P\{(1,j) \in E^*\}}_{E(j_n^*)}\Big]$$

závěr: $E(j_n^{red}) = \dfrac{p}{1-p}\left(n - E(j_n^*)\right)$

3) __Důsledek:__ $E(\ell_{red}) \leq n(|\log p| + 2)$.

__Dk:__ platí $E(\ell_{red}) = E(j^{red}(1)) + \ldots + E(j^{red}(n)) \leq$

$$\leq n\, E(j^{red}(1)) = n\, E(j_n^{red}) = n \frac{p}{1-p}\left(n - E(j_n^*)\right)$$

odhadneme $E(j_n^*) \geq n+1 - \dfrac{|\log p|+1}{p}$

takže

$$E(\ell_{red}) \leq n \frac{p}{1-p}\left[n - \left(n+1 - \frac{|\log p|+1}{p}\right)\right] = n\left[\frac{|\log p|}{1-p} + 1\right] \leq$$

$$\overset{\text{Taylor rer.}}{\leq} n\left[\frac{(1-p) + \frac{1}{2}(1-p)^2 + \frac{1}{3}(1-p)^3 + \ldots}{1-p} + 1\right] \leq n(|\log p| + 2).$$

$|\log p| = |\log(1-(1-p))|$

<u>Důsledek</u>: $E(l_{red}) \leq O(n \log n)$

<u>Dk</u>: a) nechť $p > \frac{\log n}{n}$

pak $|\log p| < \left|\log \frac{\log n}{n}\right| = |\log \log n - \log n| \leq \log n$

podle předchozího: $E(l_{red}) \leq n \cdot (|\log p| + 2) \leq$

$\leq n(\log n + 2) \leq O(n \log n)$

b) $p \leq \frac{\log n}{n}$

$l_{red} \leq l \Rightarrow E(l_{red}) \leq E(l) \leq p n^2 \leq n \log n$

<u>Věta</u>: Původní algoritmus pro výpočet $ecl^*(v)$ a $ecl^{red}(v)$ pro vš. $v \in V$ vyžaduje průměrný čas $O(n^2 \log n)$.

Poznámka: případ b) v předchozím důkaze je odhadnutý dost hrubě a je možné, že odhad pro tyto hodnoty p by mohl být řádově lepší

což se také později potvrdilo a bylo dokázáno, že tento algoritmus má pro malé hodnoty p očekávanou složitost $O(n^2)$

Další rezerva je v tom, že jsme $E(y_n^{red})$ pouze odhadli shora, zatímco se dá spočítat přesně (Simon 1993)

Vrátíme se k rekurentnímu vztahu

$$P_{nl} = (1-r_l)P_{n-1,l} + r_l P_{n-1,l-1}, \quad P_{11} = 1$$

tj:

$$P_{nl} = (1-p)^l P_{n-1,l} + \left(1-(1-p)^{l-1}\right) P_{n-1,l-1}$$

Indukcí se dá dokázat, že výsledkem je

$$P_{nl} = (1-p)^{n-l} \prod_{i=1}^{l-1} \left(1-(1-p)^{n-i}\right).$$

Dk:

$$P_{11} = 1 \cdot \underbrace{\prod_{i=1}^{0} \left(1-(1-p)^{1-i}\right)}_{=1} = 1 \quad \text{platí}$$

$$P_{nl} = (1-p)^l P_{n-1,l} + \left(1-(1-p)^{l-1}\right) P_{n-1,l-1} \overset{I.P.}{=}$$

$$= \underbrace{(1-p)^l (1-p)^{n-1-l}}_{(1-p)^{n-l}(1-p)^{l-1}} \prod_{i=1}^{l-1} \left(1-(1-p)^{n-1-i}\right) +$$

$$+ \left(1-(1-p)^{l-1}\right) \underbrace{(1-p)^{n-1-(l-1)}}_{(1-p)^{n-l}} \prod_{i=1}^{l-2} \left(1-(1-p)^{n-1-i}\right) =$$

$$= (1-p)^{n-l} \left((1-p)^{l-1} \prod_{i=1}^{l-1} \left(1-(1-p)^{n-1-i}\right) + \left(1-(1-p)^{l-1}\right) \prod_{i=1}^{l-2} \left(1-(1-p)^{n\cdot}\right.\right.$$

