

Dátové štruktúry a algoritmy

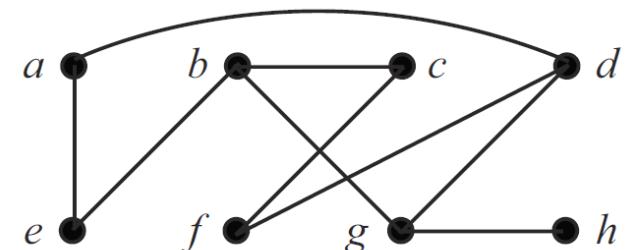
Grafové algoritmy

31. 10. 2017

zimný semester
2017/2018

Definícia

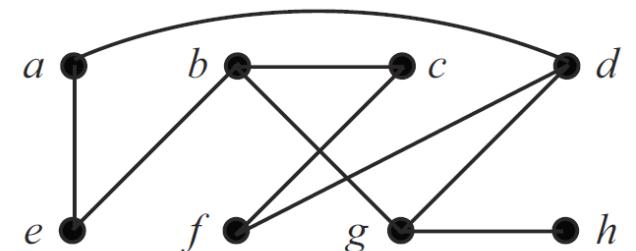
- **Graf G** je usporiadaná dvojica $G=(V, E)$,
 - V je neprázdna konečná množina (označenia vrcholov)
 - E je množina neusporiadaných dvojíc $\{u,v\}$ (koncové vrcholy hrany) takých, že $u, v \in V$.
- Prvky množiny V nazývame **vrcholy (vertices)** (počet vrcholov označujeme N)
- Prvky množiny E nazývame **hrany (edges)** (počet hrán označujeme M)
- Kreslíme ich ako diagramy:
 - bodky-krúžky (vrcholy) a čiary medzi nimi (hrany)



$$V = \{a, b, c, d, e, f, g, h\}$$
$$E = \{\{a, d\}, \{a, e\}, \{b, c\}, \{b, e\}, \{b, g\}, \{c, f\}, \{d, f\}, \{d, g\}, \{g, h\}\}$$

Základné pojmy – susednosť

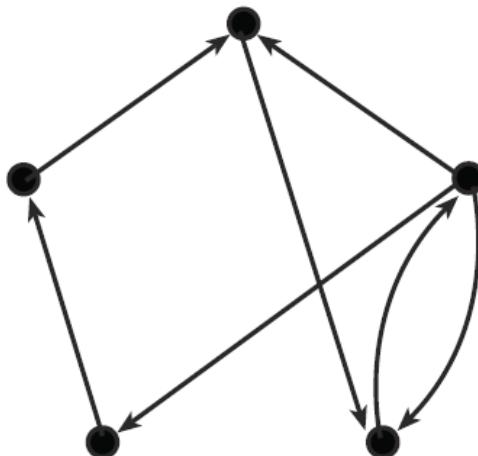
- **Vrchol v je incidentný s hranou e ak**
je jedným z vrcholov hrany e ($v \in e$).
- **Hrany e a f sú susedné (alebo prilahlé) ak**
majú spoločný jeden vrchol ($v \in e, f$).
- **Vrcholy u a v sú susedné (alebo prilahlé) ak**
existuje medzi nimi hrana ($\{u, v\} \in E$).
- **Označenie:**
 - $V(v)$, $\text{Neigh}(v)$ množina všetkých
susedných vrcholov s vrcholom v
 - $E(v)$ množina všetkých hrán
incidentných s vrcholom v



$$V = \{a, b, c, d, e, f, g, h\}$$
$$E = \{\{a, d\}, \{a, e\}, \{b, c\}, \{b, e\}, \{b, g\}, \{c, f\}, \{d, f\}, \{d, g\}, \{g, h\}\}$$

Základné pojmy – orientovaný graf (digraf)

- Ak hrany sú usporiadane dvojice dvojíc vŕcholov (u,v) dostávame **orientovaný graf**, tzv. digraf (directed graph)
- Každá hrana má orientáciu, (dovolený) smer prechodu (nazývame orientovaná hrana)



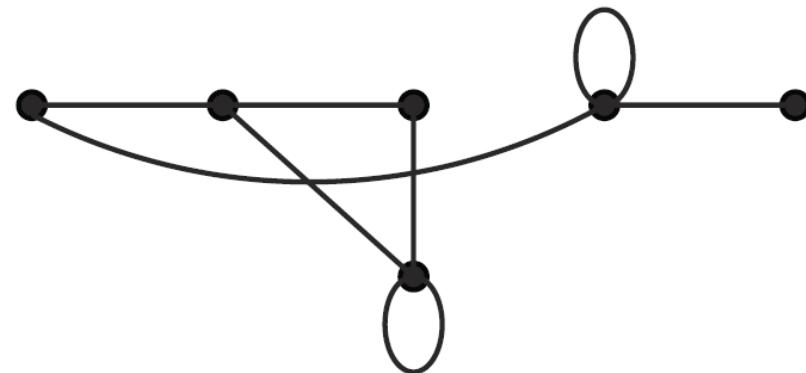
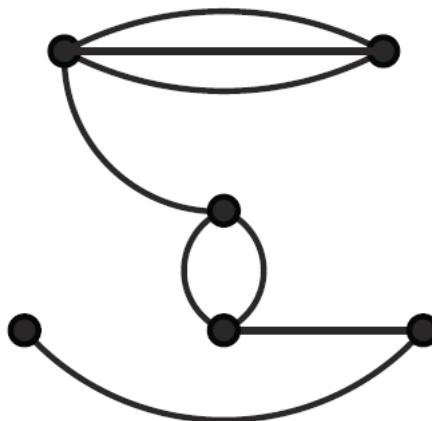
Základné pojmy – susednosť v digrafoch

- **Hrana $e=(u,v)$ vychádza z vrchola u , alebo aj vrchol u je začiatočný vrchol orientovanej hrany e .**
- **Hrana $e=(u,v)$ vchádza z vrchola v , alebo aj vrchol v je koncový vrchol orientovanej hrany e .**
- Orientovaná **hrana $e=(u,v)$ je incidentná s vrcholom x ak $u=x$ alebo $v=x$.**

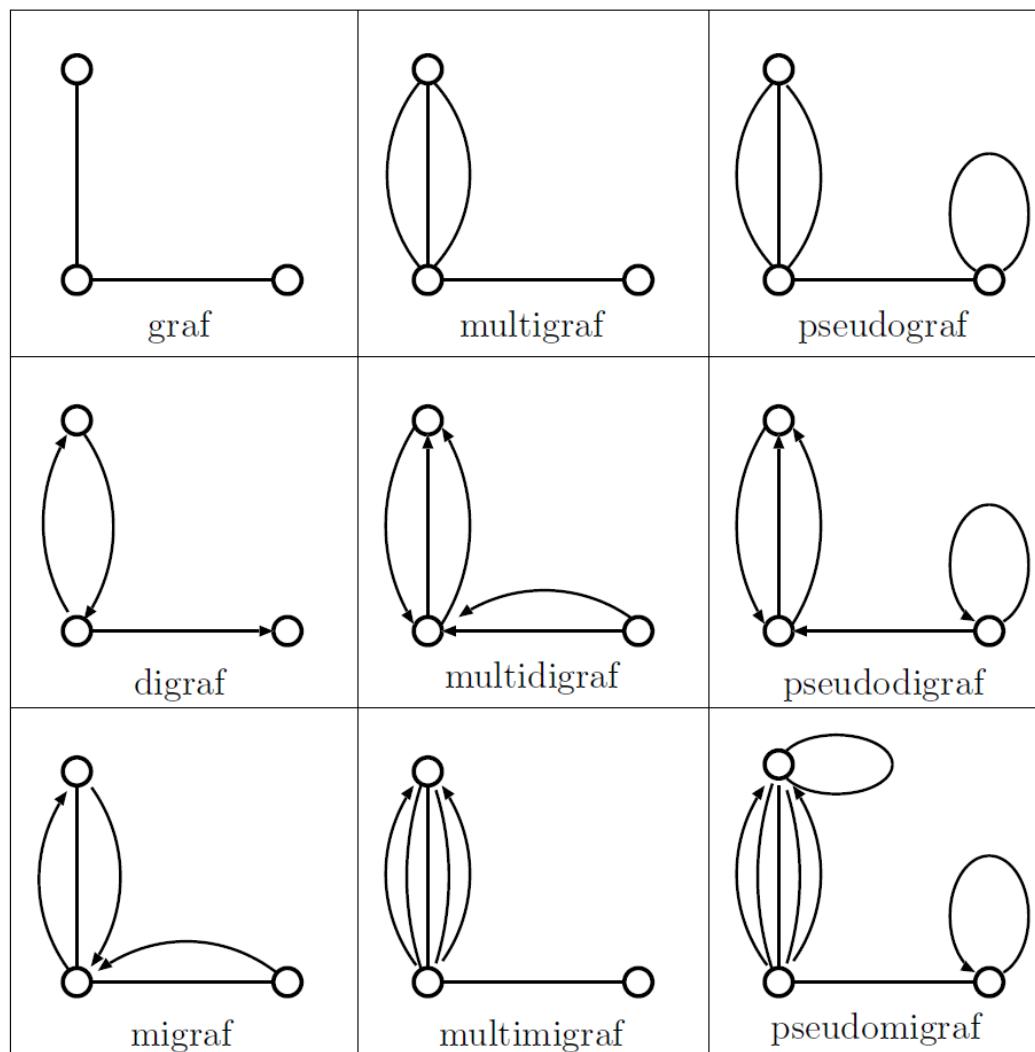
- Označenie:
 - $E^+(v)$ množina všetkých hrán vychádzajúcich z vrchola v
 - $E^-(v)$ množina všetkých hrán vchádzajúcich do vrchola v
 - $E(v) = E^+(v) \cup E^-(v)$
- Označenie:
 - $V^+(v)$ množina koncových vrcholov hrán $E^+(v)$
 - $V^-(v)$ množina začiatočných vrcholov hrán $E^-(v)$
 - $V(v) = V^+(v) \cup V^-(v)$

Základné pojmy – násobné hrany, slučky

- Ak umožníme opakujúce sa násobné hrany (množina hrán E bude multimnožina), dostaneme **multigraf**
- Ak umožníme slučky (hrana spájajúca vrchol zo sebou), dostaneme **pseudograf**



Typy grafov podľa Plesníka (1983)

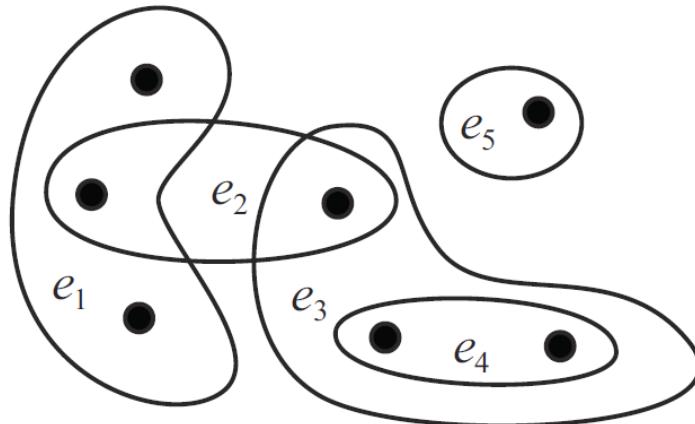


Alternatívna definícia

- **Graf G** je usporiadaná trojica $G=(V, E, f)$,
 - V je neprázdna konečná množina vrcholov
 - E je konečná množina hrán
 - f je **incidenčné zobrazenie** $f: E \rightarrow V^2$
 $f(e) = (u, v)$ potom u je začiatočný vrchol a v je koncový vrchol
- Neorientovaný graf:
Ak pre $\forall e: f(e) = (u, v)$ platí, že $\exists e': f(e') = (v, u)$ a dvojicu hrán $\{e, e'\}$ považujeme za jedinú neorientovanú hranu
- Násobné hrany: $f(e_1) = f(e_2) = (u, v)$
- Slučky: $f(e) = (u, u)$

Pokročilejšie grafy

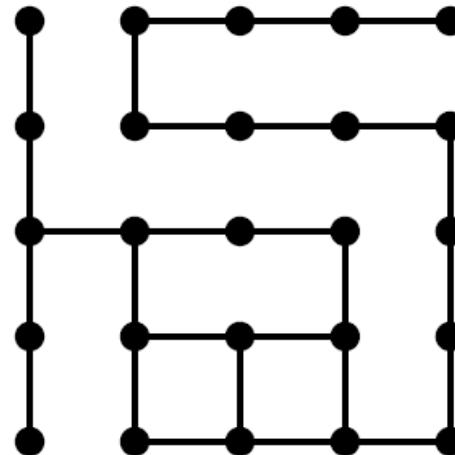
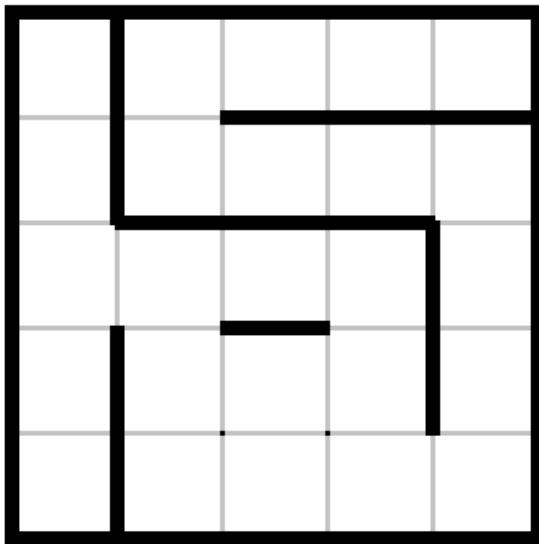
- Ak umožníme, aby hrany boli ľubovoľné podmnožiny vrcholov, dostaneme **hypergrafy**



- Ak umožníme, aby množiny V a E boli nekonečné dostaneme **nekonečné grafy (infinite graphs)**

Čo môžeme reprezentovať grafom?

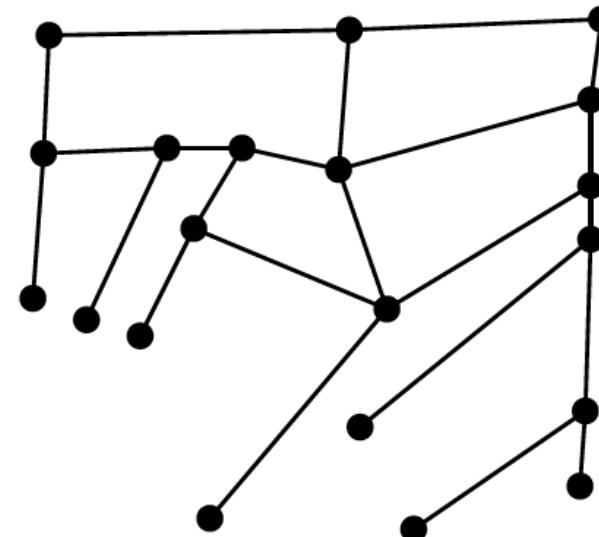
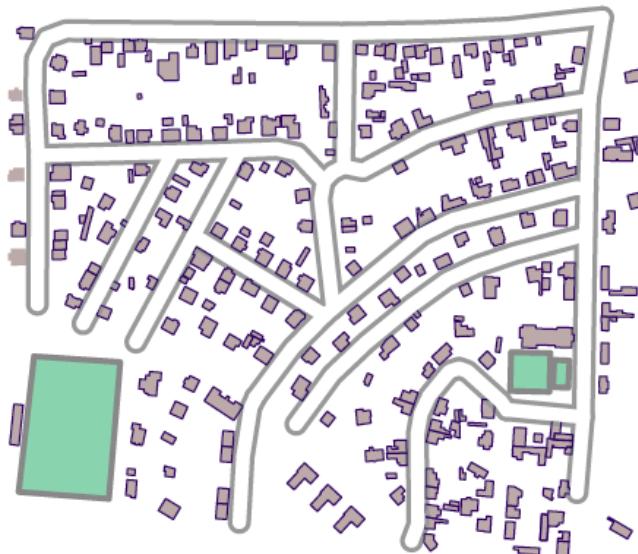
- Mapy (bludisko s miestnosťami prepojenými chodbami)



- Bludisko 5x5 ... 25 vrcholov

Čo môžeme reprezentovať grafom? (2)

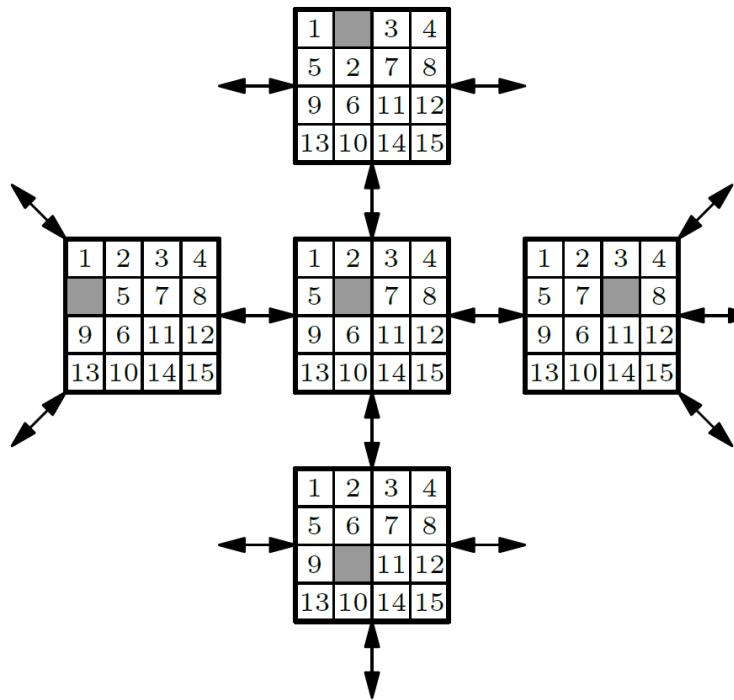
- Mapy (križovatky prepojené ulicami)



Čo môžeme reprezentovať grafom? (3)

- Hry (stavy v hre prepojené t'ahmi v hre)

- Napr. Hra 15



- Kol'ko má vrcholov a hrán?

vrcholov: $16! / 2 = 10\ 461\ 394\ 944\ 000$



Čo môžeme reprezentovať grafom? (4)

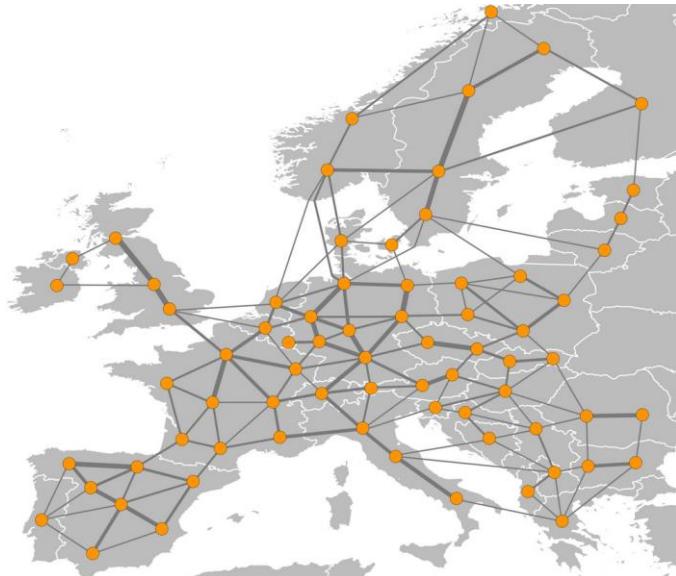
- Dopravné siete
 - Distribučné siete



Distribúcia elektriny v EÚ

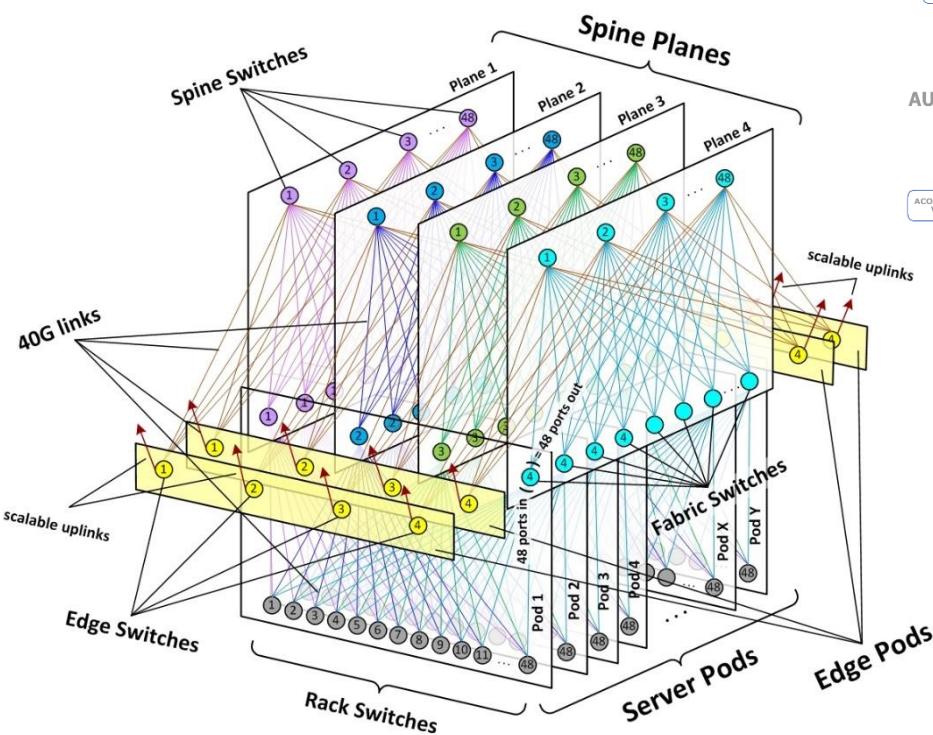


Letecké spojenia v U.S.A.



