

Strategie řešení problémů hrubou silou a úplným prohledáváním

doc. Mgr. Jiří Dvorský, Ph.D.

Stav prezentace ke dni 28. dubna 2024

Katedra informatiky

Fakulta elektrotechniky a informatiky

VŠB – TU Ostrava



Strategie řešení problémů hrubou silou a úplným prohledáváním

Třídící algoritmy

Třídění výběrem – SelectSort

Bublinové třídění – BubbleSort

Sekvenční vyhledávání

Vyhledávání podřetězce hrubou silou

Problém nejbližší dvojice bodů

Konvexní obal množiny

Úplné prohledávání

Osnova přednášky (pokrač.)

Problém obchodního cestujícího

Problém batohu

Průchody grafem

Průchod grafem do hloubky

Průchod grafem do šířky

Charakteristika

Strategie řešení problémů hrubou silou je založena na **přímočarém přístupu k řešení problému**, kdy algoritmus řešení vychází přímo ze zadání problému a pojmů obsažených v zadání.

Výpočet mocniny

Máme vypočítat mocninu a^n pro nenulové a a n přirozené číslo. Z definice mocnění platí

$$a^n = \underbrace{a \times a \times \cdots \times a}_{n \text{ krát}}$$

Řešení hrubou silou – vynásobíme číslo a mezi sebou $n - 1$ krát.

Další příklady řešení hrubou silou:

- výpočet NSD postupným dělením a
- násobení matic podle algoritmu z předchozí prezentace.

Strategie řešení problémů hrubou silou

1. obecná strategie – je obtížné najít problém, kde by „nezabrala“,
2. obecně sice nevede k efektivním algoritmům, ale pro některé problémy, např. násobení matic, pattern matching, jsou algoritmy založené na této strategii použitelné i pro větší vstupy,
3. přijatelná strategie v případě, kdy se nevyplatí zabývat se sofistikovanějším algoritmem – ad hoc řešení problému,
4. vždy použitelná strategie pro řešení problémů s malou velikostí vstupu a
5. význam i jako měřítko, kterým můžeme poměřovat efektivnější algoritmy řešící stejný problém.

Strategie řešení problémů hrubou silou
a úplným prohledáváním
Třídící algoritmy

- Máme dáno pole n prvků pro které je definována relace uspořádání (čili vztah „menší než“), pro ukázkou vezměme celé čísla.
- Úkolem je přeuspořádání prvků pole do neklesající posloupnosti – prvek pole na nižším indexu musí být menší nebo roven prvku na vyšším indexu.
- Otázka zní: existuje třídící algoritmus řešící problém hrubou silou, zcela přímočaře?

Třídění výběrem – SelectSort

Výchozí úvaha

Otázka: O kterém prvku z pole víme přesně kam patří?

Odpověď: O nejmenším! Patří na začátek pole, na nejnižší index! (Pozn. Totéž platí pro největší prvek.)

Princip algoritmu

1. Vybereme z n prvků pole nejmenší prvek a zaměníme jej s prvním prvkem pole.
2. Vybereme ze zbylých $n - 1$ prvků pole nejmenší prvek a zaměníme jej s druhým prvkem pole.
3. Obecně v i -tém kroku vybereme ze zbylých $n - i$ nejmenší prvek a zaměníme ho s i -tým prvkem.
4. Po $n - 1$ krocích je pole setříděno.

Třídění výběrem – ukázka

89	45	68	90	29	34	17
17	45	68	90	29	34	89
17	29	68	90	45	34	89
17	29	34	90	45	68	89
17	29	34	45	90	68	89
17	29	34	45	68	90	89
17	29	34	45	68	89	90

Třídění výběrem – pseudokód

Vstup : Pole $A[0 \dots n - 1]$ s definovaným uspořádáním
na prvcích pole

Výstup: Setříděné pole A

```
1 for  $i \leftarrow 0$  to  $n - 2$  do
2   |  $min \leftarrow i$ ;
3   | for  $j \leftarrow i + 1$  to  $n - 1$  do
4   |   | if  $A[j] < A[min]$  then
5   |   |   |  $min \leftarrow j$ ;
6   |   | end
7   | end
8   | Swap ( $A[i], A[min]$ );
9 end
```

Funkce pro výměnu hodnot dvou proměnných

1 Procedure *Swap*(*x*, *y*)

 Vstup : Parametry *x* a *y* shodného datového typu

 Výstup: Navzájem přehozené hodnoty *x* a *y*

2 *aux* ← *x*;

3 *x* ← *y*;

4 *y* ← *aux*;

5 end

Třídění výběrem – analýza

1. Velikost vstupu – počet prvků n .
2. Základní operace – porovnání prvků (někdy i počet výměn prvků).
3. Počet základních operací závisí pouze na n – nejhorší, nejlepší a průměrný případ splývají.
4. Sestavení vztahů

$$\begin{aligned}C(n) &= \sum_{i=0}^{n-2} \sum_{j=i+1}^{n-1} 1 = \sum_{i=0}^{n-2} [(n-1) - (i+1) + 1] \\ &= \sum_{i=0}^{n-2} (n-1-i) = \frac{1}{2}n(n-1) \approx \frac{1}{2}n^2 = \Theta(n^2)\end{aligned}$$

Poznámky

- Podrobný výpočet sumy lze najít v příkladu „Unikátnost prvků“ v minulé lekci.
- Složitost algoritmu nijak nezávisí na míře neseříděnosti vstupního pole. Algoritmus tedy **není přirozený**.
- Algoritmus je ale **stabilní** – jako minimum je vždy brán první prvek z několika shodných.
- Algoritmus je **in situ** – vystačíme si s konstantním rozsahem paměti navíc.

Animace

K algoritmu třídění výběrem je k dispozici animace v samostatném souboru.

Bublinové třídění – BubbleSort

- Vezmu dvojici sousedních prvků A_i a A_{i+1} .
- Pokud jsou v nesprávném pořadí, tak je vyměním.
- Zvýším i o jedna a pokračuji další dvojicí prvků.
- Po každém průchodu polem A se dostane jeden prvek určitě na své místo.

$$A_0, \dots, A_j \leftrightarrow A_{j+1}, \dots, A_{n-i-1} \mid \underbrace{A_{n-i} \leq \dots \leq A_{n-1}}_{\text{setříděno}}$$

V dalším průchodu polem už procházím o jeden prvek méně.