$$= (1-p)^{n-l} \underbrace{\left(\prod_{i=1}^{l-2} \left(1-(1-p)^{n-1-i}\right) \right)}_{\prod_{i=2}^{l-1} \left(1-(1-p)^{n-i}\right)} \left(\underbrace{(1-p)^{l-1} \left(1-(1-p)^{n-1-(l-1)}\right)}_{(1-p)^{l-1} - (1-p)^{n-1}} + 1-(1-p)^{l\cdot}\right.$$

$$= (1-p)^n \prod_{i=1}^{l-1} \left(1-(1-p)^{n-i}\right)$$

Z toho se dají spočítat střední hodnoty

$$E(J_n^*(1)) = n - \sum_{j=1}^{n-1} (1-p)^j \sum_{l=1}^{j} \prod_{i=1}^{l-1} \left(1-(1-p)^{j-i}\right) =$$

$$\stackrel{?}{=} \sum_{l=1}^{n} \prod_{i=1}^{l-1} \left(1-(1-p)^{n-i}\right) = a_n$$

$$E(J_n^{red}(1)) = \frac{p}{1-p}\,(n - a_n)$$

Analýza výrazu a_n už je pak záležitostí teorie čísel.

Jiný postup:

1) rekurentní vztah pro psti $P_{n,l}$ se převede na rekurentní vztah pro vytvořující funkce

2) rekurze pro vytvořující fce se zderivuje, za argument se dosadí 1, tím získáme rekurentní vztah pro střední hodnotu

Nevede to ale k jednoduššímu výsledku.

Výpočet $E\, j_n^*(1)$:

Nejprve se vypočte $P((1,n) \in E^+)$, kde E^+ je množina hran tranzitivního, nikoli ale reflexivního uzávěru grafu G (tj. $(1,1) \notin E^+$).

Pak se použije vztah, že $E(j_n^*(1)) = 1 + E(j_n^+(1))$.

$$P((1,n) \in E^+) = \sum_{h=1}^{n-1} \underbrace{P((1,n) \in E^+ \mid j_{n-1}^*(1) = h)}_{1-(1-p)^h} \underbrace{P(j_{n-1}^*(1) = h)}_{P_{n-1,h}} =$$

$$= \sum_{h=1}^{n-1} (1-(1-p)^h) P_{n-1,h} = \underbrace{\sum_{h=1}^{n-1} P_{n-1,h}}_{1} - \sum_{h=1}^{n-1} (1-p)^h (1-p)^{n-1-h} \prod_{i=1}^{h-1} (1-(1-p)^{n-1-i}) =$$

$$= 1 - (1-p)^{n-1} \sum_{h=1}^{n-1} \prod_{i=1}^{h-1} (1-(1-p)^{n-1-i})$$

Nechť $Y_j = \begin{cases} 1 & \text{když } (1,j) \in E^+ \\ 0 & \text{jinak} \end{cases}$

Pak $E(j_n^*(1)) = 1 + E(j_n^+(1)) = 1 + \sum_{j=2}^{n} P((1,j) \in E^+) =$

$$= 1 + \sum_{j=2}^{n} \left(1 - (1-p)^{j-1} \sum_{h=1}^{j-1} \prod_{i=1}^{h-1} (1-(1-p)^{j-1-i})\right) =$$

$$= n - \sum_{j=1}^{n-1} (1-p)^j \underbrace{\sum_{h=1}^{j} \prod_{i=1}^{h-1} (1-(1-p)^{j-i})}_{a_j}$$

Rovnost $E(j_n^*(1)) = a_n$ se dostane z vlastností výrazů a_j.

Simon 1993: Pro $p = \frac{(\log n)^{\alpha}}{n}$ a $\alpha \le 1$ je očekávaná doba výpočtu algoritmu $O(n^2)$.