Čo môžeme reprezentovať grafom? (5)

■ Dátové siete



Zapojenie v dátovom centre Facebook-u



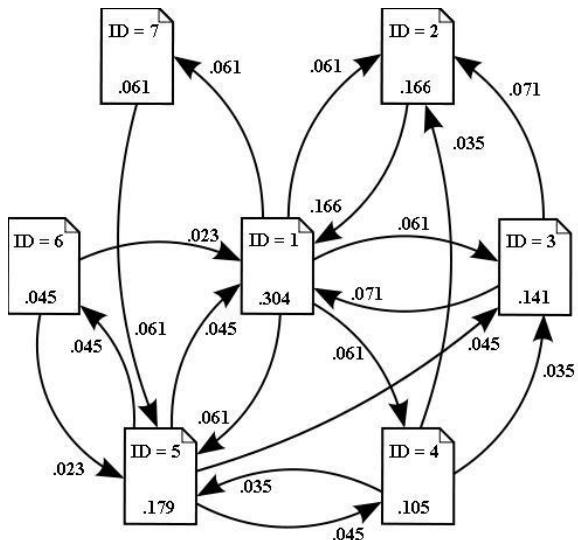
SANET - Slovak Academic Network



Infraštruktúra SANET-u

Čo môžeme reprezentovať grafom? (6)

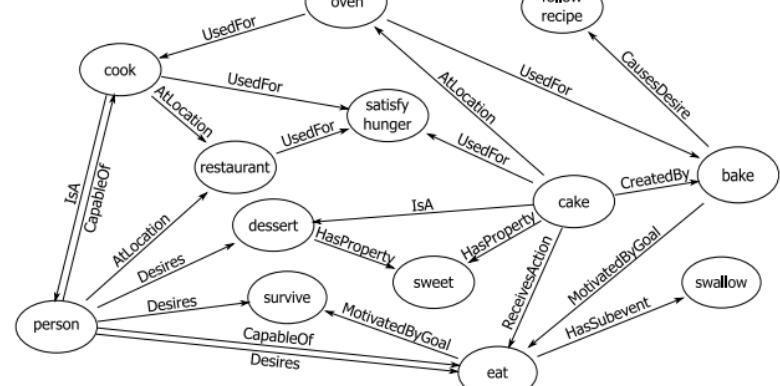
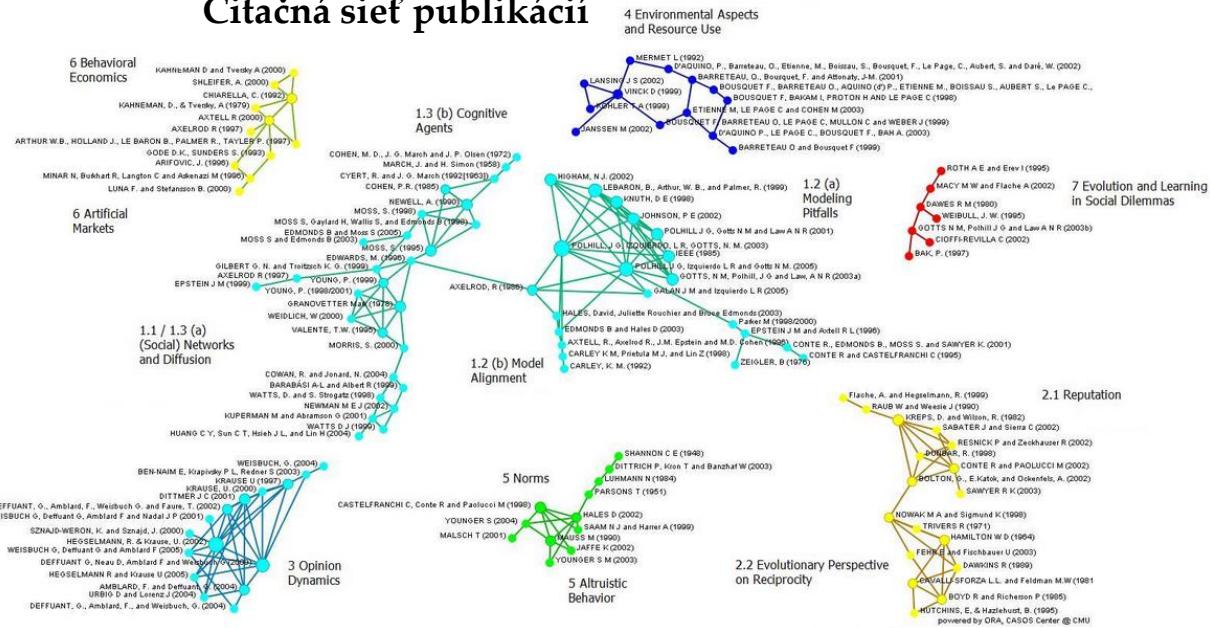
- Informačné siete
- Sémantické siete



$$PR(u) = \sum_{v \in B_u} \frac{PR(v)}{L(v)}$$

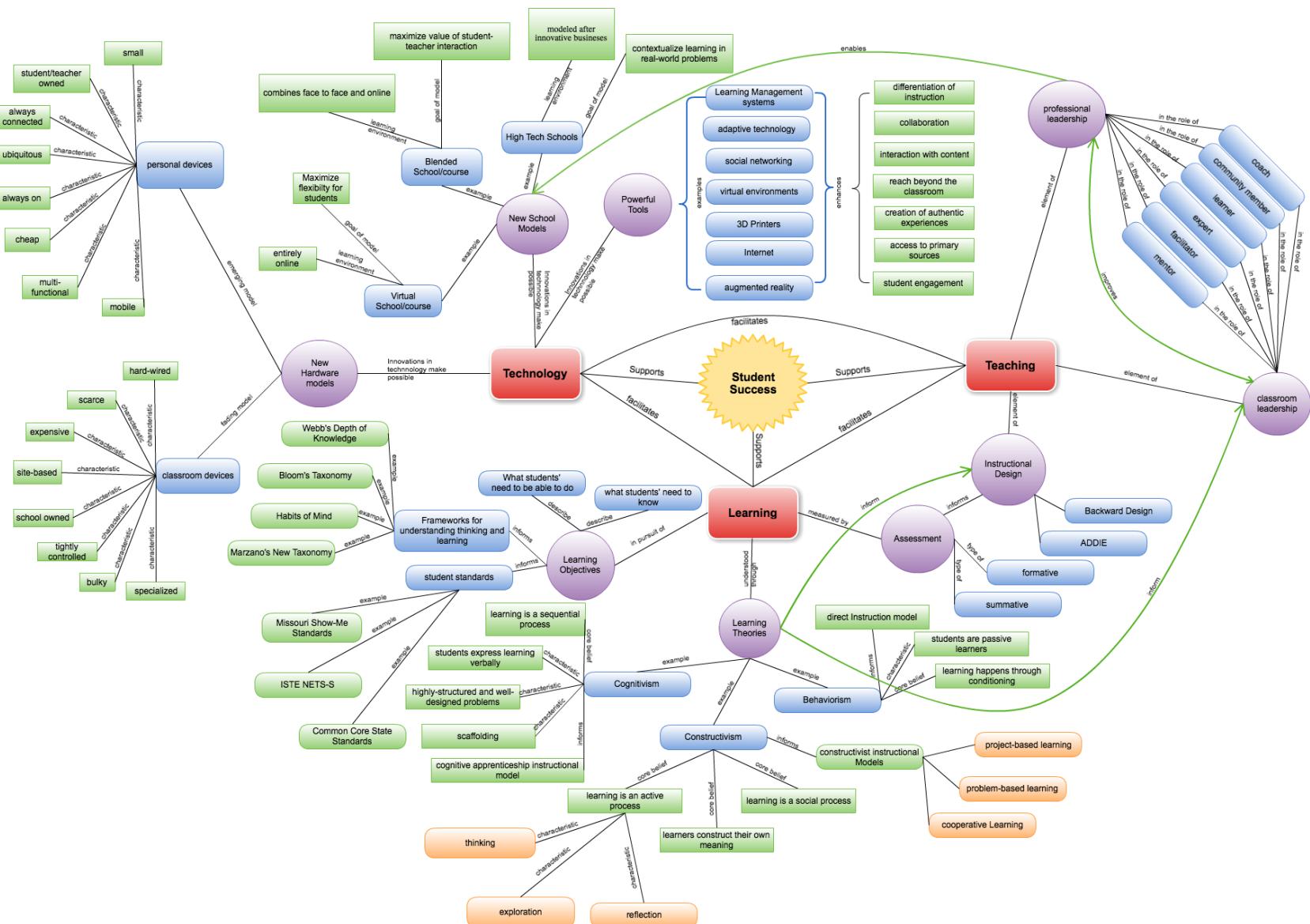
$$PR(A) = \frac{1-d}{N} + d \left(\frac{PR(B)}{L(B)} + \frac{PR(C)}{L(C)} + \frac{PR(D)}{L(D)} + \dots \right)$$

Citačná sieť publikácií



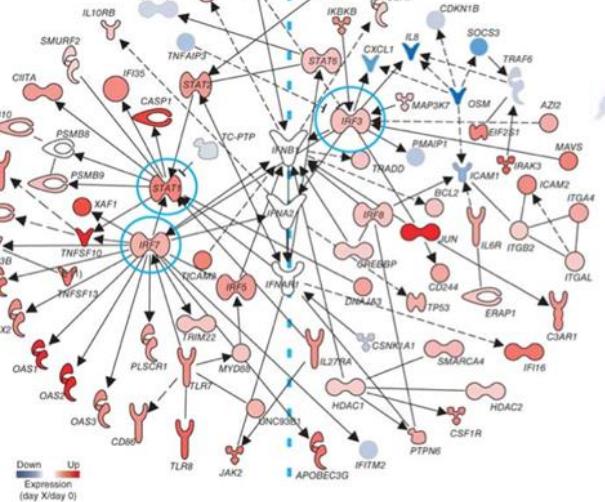
Konceptová mapa stravovania

Čo môžeme reprezentovať grafom? (7)



Čo môžeme reprezentovať grafom? (8)

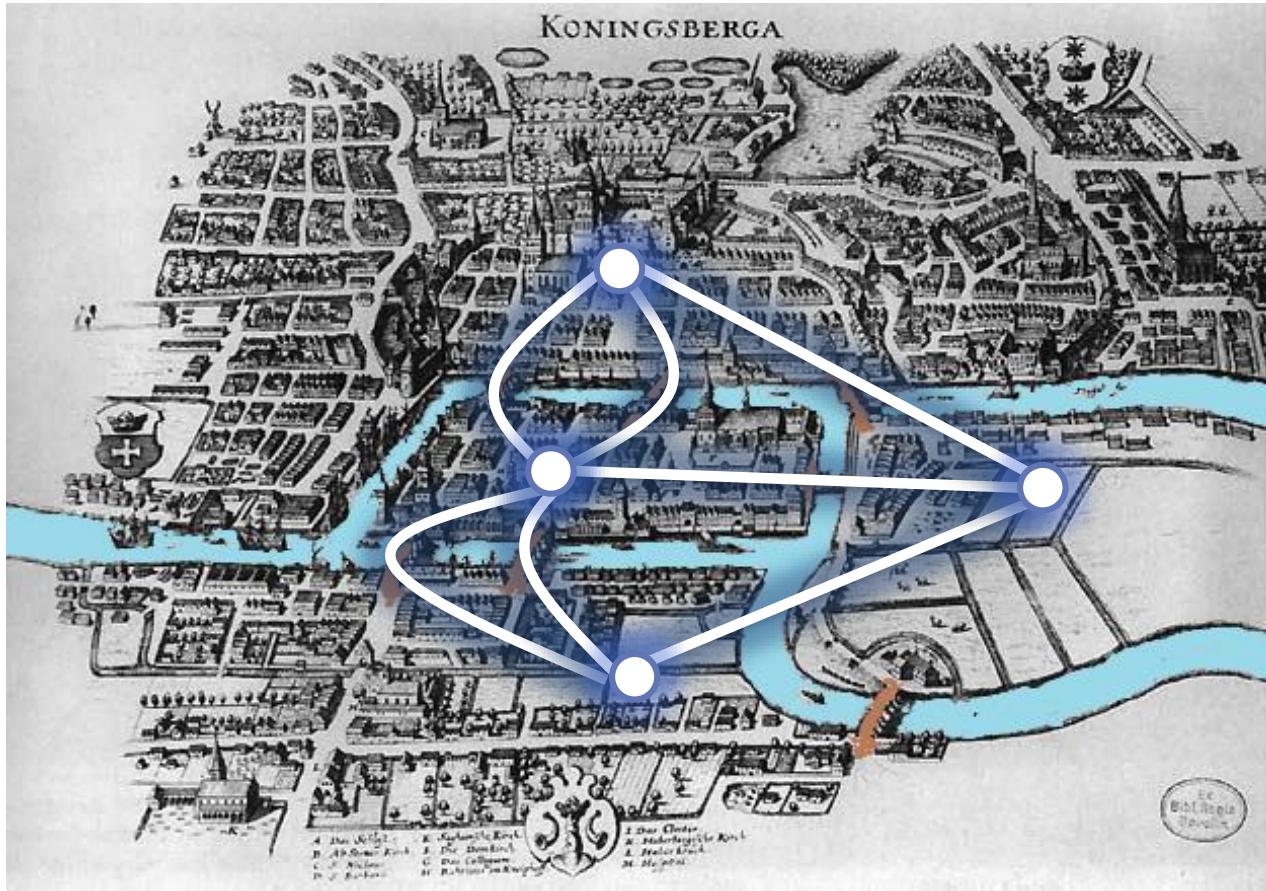
- Sociálne siete
- Biologické siete



Odpoveď imunitného systému
na očkovanie proti chrípke

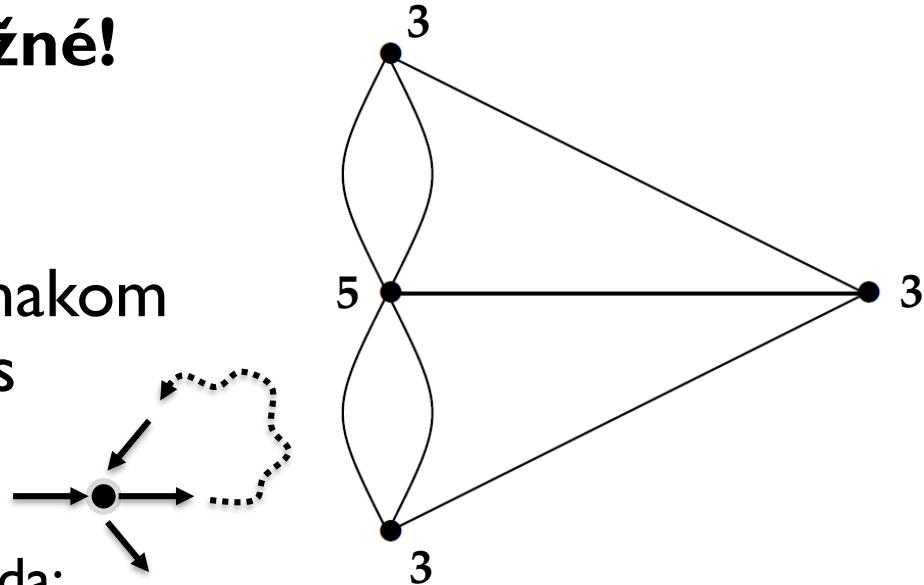
Teória grafov – 1735 – Kalingradské mosty

- Nájst' prechádzku mestom, ktorá prejde po každom moste práve raz; začiatok a koniec môžu byť rôzne.



Teória grafov – 1736 – Leonhard Euler

- Nájst' prechádzku mestom, ktorá prejde po každom moste práve raz; začiatok a koniec môžu byť rôzne.
- **Riešenie = Nie je to možné!**
- **Prečo?**
- Uvažujme taký prechod, že začneme aj skončíme v rovnakom vrchole – tzv. Eulerov cyklus
 - Ked' prechádzame nejakým vrcholom, tak do neho aj vchádzame aj vychádzame, teda:
 - **Počet hrán pri každom vrchole musí byť párny!**
 - Platí to aj naopak: **Ak je počet hrán pri každom vrchole párny tak, existuje Eulerov cyklus!** (stačí nadpájať akékoľvek cykly z neneavštívených hrán do jedného veľkého cyklu)



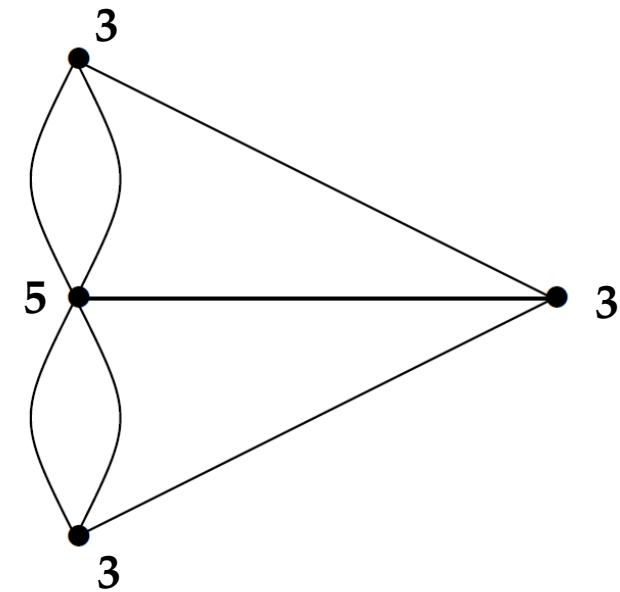
Teória grafov – 1736 – Leonhard Euler

- Uvažujme teraz možnosť, že prechod nebude začínať a končiť v jednom vrchole tzv. **Eulerov t'ah** (Euler tour)
- Je možné takýto t'ah zstrojíť?

Súvislý graf obsahuje Eulerov t'ah
práve vtedy, keď práve dva vrcholy
majú pri sebe nepárny počet hrán

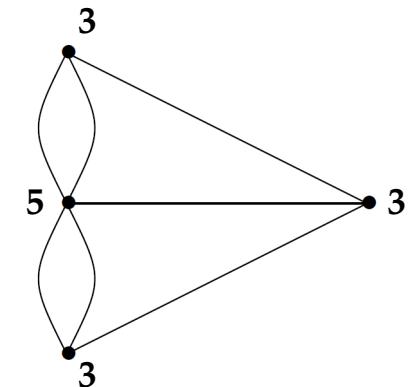
⇒ pridáme hranu medzi tieto dva nepárne
vrcholy, z t'ahu vznikne cyklus, pričom vieme,
že pri vrcholoch bude len párny počet hrán,
odobratím hrany zostanú práve dva nepárne.

⇐ ak máme práve dva nepárne, pridáme medzi
nimi hranu, nájdeme Eulerov cyklus (vieme, že
existuje), odobratím pridanéj hrany vznikne
hľadaný Eulerov t'ah.



Teória grafov – stupeň vrcholu

- **Stupeň vrcholu $\deg(v)$** nazveme počet kol'ko krát sa vrchol v nachádza ako koncový vrchol nejakej hrany – **počet hrán incidentných s vrcholom v**
- V orientovaných grafoch rozlišujeme
 - **výstupný stupeň $\deg^+(v) = |E^+(v)|$**
 - **vstupný stupeň $\deg^-(v) = |E^-(v)|$**



Súčet stupňov všetkých vrcholov sa rovná dvojnásobku počtu hrán.

$$\sum_{v \in V} \deg(v) = 2 \cdot |E|$$

Každá hrana je incidentná s dvomi vrcholmi, a do súčtu prispieva počtom 2.

Dôsledok: Počet vrcholov s nepárnym stupňom je párny.

Teória grafov – stupeň vrcholu

Súčet stupňov všetkých vrcholov grafu je ohraničený $N(N-1)$.

$$\sum_{v \in V} \deg(v) \leq N(N - 1)$$

Počet hrán (jednoduchého) grafu je nanajvýš počet všetkých kombinácií dvoch prvkov z N prvkov: $|E| \leq \binom{N}{2} = \frac{N(N-1)}{2}$. Využitím predchádzajúcej vety $\sum_{v \in V} \deg(v) = 2 \cdot |E|$ vyplýva tvrdenie $\sum_{v \in V} \deg(v) \leq N(N - 1)$.

Graf $G=(V,E)$, v ktorom $|V| \geq 2$, obsahuje aspoň dva vrcholy rovnakého stupňa.

Stupeň vrcholov v G nadobúdajú hodnoty $0, 1, \dots, N-1$, ale nemôže byť súčasne 0 aj $N-1$, lebo vrchol so stupňom $N-1$ musí mať všetky ostatné vrcholy susedné. Teda N vrcholov v grafe môže mať stupne s nanajvýš $N-1$ rôznymi hodnotami, z čoho vyplýva, že aspoň dva vrcholy musia mať rovnakú hodnotu (stupňa).

Teória grafov – postupnosť stupňov vrcholov

- Postupnosť stupňov vrcholov (valenčná postupnosť)
 $p = (\deg(v_1), \deg(v_2), \dots, \deg(v_N))$ je **grafová**, ak existuje graf s vrcholmi zodpovedajúcich stupňov
- Poradie nie je veľmi dôležité, zvyčajne ju uvádzame v neklesajúcom poradí: $\deg(v_1) \geq \deg(v_2) \geq \dots \geq \deg(v_N)$
- **Erdős–Gallai 1960**

Postupnosť $\deg(v_1) \geq \deg(v_2) \geq \dots \geq \deg(v_N)$ je grafová práve vtedy keď súčet prvkov je párny, a pre všetky $k \in \{1, 2, \dots, N\}$ platí

$$\sum_{i=1}^k \deg(v_i) \leq k(k - 1) + \sum_{i=k+1}^n \min(\deg(v_i), k)$$

$$\begin{array}{ccc} \text{susednosti vrchol-hrana} \\ \text{medzi k vrcholmi} \\ \text{s najvyšším stupňom} & \leq & \text{susednosti} \\ & & \text{medzi vrcholmi} \\ & & \text{„vysokého“ stupňa} & + & \text{horný odhad počtu hrán,} \\ & & & & \text{ktorých jeden vrchol má} \\ & & & & \text{„vysoký“ stupeň} \end{array}$$

Teória grafov – postupnosť stupňov vrcholov

- Pre danú nerastúcu postupnosť stupňov, zostrojiť graf
- **Havel–Hakimi algoritmus:**
Postupnosť $(\deg(v_1), \deg(v_2), \dots, \deg(v_N))$ pre $\deg(v_1) \geq \deg(v_2) \geq \dots \geq \deg(v_N)$ je grafová, keď postupnosť $(\deg(v_1)-1, \dots, \deg(v_{\deg(v_1)+1})-1, \deg(v_{\deg(v_1)+2}), \dots, \deg(v_N))$ neobsahuje záporné čísla a je grafová.

Postup: Vytvárame graf pridávaním vrcholov po jednom.
V jednom kroku pridáme vrchol s najvyšším stupňom a spojíme ho hranami s vrcholmi s najvyššími stupňami.

Havel, Václav (1955), "[A remark on the existence of finite graphs](#)", Časopis pro pěstování matematiky, 80: 477–480

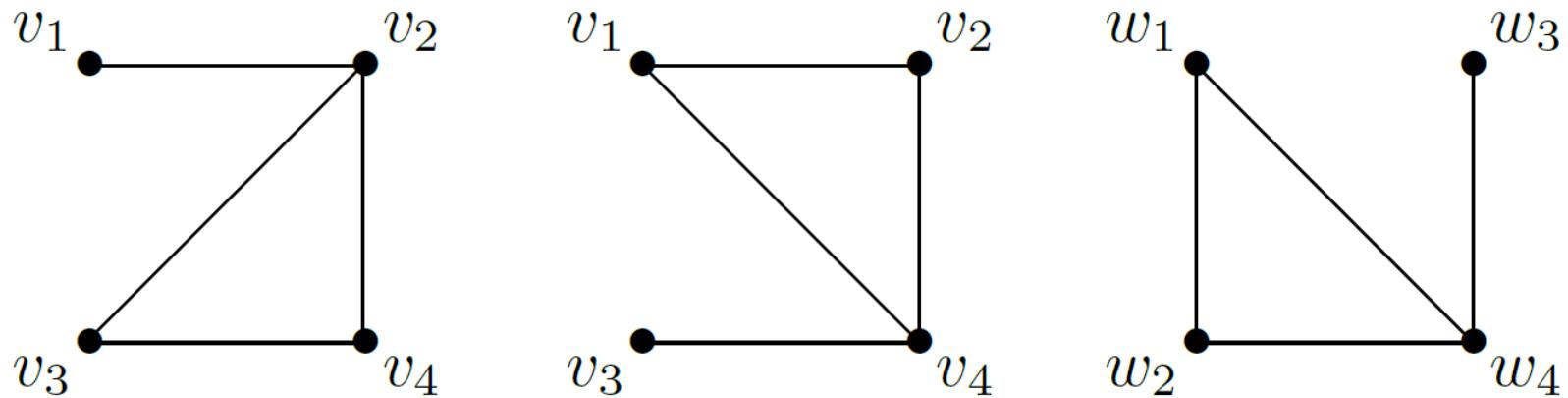
Čo to znamená, že grafy sú rovnaké?

