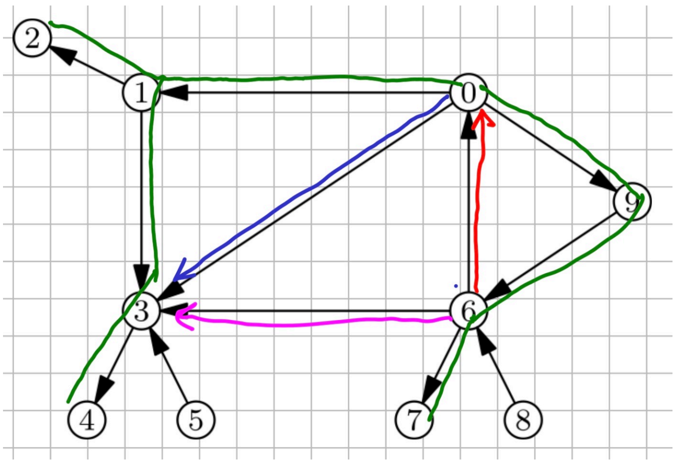
Průběh DFS se dá napsat jako uzávorkování:

$$(_0\, (_1\, (_2\,)_2\, (_3\, (_4\,)_4\,)_3\,)_1\, (_9\, (_6\, (_7\,)_7\,)_6\,)_9\,)_0$$

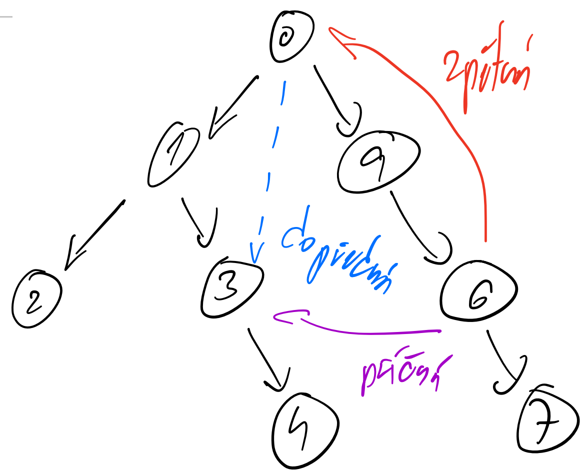
– in/out je levá/pravá závorka



– je to validní uzávorkování

dobré uzávorkování odpovídá stromům:

Strom DFS



toto je kostra grafu

Všechny typy hran $xy \in E$:

dopředná
zpětná
příčná

(DFS klasifikace)

$(_x \ldots (_y \ldots)_y \ldots)_x$ = stromová –> y neobjevené

= dopředná –> y uzavřené

$(_y \ldots (_x \ldots)_x \ldots)_y$ = zpětná

$(_y \ldots)_y \ldots (_x \ldots)_x$ = příčná

$(_x \ldots)_x \ldots (_y \ldots _y)$ = nenastane

Co když neorientovaný?

poprvé: stromová, zpětná

podruhé: zpětná, dopředná

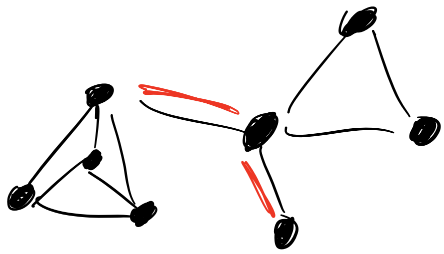
jiné nejsou

Typ hrany zjistíme v $O(1)$ čase – stačí in/out a stavy vrcholů.

DFS v čase $\Theta(n+m)$ a prostoru $\Theta(n+m)$ najde dosažitelné vrcholy a klasifikuje dosažitelné hrany.

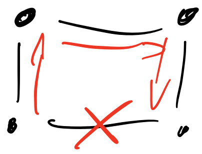
Hledání mostu v neorientovaných grafech:

Df: Hrana $e \in E(G)$ je most $\Leftrightarrow$ $G - e$ má více komp. souv.