(speciálně tedy pro $0 < p \le \frac{\log n}{n}$)

Tranzitivní uzávěr pro cyklické grafy

1) graf se rozloží na silně souvislé komponenty
(pro každé 2 vrcholy platí, že vede cesta z i do j a zároveň vede cesta z j do i)
to se dá systematickým prohledáváním (do hloubky nebo do šířky) udělat v čase $O(n^2)$

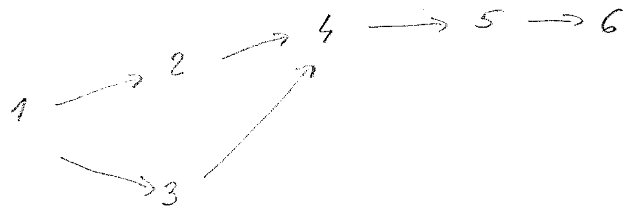
2) protože všechny vrcholy patřící do téže komponenty jsou navzájem dosažitelné, dají se "ztotožnit" – vznikne graf silně souvislých komponent, který je acyklický
najde se tranzitivní uzávěr grafu silně souvislých komponent

3) provede se konverze tohoto tranz. uzávěru na tranzitivní uzávěr původního grafu – to lze v čase $O(n^2)$

Jinak se dají použít algoritmy založené na násobení (umocňování) matice sousednosti. Jejich složitost je daná složitostí použitého algoritmu na násobení matic ($O(n^3)$ klasicky, $O(n^{2,81})$ podle Strassena atd.)

Předchozí algoritmus (G-k) má tu nevýhodu, že některé vrcholy přidává do tranzitivního uzávěru opakovaně.

Příklad:



$out^*(2) = \{2\} \cup out^*(4)$

$out^*(3) = \{3\} \cup out^*(4)$

$out^*(1) = \{1\} \cup out^*(2) \cup out^*(3)$

tj. $out^*(4)$ se v algoritmu přidá dvakrát

Následující algoritmus tento nedostatek nemá.

Ukážeme vylepšenou verzi algoritmu. (Simon 1988)

Nechť $G=(V,E)$ je acyklický graf. Rozklad $Z_1,\dots,Z_k$ množiny V (tj. $Z_i \neq \emptyset$ pro $1 \le i \le k$, $Z_i \cap Z_j = \emptyset$ pro $i \neq j$, $Z_1 \cup \dots \cup Z_k = V$) se nazývá rozklad na řetězce, jestliže každé Z_i je cesta v grafu G.

Protože G je topologicky uspořádaný, máme

$$Z_i = \{v_1 < \dots < v_s\}, \text{ kde } (v_j, v_{j+1}) \in E \ \forall\, 1 \le j \le k.$$

Číslo k se nazývá šířka dekompozice.

Označíme

$id(v)$ číslo řetězce obsahujícího v, tj. $id(v) = i \iff v \in Z_i$

a pro množinu $A \subseteq V$

$niv_j(A) = \min(A \cap Z_j)$ tj. nejmenší bod v řetězci Z_j, který je zároveň prvkem A

$niv(A) = \{niv_j(A),\ j=1,\dots,k\}$.

Speciálně pro $A = out^*(v)$ budeme značit

$$niv_j(v) = niv_j(out^*(v)), \quad niv(v) = niv(out^*(v)).$$

Platí:

$$out^*(v) = \bigcup_{1 \le j \le k} (out^*(v) \cap Z_j) = \bigcup_{1 \le j \le k} \{w \in Z_j;\ w \ge niv_j(v)\}$$

Tímto způsobem dostanu tranzitivní uzávěr, pokud budu umět spočítat $niv_j(v)$

platí $out^*(v) = \{v\} \cup \underbrace{out^*(w_1) \cup \dots \cup out^*(w_s)}_{A}$

takže

$$niv_j(v) = niv_j(out^*(v)) = niv_j(\{v\} \cup A) =$$

$$= \min((\{v\} \cup A) \cap Z_j) = \min((\{v\} \cap Z_j) \cup (A \cap Z_j)) =$$

$$= \min((\{v\} \cap Z_j), \underbrace{\min(A \cap Z_j)}_{niv_j(A)}) = \begin{cases} v, & \text{když } v \in Z_j \\ \min(niv_j(w_1), \dots, niv_j(w_s)) & \text{jinak} \end{cases}$$

protože

$$niv_j(A) = \min((out^*(w_1) \cup \dots \cup out^*(w_s)) \cap Z_j) =$$

$$= \min((out^*(w_1) \cap Z_j) \cup \dots \cup (out^*(w_s) \cap Z_j)) =$$

$$= \min(\underbrace{\min(out^*(w_1) \cap Z_j)}_{niv_j(w_1)}, \dots, \underbrace{\min(out^*(w_s) \cap Z_j)}_{niv_j(w_s)})$$

z toho se pak spočítá $niv(v) = niv(out^*(v))$.