- Uvažujme grafy:

$$G_1 = (\{v_1, v_2, v_3, v_4\}, \{\{v_1, v_2\}, \{v_2, v_3\}, \{v_3, v_4\}, \{v_2, v_4\}\})$$

$$G_2 = (\{v_1, v_2, v_3, v_4\}, \{\{v_1, v_2\}, \{v_1, v_4\}, \{v_3, v_4\}, \{v_2, v_4\}\})$$

$$G_3 = (\{w_1, w_2, w_3, w_4\}, \{\{w_1, w_2\}, \{w_1, w_4\}, \{w_3, w_4\}, \{w_2, w_4\}\})$$



- Sú to „trojuholníky“ s príveskom (pendant vertex)

Teória grafov – izomorfizmus grafov

- **Graf $\mathbf{G}=(V,E)$ je izomorfný s grafom $\mathbf{G}'=(V',E')$ ak existuje taká bijekcia (vzájomné jednoznačné zobrazenie) vrcholov $f:V \rightarrow V'$ že pre každú dvojicu vrcholov $u,v \in V$: $\{u,v\} \in E$ práve vtedy ked' $\{f(u),f(v)\} \in E'$ zobrazenie f nazývame **izomorfizmus grafov \mathbf{G} a \mathbf{G}'****

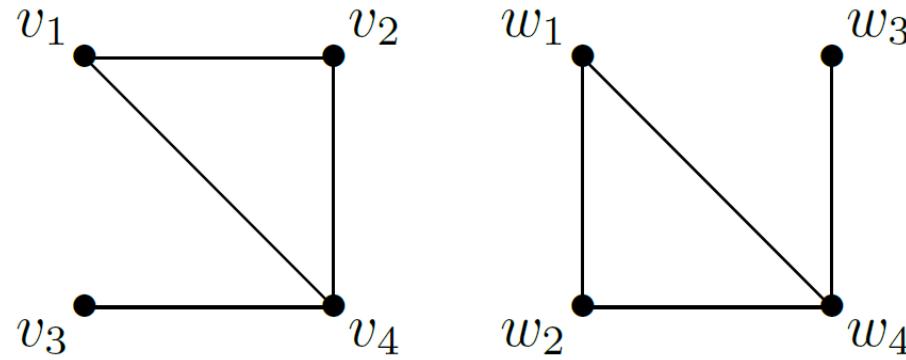
- **Napr.**

$$f(v_1) = w_2$$

$$f(v_2) = w_1$$

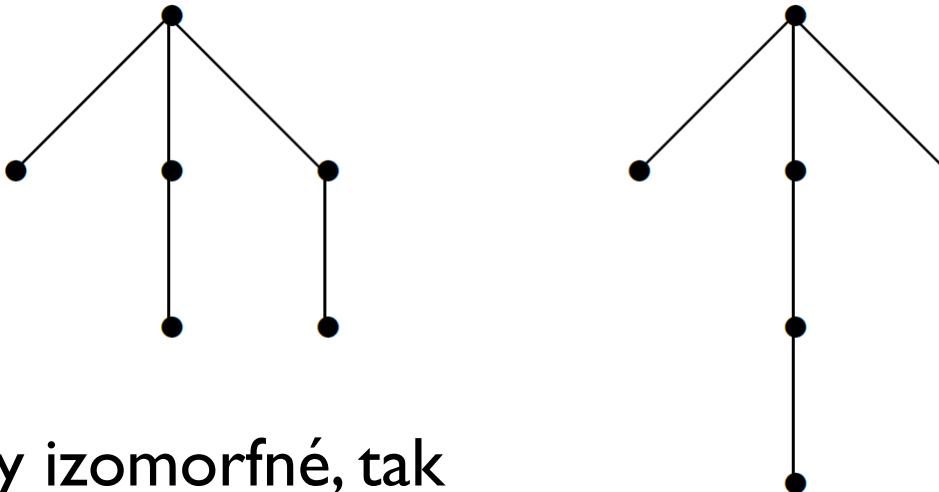
$$f(v_3) = w_3$$

$$f(v_4) = w_4$$



Teória grafov – izomorfizmus grafov (2)

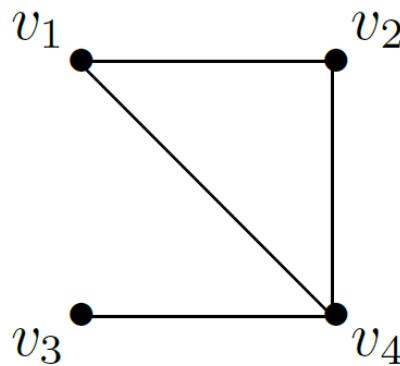
- Ked' G a G' sú izomorfné, tak ich postupnosti stupňov vrcholov sú rovnaké, platí to aj opačne?
- Opačne to neplatí: napr. 3,2,2,1,1,1



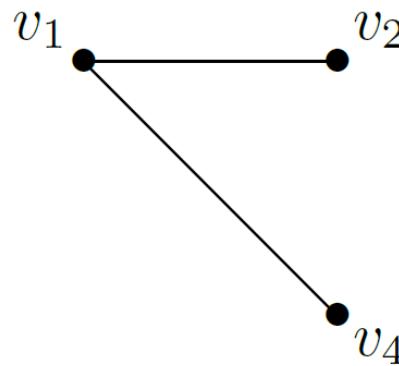
- Ked' sú grafy izomorfné, tak zdieľajú „grafové vlastnosti“ (invarianty izomorfizmu), iné vlastnosti nemusia byť zachované, napr. „počet kolko krát sa hrany pretnú keď graf nakreslíme v rovine“

Teória grafov – podgraf, indukovaný podgraf

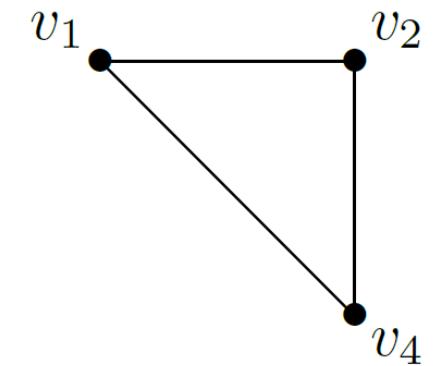
- Graf $H=(W,F)$ je **podgraf** grafu $G=(V,E)$ ak $W \subseteq V$ a $F \subseteq E$
- H nazývame **indukovaný podgraf**, ked' F obsahuje všetky hrany v E , ktoré majú koncové vrcholy v množine W



graf G



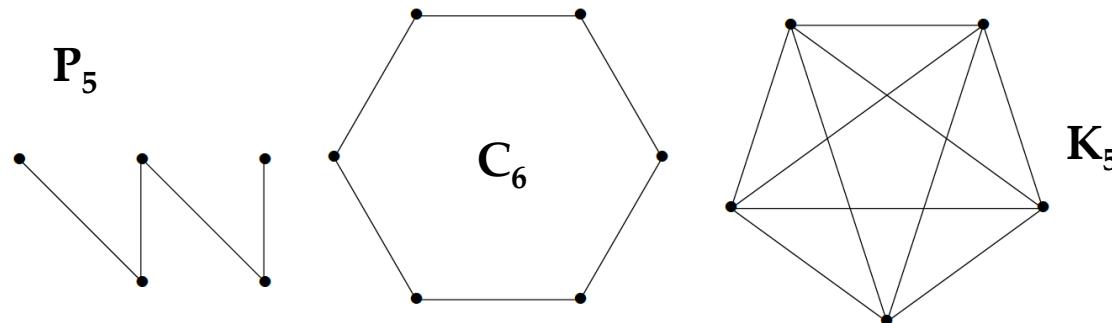
podgraf grafu G



indukovaný
podgraf grafu G

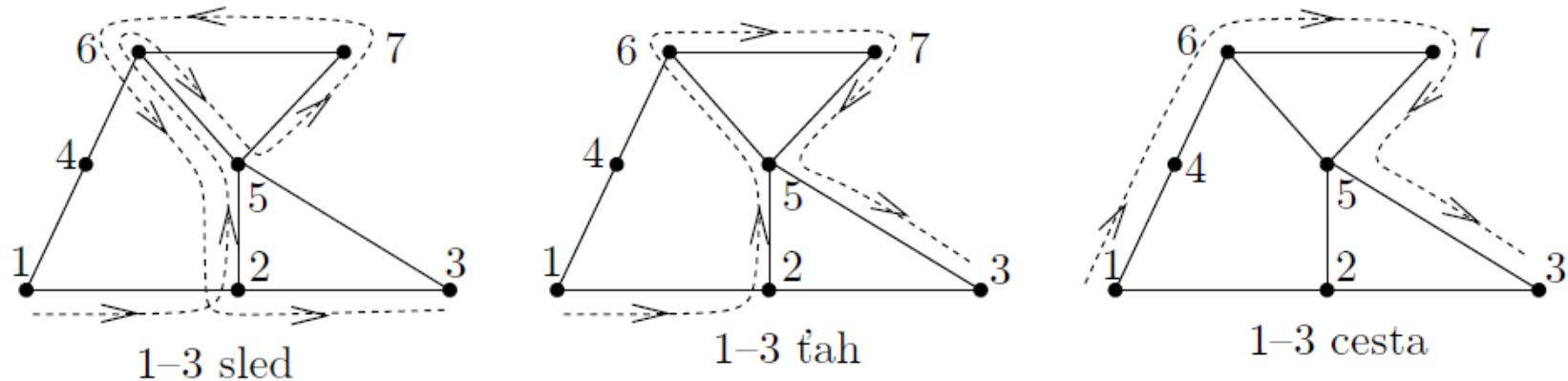
Teória grafov – jednoduché typy grafov

- **Ret'az (chain)** alebo niekedy **cesta (path)** je graf z vrcholov v_1, v_2, \dots, v_N s hranami $\{v_i, v_{i+1}\}$ pre $1 \leq i \leq N-1$ a žiadnymi inými, označujeme P_N
- **Kružnica (cycle)** je graf z vrcholov v_1, v_2, \dots, v_N s hranami $\{v_i, v_{1+(i \bmod N)}\}$ pre $1 \leq i \leq N$ a žiadnymi inými, označujeme C_N
- **Úplný (complete) graf** s vrcholmi v_1, v_2, \dots, v_N má spojený každý vrchol s každým, označujeme K_N



Teória grafov – sled, t'ah, cesta

- **Sled** (walk) v grafe je ľubovoľná striedajúca sa postupnosť vrcholov a hrán grafu: $v_1, (v_1, v_2), v_2, (v_2, v_3), v_3, \dots, (v_{k-1}, v_k), v_k$
- **Ťah** (tour) je taký sled, v ktorom sa žiadna hrana neopakuje
- **Cesta** (path) je taký sled, v ktorom sa žiadnen vrchol neopakuje



1–3 sled: $(1, \{1, 2\}, 2, \{2, 5\}, 5, \{5, 6\}, 6, \{6, 5\}, 5, \{5, 7\}, 7, \{7, 6\}, 6, \{6, 5\}, 5, \{5, 2\}, 2, \{2, 3\}, 3)$.

1–3 ťah: $(1, \{1, 2\}, 2, \{2, 5\}, 5, \{5, 6\}, 6, \{6, 7\}, 7, \{7, 5\}, 5, \{5, 3\}, 3)$.

1–3 cesta: $(1, \{1, 4\}, 4, \{4, 6\}, 6, \{6, 7\}, 7, \{7, 5\}, 5, \{5, 3\}, 3)$.

Teória grafov – cyklus, súvislosť

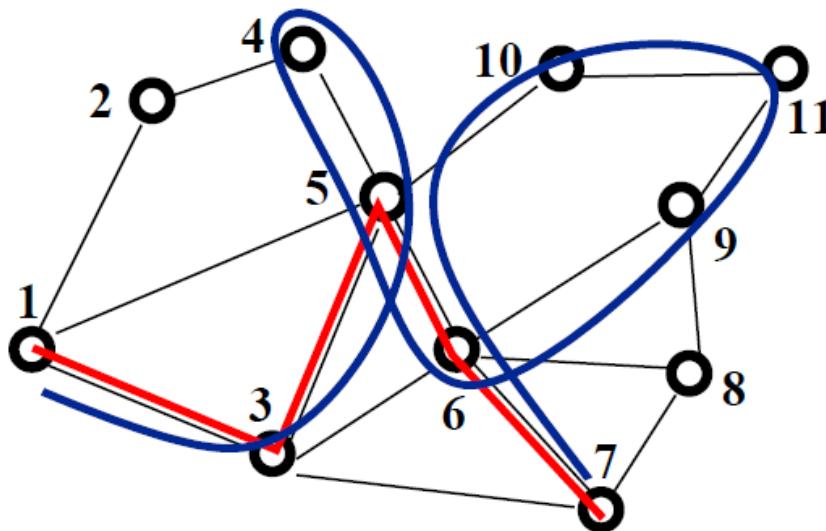
- Sled alebo t'ah je **uzavretý**, ak $v_1 = v_k$, inak je **otvorený**.
- Uzavretý t'ah, v ktorom sa okrem prvého a posledného vrchola žiadnen vrchol nevyskytuje viac než raz nazývame **cyklus**.
- Graf $G=(V,E)$ je **súvislý** (connected) ak pre každú dvojicu vrcholov $u,v \in V$ existuje $u-v$ cesta.
Inak je graf G **nesúvislý** (disconnected).
- **Acyklický graf**, je taký graf, ktorý neobsahuje (ako podgraf) kružnicu.
 - Dôsledok: Acyklický graf neobsahuje cyklus.

Teória grafov – dosiahnutelnosť, sled → cesta

- Vrchol y je **dosiahnuteľný** z vrcholu x v grafe G , ak existuje $x-y$ sled v grafe G .

Ak existuje $x-y$ sled, potom existuje aj $x-y$ cesta.

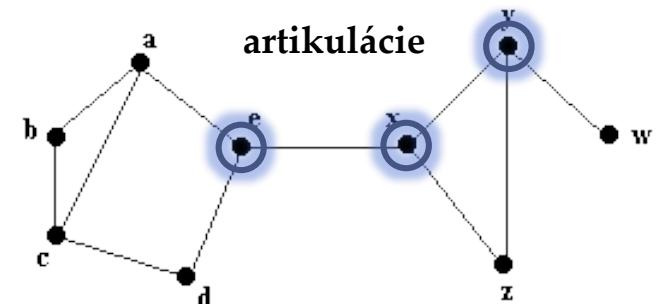
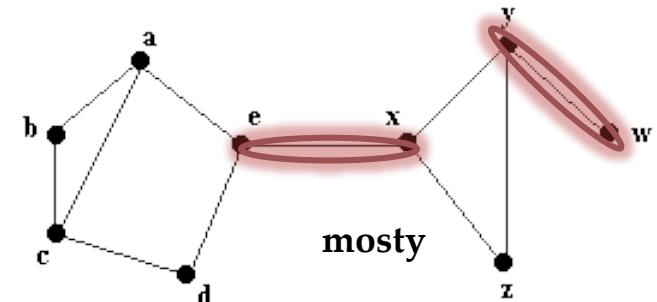
Zo sledu odstráime cykly:



Dôsledok: Vrchol y je dosiahnuteľný z vrcholu x , ak existuje $x-y$ cesta.

Teória grafov – komponent, most, artikulácia

- **Komponent grafu** G je jeho ľubovoľný maximálny súvislý podgraf.
- **Most (bridge)** je taká hrana, po odstránení ktorej vzrastie počet komponentov.
- **Artikulácia (cutpoint)** je taký vrchol, po odstránení ktorého (spolu s incidentnými hranami) vzrastie počet komponentov.



Teória grafov – Strom, les

- **Strom** je súvislý acyklický graf. **Les** je acyklický graf.

Strom, ktorý má aspoň dva vrcholy,
obsahuje aspoň dva vrcholy stupňa 1.

Všimneme si najdlhšiu cestu v strome. Je možné ju predĺžiť?
Aký stupeň majú krajné vrcholy na tejto ceste?

- Nasledovné tvrdenia o stromoch sú ekvivalentné:
 - a) Graf $G = (V, E)$ je strom.
 - b) V grafe $G = (V, E)$ existuje pre každé $u, v \in V$ jediná $u-v$ cesta.
 - c) Graf $G = (V, E)$ je súvislý a každá hrana $e \in E$ je mostom.
 - d) Graf $G = (V, E)$ je súvislý a $|E| = |V|-1$.
 - e) V grafe $G = (V, E)$ platí $|E| = |V|-1$ a je acyklický.
- Dôkaz ako cvičenie :) ... (dokazujte $a \Rightarrow b \Rightarrow c \Rightarrow d \Rightarrow e \Rightarrow a$)

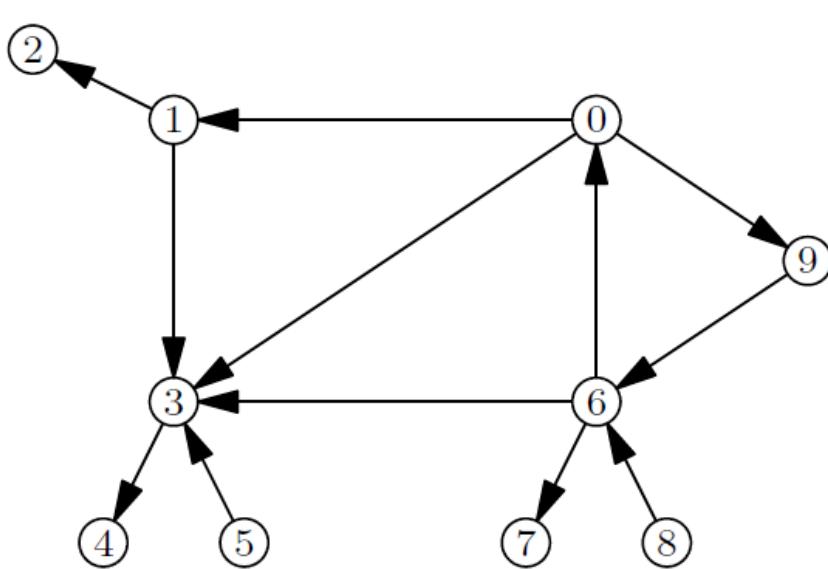
Reprezentácia grafov v počítači

- Vhodný spôsob reprezentácie závisí od problému, ktorý potrebujeme vyriešiť
- **Zoznam hrán** – všetky hrany si pamätáme ako zoznam dvojíc vrcholov.
 - Jednoduchá implementácia, pamäťovo efektívne
 - Neefektívne vyhľadávanie hrán pre konkrétny vrchol
- **Matica susednosti** – hrany grafu si pamätáme v matici $N \times N$, prvok i,j obsahuje 1 ak graf obsahuje hranu (i,j) , inak obsahuje hodnotu 0.
 - Jednoduchá implementácia, pamäťovo náročné pre riedke grafy, vyhľadanie hrán pre vrchol preskúma jeden riadok

Reprezentácia grafov v počítači (2)

- **Zoznamy susednosti** – pre každý vrchol si pamäťame zoznam z neho odchádzajúcich hrán
 - Náročnejšia implementácia, časovo aj pamäťovo efektívne
 - Efektívna implementácia pomocou dvoch polí: 1) prefixové sumy stupňov vrcholov, 2) koncové vrcholy hrán
- **Implicitne reprezentované grafy** – zadanie problému si niekedy nevyžaduje, aby sme si graf explicitne pamätali, pričom môžeme generovať graf podľa dostupných údajov z úlohy
 - Napr. graf vzdušných vzdialenosí miest, ak vieme ich GPS súradnice, graf je kompletný (obsahuje všetky hrany), pričom dĺžku hrany určíme výpočtom zo súradníc

Ukážka reprezentácie



matica susednosti

0	1	2	3	4	5	6	7	8	9
0	0	1	0	1	0	0	0	0	1
1	0	0	1	1	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	0	0	0	0	1	0	0	0	0
4	0	0	0	0	0	0	0	0	0
5	0	0	0	0	1	0	0	0	0
6	1	0	0	1	0	0	1	0	0
7	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	1	0	0	0
9	0	0	0	0	0	0	0	1	0

zoznamy susednosti

0: 1, 3, 9

5:

1: 2, 3

6: 0, 3, 7

2:

7:

3: 4

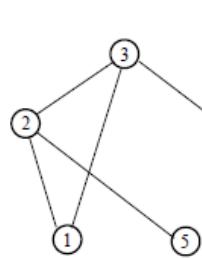
8: 6

4:

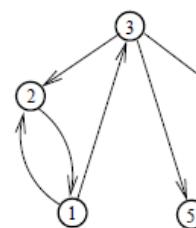
9: 6

Ukážka reprezentácie (2)

- **Neorientované vs orientované grafy**



	1	2	3	4	5
1	-	1	1	-	-
2	1	-	1	-	1
3	1	1	-	1	-
4	-	-	1	-	-
5	-	1	-	-	-



	1	2	3	4	5
1	-	1	1	-	-
2	1	-	-	-	-
3	-	1	-	1	1
4	-	-	-	-	-
5	-	-	-	-	-

- Neorientovaný graf je reprezentovaný symetrickou maticou

Porovnanie reprezentácií grafov v počítači

Efektívnosť	Zoznamy hrán	Matica susednosti	Zoznamy susednosti
Pamäťové nároky	M	N^2	$2*M$
Sú dva vrcholy susedné?	M	1	\deg_{\max}
Všetky susedné vrcholy pre daný vrchol	M	N	\deg_{\max}
Pridaj hranu do grafu	1	2	2
Odstráň hranu z grafu	M	2	$2*\deg_{\max}$

Prehľadávanie grafov

- Prehľadávanie grafu je úloha, v ktorej chceme navštíviť každý vrchol (príp. každú hranu)
- Schéma: zvyčajne z počiatočného vrcholu „vyšleme správu“ do ďalších vrcholov, ktoré ju „pošlú ďalej“, až kým nie je možné ďalej poslat’.
- Základné algoritmy sú:
 - Tarryho prieskum
 - Prehľadávanie do hĺbky
 - Prehľadávanie do šírky

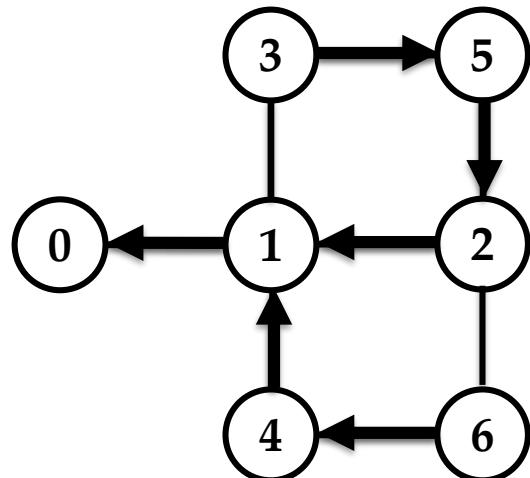
Prehľadávanie grafu Tarryho prieskumom

- Označenia:
 - Označme **rodiča** vrcholu v ten vrchol, z ktorého prišla správa do vrcholu prvý krát.
 - **Koreň** je počiatočný vrchol, z ktorého začíname prehľadávanie, a nemá rodiča.
 - Ostatných susedov vrcholu v nazývame len **sused**.
- Algoritmus je definovaný dvomi pravidlami:
 - **Pravidlo 1:** Pošli správu ďalej do každého **suseda** najviac raz.
 - **Pravidlo 2:** Ak nie je možné aplikovať pravidlo 1, pošli správu (naspäť) do **rodiča**.

Tarry-ho prieskum – príklad

- Algoritmus je definovaný dvomi pravidlami:
 - **Pravidlo 1:** Pošli správu ďalej do každého suseda najviac raz.
 - **Pravidlo 2:** Ak nie je možné aplikovať pravidlo 1, pošli správu (naspäť) do rodiča.
- Napr. $0 \rightarrow 1 \rightarrow 2 \rightarrow 5 \rightarrow 3 \rightarrow 1 \rightarrow 4 \rightarrow 6 \rightarrow 2 \rightarrow 6 \rightarrow$
 $4 \rightarrow 1 \rightarrow 3 \rightarrow 5 \rightarrow 2 \rightarrow 1 \rightarrow 0$

Pravidlo 2



Neorientovaný graf

→ šípka označuje
rodiča pre vrchol

Prehľadávanie grafu Tarryho prieskumom

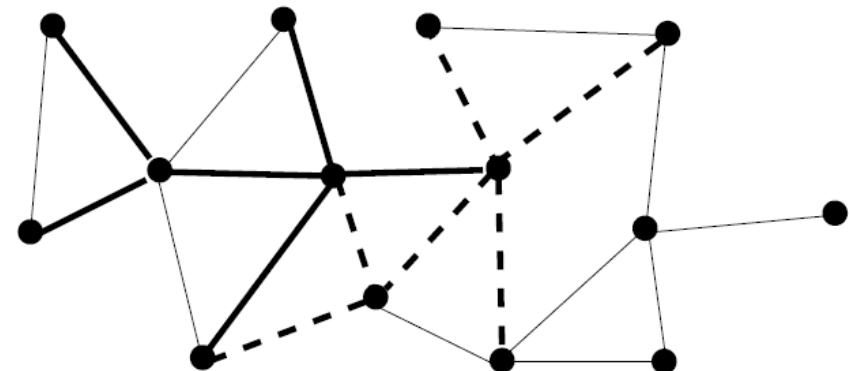
- Správa sa môže posúvať až kým sa nevráti naspäť do koreňa: Ked' sme v korení, aplikujeme pravidlo 1, inak, ak sa nedá aplikovať pravidlo 1, tak pravidlo 2 sa musí dať aplikovať (návrat do rodiča).
- Správa eventuálne navštívi všetky vrcholy grafu: Ak niektorý z vrcholov neboli navštívený, tak uvažujme, jeho suseda... Prečo sa nenavštívil z tohto suseda?
- **Tarryho prieskum je teda korektný algoritmus prehľadávania grafov**
- Inak povedané: Tarryho prieskum zostrojí uzavretý sled (Tarryho sled) obsahujúci každú hranu práve dvakrát.