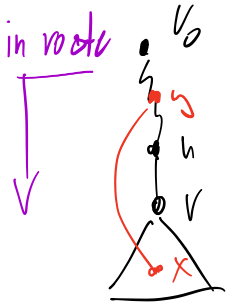


e není most $\Leftrightarrow$ $e \in$ kružnici:

Každá hrana po smazání z kružnice nezmění dosažitelnost, jelikož to můžu po kružnici objet.



Každá zpětná leží na kružnici $\Rightarrow$ (není most) stačí tedy koukat na stromové hrany.

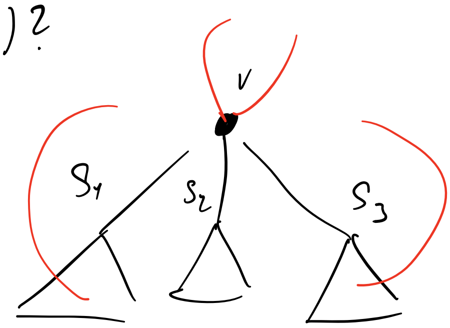


uv leží na kružnici $\Leftrightarrow$ $\exists\, xy \in E$ zpětná t.ž.
x je potomkem v
y je předkem v $\Big\}$ $in(y) < in(v)$

$low(v) < in(v)$ $\Leftrightarrow$

Df: $low(v) := \min \{ in(y) \mid xy$ je zpětná & x neostře pod $v \}$

Jak počítám $low(v)$?

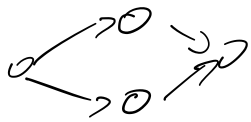


$$low(v) = \min \begin{cases} low(s) \mid s \text{ syn } v \\ in(y) \mid vy \text{ zpětná hrana} \end{cases}$$

trvá $O(\deg(v))$

Alg. nalezne všechny mosty v čase a prostoru $\Theta(n+m)$.

Acyklický orientovaný graf (DAG):



- nemá tedy žádnou zpětnou hranu.

→ pak opakujeme DFS na neobjevené vrcholy

$\exists$ dosažitelný cyklus $\iff$ DFS najde zpětnou hranu.

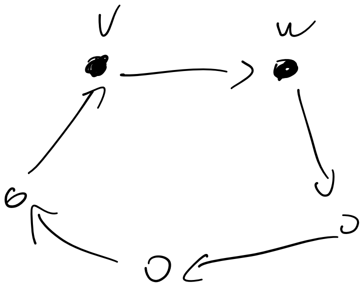
$\Leftarrow$ triviální

$\Rightarrow$ nechť $\exists$ cyklus C, zvolíme $v \in C$ s min $out(v)$

$w :=$ následník v na cyklu

na hraně vw roste out.

tohle je ale pouze u zpětné hrany.



Pro $\forall v \in V(G)$, pokud $stav(v) =$ neobjeven, volám DFS2(v)

Topologické uspořádání:

Lineární uspořádání $\leq$ na $V(G)$ t. ž.

$$\forall xy \in E(G) : x \leq y$$

↳ alternativní je to očíslování vrcholů t. ž. $v_i v_j \in E : i < j$

cyklus nemá topologické uspořádání

Graf má TU $\Leftrightarrow$ Graf je DAG:

Zdroj: $v \in V$ je zdroj $\Leftrightarrow$ $\deg^{in}(v) = 0$ [stok $= \deg^{out}(v) = 0$]

Každý DAG má zdroj.

- vezmu vrchol. Pokud mám vrchol, je to stok, nebo do něj vede nějaká hrana. Pokud vede, jdu zpět, až narazím na zdroj.

$\Leftarrow$ - Stačí tedy odtrhávat zdroje.

$\Rightarrow$ Viz. výše TU

Pořadí, v němž DFS opouští vrcholy, je opačné topologické.