- Proč bublinové třídění – největší prvek z nesetříděné části takto „probublá“ směrem ke konci pole.

Bublinové třídění – ukázka

89	↔	45		68		90		29		34		17
45		89	↔	68		90		29		34		17
45		68		89	↔	90		29		34		17
45		68		89		90	↔	29		34		17
45		68		89		29		90	↔	34		17
45		68		89		29		34		90	↔	17
45		68		89		29		34		17		90
45	↔	68	↔	89	↔	29		34		17		90
45		68		29		89	↔	34		17		90
45		68		29		34		89	↔	17		90
45		68		29		34		17		89		90

a tak dále

Bublinové třídění – pseudokód

Vstup : Pole $A[0 \dots n - 1]$ s definovaným uspořádáním
na prvcích pole

Výstup: Setříděné pole A

```
1 for  $i \leftarrow 0$  to  $n - 2$  do
2   | for  $j \leftarrow 0$  to  $n - i - 2$  do
3     | | if  $A[j] > A[j + 1]$  then
4       | |   |  $Swap(A[j], A[j + 1]);$ 
5       | |   | end
6     | | end
7   | end
8 end
```

Bublinové třídění – analýza

1. Velikost vstupu – počet prvků n .
2. Základní operace – porovnání prvků.
3. Počet základních operací závisí pouze na n – nejhorší, nejlepší a průměrný případ splývají.
4. Sestavení vztahů

$$\begin{aligned}C(n) &= \sum_{i=0}^{n-2} \sum_{j=0}^{n-i-2} 1 = \sum_{i=0}^{n-2} [(n-i-2) - 0 + 1] \\ &= \sum_{i=0}^{n-2} (n-1-i) = \frac{1}{2}n(n-1) \approx \frac{1}{2}n^2 = \Theta(n^2)\end{aligned}$$

Bublinové třídění – analýza (pokrač.)

Počet základních operací $C(n)$ je shodný s tříděním výběrem.

Počet vzájemných výměn prvků $S(n)$ ale už na vstupu závisí a je v nejhorším případě roven

$$S_{\text{worst}}(n) = C(n) = \frac{1}{2}n(n - 1) \in \Theta(n^2)$$

Bublinové třídění – další vlastnosti

- Algoritmus je **in situ** – vystačíme si s konstantním rozsahem paměti navíc.
- Algoritmus je **stabilní** – k výměně dojde jen pokud je $A[j] > A[j + 1]$, jinak ne.
- Složitost této verze algoritmu nijak nezávisí na míře nesetříděnosti vstupního pole. Algoritmus tedy **není přirozený**.
- Modifikované verze (viz dále) už ale přirozené jsou.

- Počet opakování vnějšího cyklu by měl záviset na provedení či neprovedení výměny prvků.
- Pokud se během vnitřního cyklu neprovedla žádná výměna prvků není nutné toto pole dále procházet – pole je evidentně setříděno.
- Zavedeme si logickou proměnnou ***AnySwap***, kam si budeme ukládat informaci o případné výměně.

Bublinové třídění – alternativní konstrukce algoritmu I

Vstup : Pole $A[0 \dots n - 1]$ s definovaným uspořádáním
na prvcích pole

Výstup: Setříděné pole A

```
1 do
2   AnySwap ← false;
3   for  $i \leftarrow 0$  to  $n - 2$  do
4     if  $A[i] > A[i + 1]$  then
5       Swap ( $A[i], A[i + 1]$ );
6       AnySwap ← true;
7     end
8   end
9 while AnySwap;
```

- Kombinovaná konstrukce algoritmu:
 - nedošlo k výměně prvků a
 - zkracování posloupnosti o jeden prvek zprava.
- Zavedeme si logickou proměnnou **AnySwap**, kam si budeme ukládat informaci o případné výměně.
- Proměnná **Right** bude označovat pravý okraj tříděné posloupnosti.

Bublinové třídění – alternativní konstrukce algoritmu II

Vstup : Pole $A[0 \dots n - 1]$ s definovaným uspořádáním
na prvcích pole

Výstup: Setříděné pole A

```
1 Right ←  $n - 2$ ;  
2 do  
3   AnySwap ← false;  
4   for  $i \leftarrow 0$  to Right do  
5     if  $A[i] > A[i + 1]$  then  
6       Swap ( $A[i], A[i + 1]$ );  
7       AnySwap ← true;  
8     end  
9   end  
10  Right ← Right - 1;  
11 while AnySwap;
```

Proměnná *LastSwapIndex*

- bude označovat index poslední výměny prvků,
- jinak řečeno pravý okraj tříděné posloupnosti v následujícím průchodu polem.

Bublinové třídění – alternativní konstrukce algoritmu III

Vstup : Pole $A[0 \dots n - 1]$ s definovaným uspořádáním
na prvcích pole

Výstup: Setříděné pole A

```
1 Right ← n - 2;  
2 do  
3   | LastSwapIndex ← 0;  
4   | for i ← 0 to Right do  
5   |   | if  $A[i] > A[i + 1]$  then  
6   |   |   | Swap ( $A[i], A[i + 1]$ );  
7   |   |   | LastSwapIndex ← i + 1;  
8   |   | end  
9   | end  
10  | Right ← LastSwapIndex;  
11 while LastSwapIndex > 0;
```

Animace

K algoritmu bublinového třídění je k dispozici animace v samostatném souboru. V animaci je znázorněn algoritmus podle snímku 222.

Třídění přetřásáním – ShakerSort

U BubbleSortu lze pozorovat dva efekty:

1. „velké“ prvky, **zajíci**, se velice rychle posunují ke konci pole, ale
2. „malé“ prvky, **želvy**, se posunují na začátek pole jen jako důsledek rychlého přesunu velkých prvků.

Modifikace BubbleSortu:

- pole budeme procházet střídavě z obou stran
- po každém průchodu pole se vymění role zajíců a želv, cimrmanovská „výměna mečů“ ze hry Blaník
- **ShakerSort** – pole je „přestřásáno, setřepáváno“ jako v barovém shakeru.