Prehľadávanie grafu do hĺbky a šírky

- Systematicky prehľadat' vrcholy
- Začíname v niektorom vrchole $v \in V$
- Množina preskúmaných vrcholov V'
- Potrebujeme pravidlo určujúce, ktorý nepreskúmaný vrchol budeme prehliadať ako ďalší
- Pri prehľadávaní do hĺbky aj pri prehľadávaní do šírky vytvárame podstrom T grafu G .

Prehľadávanie grafu do hĺbky a šírky

- Pri prehľadávaní do hĺbky aj pri prehľadávaní do šírky vytvárame podstromom T grafu G .
- Poradie pridávania vrcholov do stromu T je poradím, v ktorom sa postupne preskúmajú vrcholy grafu G
- Strom $T = (V_T, E_T)$ je podgraf $G = (V, E)$, hrana $e=\{u,v\} \in E$ je **hraničná hrana** ak
 $u \in V_T$ a $v \notin V_T$,
u je zaraďený vrchol,
v je voľný vrchol.



Hrubo – hrany stromu T , hrubo čiarkovane – hraničné hrany, tenko – ostatné hrany grafu G .

Prehľadávanie grafu do hĺbky a šírky

- Začíname s vrcholom obsahujúcim len počiatočný vrchol.
- Opakujeme:
 - Ak už máme nejaký strom T , rozšírime ho tak, že do jeho množiny hrán pridáme nejakú hraničnú hranu $e = \{u,v\}$, $u \in V_T$ a $v \notin V_T$ a množinu vrcholov T rozšírime o vrchol v .
 - Pridanému vrcholu v určíme značku $p(v)$ – poradie, v ktorom sme ho objavili (pridali do stromu).
- Jediný rozdiel medzi prehľadávaním do hĺbky a prehľadávaním do šírky je spôsob výberu hraničnej hrany
 - **Prehľadávanie do hĺbky (depth-first search):** vyberá hraničnú hranu s najvyššou značkou p zaradeného konca hrany
 - **Prehľadávanie do šírky (breadth-first search):** vyberá hraničnú hranu s najnižšou značkou p zaradeného konca hrany

Prehľadávanie grafu do hĺbky (DFS)

Prehľadávanie grafu $G = (V, E)$ do hĺbky.

- **Krok 1. Inicializácia.**

Nech strom T je triviálny strom obsahujúci jediný vrchol $v \in V$.

Polož $p(v) := 1$, $k := 1$.

- **Krok 2.** Ak T ešte neobsahuje všetky vrcholy grafu, GOTO Krok 3.
Inak STOP.
- **Krok 3.** V grafe G so stromom T nájdi hraničnú hranu $e = \{u, v\}$ s **maximálnou značkou** $p(u)$ zaradeného vrchola u .
- **Krok 4.** Polož $T := T \cup \{e\} \cup \{v\}$, $k := k + 1$, $p(v) := k$.
GOTO Krok 2.

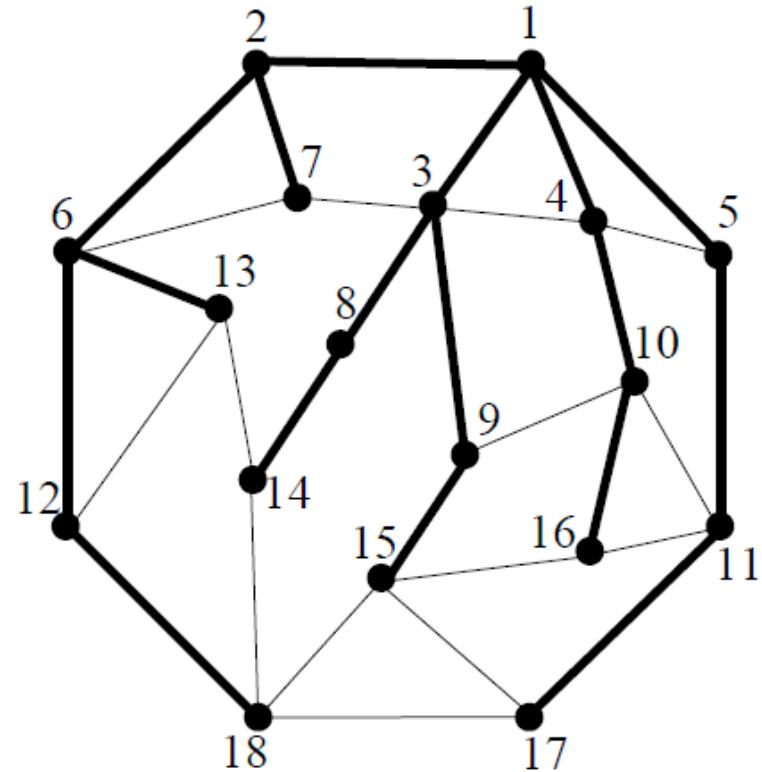
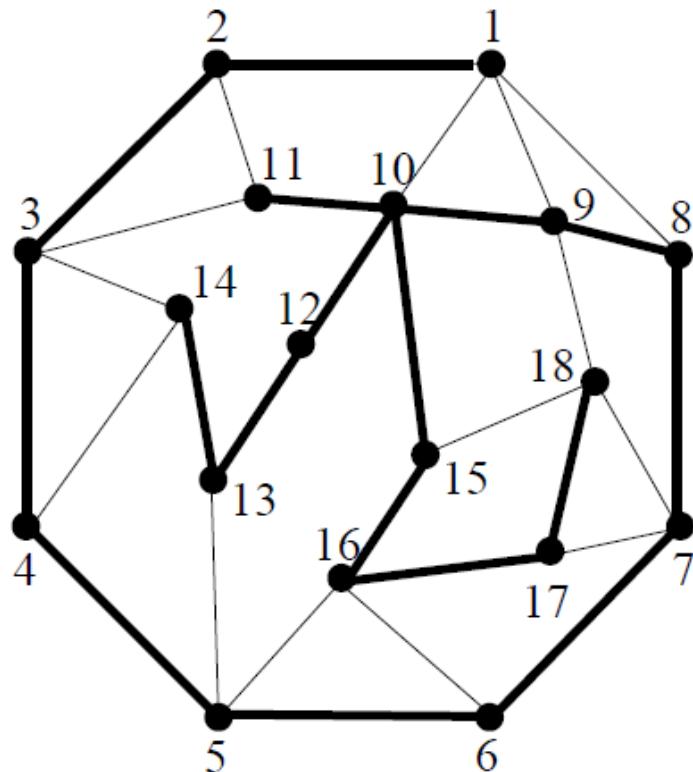
Prehľadávanie grafu do šírky (BFS)

Prehľadávanie grafu $G = (V, E)$ do šírky.

- **Krok 1.** Inicializácia. Nech strom T je triviálny strom obsahujúci jediný vrchol $v \in V$. Polož $p(v) := 1$, $k := 1$.
- **Krok 2.** Ak T ešte neobsahuje všetky vrcholy grafu, GOTO Krok 3. Inak STOP.
- **Krok 3.** V grafe G so stromom T nájdi hraničnú hranu $e = \{u, v\}$ s **minimálnou značkou** $p(u)$ zaradeného vrchola u .
- **Krok 4.** Polož $T := T \cup \{e\} \cup \{v\}$, $k := k + 1$, $p(v) := k$. GOTO Krok 2.

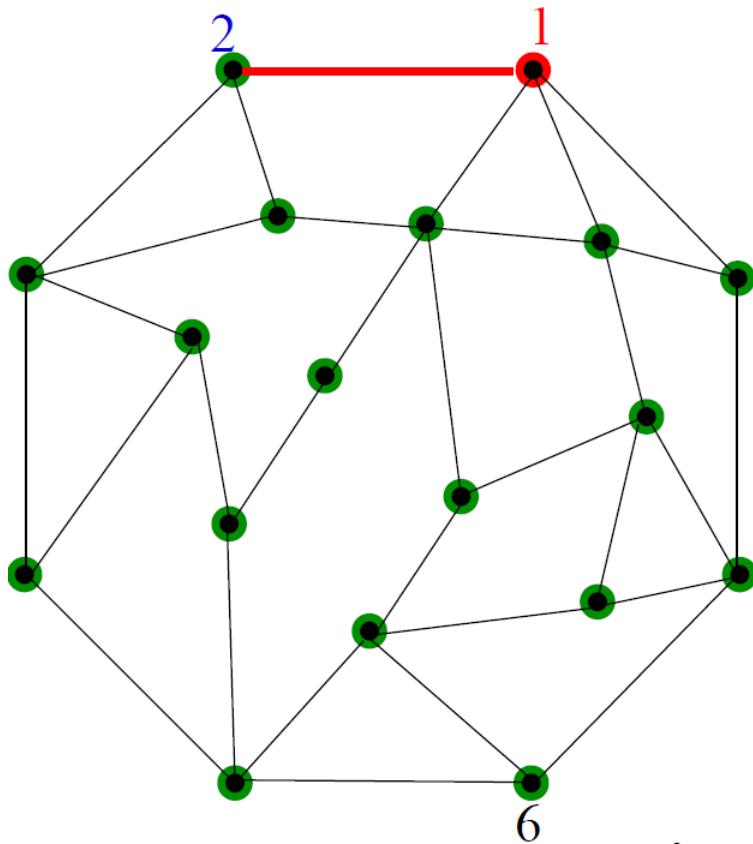
Prehľadávanie grafu do hĺbky vs šírky

- ## ■ Hrany stromu prehľadávania:

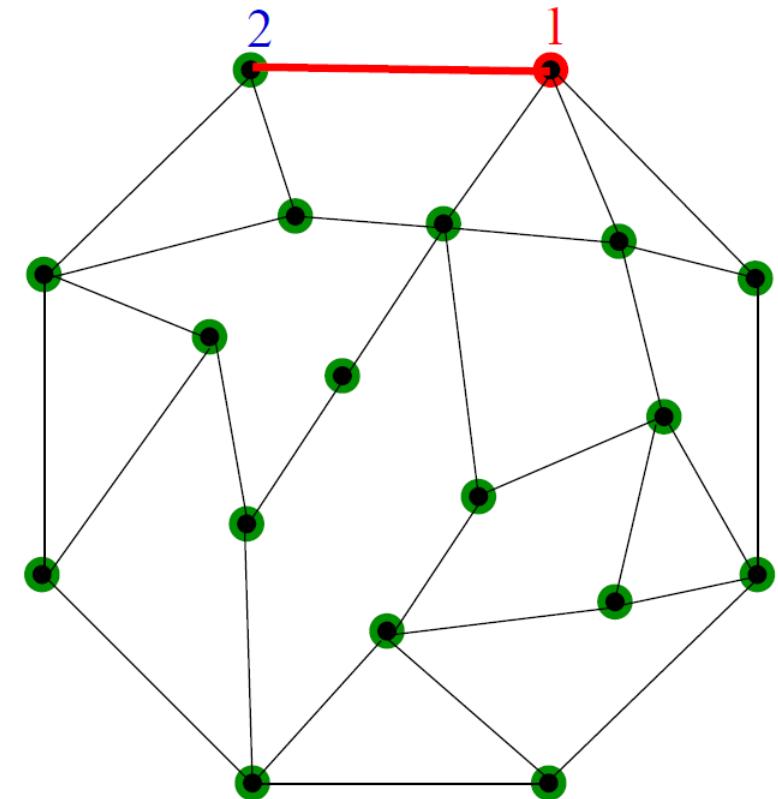


- ## ■ Ktoré je ktoré?