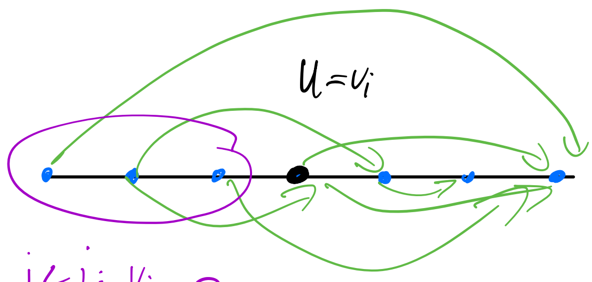
na každé klesá out.

jediný, kde neklesá out, jsou zpětné a ty se nevyskytují v DAG.

Topologická indukce

Zvolme u, chceme počet cest z u do v. $C(v) := \#$cest z u do v

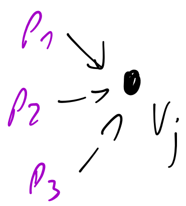
Nechť $v_1 — v_n$ je topologické pořadí.



Pro $j < i$: $v_j = 0$

$v_i = 1$

$\rightarrow$ Pro $j > i$, ind.



$\rightarrow$ počet cest bude součet všech předchozích cest $p_1 — p_3 \dots$

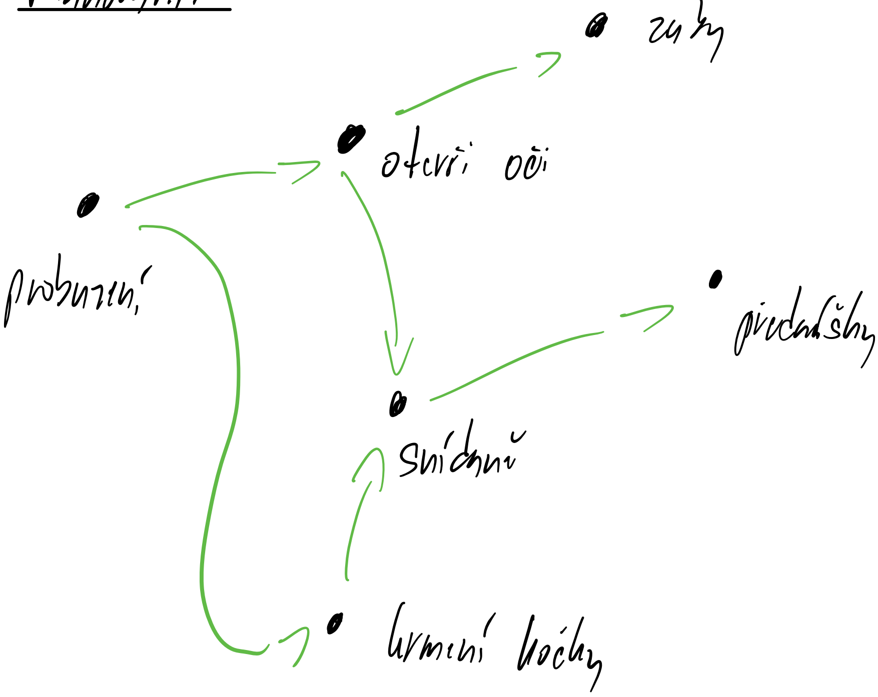
$$\#\text{cest do } v_j = \sum_{p:\, pv_j \in E} c(p)$$

$\rightarrow$ to jde dělat postupně zleva lineárně.

u jednoho vrcholu: $\Theta(\deg(v))$

celkem tedy $\Theta(m+n)$, včetně topo. uspoř.

Plánování:



Nesmí tam být smyčky.

Silná souvislost:

(Slabá: $\bullet \to \bullet \leftarrow \bullet$)

Relace $\rightsquigarrow$ na V: $u \rightsquigarrow v \equiv \exists$ sled z u do v

dosaž.