Vylepšení algoritmu spočívá v tom, že se operace

$out^*(v) \leftarrow out^*(v) \cup out^*(w)$ ... čas $O(n)$

nahradí operací

$niv(out^*(v)) \longleftarrow niv(out^*(v) \cup out^*(w))$ ... čas $O(k)$

Obvykle iterací bývá podstatně méně než vrcholů.

Test $w \in out^*(v)$ se pak dělá na základě vlastnosti

$$w \notin out^*(w_1) \cup \dots \cup out^*(w_s) \iff w < \min(niv_{id(w)}(w_1), \dots$$

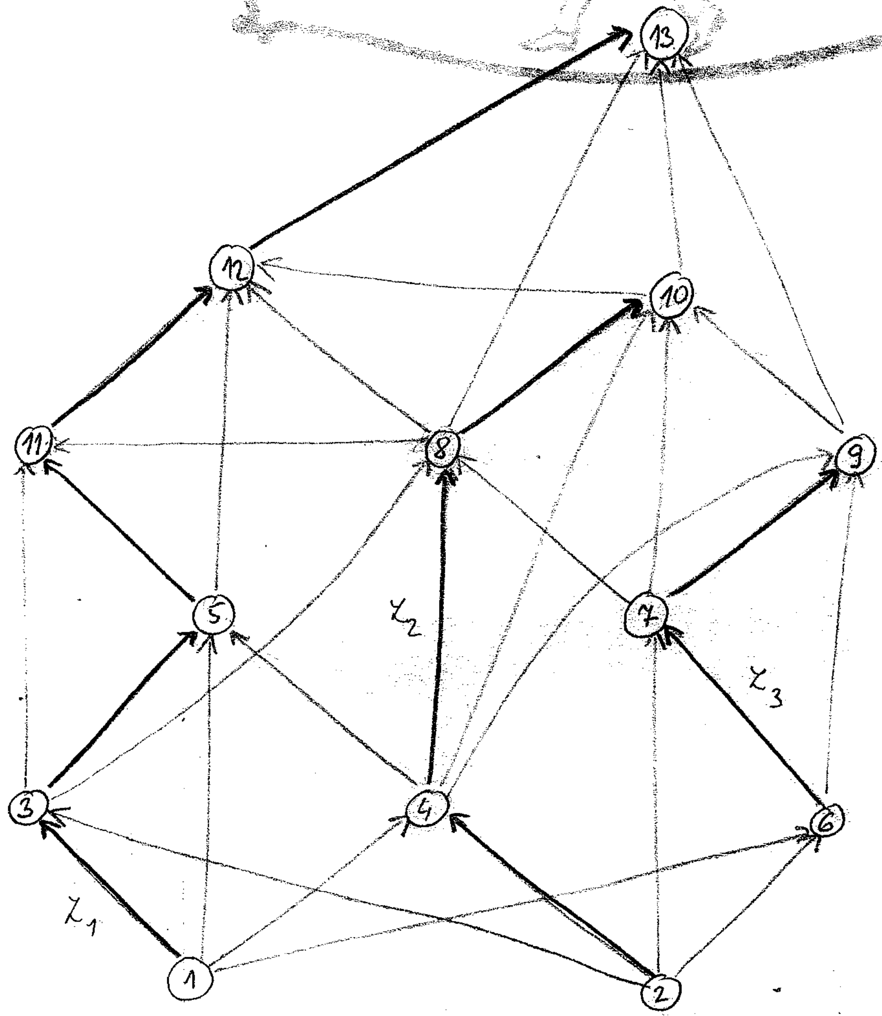
$$\dots niv_{id(w)}(w_s))$$

Postup: 1) Spočítá se řetězcová dekompozice (hladově)