Prehľadávanie grafov

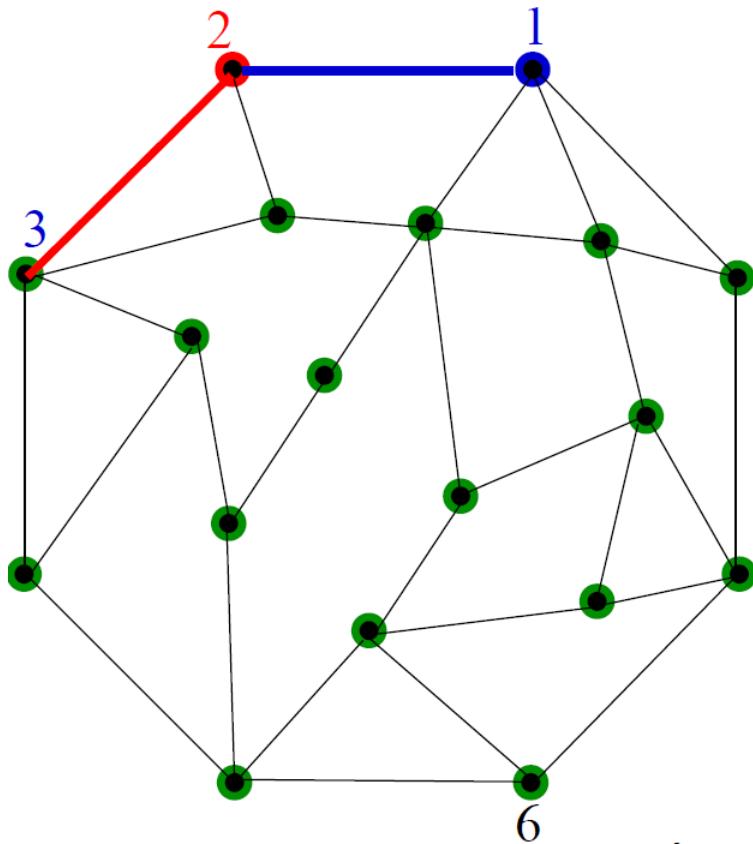


Prehľadávanie do hĺbky

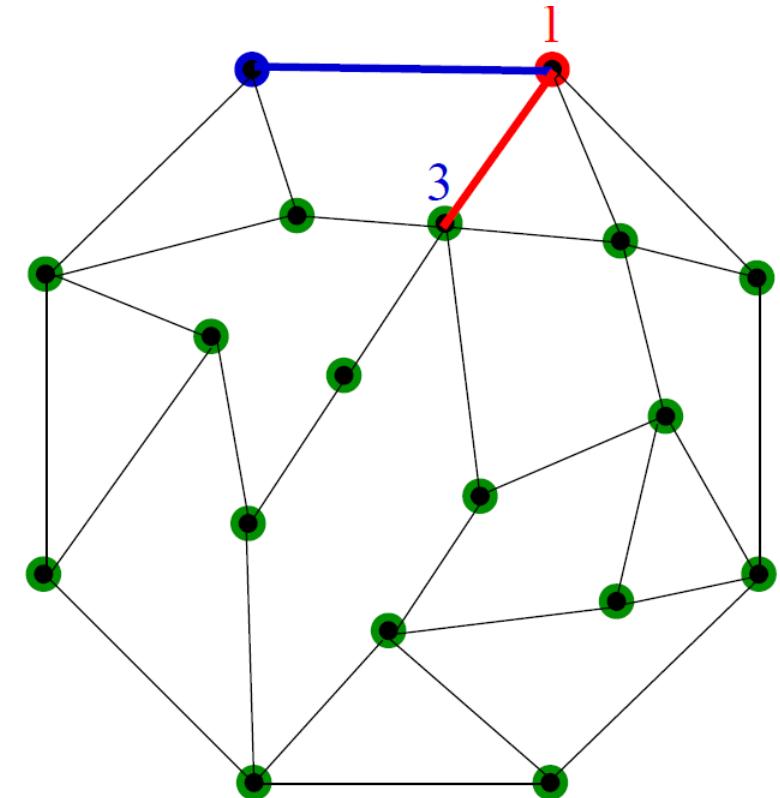


Prehľadávanie do šírky

Prehľadávanie grafov

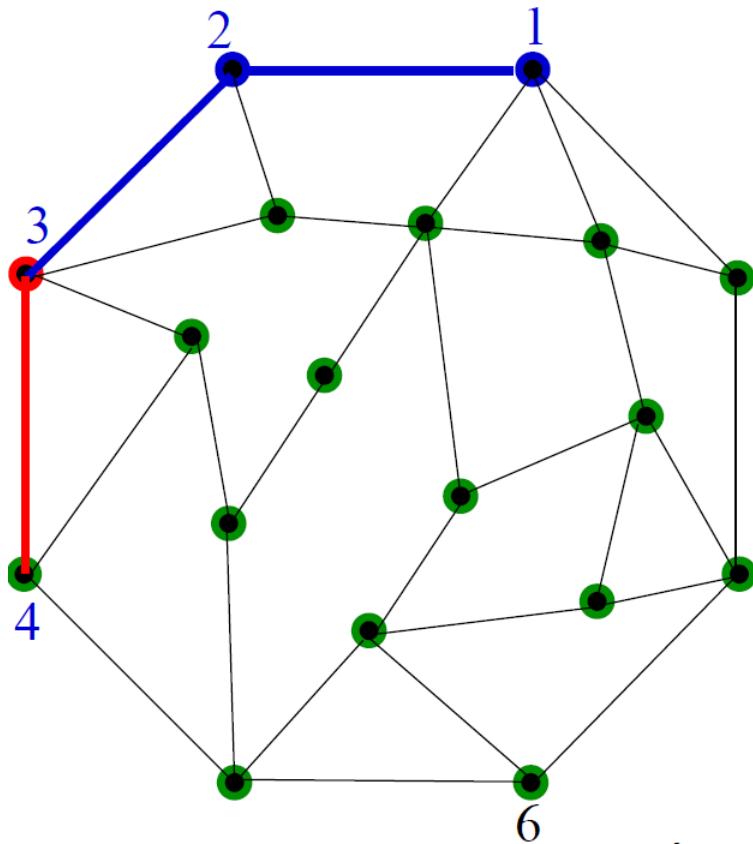


Prehľadávanie do hĺbky

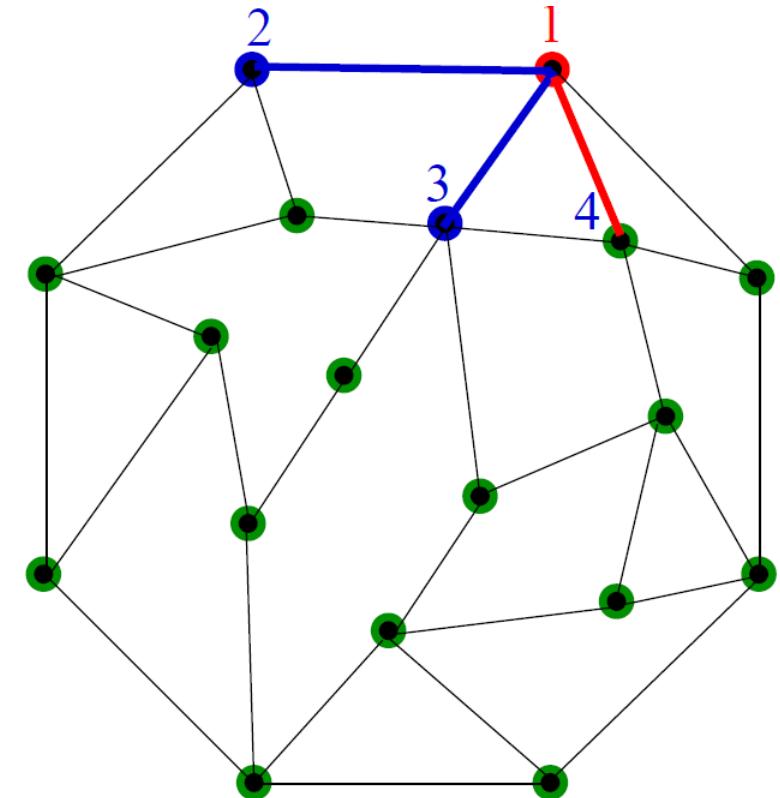


Prehľadávanie do šírky

Prehľadávanie grafov

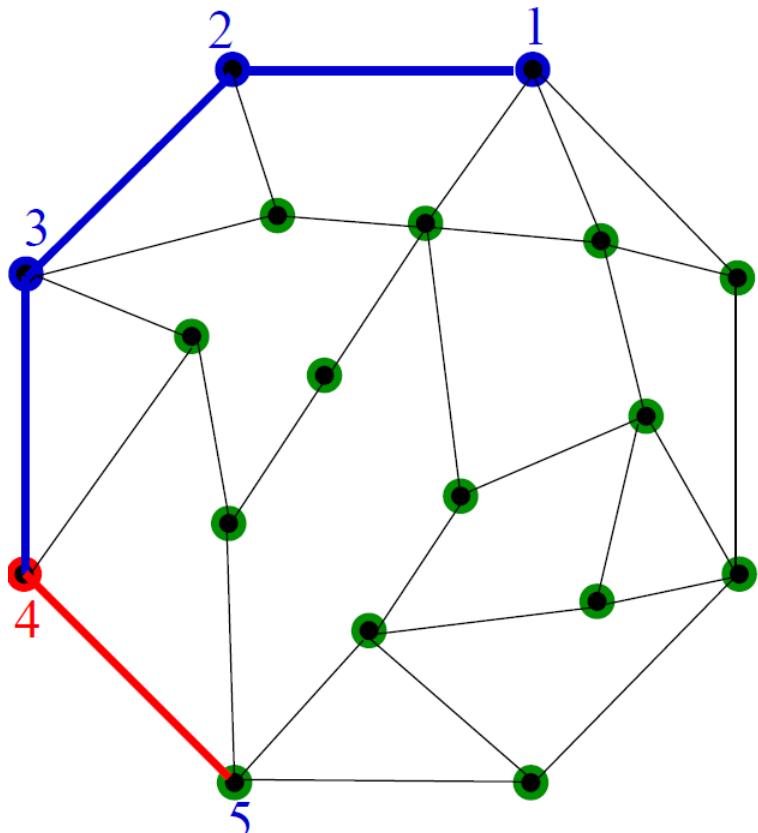


Prehľadávanie do hĺbky

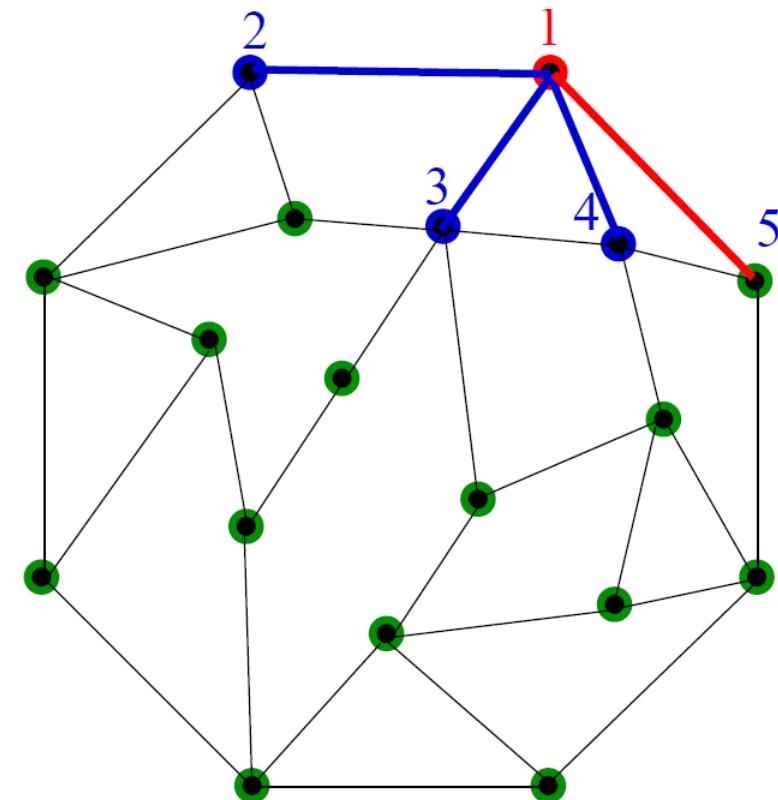


Prehľadávanie do šírky

Prehľadávanie grafov

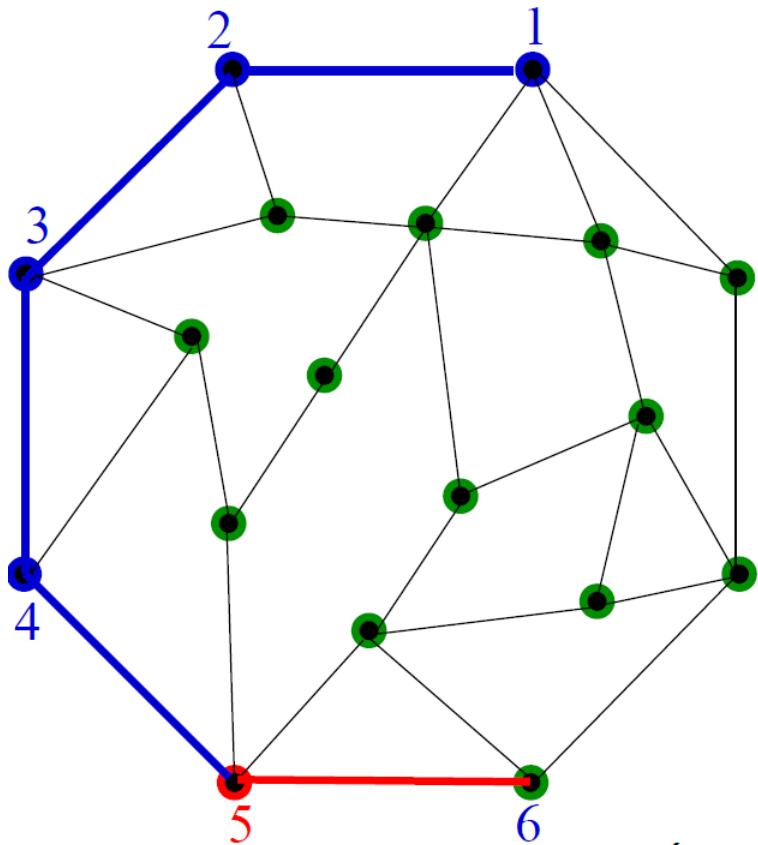


Prehľadávanie do hĺbky

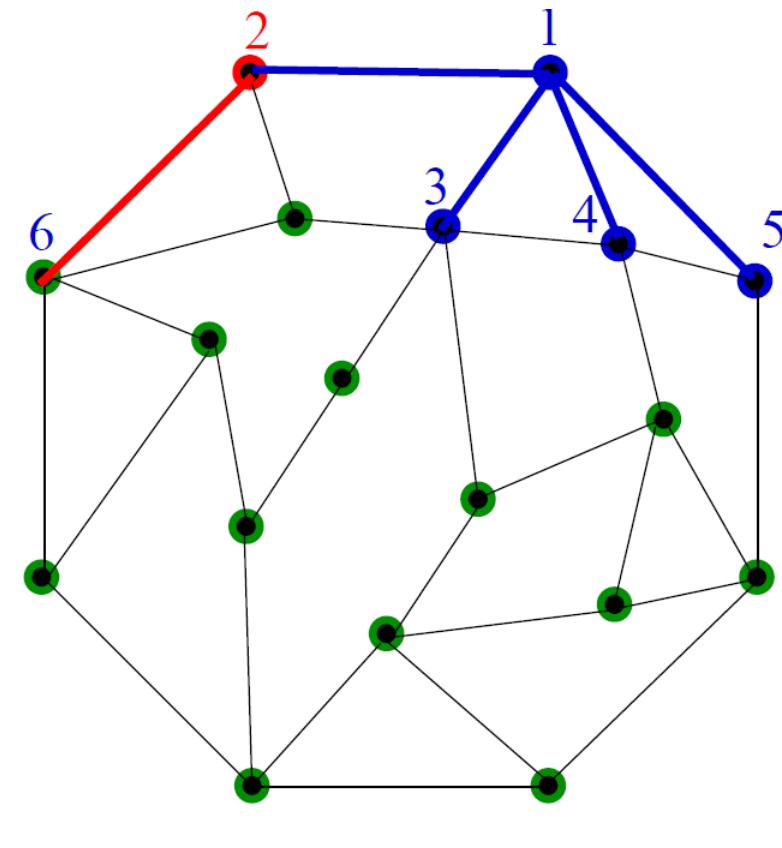


Prehľadávanie do šírky

Prehľadávanie grafov

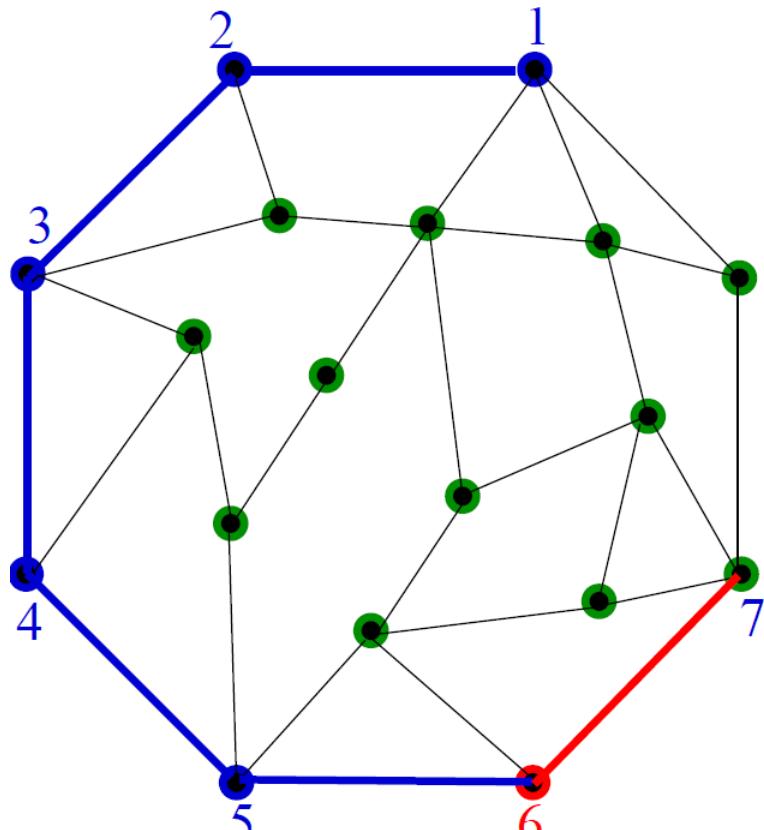


Prehľadávanie do hĺbky

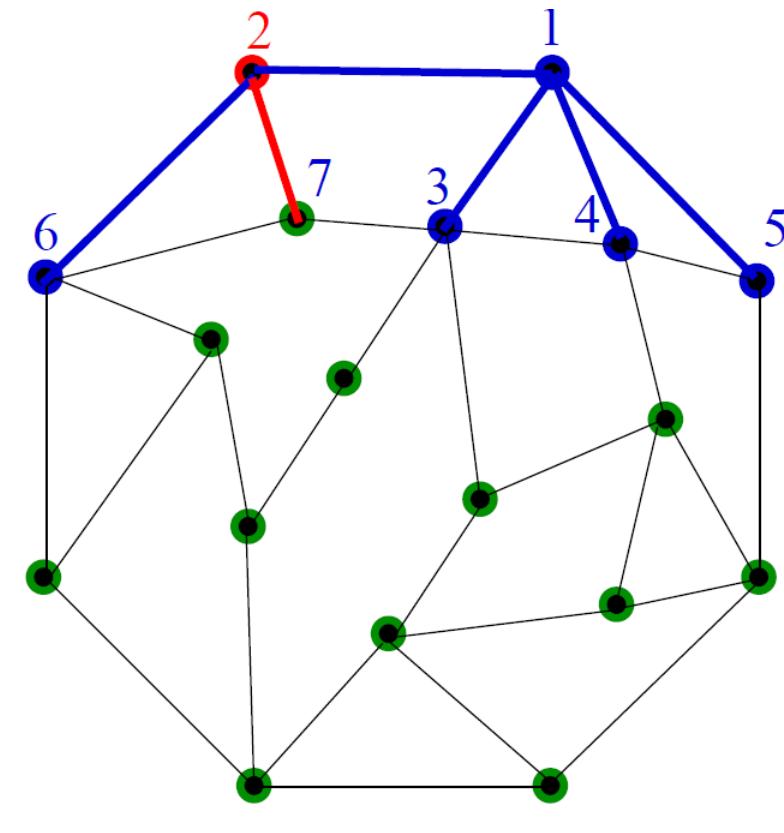


Prehľadávanie do šírky

Prehľadávanie grafov

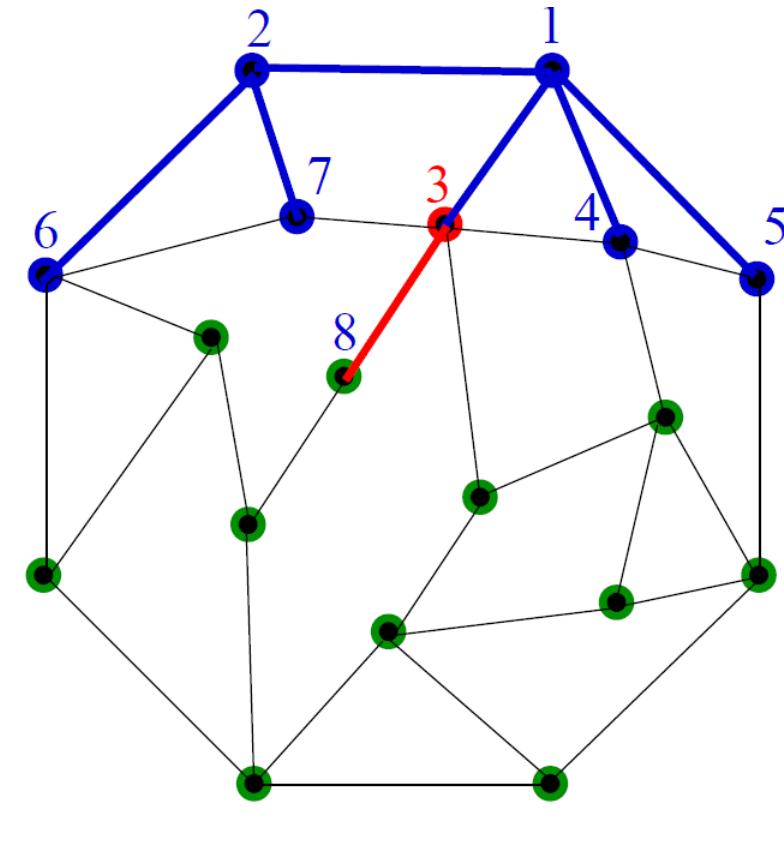
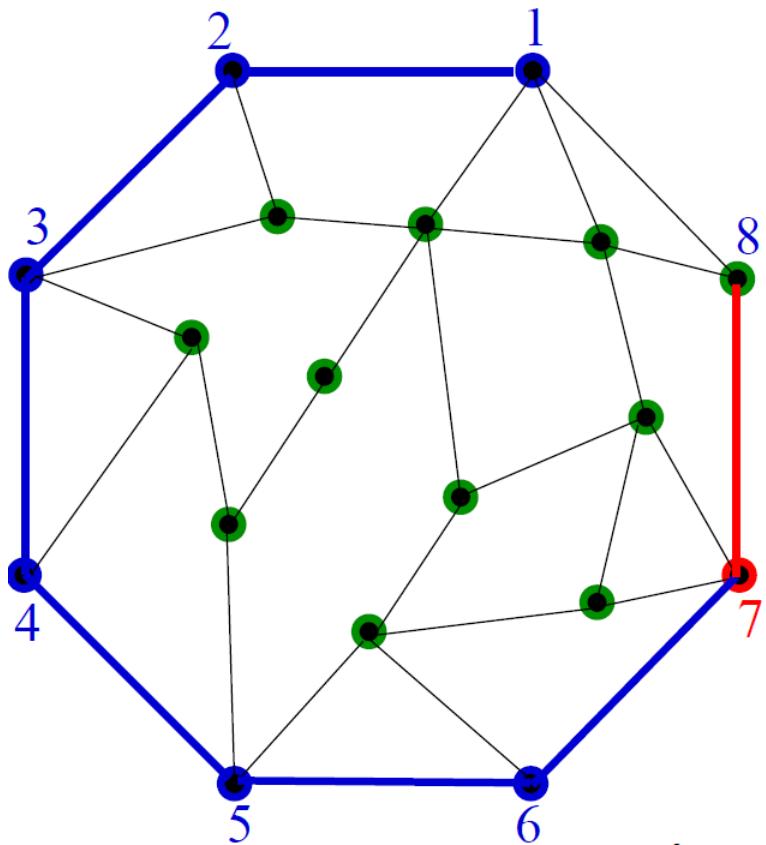