Třídění přetřásáním – ukázka

89	↔	45		68		90		29		34		17
45		89	↔	68		90		29		34		17
45		68		89	↔	90		29		34		17
45		68		89		90	↔	29		34		17
45		68		89		29		90	↔	34		17
45		68		89		29		34		90	↔	17
45		68		89		29		34		17		90
45		68		89		29		34	↔	17		90
45		68		89		29	↔	17		34		90
45		68		89	↔	17		29		34		90
45		68	↔	17		89		29		34		90
45	↔	17		68		89		29		34		90
17		45		68		89		29		34		90

a tak dále

Třídění přetřásáním – pseudokód

Vstup : Pole $A[0 \dots n - 1]$ s definovaným uspořádáním
na prvcích pole

Výstup: Setříděné pole A

```
1 Left ← 0;  
2 Right ←  $n - 1$ ;  
3 do  
4   |   LeftToRight (A, Left, Right);  
5   |   RightToLeft (A, Left, Right);  
6 while Left < Right;
```

Třídění přetřásáním – pseudokód (pokrač.)

```
1 Procedure LeftToRight(A, Left, Right)
2    $j \leftarrow 0$ ;
3   for  $i \leftarrow \textit{Left}$  to  $\textit{Right} - 1$  do
4     if  $A[i] > A[i + 1]$  then
5       Swap ( $A[i]$ ,  $A[i + 1]$ );
6        $j \leftarrow i$ ;
7     end
8   end
9    $\textit{Right} \leftarrow j$ ;
10 end
```


Třídění přetřásáním – pseudokód (pokrač.)

```
1 Procedure RightToLeft(A, Left, Right)
2    $j \leftarrow 0$ ;
3   for  $i \leftarrow \textit{Right}$  downto  $\textit{Left} + 1$  do
4     if  $A[i - 1] > A[i]$  then
5        $\textit{Swap}(A[i - 1], A[i])$ ;
6        $j \leftarrow i$ ;
7     end
8   end
9    $\textit{Left} \leftarrow j$ ;
10 end
```

Animace

K algoritmu třídění přetřásáním je k dispozici animace v samostatném souboru.

Strategie řešení problémů hrubou silou
a úplným prohledáváním
Sekvenční vyhledávání

Sekvenční vyhledávání

- Typická ukázka strategie řešení hrubou silou.
- Silná stránka algoritmu – jednoduchost (simplicity).
- Slabá stránka algoritmu – vysoká složitost.

Vstup : Pole $A[0 \dots n - 1]$ a hledaný prvek x

Výstup: Index prvního výskytu prvku x v poli A , jinak -1

```
1 for  $i \leftarrow 0$  to  $n - 1$  do
2   |   if  $A[i] = x$  then
3     |   |   return  $i$ ;
4   |   end
5 end
6 return -1;
```

Algoritmus bývá také označován jako **lineární vyhledávání**.

Lineární vyhledávání – složitost

	Počet porovnání
Nejhorší případ	n
Nejlepší případ	1
Průměrný případ	$\frac{1}{2}p(n + 1) + n(1 - p)$

kde p je pravděpodobnost úspěšného vyhledání

Lineární vyhledávání – využití záložky (angl. sentinel)

Vstup : Pole $A[0 \dots n]$ a hledaný prvek x

Výstup: Index prvního výskytu prvku x v poli A , jinak -1

```
1  $A[n] \leftarrow x$ ;  
2  $i \leftarrow 0$ ;  
3 while  $A[i] \neq x$  do  
4   |  $i \leftarrow i + 1$ ;  
5 end  
6 if  $i < n$  then return  $i$ ;  
7 return -1;
```

Strategie řešení problémů hrubou silou a úplným prohledáváním

Vyhledávání podřetězce hrubou silou

Vyhledávání podřetězce hrubou silou

Zadání – najít vzorek p v textu t .

Řešení hrubou silou

1. Přiložíme vzorek na začátek textu.
2. Začneme porovnávat znaky ve vzorku a textu.
3. Pokud se shodují všechny znaky vzorku s textem – nalezeno.
4. Pokud nalezneme neshodu, posuneme vzorek o jednu pozici dopředu a pokračujeme bodem 2

$$\begin{array}{ccccccc} t_0 & \cdots & t_i & \cdots & t_{i+j} & \cdots & t_{i+m-1} & \cdots & t_{n-1} \\ & & \downarrow & & \downarrow & & \downarrow & & \\ & & p_0 & \cdots & p_j & \cdots & p_{m-1} & & \end{array}$$

Vyhledávání hrubou silou – pseudokód

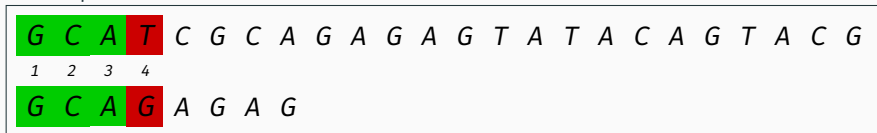
Vstup : Vzorek p , text t a počáteční pozice s

Výstup: Pozice prvního výskytu p v textu t nebo -1

```
1 for  $i \leftarrow s$  to  $|t| - |p|$  do
2    $j \leftarrow 0$ ;
3   while  $j < |p|$  do
4     if  $p[j] \neq t[i + j]$  then break;
5      $j \leftarrow j + 1$ ;
6   end
7   if  $j = |p|$  then return  $i$ ;
8 end
9 return -1;
```

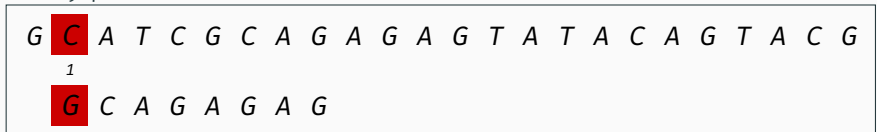
Vyhledávání hrubou silou – příklad

První pokus



Posun o 1 znak

Druhý pokus



Posun o 1 znak

Vyhledávání hrubou silou – příklad (pokrač.)