2) Pro každé $v \in V$ se spočítá $niv(v) = (niv_1(v), \ldots, niv_k(v))$

3) Sestrojí se E^*

Obrázek:



hledám $suc^*(4)$: $niv_1(4) = 5$ ... vezmu 5, 11, 12, 13

$niv_2(4) = 4$ ... vezmu 4, 8, 10

$niv_3(4) = 9$ ... vezmu 9

Složitost 1) je zřejmě $O(n + |E|)$

Složitost 2):

Řetězec, ve kterém vrchol v leží, se bere od tohoto vrcholu.

V jiném řetězci se najde vrchol takový, do kterého z v vede hrana, ale nikoli cesta délky ≥ 2.

Neboli dělá se průnik řetězce s tranzitivním reduktem vrcholu v.

To se dělá pro každý vrchol a každý řetězec, složitost je tedy $O(|E| + k \cdot |E^{red}|)$.

(Tranzitivní redukt se opět konstruuje tak, že se vrcholy procházejí v opačném pořadí vzhledem k topologickému uspořádání, opakované přidávání týchž vrcholů tedy nehrozí).

Složitost 3) je zřejmě $O(|E^*|)$.

Celková složitost: $O(|E^*| + k \cdot |E^{red}|)$

Platí: $E(k) \leq \frac{\log(p \cdot n)}{p} + 1 = O\left(\frac{\lg(pn)}{p}\right)$

$$E(k \cdot |E^{red}|) = O(n^2) \text{ pro } \frac{\log^2 n}{n} \leq p < 1$$

$$= O(n^2 \log\log n) \text{ jinak}$$

Při výpočtu pozor - veličiny k a $|E^{red}|$ nejsou nezávislé!

Výpočet silnězcové dekompozice

vstup: $G = (V, E)$

výstup: $Z_1, Z_2, \dots, Z_k$ $id(v)$ pro $v \in V$

$i \leftarrow 1$

for all $v \in V$ **do** $id(v) \leftarrow 0$ **od**

$V_i \leftarrow V$

while $V_i \neq \emptyset$ **do**

  $x \leftarrow min(V_i)$

  $Z \leftarrow \{x\}$

  while existuje $y \in V_i$ tak, že $(x, y) \in E$ **do**

    vezmeme minimální takové y

    $Z \leftarrow Z \cup \{y\}$

    $x \leftarrow y$

  od

  $Z_i \leftarrow Z$

  $V_{i+1} \leftarrow V_i - Z_i$

  for all $v \in Z_i$ **do** $id(v) \leftarrow i$ od

  $i \leftarrow i+1$

od

Čas: $O(n+e)$

vnitřní cyklus: $O(x(v))$

Výpočet niv(v), out^red(v)

vstup: G = (V, E), id : V → {1, .., k}

výstup: niv(v), out^red(v) ∀ v ∈ V

```
1)  for s ← 1 to k do nivfield[s] ← ∞ od
    for v ← n downto 1 do
        out^red(v) ← ∅ ; nivlist(v) ← ∅
        for all w ∈ out(v) do            { v rostoucím uspořádání }
            if w < nivfield(id(w)) then
                out^red(v) ← out^red(v) ∪ {w}
(3)             for all p ∈ nivlist(w) do
            (4)     nivfield(id(p)) ← min(nivfield(id(p)), p)
                od
            fi
        od
        nivfield(id(v)) ← v
        for s ← 1 to k do
            if nivfield(s) ≠ ∞ then
(2)             nivlist(v) ← nivlist(v) ∪ nivfield(s) fi
            nivfield(s) ← ∞
        od
    od
```

nivfield(id(p)) ... odpovídá $niv_{id(p)}$

nivlist(v) ... niv(v)

Korektnost algoritmu se dokazuje indukcí.

Čas $O(n^2)$ pro násobení matic se deterministickým algoritmem nepodařilo dosáhnout. Existuje pravděpodobnostní algoritmus, (O'Neil 1973), který počítá Booleův součin dvou Booleových matic typu $n \times n$ v očekávaném čase $O(n^2)$.