Prehľadávanie do híbky

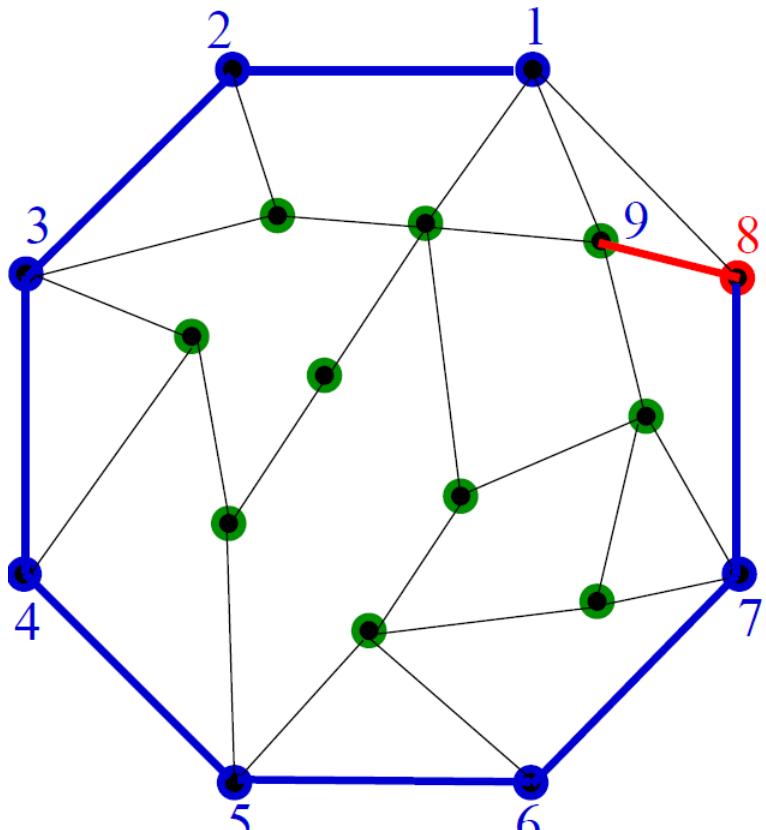


Prehľadávanie do šírky

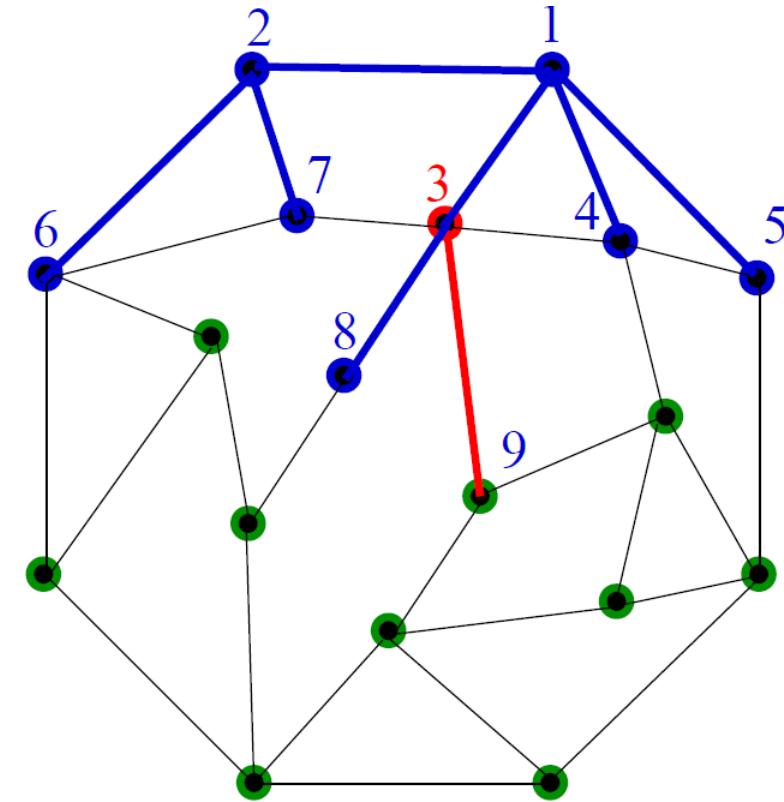
Prehľadávanie grafov



Prehľadávanie grafov

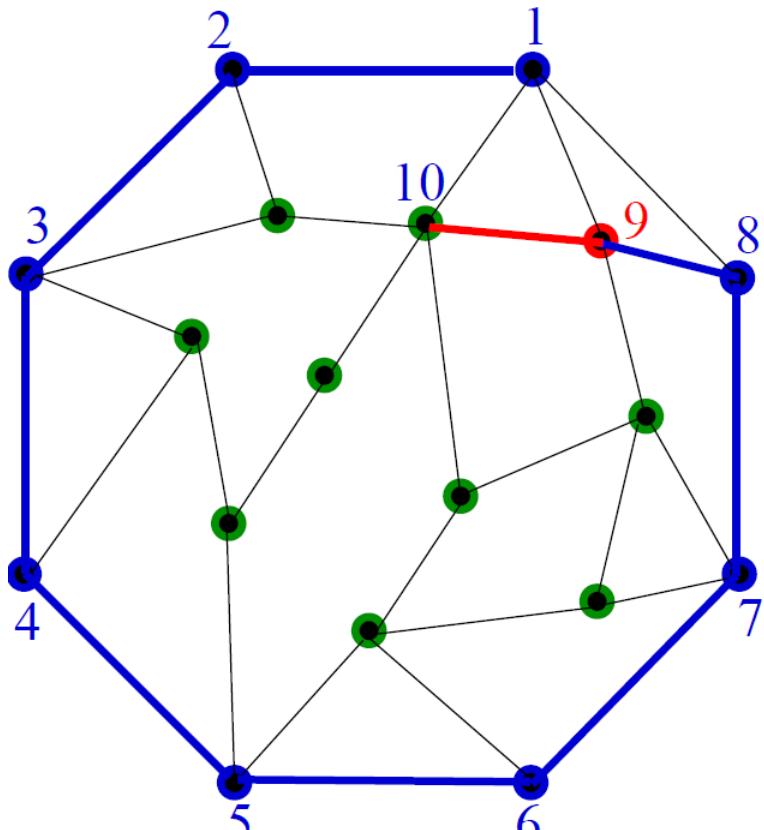


Prehľadávanie do hĺbky

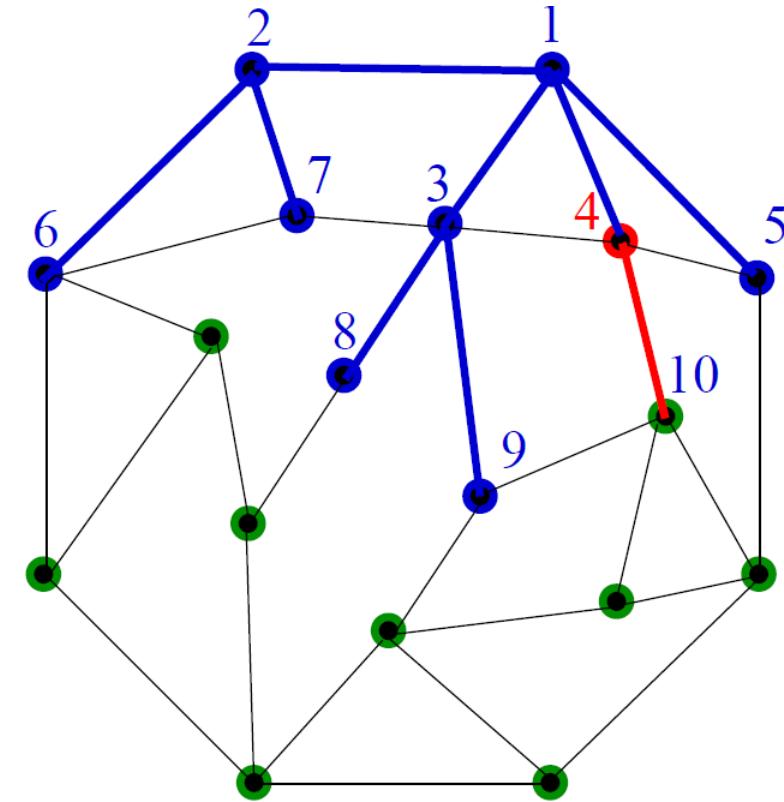


Prehľadávanie do šírky

Prehľadávanie grafov

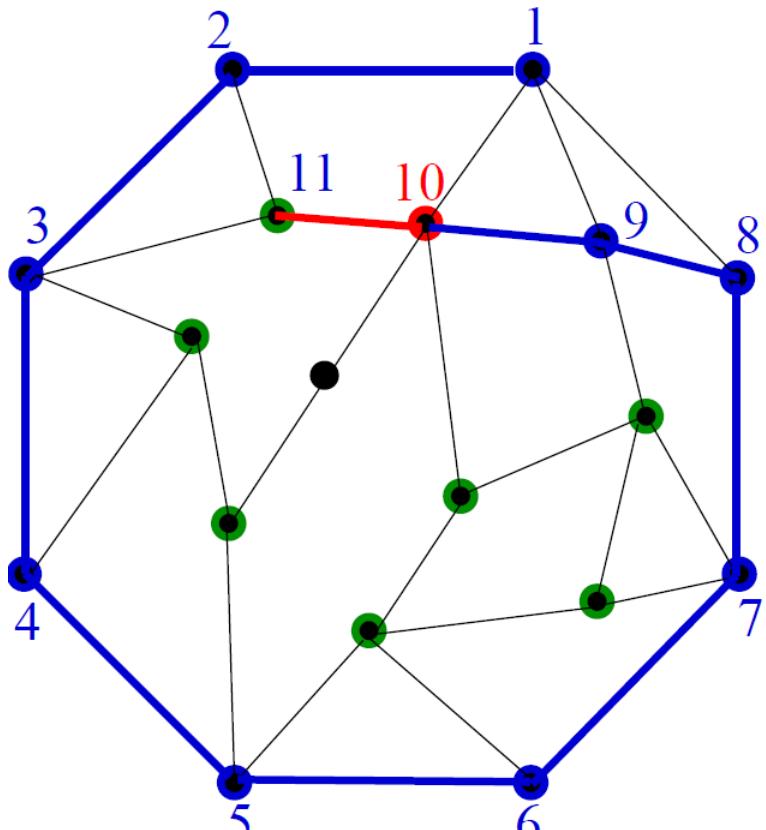


Prehľadávanie do hĺbky

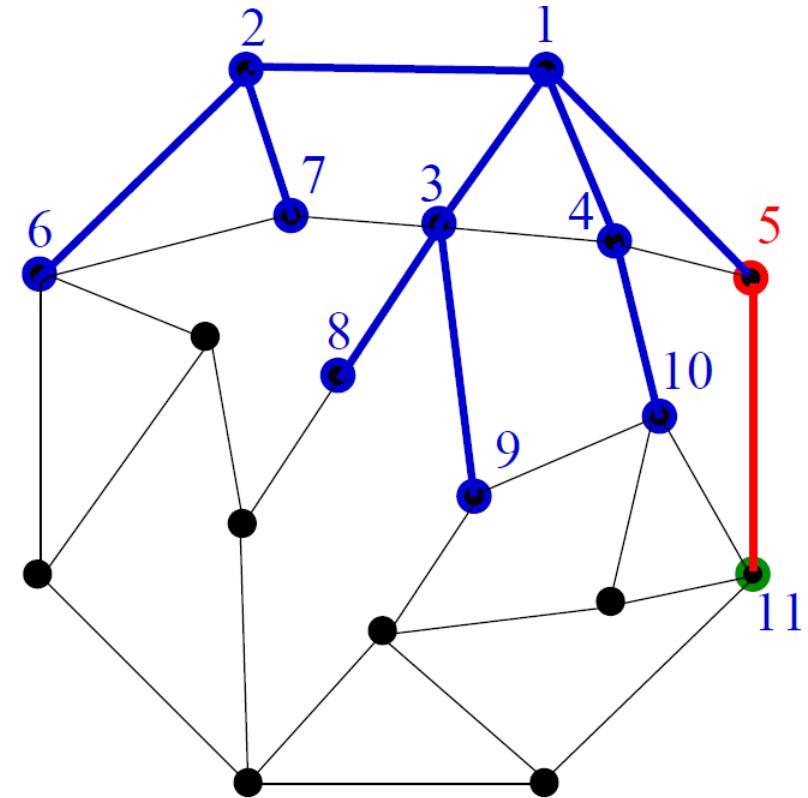


Prehľadávanie do šírky

Prehľadávanie grafov

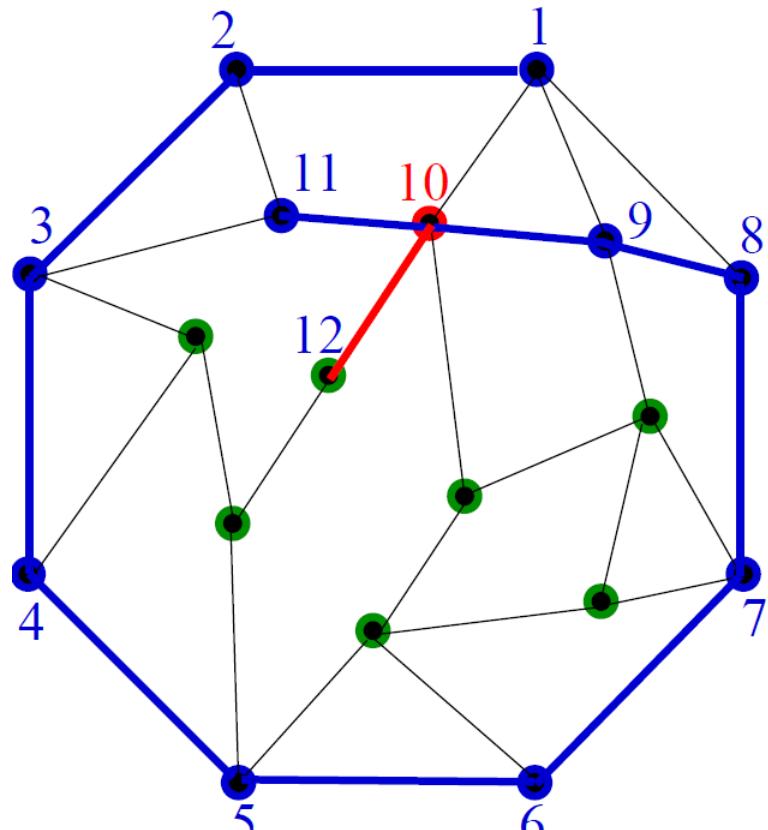


Prehľadávanie do hĺbky

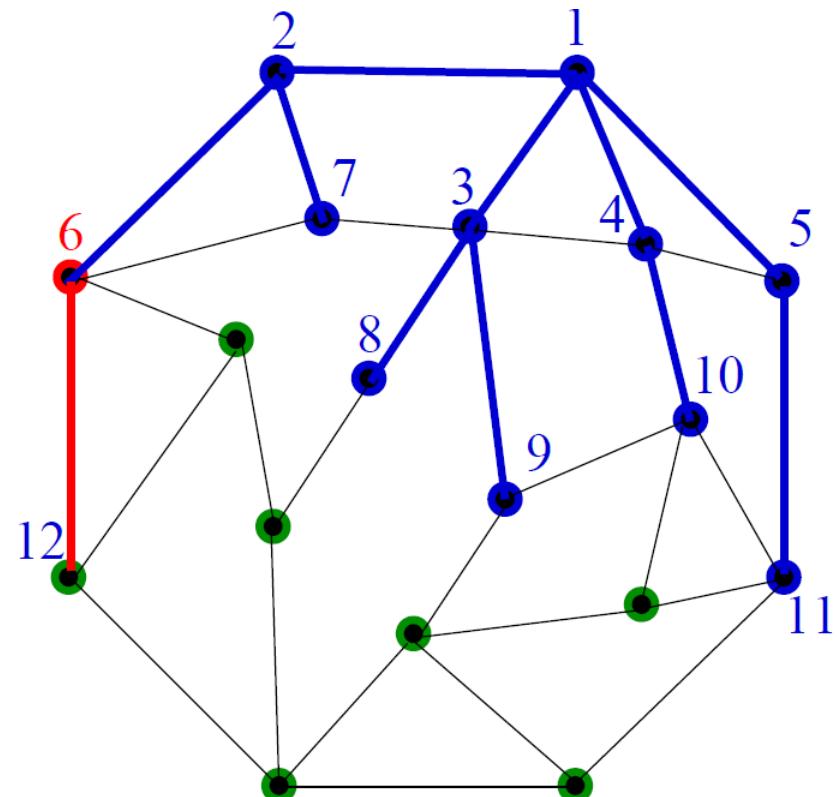


Prehľadávanie do šírky

Prehľadávanie grafov

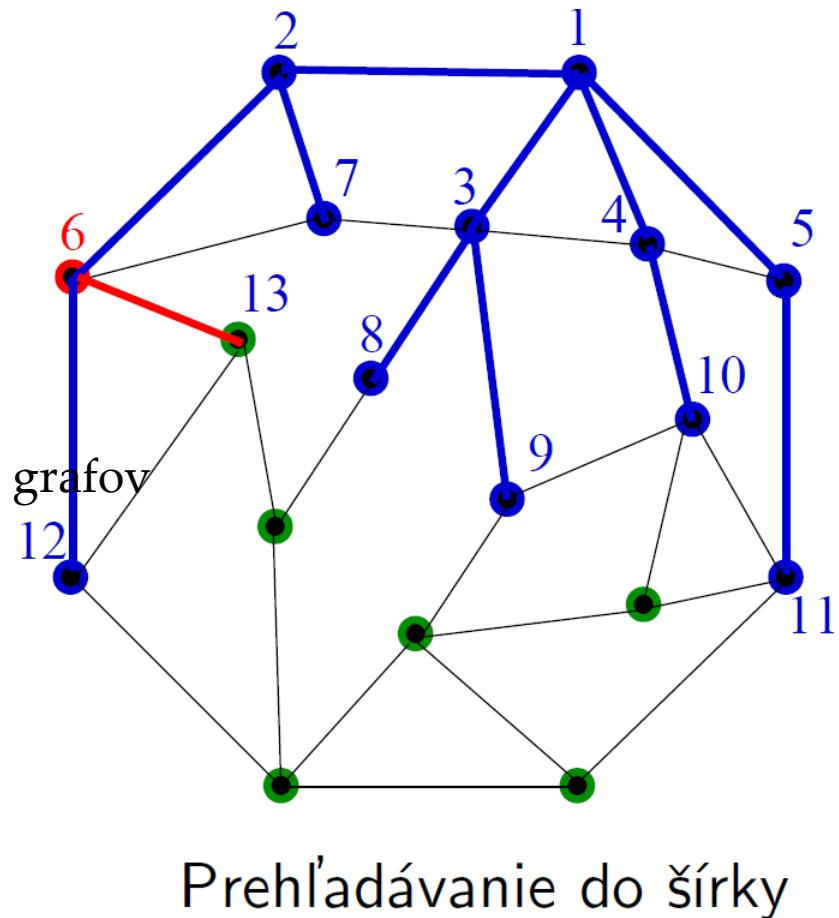
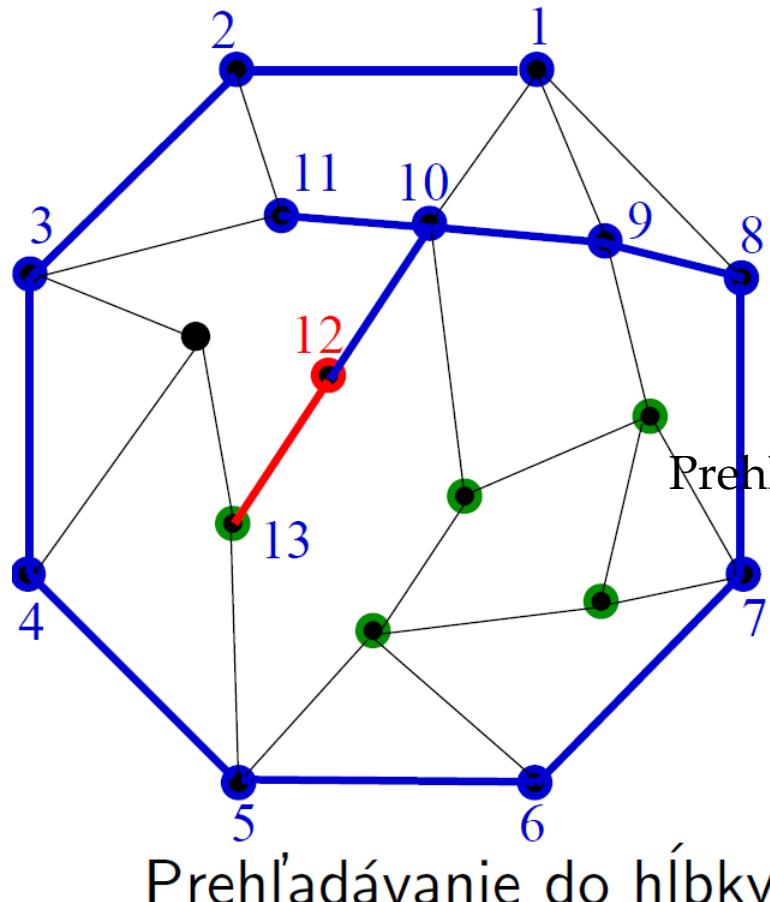