Třetí pokus

G	C	A	T	C	G	C	A	G	A	G	A	G	T	A	T	A	C	A	G	T	A	C	G
		1																					
		G	C	A	G	A	G	A	G														

Posun o 1 znak

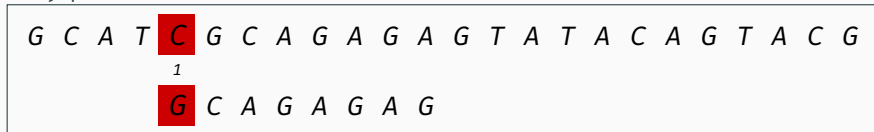
Čtvrtý pokus

G	C	A	T	C	G	C	A	G	A	G	A	G	T	A	T	A	C	A	G	T	A	C	G
			1																				
			G	C	A	G	A	G	A	G													

Posun o 1 znak

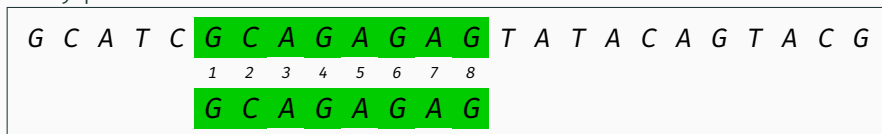
Vyhledávání hrubou silou – příklad (pokrač.)

Pátý pokus



Posun o 1 znak

Šestý pokus



Posun o 1 znak

Vyhledávání hrubou silou – příklad (pokrač.)

Sedmý pokus

G	C	A	T	C	G	C	A	G	A	G	A	G	T	A	T	A	C	A	G	T	A	C	G	
						1																		
						G	C	A	G	A	G	A	G											

Posun o 1 znak

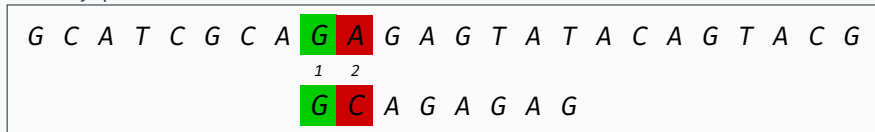
Osmý pokus

G	C	A	T	C	G	C	A	G	A	G	A	G	T	A	T	A	C	A	G	T	A	C	G	
							1																	
							G	C	A	G	A	G	A	G										

Posun o 1 znak

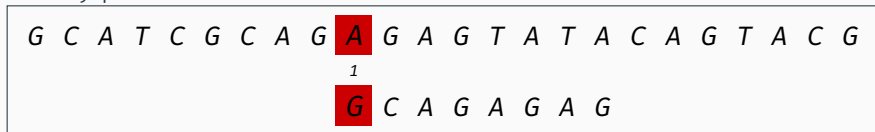
Vyhledávání hrubou silou – příklad (pokrač.)

Devátý pokus



Posun o 1 znak

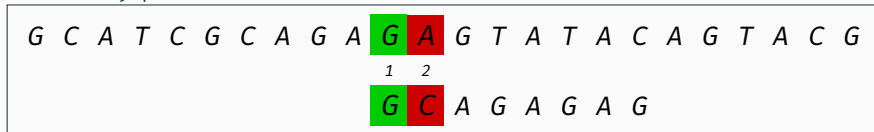
Desátý pokus



Posun o 1 znak

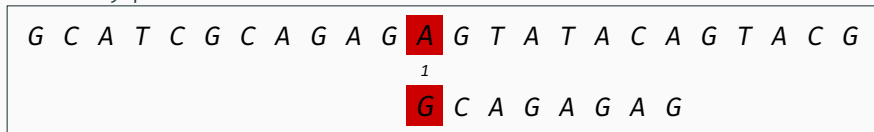
Vyhledávání hrubou silou – příklad (pokrač.)

Jedenáctý pokus



Posun o 1 znak

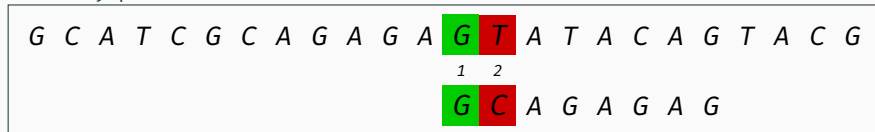
Dvanáctý pokus



Posun o 1 znak

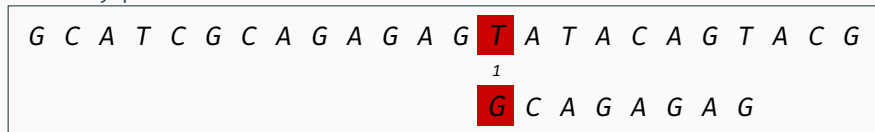
Vyhledávání hrubou silou – příklad (pokrač.)

Třináctý pokus



Posun o 1 znak

Čtrnáctý pokus



Posun o 1 znak

Vyhledávání hrubou silou – příklad (pokrač.)

Patnáctý pokus

G	C	A	T	C	G	C	A	G	A	G	A	G	T	A	T	A	C	A	G	T	A	C	G	
														1										
														G	C	A	G	A	G	A	G			

Posun o 1 znak

Šesnáctý pokus

G	C	A	T	C	G	C	A	G	A	G	A	G	T	A	T	A	C	A	G	T	A	C	G	
															1									
															G	C	A	G	A	G	A	G		

Posun o 1 znak

Vyhledávání hrubou silou – příklad (pokrač.)

Sedmnáctý pokus

G	C	A	T	C	G	C	A	G	A	G	A	G	T	A	T	A	C	A	G	T	A	C	G	
																	1							
																	G	C	A	G	A	G	A	G

Posun o 1 znak

Algoritmus provedl celkem 30 porovnání znaků.

Vyhledávání podřetězce hrubou silou – složitost algoritmu

- Velikost vstupu – délka textu n a délka vzorku m .
- Základní operace – porovnání znaku vzorku a textu.
- Závislost jen na velikosti vstupu – ne, záleží i na tom, kdy se najde první neshoda.
- Nejhorší případ – text $a^{n-1}b$, vzorek $a^{m-1}b$
 - v každém pokusu provedeme všech m porovnání vzorku s textem
 - současně provedeme všech $n - m + 1$ pokusů.
 - Celkem provedeme $m(n - m + 1)$ porovnání, algoritmus spadá do $O(mn)$.
- Nejlepší případ – vzorek je nalezen na začátku textu, složitost $O(m)$.
- Přirozené jazyky – posun nastane po „několika“ (k_L) porovnáních, nejhorší složitost $O(k_L n) = O(n)$.