Protože v mnoha aplikacích (relační databáze atd.) se vyskytují tzv. řídké grafy (mají velmi málo hran), má smysl používat algoritmy pro násobení řídkých matic (mají velmi málo nenulových prvků). Šetří to jednak paměť (ukládají se jen nenulové prvky), jednak čas (nulami se nenásobí).

Základem pro rekurzivní redukci bude i v tomto případě půlení rozměru zadání. Zapíšeme-li matice A, B a C prostřednictvím čtyř submatic:

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

kde "čtvrtinové" submatice A_{ik} mají rozměry

A_{11} ... ((m+1)div2)x((n+1)div2) A_{12} ... ((m+1) div 2)x(n div 2)
A_{21} ... ((mdiv2)x((n+1)div2) A_{22} ... (m div 2)x(n div 2)

a podobně jsou definovány rozměry submatic B_{kj} a C_{ij}, platí i nyní

$$C_{ij} = A_{i1}*B_{1j} + A_{i2}*B_{2j} \; ; \; i, j = 1, 2, \tag{4.7}$$

kde '+' a '*' označují sčítání, resp. násobení matic. V roce 1969 uveřejnil Strassen algoritmus, který při redukci "čtvrcením" vystačí se sedmi součiny submatic, na rozdíl od formulí (4.7), jejichž použití vyžaduje osm součinů:

$M_1 := (A_{12} - A_{22}) * (B_{21} + B_{22});$
$M_2 := (A_{11} + A_{22}) * (B_{11} + B_{22});$
$M_3 := (A_{11} - A_{21}) * (B_{11} + B_{12});$
$M_4 := (A_{11} + A_{12}) * B_{22};$
$M_5 := A_{11} * (B_{12} - B_{22});$
$M_6 := A_{22} * (B_{21} - B_{11});$
$M_7 := (A_{21} + A_{22}) * B_{11};$

$C_{11} := M_1 + M_2 - M_4 + M_6;$
$C_{12} := M_4 + M_5;$
$C_{21} := M_6 + M_7;$
$C_{22} := M_2 - M_3 + M_5 - M_7;$

4.4.1. Poznámka

Nejsou-li rozměry m, n, p mocninami dvojky, musíme si ovšem pokaždé, kdy při půlení narazíme na lichý rozměr, pomoci přidáním nulového řádku a/nebo sloupce.

Omezíme-li se na čtvercové matice ($m = n = p$) a uvažujeme-li pouze aritmetické operace (t.j. zanedbáváme nutné přesuny), dostaneme pro $n = 2^k$, $k > 0$ rekurentní vztah

$$T(n) = 7T(n/2) + 18(n/2)^2$$

Předpoklady tvrzení 4.1 jsou tedy splněny pro $a = 7$, $c = 2$, $d = 2$, odkud $T(n) = O(n^{\log 7})$, přibližně tedy $O(n^{2.81})$.

Algoritmus pro násobení řídkých matic (Schorr 1982)

Máme matici A typu $M \times N$ s hustotou D_1 a matici B typu $N \times K$ s hustotou D_2 (Hustota je podíl nenulových prvků, tj.

$$D_1 = \frac{\text{počet nenulových prvků v matici } A}{MN}, \quad \text{podobně } D_2$$

jinak: počet nenulových prvků $A = D_1 MN$

—"— $B = D_2 NK$

u řídkých matic je $D_1, D_2 \ll 1$ (obsahují hodně 0)

počítáme matici $C = A \times B$

Klasický algoritmus na násobení matic prochází současně i-tý řádek matice A a j-tý sloupec B a počítá $c_{ij} = \sum_{t=1}^{N} A_{it} B_{tj}$. Vyžaduje čas $O(MNK)$ a nevyužívá toho, že mnoho prvků je nulových a příslušné součiny se tedy vůbec nemusí počítat.