Prehľadávanie do hĺbky

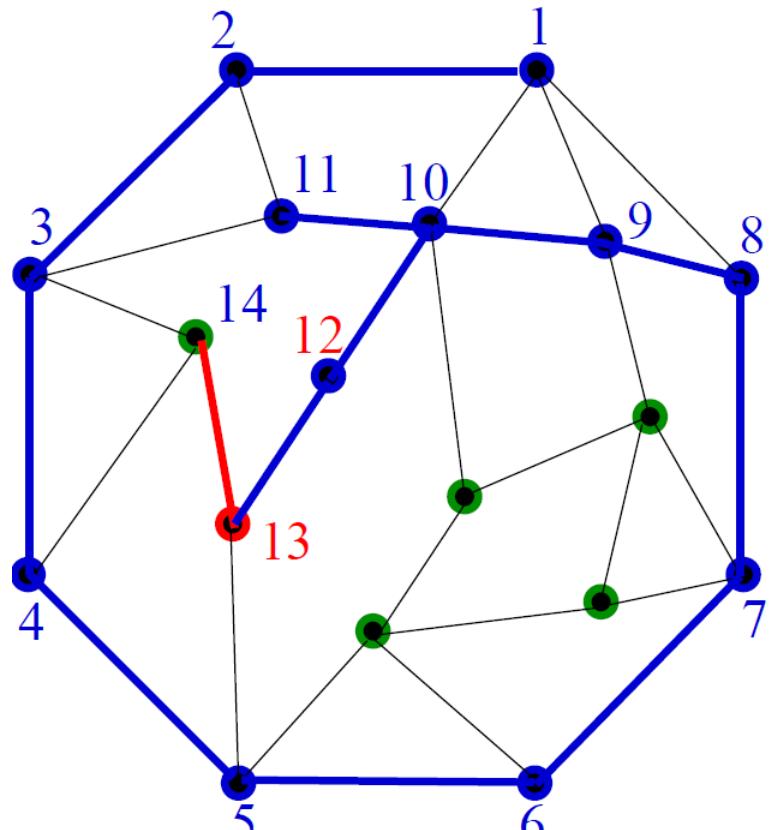


Prehľadávanie do šírky

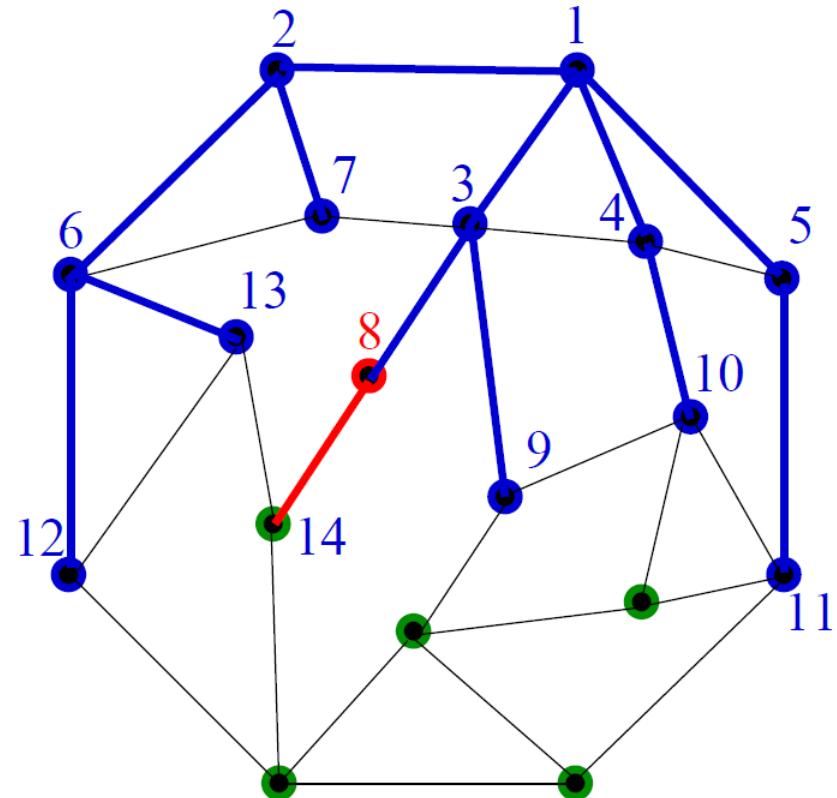
Prehľadávanie grafov



Prehľadávanie grafov

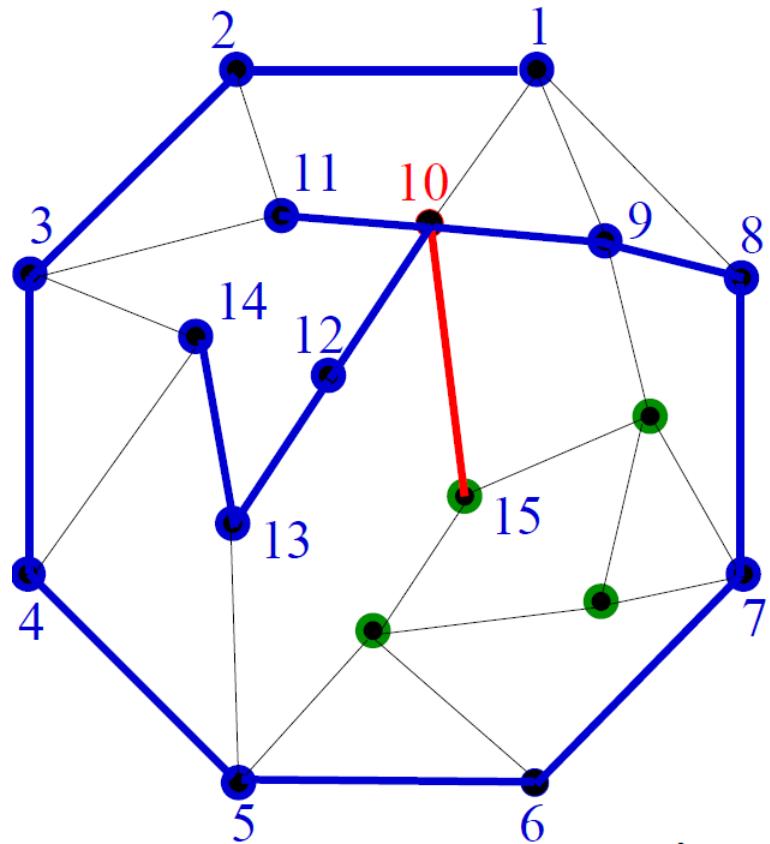


Prehľadávanie do hĺbky

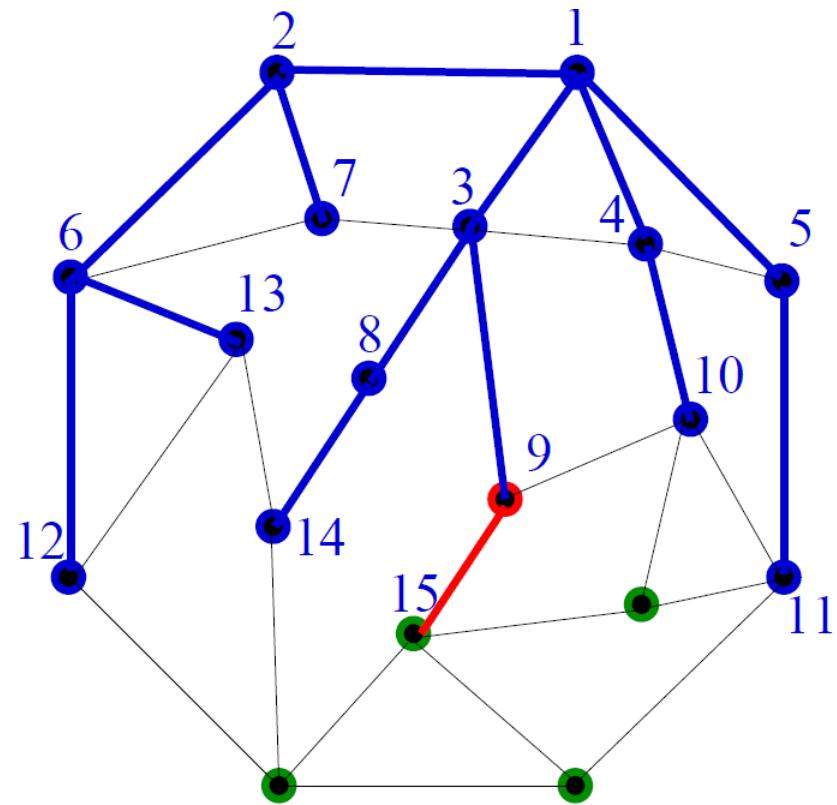


Prehľadávanie do šírky

Prehľadávanie grafov

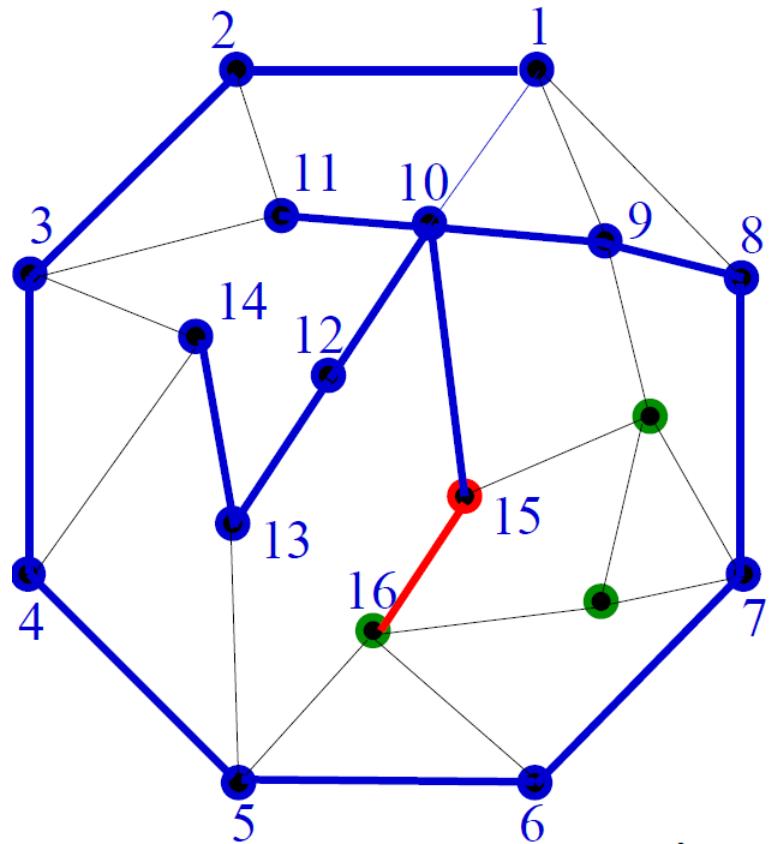


Prehľadávanie do hĺbky

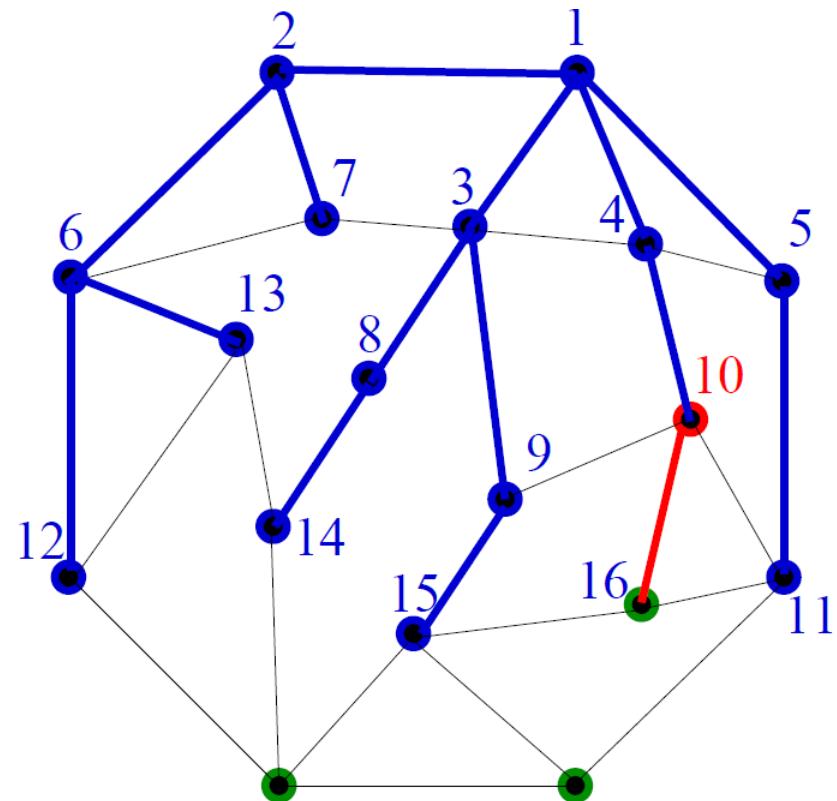


Prehľadávanie do šírky

Prehľadávanie grafov

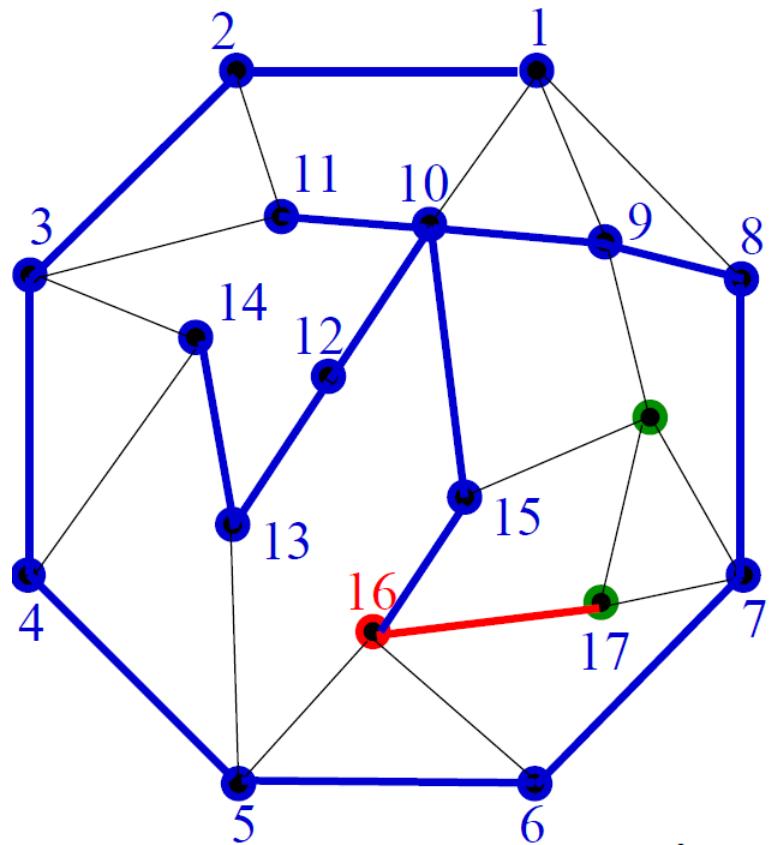


Prehľadávanie do hĺbky

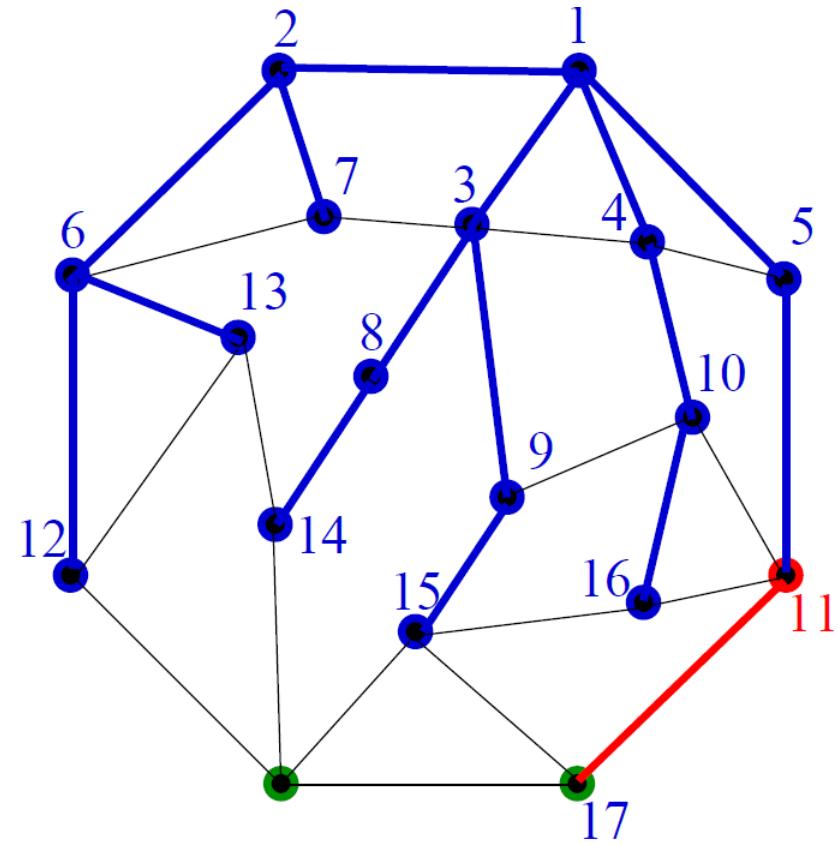


Prehľadávanie do šírky

Prehľadávanie grafov

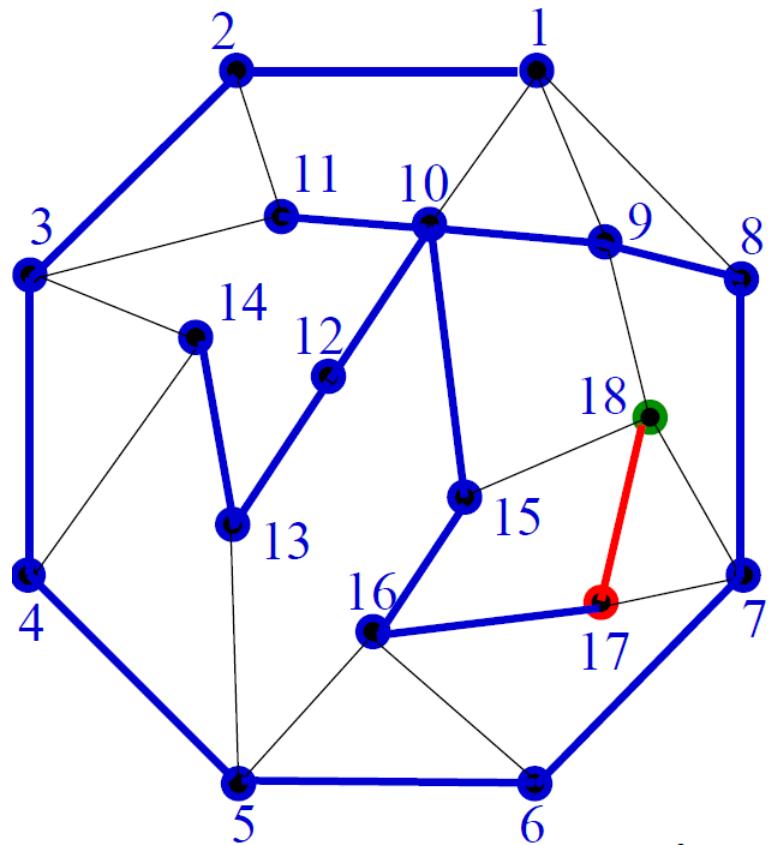


Prehľadávanie do hĺbky

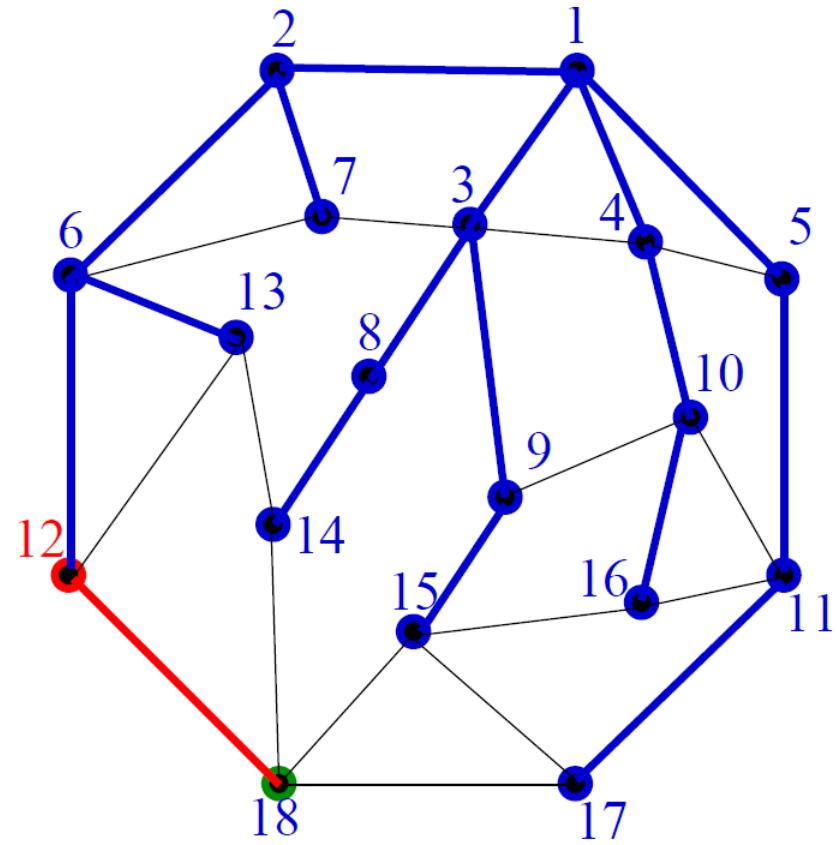


Prehľadávanie do šírky

Prehľadávanie grafov



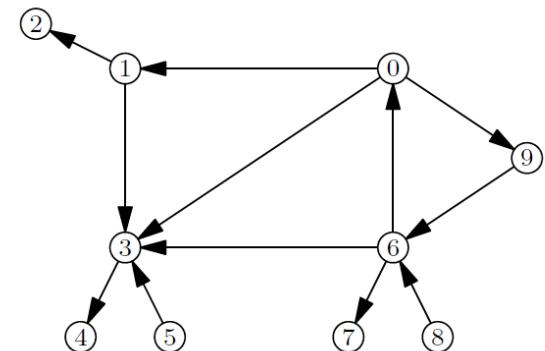
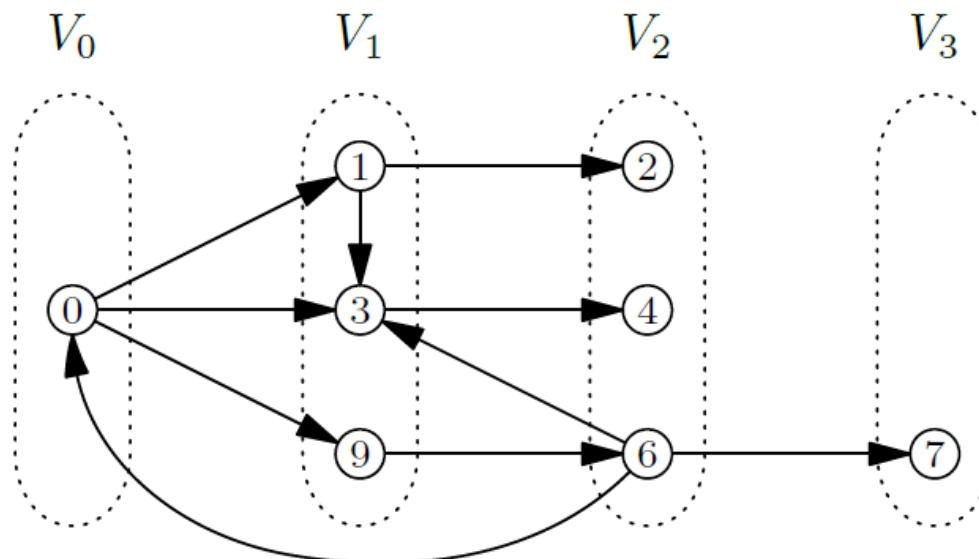
Prehľadávanie do hĺbky



Prehľadávanie do šírky

Prehľadávanie grafu do šírky – ukážka

- Vrcholy možno rozdeliť do vrstiev

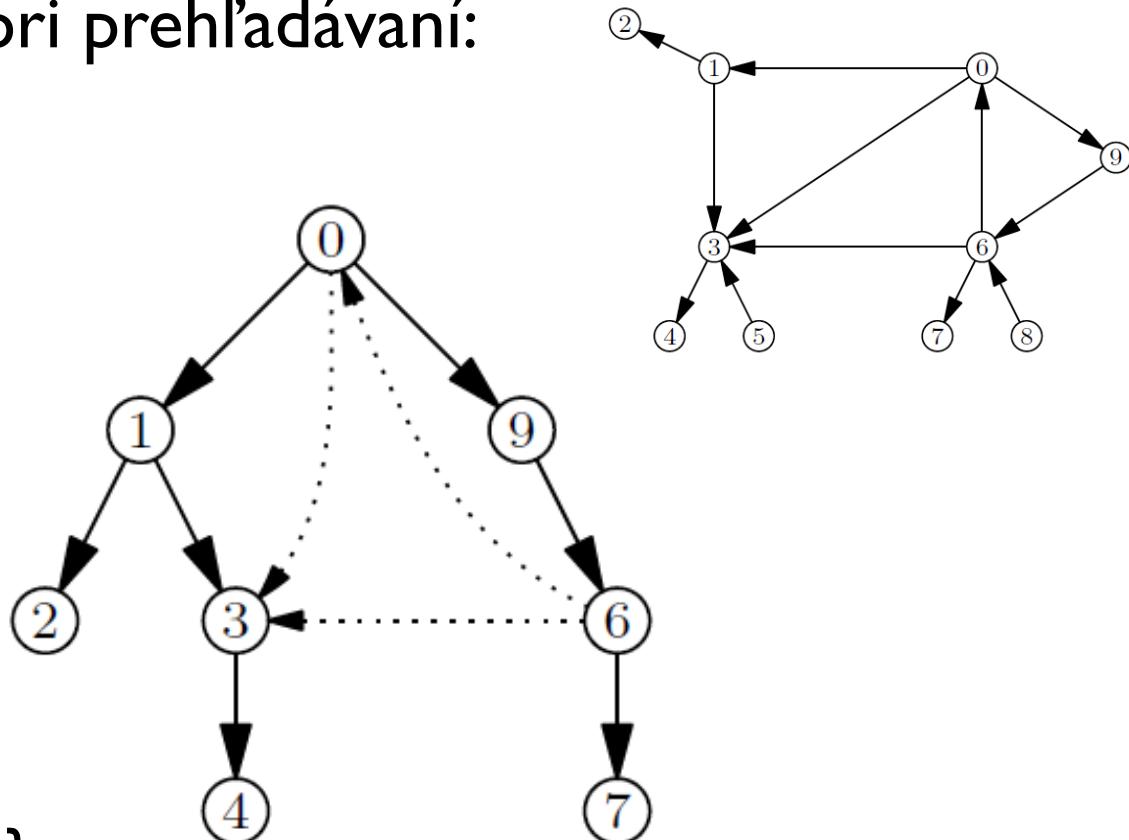


Prehľadávanie grafu do hĺbky – ukážka

- Rôzne typy hrán pri prehľadávaní:

- Stromové
- Dopredné
- Spätné
- Priečne

- Typ hrany určíme podľa značiek $p(u)$ a $p(v)$ pre koncové vrcholy hrany $\{u,v\}$ ($u \in V_T$ a $v \notin V_T$)



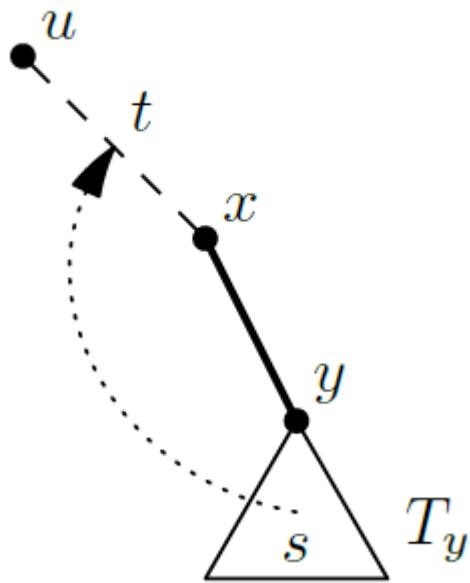
Prehľadávanie do hĺbky – Implementácia

```
// prehladaj z vrcholu v
void visit(int v)
{
    color[v] = GRAY; // zaciatok
    for (int i = 0; i < N; i++)
        if (graph[v][i] != 0) // sused i
            if (color[i] == WHITE)
                visit(i);           // prehľadavanie do hlbky
    color[v] = BLACK; // koniec
}
void dfs()
{
    // zafarbime kazdy vrchol na bielo
    for (int i = 0; i < N; i++)
        color[i] = WHITE;
    // ofarbovat zacneme z bielych vrcholov
    for (int i = 0; i < N; i++)
        if (color[i] == WHITE)
            visit(i);
}
```



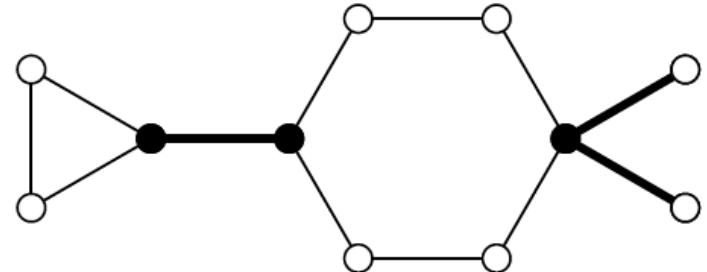
Hľadanie mostov, artikulácií

- Využijeme prehľadávanie do hĺbky
- Všímame si typy hrán, ktoré navštívime pri prehľadávaní



Spätná s-t hrana spôsobuje, že x-y nie je most.

Podobne pri určovaní, či je vrchol artikulácia.

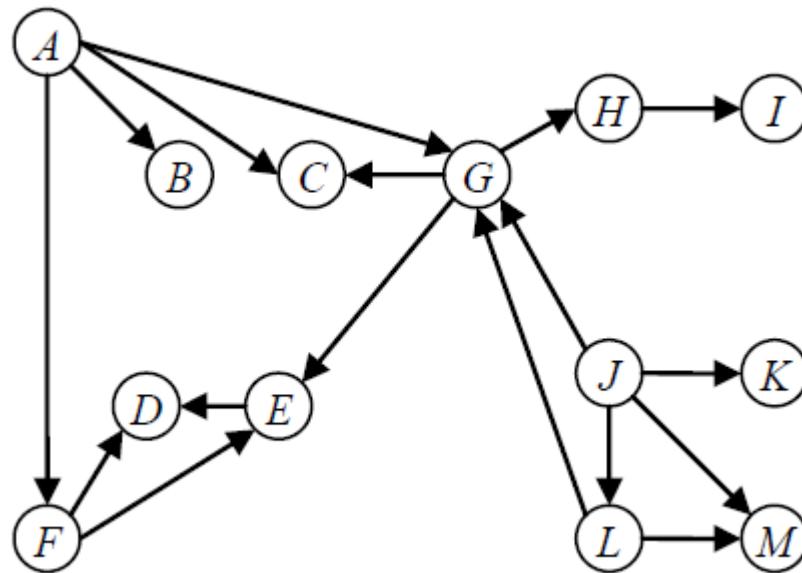


Topologické usporiadanie

- Orientované acyklické grafy (directed acyclic graphs)
- Reprezentácia závislostí medzi činnosťami:
 - Orientovaný graf bez (orientovaných) cyklov
 - Cyklus (v závislostiach) je problém
- Topologické usporiadanie vrcholov acyklického grafu
 - Také poradie vrcholov, že žiaden vrchol nie je spracovaný skôr ako vrchol, ktorý na neho ukazuje
- Úlohy v (orientovaných) acyklických grafoch
 - Nájdenie najkratšej cesty
 - Nájdenie najdlhšej cesty
 - ...

Topologické usporiadanie

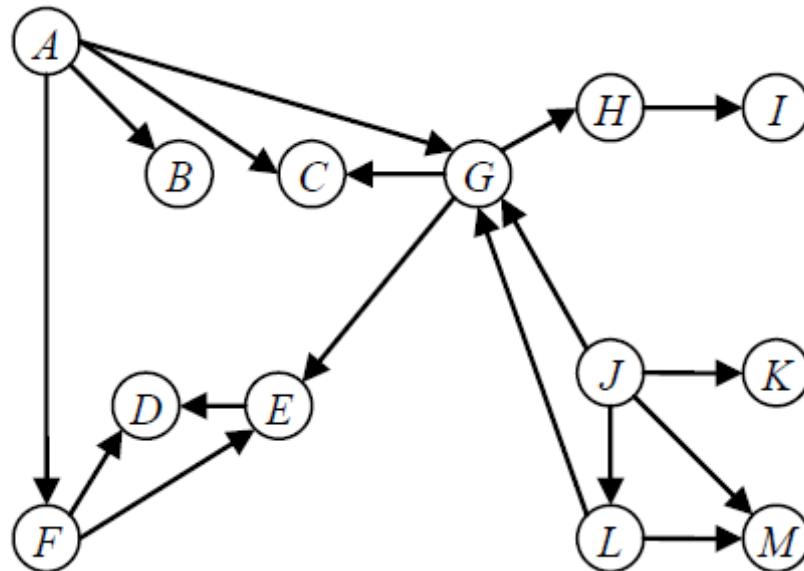
- Topologické usporiadanie vrcholov – žiadnen vrchol nie je spracovaný skôr ako vrchol, ktorý na neho ukazuje
- Napr. J K L M A G H I F E D B C



- Nie je jediné: A J L G F K E M B H C I D

Reverzné topologické usporiadanie

- Uvažujme opačný problém:
 - hrana (x,y) znamená, že vrchol x závisí na vrchole y .
- D E F C B I H G A K M L J



- Prehľadávanie do hĺbky: vždy ked' prehľadávanie v rekurzií „odchádza z vrcholu“, vypíšeme číslo vrcholu

Topologické usporiadanie – Implementácia

```
int graph[N][N];
int saw[N]; // 1=navstivili sme vrchol
int rev[N], nrev; // reverzne top. usporiadanie

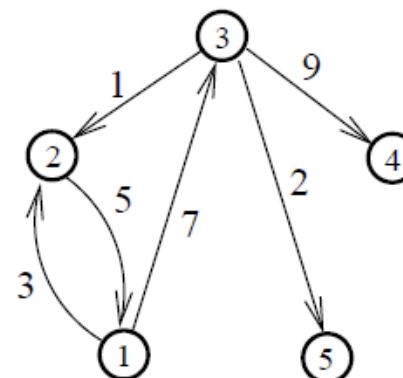
void dfs_rts(int v)
{
    saw[v] = 1;
    for (int i = 0; i < N; i++)
        if (graph[v][i]) // sused i
            if (saw[i] == 0)
                dfs_rts(i);
    rev[nrev++] = v;
}

void tsort()
{
    nrev = 0;
    // najdeme reverzne top. usporiadanie
    for (int i = 0; i < N; i++)
        if (saw[i] == 0)
            dfs_rts(i);
    // vypiseme opacne
    for (int i = N-1; i >= 0; i--)
        printf("%d\n", rev[i]);
}
```



Teória grafov – ohodnotený graf

- Hranám priradíme ohodnenie, zobrazenie w
 - w je hranové ohodnenie $w: E \rightarrow \mathbb{R}$
 $w(e) = c$ (hovoríme, že hrana e má ohodnenie c)
 - Graf $G = (V, E, w)$ nazývame **ohodnotený graf** (weighted graph)
- Ohodnenie nazývame aj
 - Váha (weight)
 - Cena (cost)
 - Dĺžka (length)
 - ...
- Rôzne typy ohodnení
 - Vrcholové ohodnenie (vertex weight)
 - Viacero ohodnení zároveň (napr. dĺžka hrany a cena hrany)