Strategie řešení problémů hrubou silou
a úplným prohledáváním
Problém nejbližší dvojice bodů

Problém nejbližší dvojice bodů

Zadání problému

Nalezněte dva navzájem nejbližší body z množiny n bodů.

- Jde o jeden z problémů **výpočetní geometrie**.
- Body mohou ležet na rovině nebo obecně v nějakém mnohodomenzionálním prostoru.
- Body mohou reprezentovat objekty reálného světa nebo záznamy v databázi, texty...
- Příklady aplikací:
 - Bezpečnost letového provozu – hledáme dvě nejbližší letadla ve vzdušném prostoru.
 - Shlukování – hierarchické shlukovací algoritmy postupně spojují sobě nejbližší shluky do jednoho, většího shluku.

Problém nejbližší dvojice bodů – předpoklady

Předpokládejme množinu n bodů $\{P_1, \dots, P_n\}$, každému bodu P_i odpovídá vektor \vec{p}_i se složkami

$$\vec{p}_i = (x_i, y_i)$$

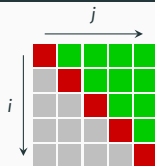
v obvyklých kartézských souřadnicích.

Vzdálenost bodů \vec{p}_i a \vec{p}_j budeme počítat pomocí Euklidovské vzdálenosti

$$d(\vec{p}_i, \vec{p}_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

Problém nejbližší dvojice bodů – řešení hrubou silou

- Vypočteme vzdálenost všech dvojic bodů \vec{p}_i a \vec{p}_j a nalezneme minimum.
- Stačí počítat jen dvojice bodů pro $j = i + 1, \dots, n$.



Vstup : Množina bodů $\{\vec{p}_1, \vec{p}_2, \dots, \vec{p}_n\}$

Výstup: Vzdálenost dvou nejbližších bodů

```
1 MinDist ← ∞;  
2 for i ← 1 to n - 1 do  
3   | for j ← i + 1 to n do  
4   |   | MinDist ← min(MinDist,  $\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$ );  
5   |   | end  
6   | end  
7 return MinDist;
```

1. Velikost vstupu – počet bodů n
2. Základní operace
 - Výpočet odmocniny – nejde o triviální záležitost¹.
 - Výpočtu odmocniny se lze vyhnout – jde o rostoucí funkci, lze hledat minimum „čtverců“ vzdáleností.
 - Za základní operaci vezmeme umocňování rozdílů souřadnic.
3. Počet základních operací závisí pouze na n – nejhorší, nejlepší a průměrný případ splývají.

Problém nejbližší dvojice bodů – řešení hrubou silou, složitost (pokrač.)

4. Sestavení vztahů

$$\begin{aligned}C(n) &= \sum_{i=1}^{n-1} \sum_{j=i+1}^n 2 = 2 \sum_{i=1}^{n-1} (n - i) \\ &= 2[(n - 1) + (n - 2) + \dots + 1] = (n - 1)n \in \Theta(n^2)\end{aligned}$$

5. Odstranění odmocniny – snížení složitosti o konstantní faktor, nedošlo k zlepšení asymptotické složitosti, stále je to $\Theta(n^2)$ algoritmus.

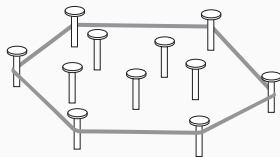
6. Později si ukážeme algoritmus s lineárně logaritmickou složitostí.

¹https://en.wikipedia.org/wiki/Methods_of_computing_square_roots

Strategie řešení problémů hrubou silou
a úplným prohledáváním
Konvexní obal množiny

Zadání problému

Úkolem je najít konvexní obal (obálku) množiny bodů v prostoru.



- Jde o jeden z problémů **výpočetní geometrie**.
- Body mohou ležet na rovině nebo obecně v nějakém mnohodomenzionálním prostoru.
- Příklady aplikací:
 - Detekce kolizí – počítačová grafika, autonomní vozidla,
 - GIS – bodové senzory, vytvoření oblasti z těchto dat,
 - Optimalizační úlohy – vrcholy konvexního obalu jsou jistým způsobem extrémní; konvexní mnohoúhelník vzniká jako průnik konečného množství polorovin; polorovina je definována nerovnicí...

Definice

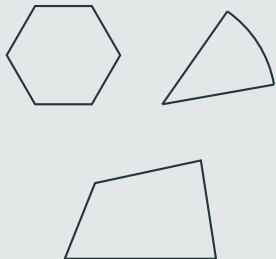
Množina bodů M v rovině se nazývá **konvexní**, jestliže pro libovolnou dvojici bodů $p, q \in M$ úsečka spojující body p a q náleží do množiny M .

Množina, které není konvexní se nazývá **nekonvexní**.

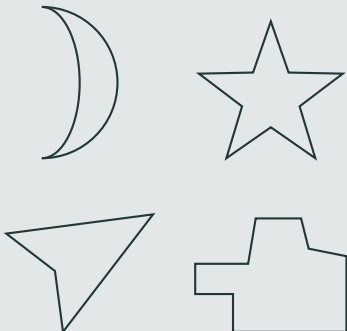
Představíme-li si hranici množiny jako neprůhlednou, znamená konvexita množiny názorně to, že z každého jejího bodu je vidět každý její bod.

Konvexní množina bodů – příklady

Konvexní útvary



Nekonvexní útvary



Definice

Konvexním obalem množiny bodů M nazýváme nejmenší konvexní množinu, která obsahuje M .

Výraz „nejmenší“ znamená, že konvexní obal množiny M musí být podmnožinou jakékoliv jiné konvexní podmnožiny obsahující množinu M .