$\leftrightsquigarrow$ $\quad u \leftrightsquigarrow v \equiv u \rightsquigarrow v \;\&\; v \rightsquigarrow u$

$\leftrightsquigarrow$ je ekvivalence

L> ekv. třídy: komponenty silné souvislosti

Graf je silně souvislý $\iff$ # komponent silné souvislosti $= 1$

Graf komponent $C(G)$: (kondenzace [kontrakce])

vrcholy: komponenty grafu G

hrany: $(C_i, C_j) : \exists x \in C_i, \exists y \in C_j : \exists xy \in E(G)$

když vede hrana mezi komponentami

Graf komponent je DAG:

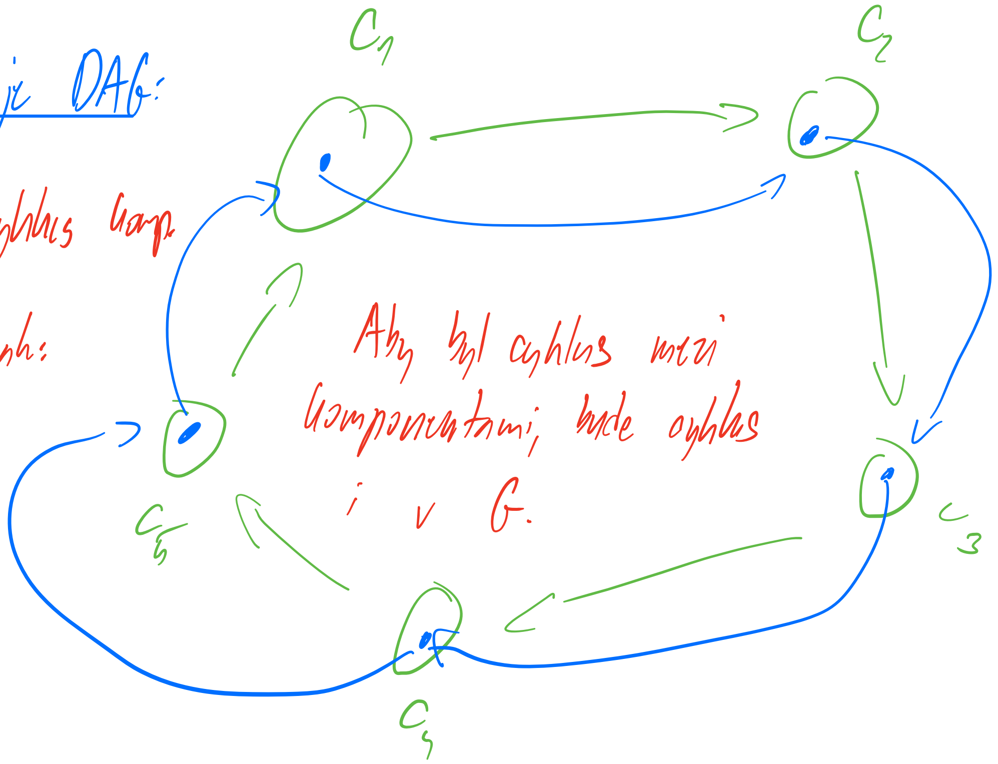
Kdyby existoval cyklus komp. $C_1 - C_k, C_1$, pak:

$\exists\ u_1 - u_k$

$v_1 - v_k$:

$\forall i \quad u_i, v_i \in C_i$

$u_i, v_{i+1} \in E$



Aby byl cyklus mezi komponentami, bude cyklus i v G.

Nyní v_1 (cesta v C_1) $u_1 v_2$ (cesta v C_2) $u_2 v_3 \dots$

$\dots v_k$ (cesta v C_k) $u_k\ v_1$

je cyklus v G $\Rightarrow$ spor s růzností $C_1 - C_k$.