Následující algoritmus využívá speciální datovou strukturu pro reprezentaci řídkých matic, která uchovává pouze jejich nenulové prvky. V této struktuře každý prvek A_{ij} matice A je reprezentován záznamem $a_{ij} = (A_{ij}, i, j, p_i, p_j)$, kde

- A_{ij} je hodnota prvku
- i je řádkový index
- j je sloupcový —"—
- p_i je ukazatel na následující nenulový prvek v i-tém řádku
- p_j —— " —— j-tém sloupci

(pro poslední prvek v řádku, resp. sloupci, je $p_i = NIL$, resp. $p_j = NIL$)

dále jsou dána pole adres prvních nenulových prvků v každém řádku, resp. sloupci, Row_A, resp. Column_A

Takže matice typu $M \times N$, která má $D_1 MN$ nenulových prvků, potřebuje k reprezentaci $5D_1 MN + M + N$ paměti.

Př:

$$A = \begin{pmatrix} 1 & 0 & 2 \\ 0 & 0 & 3 \\ 4 & 0 & 0 \end{pmatrix}, \quad B = \begin{pmatrix} 2 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix}, \quad C = A \times B = \begin{pmatrix} 2 & 2 \\ 0 & 3 \\ 8 & 0 \end{pmatrix}$$

$a_{11} = (1, 1, 1,$ ukaz. na 2, ukaz. na 4$)$

$a_{13} = (2, 1, 3,$ NIL, ukaz. na 3$)$

$a_{23} = (3, 2, 3,$ NIL, NIL$)$

$a_{31} = (4, 3, 1,$ NIL, NIL$)$

Row_A = (adr. 1, adr. 3, adr. 4)

Column_A = (adr. 1, NIL, adr. 2)

$b_{11} = (2, 1, 1,$ NIL, NIL$)$

$b_{32} = (1, 3, 2,$ NIL, NIL$)$

Row_B = (adr. 2, NIL, adr. 1)

Column_B = (adr. 2, adr. 1)

Popis algoritmu: je založen na tom, že každý nenulový prvek A_{ij} matice A se násobí všemi nenulovými prvky j-tého řádku matice B a pouze jimi

algoritmus prochází všechny řádky matice A

v každém řádku prochází všechny jeho nenulové prvky

pro každý takový prvek A_{it} prochází t-tý řádek B, a

násobí A_{it} všemi nenulovými prvky B_{tj} tohoto řádku každý součin $A_{it} B_{tj}$ přičte k (i,j)-tému prvku výsledné matice C

Doba výpočtu algoritmu bude úměrná počtu provedených násobení, tj. bude $O(R)$, kde $R = \sum_{t=1}^{N} X_t Y_t$, kde dále

X_t je počet nenulových prvků v t-tém sloupci A

Y_t je počet nenulových prvků v t-tém řádku B

Jsou-li nenulové prvky v maticích rozloženy "rovnoměrně" v tom smyslu, že $X_t = D_1 M \ \forall t$ a $Y_t = D_2 K \ \forall t$, pak $R = D_1 D_2 MNK$

Předpokládejme nyní, že vstupem pro algoritmus jsou matice generované náhodně tak, že

$$P(A_{ij} \neq 0) = D_1 \quad \forall i,j$$

$$P(A_{ij} = 0) = 1 - D_1 \quad \forall i,j$$

$$P(B_{ij} \neq 0) = D_2 \quad \forall i,j$$

$$P(B_{ij} = 0) = 1 - D_2 \quad \forall i,j$$

(hodnoty prvků matice A jsou navzájem nezávislé, podobně v matici B)

a rozdělení prvků matice A je nezávislé na rozdělení prvků matice B.

Definujme náh. veličiny V_{ij} a W_{ij} tak, že

$$V_{ij} = \begin{cases} 1 & \text{když } A_{ij} \neq 0 \\ 0 & \text{když } A_{ij} = 0 \end{cases} \qquad W_{ij} = \begin{cases} 1 & \text{když } B_{ij} \neq 0 \\ 0 & \text{když } B_{ij} = 0 \end{cases}$$

pak $X_t = \sum_{i=1}^{M} V_{it}$, $Y_t = \sum_{i=1}^{K} W_{ti}$

průměrná doba výpočtu:

$$ER = \sum_{t=1}^{N} E(X_t Y_t) = \sum_{t=1}^{N} EX_t \, EY_t \quad \text{neboť } X_t \text{ a } Y_t \text{ jsou nezávislé}$$