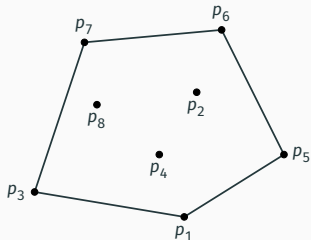
Konvexní obal

- dvouprvkové množiny – úsečka spojující oba body.
- tříprvkové množiny – trojúhelník, pokud body neleží na přímce, jinak je to úsečka spojující nejvzdálenější body.

Konvexní obal (pokrač.)

Věta

Konvexní obal množiny bodů M s více než dvěma body, které neleží na jedné přímce, je konvexní mnohoúhelník, jehož vrcholy náležejí do M .

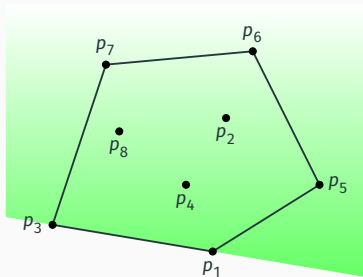
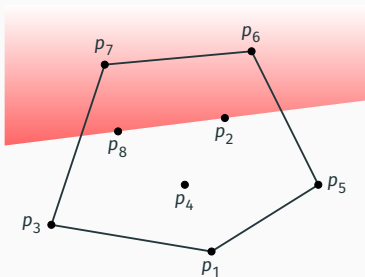


Body množiny M , které specifikují konvexní obal M se nazývají **extrémní body**.

Problém nalezení konvexního obalu množiny M redukuje se na nalezení extrémních bodů.

Konvexní obal – řešení hrubou silou

Úsečka $\vec{p}_i\vec{p}_j$ náleží do konvexního obalu množiny M , právě když všechny ostatní body z M leží v jedné z polorovin definovaných přímkou $\vec{p}_i\vec{p}_j$.



Obecnou rovnici přímky procházející body \vec{p}_i a \vec{p}_j lze psát jako

$$ax + by + c = 0,$$

kde

$$a = y_j - y_i$$

$$b = x_i - x_j$$

$$c = y_i x_j - x_i y_j$$

Přímka definuje dvě poloroviny:

$$ax + by + c < 0 \quad (14)$$

$$ax + by + c > 0 \quad (15)$$

Stačí tedy ověřit, zda pro zbývajících $n - 2$ bodů platí buď nerovnice (14) nebo (15).

- Musíme prověřit všech $\frac{1}{2}n(n - 1)$ dvojic bodů a současně
- pro každou přímku definovanou jednou dvojicí bodů musíme ověřit platnost nerovnic (14) a (15) pro zbývajících $n - 2$ bodů.
- Celkově tedy $\left[\frac{1}{2}n(n - 1)\right](n - 2) \in O(n^3)$.

Strategie řešení problémů hrubou silou a úplným prohledáváním

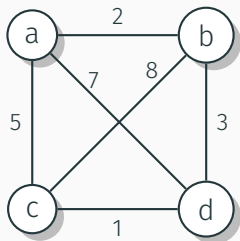
Úplné prohledávání

Úplné prohledávání (Exhaustive search)

- Součástí řešení mnoha problémů – nalezení **jednoho prvku**, s nějakou **specifickou vlastností**, z množiny prvků, tzv. domény, která **exponenciálně** nebo rychleji roste s velikostí problému.
- Hledaný element je typicky **kombinatorické povahy** – permutace, kombinace, podmnožina.
- Typicky jde o **optimalizační úlohy** – typicky hledáme maximum, minimum. Například minimalizujeme délku cesty, maximalizujeme zisk.

Úplné prohledávání je strategie řešení problému hrubou silou spočívající v otestování všech prvků uvažované domény.

Problém obchodního cestujícího – příklad



Trasa	Délka trasy l
$a \rightarrow b \rightarrow c \rightarrow d \rightarrow a$	$2 + 8 + 1 + 7 = 18$
$a \rightarrow b \rightarrow d \rightarrow c \rightarrow a$	$2 + 3 + 1 + 5 = 11$
$a \rightarrow c \rightarrow b \rightarrow d \rightarrow a$	$5 + 8 + 3 + 7 = 23$
$a \rightarrow c \rightarrow d \rightarrow b \rightarrow a$	$5 + 1 + 3 + 2 = 11$
$a \rightarrow d \rightarrow b \rightarrow c \rightarrow a$	$7 + 3 + 8 + 5 = 23$
$a \rightarrow d \rightarrow c \rightarrow b \rightarrow a$	$7 + 1 + 8 + 2 = 18$



Problém obchodního cestujícího (Traveling Salesman Problem)





Strategie řešení problémů hrubou silou a úplným prohledáváním

Průchody grafem

Průchod grafem do hloubky a do šířky



Algoritmus průchodu grafem do hloubky

Input : Graf $G(V, E)$, počáteční vrchol $s \in V$

Output: DF-strom

```
1 Init( $V, s$ );
2 while stack  $\neq \emptyset$  do
3    $u \leftarrow \text{Top}(\text{stack})$ ;
4   switch state [ $u$ ] do
5     case discovered do
6       | ProcessDiscoveredVertex( $u$ );
7     end
8     case current do
9       | ProcessCurrentVertex( $u$ );
10    end
11  end
12 end
```

Algoritmus průchodu grafem do hloubky (pokrač.)

```
1 Procedure Init( $V, s$ )
  Input : Množina vrcholů  $V$ , počáteční vrchol  $s \in V$ 
2  foreach  $u \in V$  do
3    state [ $u$ ]  $\leftarrow$  unknown;
4    d [ $u$ ]  $\leftarrow$  f [ $u$ ]  $\leftarrow$   $\infty$ ;
5     $\pi$  [ $u$ ]  $\leftarrow$  nothing;
6  end
7  state [ $s$ ]  $\leftarrow$  discovered;
8  stack  $\leftarrow$   $\emptyset$ ;
9  Push(stack,  $s$ );
10 time  $\leftarrow$  0;
11 end
```

Algoritmus průchodu grafem do hloubky (pokrač.)

```
1 Procedure ProcessDiscoveredVertex(u)
  Input : Vrchol  $u \in V$ 
2   state [u]  $\leftarrow$  current;
3   d [u]  $\leftarrow$  time  $\leftarrow$  time + 1;
4   foreach  $v \in Adj(G, u)$  do
5     if state [v] = unknown then
6       state [v]  $\leftarrow$  discovered;
7        $\pi$  [v]  $\leftarrow$  u;
8       Push(stack, v);
9     end
10  end
11 end
```

