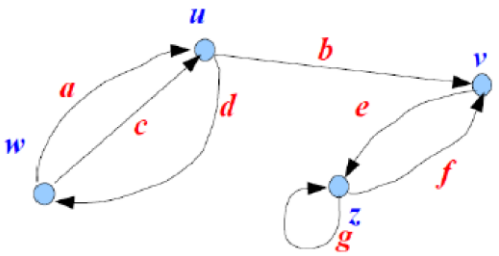
10. Orientované grafy (orientované cesty a kružnice, souvislost a silná souvislost, turnaj, eulerovský graf, Dijkstrův a Floyd-Warshallův algoritmus pro hledání cesty minimální délky).

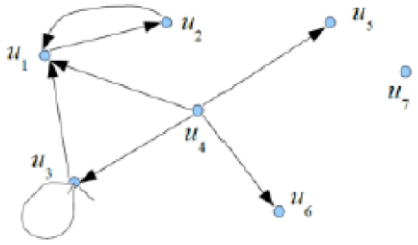
Orientované grafy

Orientovaný graf je trojice $G = (U, H, \varepsilon)$, kde U je konečná množina vrcholů, H je konečná množina orientovaných hran. Přitom ε: $H \to \{(u, v) \mid u, v \in U\} \cup \{u \mid u \in U\}$, je zobrazení, které každé hraně přiřadí buď orientovanou dvojici uzlů (u, v) nebo uzel u. V prvním případě říkáme, že hrana vede z uzlu u do v, v druhém případě říkáme, že hrana tvoří smyčku v uzlu u.



Nechť trojice $G = (U, H, \varepsilon)$ je orientovaný graf. Pak definujeme $u \in U$ pro uzel dvě čísla:

- $\deg_+(u) = |\{h \in H \mid \exists v \in H: \varepsilon(h) = (v, u)\}|$, které nazýváme **vstupním stupněm uzlu**;
- $\deg_-(u) = |\{h \in H \mid \exists v \in H: \varepsilon(h) = (u, v)\}|$, které nazýváme **výstupním stupněm uzlu**;



Číslo $\deg_+(u)$ se rovná počtu hran, které vedou z nějakého uzlu do u, číslo $\deg_-(u)$ se rovná počtu hran, které vedou z uzlu u do nějakého uzlu.

Počteční uzel grafu má $\deg_+(u) = 0$, **koncový uzel grafu** má $\deg_-(u) = 0$.

Uzel	$\deg_+$	$\deg_-$
u_1	3	1
u_2	1	1
u_3	2	2
u_4	0	4
u_5	1	0
u_6	1	0
u_7	0	0

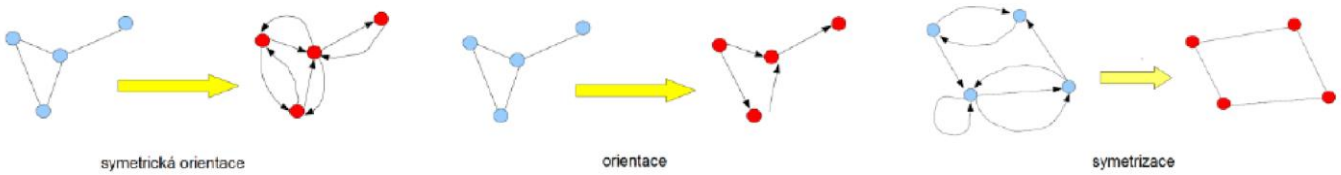
Analogicky k obyčejnému grafu, lze definovat varianty orientovaného sledu, orientovaného tahu, orientované cesty a orientované kružnice.

Máme-li zadán obyčejný graf $G = U, H$ je k němu možné definovat orientovaný graf $G' = (U, H', \varepsilon)$ tak, že pro každou hranu $\{u, v\} \in H$ existují v H' hrany h_1 a h_2 takové, že $\varepsilon(h_1) = (u, v) \wedge \varepsilon(h_2) = (v, u)$. Takovýto graf G' se nazývá **symetrickou orientací grafu** G. Jinými slovy hrana v obyčejném grafu mezi uzly u a v sebe nahradí oběma orientovanými hranami mezi těmito uzly v novém grafu.

Máme-li zadán obyčejný graf $G = (U, H)$ je k němu možné definovat orientovaný graf $G' = (U, H', \varepsilon)$ tak, že pro každou hranu $\{u, v\} \in H$ existují v H' hrana h taková, že $\varepsilon(h) = (u, v)$ a neexistuje hrana h' taková, že $\varepsilon(h) = (v, u)$. Tento graf se nazývá **orientace grafu** G. Je zřejmé, že na rozdíl od symetrické

orientace grafu, která je jednoznačně definována, může existovat k obyčejnému grafu více jeho orientací a navíc orientace grafu neobsahuje smyčky.