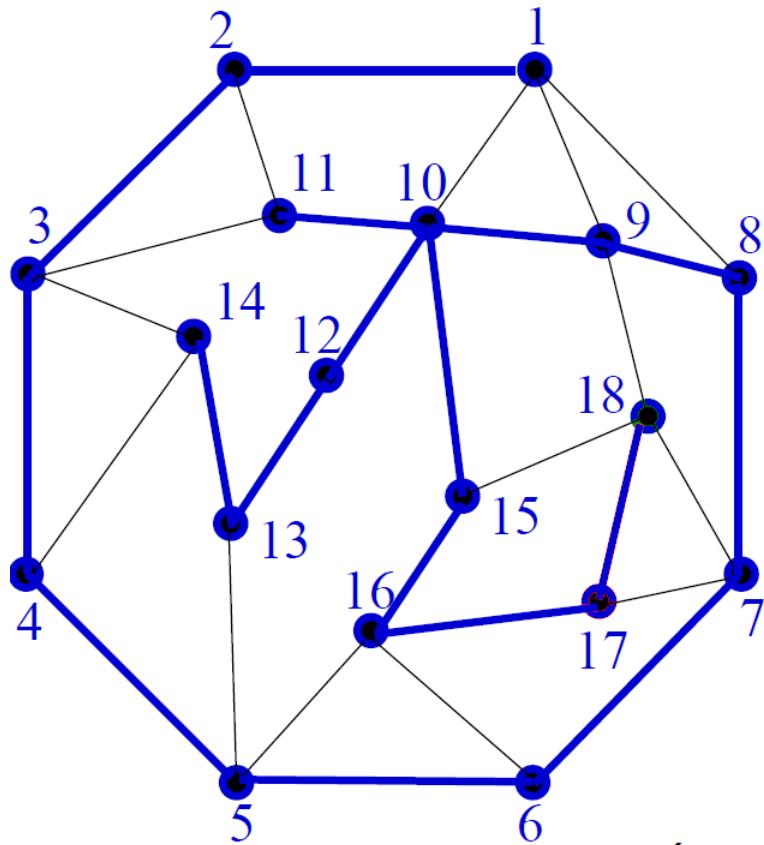


	1	2	3	4	5
1	-	3	7	-	-
2	5	-	-	-	-
3	-	1	-	9	2
4	-	-	-	-	-
5	-	-	-	-	-

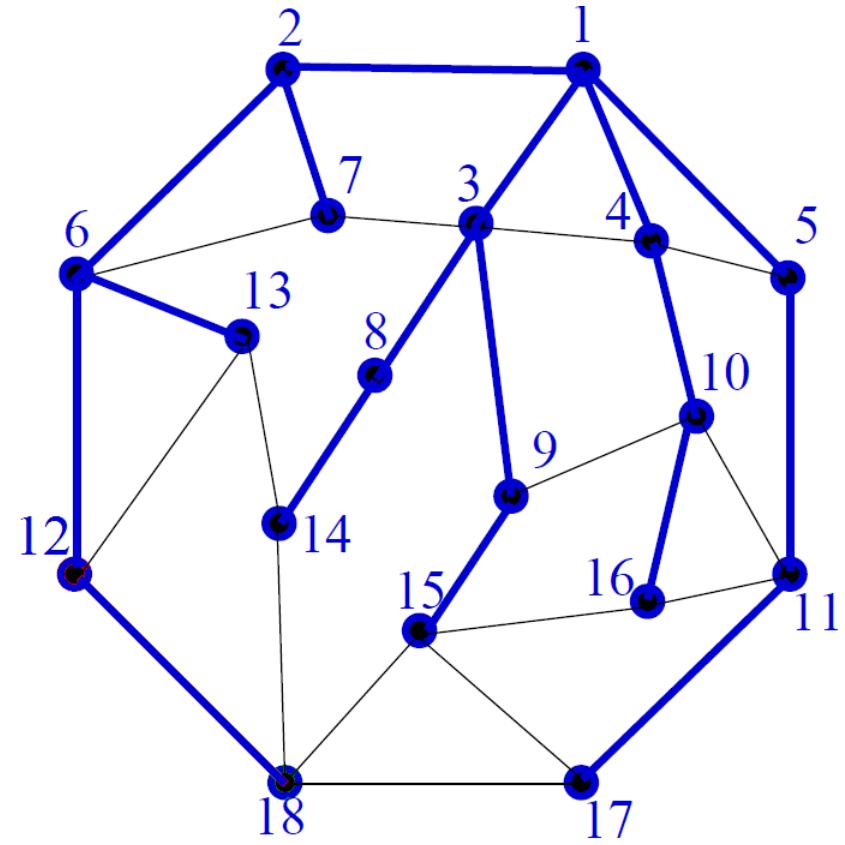
Teória grafov – najkratšia cesta

- Dĺžka sledu, tåhu, cesty v hranovo ohodnotenom grafe nazveme súčet ohodnotení jeho hrán
 - Ak máme sled, tak ohodnenie každej hrany započítame toľkokrát, koľkokrát sa hrana v slede nachádza
 - Dĺžka sledu, tåhu, cesty ak je to len jeden vrchol je 0.
 - V prípade neohodnoteného grafu, je dĺžka počet hrán.
- **Najkratšia x-y cesta** v hranovo ohodnotenom grafe G je tá zo všetkých x-y ciest v G, ktorá má najmenšiu dĺžku
- Najkratšia x-y cesta v neohodnotenom grafe?
 - Najmenšia dĺžka (počet hrán) cesty z vrcholu x do vrcholu
 - Prehľadávanie do šírky

Najkratšia cesta v neohodnotenom grafe



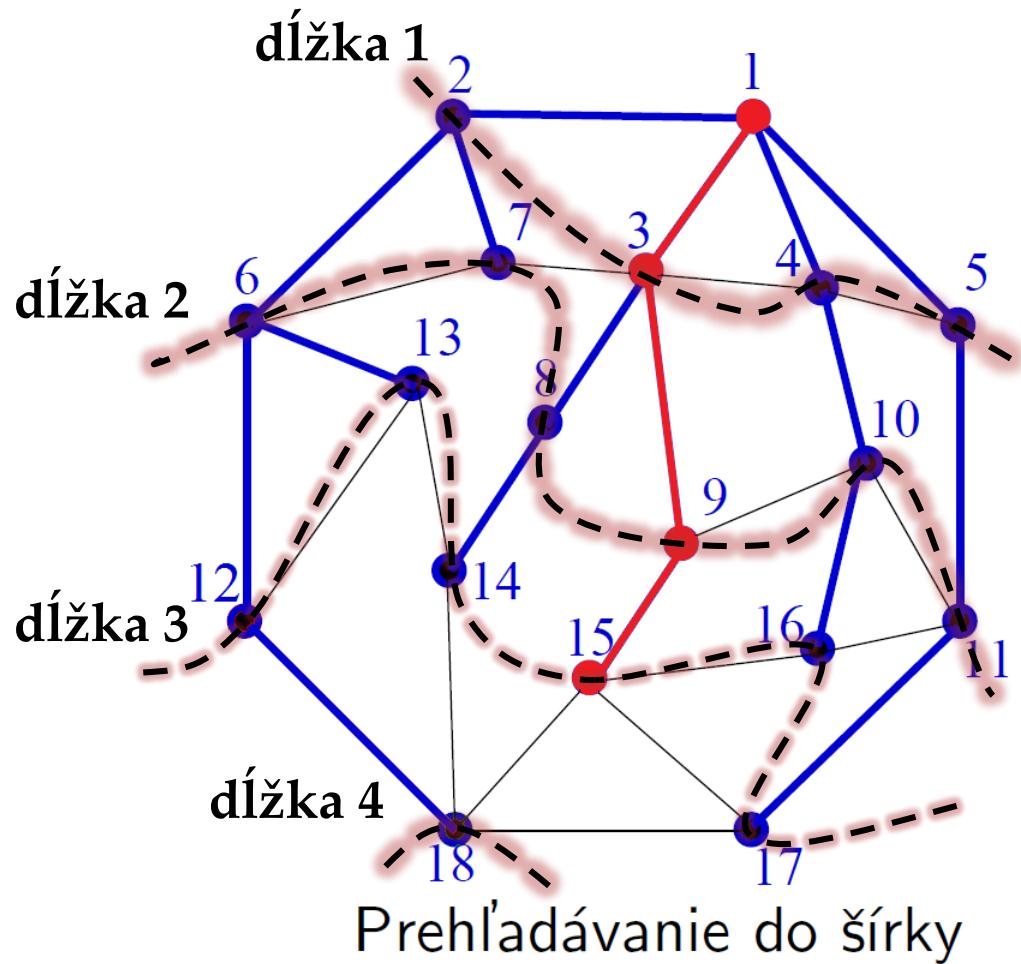
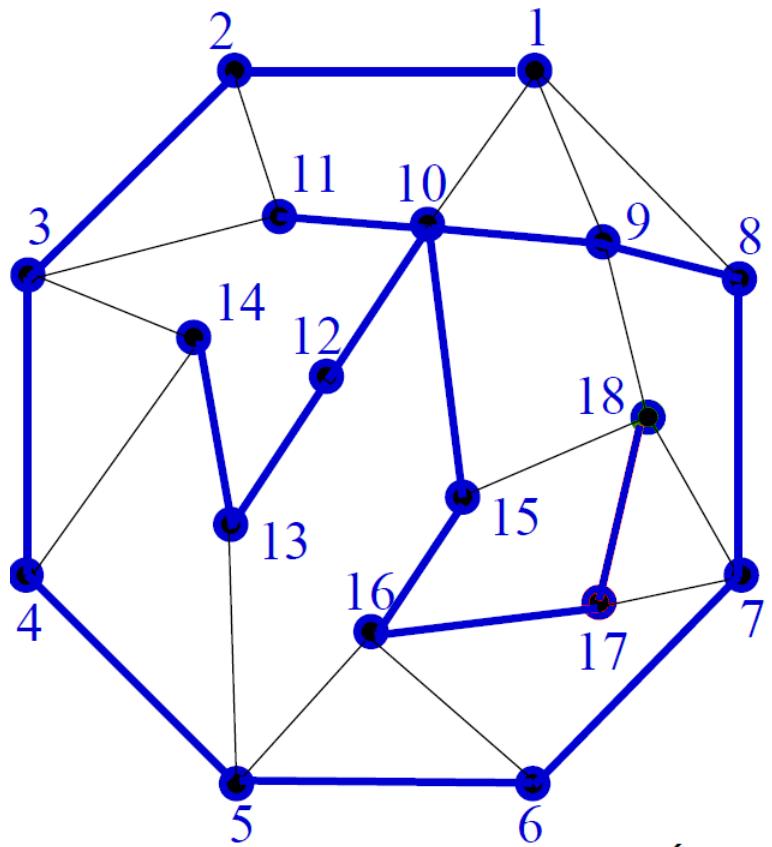
Prehľadávanie do hĺbky



Prehľadávanie do šírky

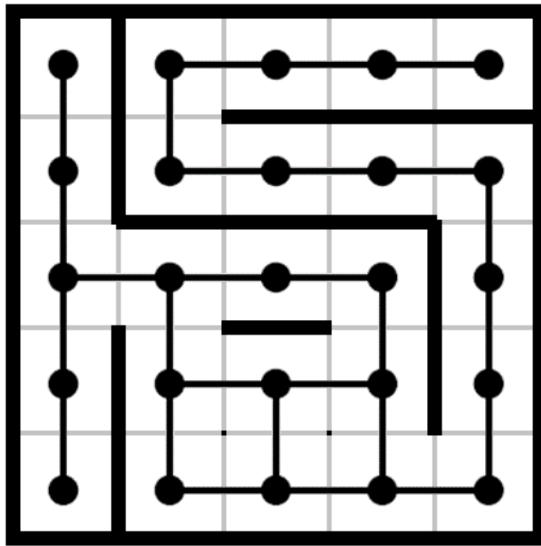
Strom prehľadávania do šírky (z vrcholu x)
zodpovedá **stromu najkratších ciest** (z vrcholu x)

Najkratšia cesta z I do 15



Aplikácia: Bludiská

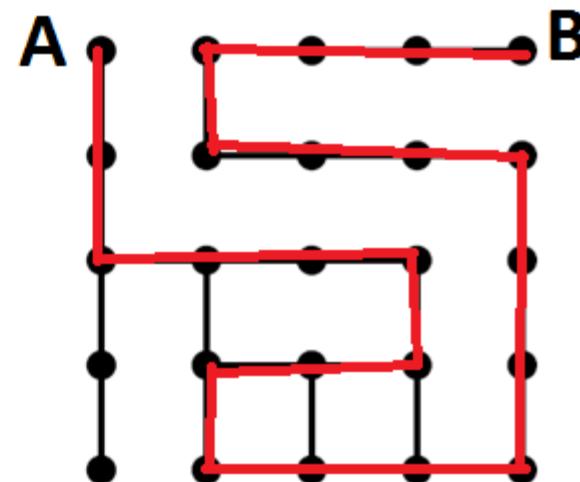
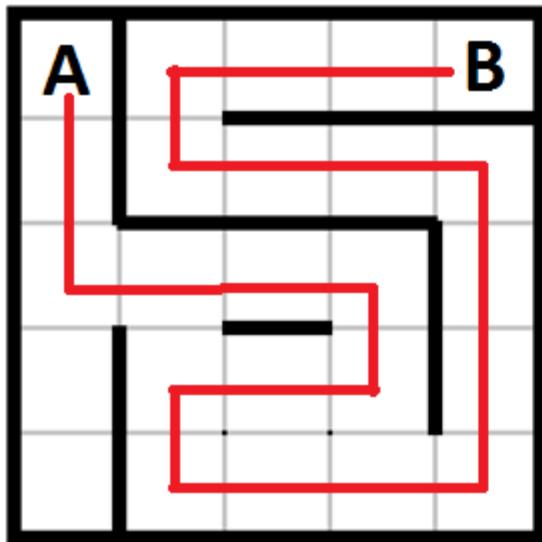
- Čo môžeme reprezentovať grafom?
- Mapy (bludisko s miestnosťami prepojenými chodbami)
 - Graf: vrchol = poličko, hrana = dá sa prejsť medzi poličkami



- Postačuje nám LEN grafová reprezentácia!

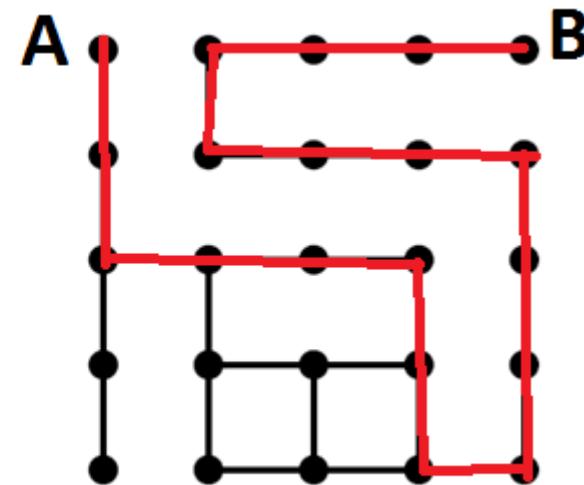
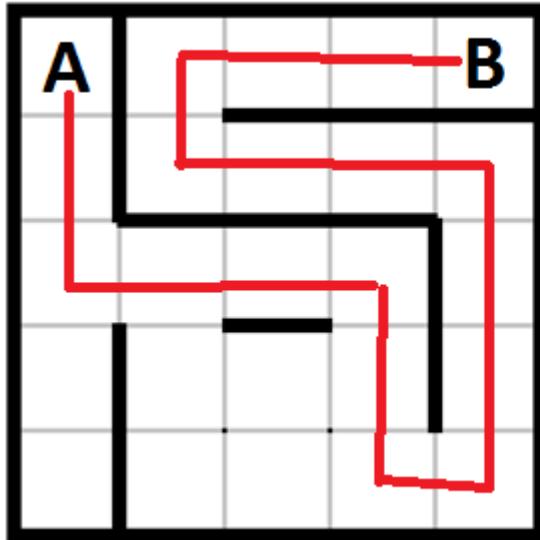
Prejst' bludiskom znamená prejst' grafom

- Prehľadávanie grafu je úloha v ktorej chceme navštíviť každý vrchol (príp. každú hranu)
- Riešime ale prechod bludiskom (z bodu A do bodu B) ...
 - Môžeme použiť prehľadávanie do hĺbky?
 - Akú dostaneme cestu? Nejakú.



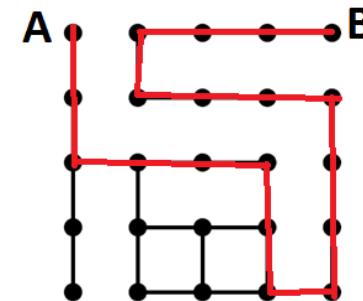
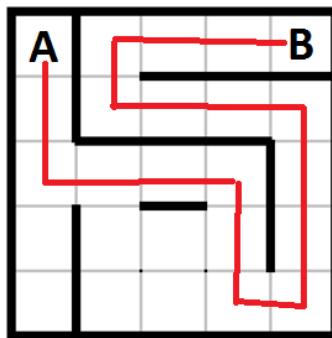
Najkratšia cesta v bludisku

- Ako nájdem v bludisku najkratšiu cestu z bodu A do B?
- Neorientovaný neohodnotený graf
 - Prehľadávanie do šírky



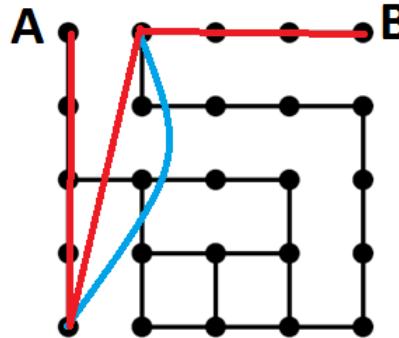
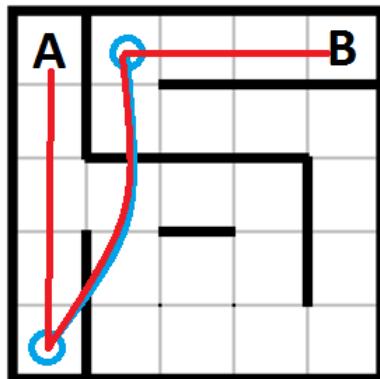
Čo ked' by sme chceli ešte kratšiu cestu?

- Najkratšia cesta:



- Variant úlohy:

- existuje **teleportovacia miestnosť**, môžeme „skočiť“ na vzdialené poličko“
- Úloha: **Ako sa zmení graf ?**



Stačí pridať hranu pre každý teleport

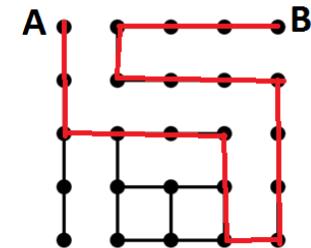
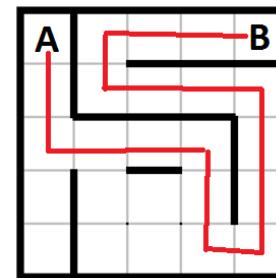
Čo ked' by sme chceli ešte kratšiu cestu? (2)

- Najkratšia cesta:

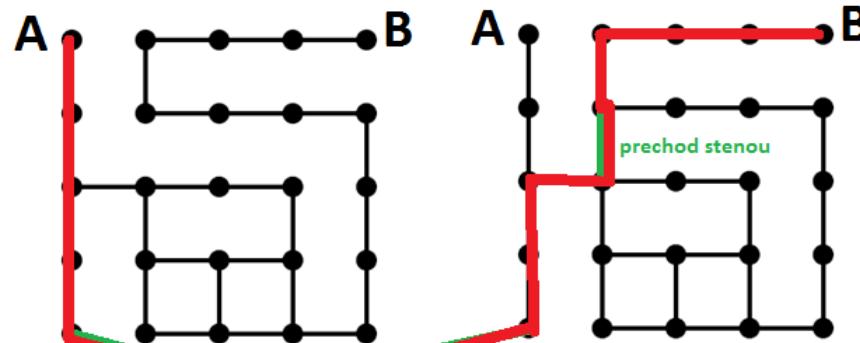
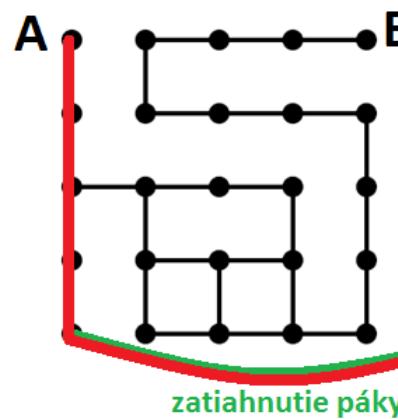
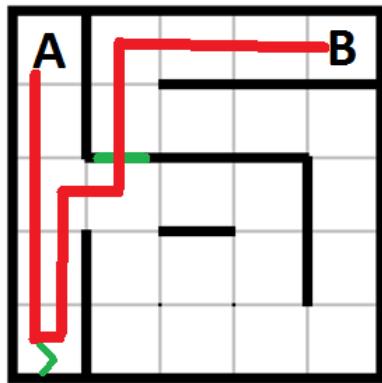


- Variant úlohy:

- nájdeme **páku**, ktorá otvorí tajné dvere



- Ako sa zmení graf, aby nám stačilo jedno prehľadávanie ?

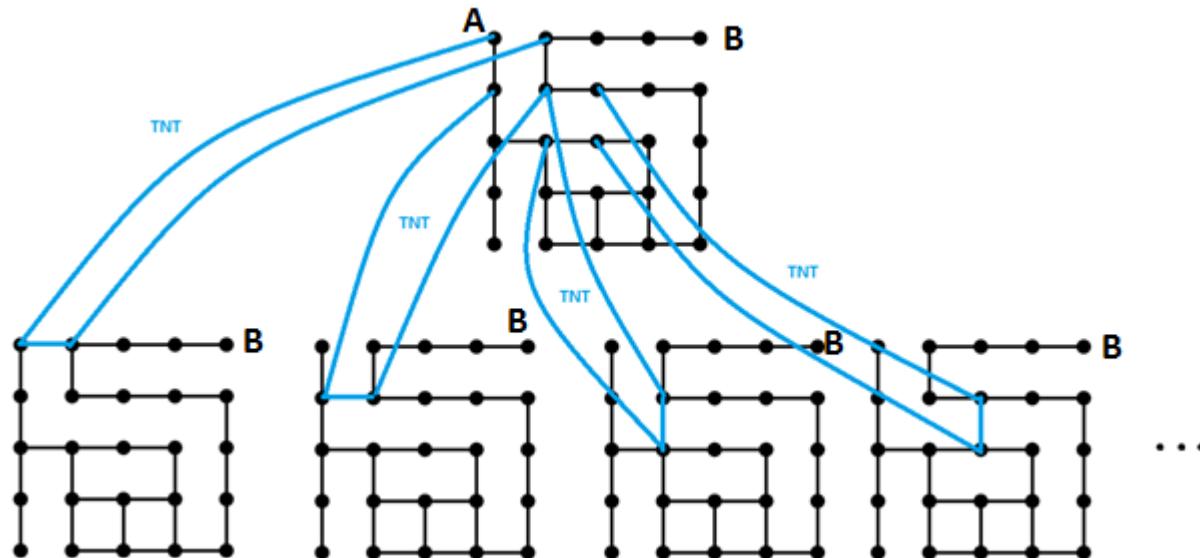
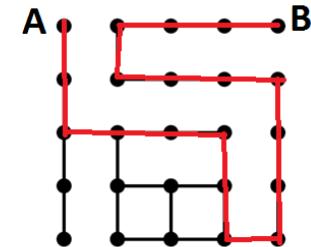
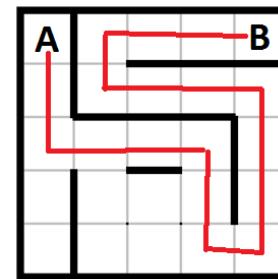
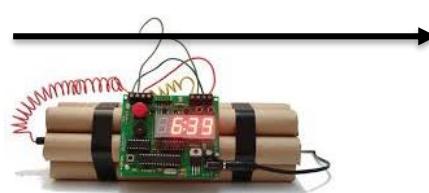


- Zdvojíme graf:

- Prvá časť je pôvodné bludisko (prechod do druhej časti je zatiahnutie páky)
- Druhá časť obsahuje chodby bludiska po zatiahnutí páky.

Čo ked' by sme chceli ešte kratšiu cestu? (3)

- Najkratšia cesta:
- Variant úlohy:
 - máme **dynamit**, môžeme jednu stenu odstrelit'
 - Ako vytvorit' graf, aby nám stačilo jedno prehľadávanie ?



- Pre každú stenu: vytvor kópiu bludiska s prechodom cez zbúranú stenu
 - prechod z pôvodného bludiska do zbúranej časti zodpovedá použitiu dynamitu