Algoritmus průchodu grafem do hloubky (pokrač.)

```
1 Procedure ProcessCurrentVertex ( $u$ )  
   Input : Vrchol  $u \in V$   
2   state [ $u$ ]  $\leftarrow$  finished;  
3   f [ $u$ ]  $\leftarrow$  time  $\leftarrow$  time + 1;  
4   Pop(stack);  
5 end
```

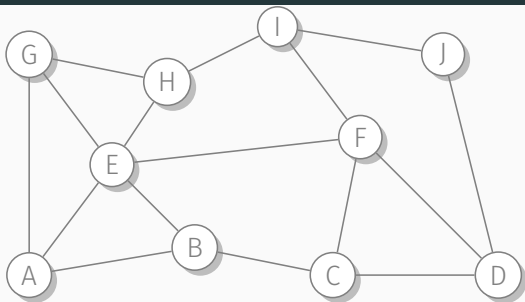
Vrcholy grafu:

- šedá vrchol ve stavu unknown
- žlutá vrchol ve stavu discovered
- červená vrchol ve stavu current
- modrá vrchol ve stavu finished

Hrany grafu:

- šedá hrana mezi vrcholy ve stavu unknown nebo hrana nepatřící do DF-stromu
- žlutá hrana incidentní s vrcholy ve stavu discovered
- červená hrana incidentní s vrcholem ve stavu current
- modrá hrana mezi vrcholy ve stavu finished