Máme-li zadán orientovaný graf $G = (U, H, \varepsilon)$, potom k němu můžeme sestrojit obyčejný graf $G' = (U, H')$, který se nazývá **symetrizací grafu** G, kde $H' = \{\{u, v\} \mid u, v \in H, u \neq v, \exists h \in H: \varepsilon(h) = (u, v) \vee \varepsilon(h) = (v, u)\}$. Jinými slovy symetrizace vznikne „zanedbáním“ šipek, vícenásobných hran a smyček v původním grafu.

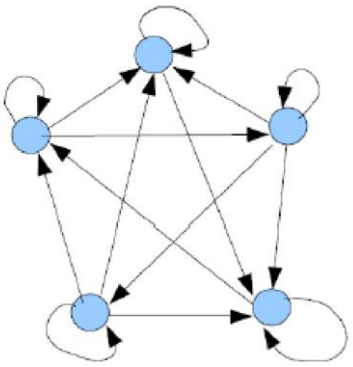


Říkáme, že orientovaný graf je **slabě souvislý**, jestliže jeho symetrizací vznikne obyčejný souvislý graf. Říkáme, že orientovaný graf $G = (U, H, \varepsilon)$ je **silně souvislý**, jestliže pro libovolné dva uzly $u, v \in U$ existuje orientovaná cesta z u do v.

Orientovaný graf $G = (U, H, \varepsilon)$ se nazývá **turnaj**, když pro:

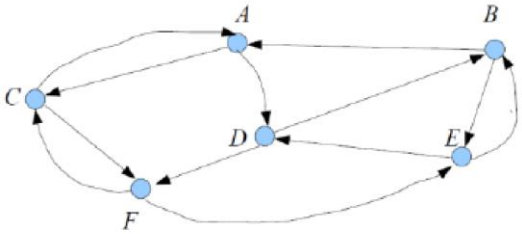
- každou množinu uzlů $\{u, v\}$, že $u, v \in U, u \neq v$, kde existuje právě jedna hrana $h \in H$ taková, že platí $\varepsilon(h) = (u, v) \vee \varepsilon(h) = (v, u)$;
- každý uzel $u \in U$ existuje právě jedna hrana (smyčka) $h \in H$ taková, že platí $\varepsilon(h) = (u, u)$;

Řečeno jinak existuje pro každou dvojici různých uzlů jediná orientovaná hrana jdoucího z jednoho uzlu do druhého a u každého uzlu je smyčka.



Eulerovské grafy

Orientovaný graf $G = (U, H, \varepsilon)$ se nazývá **eulerovským grafem**, jestliže v něm existuje UZAVŘENÝ tah (↭ *„nakreslí se jedním tahem a skončí se tam, kde se začalo“*) délky obsahující všechny orientované hrany. Vzhledem k tomu, že v tahu se nesmějí opakovat hrany, je orientovaný graf eulerovský právě tehdy, když se všechny jeho orientované hrany dají nakreslit ve směru šipek jedním tahem, aniž zvedneme tužku s papíru a po jedné hraně táhneme právě jednou.



C,A,D,F,C,F,E,B,E,D,B,A,C

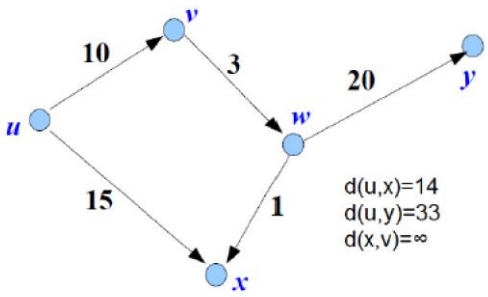
Platí věta, že souvislý orientovaný graf $G = (U, H, \varepsilon)$ je právě tehdy eulerovský, když platí $\deg_+(u) = \deg_-(u)$ pro $\forall u \in U$.

Délky hran a cest

Pro účely měření délek hran a cest budeme od teď pracovat orientovanými grafy bez vícenásobných

hran a smyček. „Hrana“ bude vždy znamenat orientovanou hranu a „cesta“ orientovanou cestu.

Nechť $G = (U, H)$ je graf a každé hraně $h \in H$ nechť je přiřazeno reálné číslo $l(h)$. Potom tomuto číslu budeme říkat **délka hrany** h. **Délka** $d(p)$ **cesty** p v grafu G se definuje jako součet délek všech hran obsažených v cestě p.



Nechť je dán graf $G = (U, H)$ a $u, v \in U$. Pokud existuje mezi uzly u a v cesta minimální délky, definujeme $d(u, v)$ jako délku této cesty. Pokud z uzlu u do uzlu v vůbec žádná cesta neexistuje, klademe $d(u, v) = \infty$.