$$EX_t = \sum_{i=1}^{M} EV_{it} = \sum_{i=1}^{M} \left(1 \cdot D_1 + 0 \cdot (1 - D_1)\right) = D_1 M$$

$$EY_t = \sum_{i=1}^{K} EW_{ti} = \sum_{i=1}^{K} \left(1 \cdot D_2 + 0 \cdot (1 - D_2)\right) = D_2 K$$

$$ER = D_1 D_2 MNK$$

rozptyl doby výpočtu:

$$\mathrm{var}\, R = \mathrm{var}\left(\sum_{t=1}^{N} X_t Y_t\right) = \sum_{t=1}^{N} \mathrm{var}(X_t Y_t)$$ neboť součiny $X_t Y_t$ jsou navzájem nezávislé

platí: $\mathrm{var}(X_t Y_t) = \mathrm{var}\, X_t \, \mathrm{var}\, Y_t + (EX_t)^2 \mathrm{var}\, Y_t + (EY_t)^2 \mathrm{var}\, X_t$

dk: $\mathrm{var}(X_t Y_t) = E(X_t Y_t - EX_t Y_t)^2 =$

$$= E\left(X_t^2 Y_t^2 - 2X_t Y_t EX_t Y_t + (EX_t Y_t)^2\right) =$$

$$= E(X_t^2 Y_t^2) - (EX_t Y_t)^2 = EX_t^2 EY_t^2 - (EX_t)^2 (EY_t)^2 =$$

$$= EX_t^2 \underbrace{\left[EY_t^2 - (EY_t)^2\right]}_{\mathrm{var}\, Y_t} + \underbrace{EX_t^2 (EY_t)^2 - (EX_t)^2 (EY_t)^2}_{(EY_t)^2 \underbrace{\left[EX_t^2 - (EX_t)^2\right]}_{\mathrm{var}\, X_t}} =$$

$$= EX_t^2 \mathrm{var}\, Y_t + (EY_t)^2 \mathrm{var}\, X_t =$$

$$= \mathrm{var}\, Y_t \underbrace{\left[EX_t^2 - (EX_t)^2\right]}_{\mathrm{var}\, X_t} + \mathrm{var}\, Y_t (EX_t)^2 + (EY_t)^2 \mathrm{var}\, X_t \; .$$

$$= \mathrm{var}\, X_t \, \mathrm{var}\, Y_t + (EX_t)^2 \mathrm{var}\, Y_t + (EY_t)^2 \mathrm{var}\, X_t$$

$$X_t = \sum_{i=1}^{M} V_{it} \qquad EX_t = \sum_{i=1}^{M} EV_{it} = MD_1$$

$$\mathrm{var}\, X_t = \sum_{i=1}^{M} \mathrm{var}\, V_{it} = MD_1(1-D_1)$$

$$Y_t = \sum_{i=1}^{K} W_{ti} \qquad EY_t = KD_2 \qquad \mathrm{var}\, Y_t = KD_2(1-D_2)$$

$$\operatorname{var} R = \sum_{t=1}^{N} \operatorname{var}(X_t Y_t) = N\left[\operatorname{var} X_t \operatorname{var} Y_t + (EX_t)^2 \operatorname{var} Y_t + (EY_t)^2 \operatorname{var} X_t\right] =$$

$$= N\Big[D_1\underbrace{(1-D_1)}_{\leq 1} M D_2 \underbrace{(1-D_2)}_{\leq 1} K + D_1^2 M^2 D_2 \underbrace{(1-D_2)}_{\leq 1} K + D_2^2 K^2 D_1 \underbrace{(1-D_1)}_{\leq 1} M\Big]$$

$$\leq D_1 D_2 MNK + D_1^2 D_2 M^2 NK + D_1 D_2^2 MNK^2 =$$

$$= D_1 D_2 MNK\,(1 + D_1 M + D_2 K)$$

Poznámka: Algoritmus na rozdíl od klasického pracuje po řádcích matice B, nikoliv po sloupcích, je tedy rychlejší než klasický algoritmus i v případě velkých ne-řídkých matic, které jsou v počítači uloženy po řádcích.