Algoritmy pro hledání minimální cesty

Dijkstrův algoritmus

Horní odhad vzdálenosti z uzlu s do uzlu v je číslo $D(v)$: $D(v) \geq d(s,v)$.

Pro každý uzel $v \in U$ bude symbol $\pi(v)$ označovat uzel, který bezprostředně předchází uzlu v v cestě minimální délky z s do v zkonstruované Dijsktrovým algoritmem. Pokud taková cesta dosud nebyla zkonstruována, pak $\pi(v) = \emptyset$.

Pro každý uzel $v \in U$ definujeme symbol $N(v)$ označující množinu všech uzlů spojených přímo nějakou hranou s uzlem v, tedy $N(v) = \{w \in U \mid (v,w) \in H\}$.

$S \subseteq U$ je množina všech uzlů v, pro které už byla Dijsktrovým algoritmem definitivně stanovena cesta minimální délky $p(s,v)$ odpovídající minimální vzdálenosti $c(s,v)$.

Schéma Dijsktrova algoritmu po selsku:

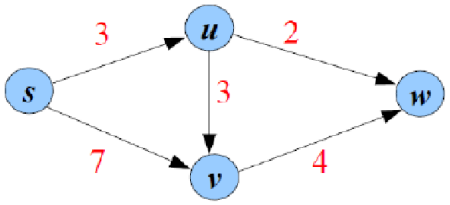
(1) Přiřaď vzdálenosti všem uzlům, počátečnímu 0 a všem ostatním nekonečno ∞;

(2) Označ všechny uzly jako nenavštívené, počáteční uzel nastav jako aktuální zpracovávaný;

(3) Pro aktuální uzel zvaž všechny jeho dosud nenavštívené sousedy a přepočítej pro ně vzdálenosti od počátečního uzlu přes aktuální. Pokud je přepočítaná vzdálenost menší, než ta současná, přiřaď mu tuto vzdálenost.

(4) Ve chvíli, kdy je přepočet vzdáleností sousedních uzlů hotov, označ aktuální uzel jako navštívený (už nikdy se nebude kontrolovat, jeho hodnota udává konečnou vzdálenost od počátečního uzlu);

(5) Z množiny dosud nenavštívených uzlů vyber ten s nejmenší vzdáleností od počátečního uzlu a pokračuj krokem (3) do doby, dokud je množina nenavštívených uzlů neprázdná.



1. krok: $S = \{s\}, Q = \{u, v, w\}, D(s) = 0, D(u) = D(v) = D(w) = \infty$

2. krok: $S = \{s, u\}, Q = \{v, w\}, D(s) = 0, D(u) = 3, D(v) = 7, D(w) = \infty$
$\pi(u) = s, \pi(v) = s$

3. krok: $S = \{s, u, w\}, Q = \{v\}, D(s) = 0, D(u) = 3, D(v) = 6, D(w) = 5$
$\pi(u) = s, \pi(v) = u, \pi(w) = u$

4. krok: $S = \{s, u, w, v\}, Q = \emptyset, D(s) = 0, D(u) = 3, D(v) = 6, D(w) = 5$
$\pi(u) = s, \pi(v) = u, \pi(w) = u$

$p(s,u) = s \rightarrow u, \; p(s,v) = s \rightarrow u \rightarrow v, \; p(s,w) = s \rightarrow u \rightarrow w$

Dijkstrův algoritmus nelze použít, pokud se v grafu objevují hrany se zápornou délkou, tento nešvar řeší...

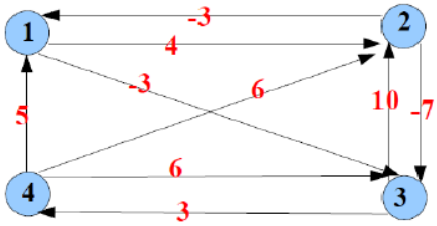
Floyd-Warshallův algoritmus

Při každém zadání délek hran tento algoritmus nalezne cestu minimální délky z každého uzlu do každého jiného uzlu, a pokud taková cesta neexistuje kvůli kružnici se zápornou délkou, tuto kružnici odhalí.

Uvažujme graf $G=(U,H)$, který má $|U|=n$ uzlů a délky hran jsou zadány maticí A, pak budeme používat matici P, kde jsou jednotlivé prvky p_{ij} nastaveny na hodnou sloupce j ve, kterém se nacházejí. Algoritmus má vždy n iterací.

Začneme s maticí $A^0=A$, $P^0=P$ a v i-té iteraci vytvoříme matice A^i, P^i pomocí matic A^{i-1}, P^{i-1}. Nakonec tedy dostaneme matice A^n, P^n. Prvky matic A^j, P^j, $j = 1, 2, \ldots, n$ se vypočítají následujícím způsobem:

- $\texttt{if}\left(a_{ik}^{j-1} \le a_{ij}^{j-1} + a_{jk}^{j-1}\right)\texttt{ then}\left\{a_{ik}^{j} = a_{ik}^{j-1};\ p_{ik}^{j} = p_{ik}^{j-1}\right\}$
- $\texttt{if}\left(a_{ik}^{j-1} > a_{ij}^{j-1} + a_{jk}^{j-1}\right)\texttt{ then}\left\{a_{ik}^{j} = a_{ij}^{j-1} + a_{jk}^{j-1};\ p_{ik}^{j} = j\right\}$



$$\Rightarrow A^0 = \begin{bmatrix} 0 & 4 & -3 & \infty \\ -3 & 0 & -7 & \infty \\ \infty & 10 & 0 & 3 \\ 5 & 6 & 6 & 0 \end{bmatrix}, P^0 = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 4 \end{bmatrix}$$

$$j=1: A^0 = \begin{bmatrix} 0 & 4 & -3 & \infty \\ -3 & 0 & -7 & \infty \\ \infty & 10 & 0 & 3 \\ 5 & 6 & 6 & 0 \end{bmatrix} \Rightarrow A^1 = \begin{bmatrix} 0 & 4 & -3 & \infty \\ -3 & 0 & -7 & \infty \\ \infty & 10 & 0 & 3 \\ 5 & 6 & \underbrace{2}_{6>(5-3)} & 0 \end{bmatrix}, P^1 = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 4 \\ 1 & 2 & 1 & 4 \end{bmatrix}$$

$$j=2: A^1 = \begin{bmatrix} 0 & 4 & -3 & \infty \\ -3 & 0 & -7 & \infty \\ \infty & 10 & 0 & 3 \\ 5 & 6 & 2 & 0 \end{bmatrix} \Rightarrow A^2 = \begin{bmatrix} 0 & 4 & -3 & \infty \\ -3 & 0 & -7 & \infty \\ \underbrace{7}_{\infty>(-3+10)} & 10 & 0 & 3 \\ \underbrace{3}_{5>(-3+6)} & 6 & \underbrace{-1}_{2>(-7+6)} & 0 \end{bmatrix}, P^2 = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 4 \\ 2 & 2 & 3 & 4 \\ 2 & 2 & 2 & 4 \end{bmatrix}$$

$$j=3: A^2 = \begin{bmatrix} 0 & 4 & -3 & \infty \\ -3 & 0 & -7 & \infty \\ 7 & 10 & 0 & 3 \\ 3 & 6 & -1 & 0 \end{bmatrix} \Rightarrow A^3 = \begin{bmatrix} 0 & 4 & -3 & \underbrace{0}_{\infty>(-3+3)} \\ -3 & 0 & -7 & \underbrace{-4}_{\infty>(3-7)} \\ 7 & 10 & 0 & 3 \\ 3 & 6 & -1 & 0 \end{bmatrix}, P^3 = \begin{bmatrix} 1 & 2 & 3 & 3 \\ 1 & 2 & 3 & 3 \\ 2 & 2 & 3 & 4 \\ 2 & 2 & 2 & 4 \end{bmatrix}$$

$$j=4: A^3 = \begin{bmatrix} 0 & 4 & -3 & 0 \\ -3 & 0 & -7 & -4 \\ 7 & 10 & 0 & 3 \\ 3 & 6 & -1 & 0 \end{bmatrix} \Rightarrow A^4 = \begin{bmatrix} 0 & 4 & -3 & \infty \\ -3 & 0 & -7 & -4 \\ \underbrace{6}_{7>(3+3)} & \underbrace{9}_{10>(3+6)} & 0 & 3 \\ 3 & 6 & -1 & 0 \end{bmatrix}, P^4 = \begin{bmatrix} 1 & 2 & 3 & 3 \\ 1 & 2 & 3 & 3 \\ 4 & 4 & 3 & 4 \\ 2 & 2 & 2 & 4 \end{bmatrix}$$

Pro iteraci j sledujeme matici A^{j-1}, a to pouze její j-tý řádek a j-tý sloupec (tedy takový kříž). Pro všechny prvky z A^{j-1} porovnáváme jejich hodnotu s průmětem na tento kříž (tedy se součtem s odpovídajícími prvky na stejném řádku a sloupci v kříži). Pokud je součet menší, v matici A^j zapíšeme součet a v matici P^j zapíšeme hodnotu j.

Pokud kdykoli na hlavní diagonále A vyjde něco jiného než 0, tak v grafu existuje záporná kružnice a tedy neexistuje cesta s minimální délkou.