Průchod grafem do hloubky, krok 1

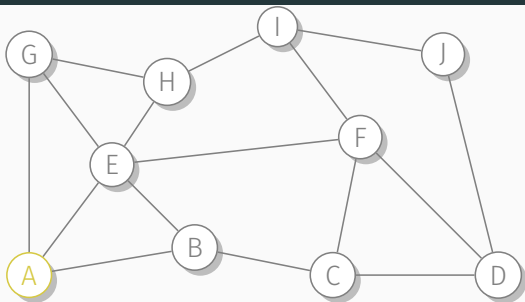


v	$d[v]$	$f[v]$	$\pi[v]$
A	∞	∞	/
B	∞	∞	/
C	∞	∞	/
D	∞	∞	/
E	∞	∞	/

v	$d[v]$	$f[v]$	$\pi[v]$
F	∞	∞	/
G	∞	∞	/
H	∞	∞	/
I	∞	∞	/
J	∞	∞	/

S

Průchod grafem do hloubky, krok 2

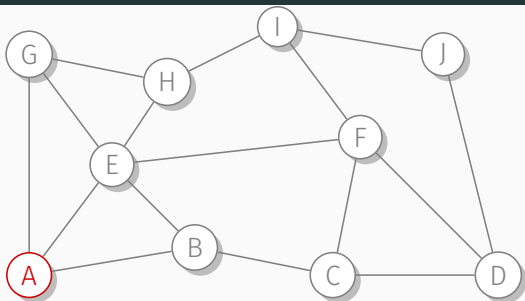


v	$d[v]$	$f[v]$	$\pi[v]$
A	∞	∞	/
B	∞	∞	/
C	∞	∞	/
D	∞	∞	/
E	∞	∞	/

v	$d[v]$	$f[v]$	$\pi[v]$
F	∞	∞	/
G	∞	∞	/
H	∞	∞	/
I	∞	∞	/
J	∞	∞	/



Průchod grafem do hloubky, krok 3

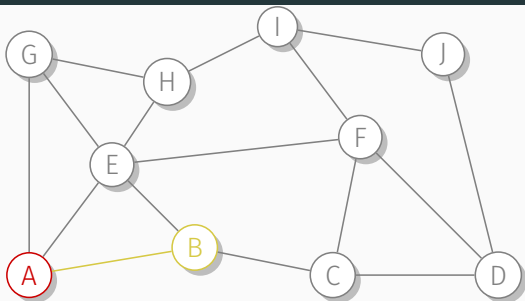


v	$d[v]$	$f[v]$	$\pi[v]$
A	1	∞	/
B	∞	∞	/
C	∞	∞	/
D	∞	∞	/
E	∞	∞	/

v	$d[v]$	$f[v]$	$\pi[v]$
F	∞	∞	/
G	∞	∞	/
H	∞	∞	/
I	∞	∞	/
J	∞	∞	/



Průchod grafem do hloubky, krok 4

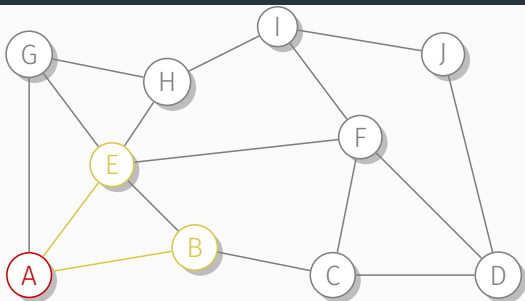


v	$d[v]$	$f[v]$	$\pi[v]$
A	1	∞	/
B	∞	∞	A
C	∞	∞	/
D	∞	∞	/
E	∞	∞	/

v	$d[v]$	$f[v]$	$\pi[v]$
F	∞	∞	/
G	∞	∞	/
H	∞	∞	/
I	∞	∞	/
J	∞	∞	/

B
A
S

Průchod grafem do hloubky, krok 5

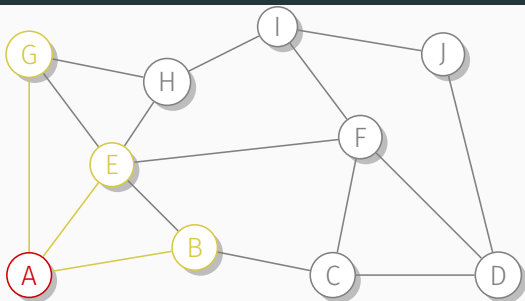


v	$d[v]$	$f[v]$	$\pi[v]$
A	1	∞	/
B	∞	∞	A
C	∞	∞	/
D	∞	∞	/
E	∞	∞	A

v	$d[v]$	$f[v]$	$\pi[v]$
F	∞	∞	/
G	∞	∞	/
H	∞	∞	/
I	∞	∞	/
J	∞	∞	/

E
B
A
S

Průchod grafem do hloubky, krok 6

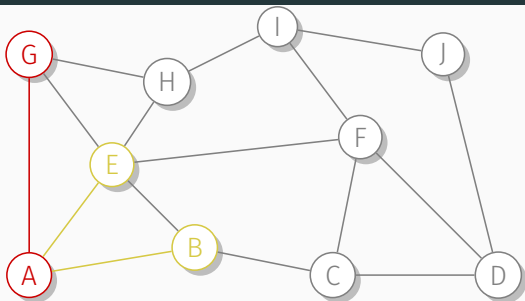


v	$d[v]$	$f[v]$	$\pi[v]$
A	1	∞	/
B	∞	∞	A
C	∞	∞	/
D	∞	∞	/
E	∞	∞	A

v	$d[v]$	$f[v]$	$\pi[v]$
F	∞	∞	/
G	∞	∞	A
H	∞	∞	/
I	∞	∞	/
J	∞	∞	/

G
E
B
A
S

Průchod grafem do hloubky, krok 7

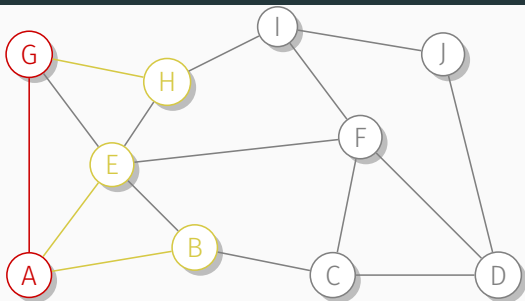


v	$d[v]$	$f[v]$	$\pi[v]$
A	1	∞	/
B	∞	∞	A
C	∞	∞	/
D	∞	∞	/
E	∞	∞	A

v	$d[v]$	$f[v]$	$\pi[v]$
F	∞	∞	/
G	2	∞	A
H	∞	∞	/
I	∞	∞	/
J	∞	∞	/

G
E
B
A
S

Průchod grafem do hloubky, krok 8

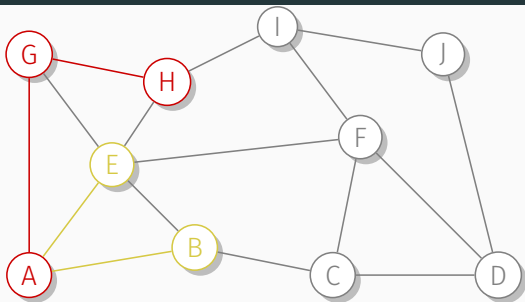


v	$d[v]$	$f[v]$	$\pi[v]$
A	1	∞	/
B	∞	∞	A
C	∞	∞	/
D	∞	∞	/
E	∞	∞	A

v	$d[v]$	$f[v]$	$\pi[v]$
F	∞	∞	/
G	2	∞	A
H	∞	∞	G
I	∞	∞	/
J	∞	∞	/

H
G
E
B
A
S

Průchod grafem do hloubky, krok 9

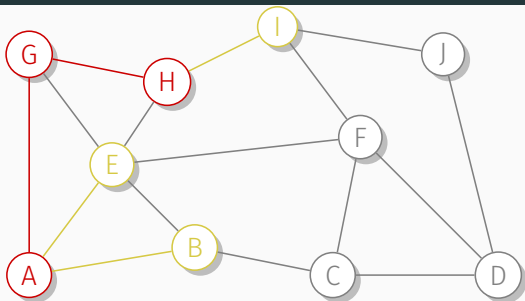


v	$d[v]$	$f[v]$	$\pi[v]$
A	1	∞	/
B	∞	∞	A
C	∞	∞	/
D	∞	∞	/
E	∞	∞	A

v	$d[v]$	$f[v]$	$\pi[v]$
F	∞	∞	/
G	2	∞	A
H	3	∞	G
I	∞	∞	/
J	∞	∞	/

H
G
E
B
A
S

Průchod grafem do hloubky, krok 10

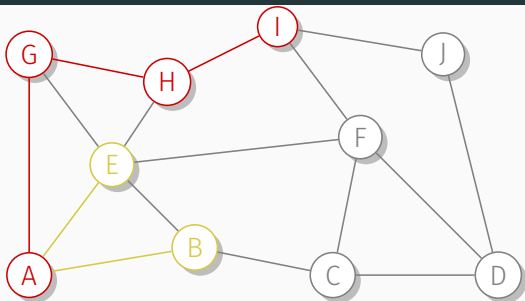


v	$d[v]$	$f[v]$	$\pi[v]$
A	1	∞	/
B	∞	∞	A
C	∞	∞	/
D	∞	∞	/
E	∞	∞	A

v	$d[v]$	$f[v]$	$\pi[v]$
F	∞	∞	/
G	2	∞	A
H	3	∞	G
I	∞	∞	H
J	∞	∞	/

I
H
G
E
B
A
S

Průchod grafem do hloubky, krok 11

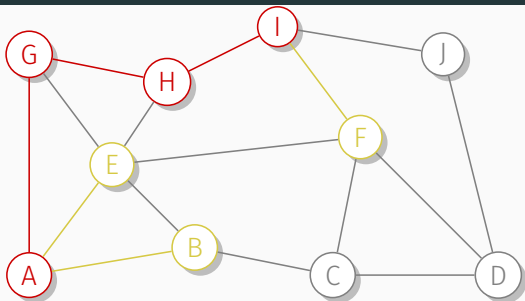


v	$d[v]$	$f[v]$	$\pi[v]$
A	1	∞	/
B	∞	∞	A
C	∞	∞	/
D	∞	∞	/
E	∞	∞	A

v	$d[v]$	$f[v]$	$\pi[v]$
F	∞	∞	/
G	2	∞	A
H	3	∞	G
I	4	∞	H
J	∞	∞	/

I
H
G
E
B
A
S

Průchod grafem do hloubky, krok 12

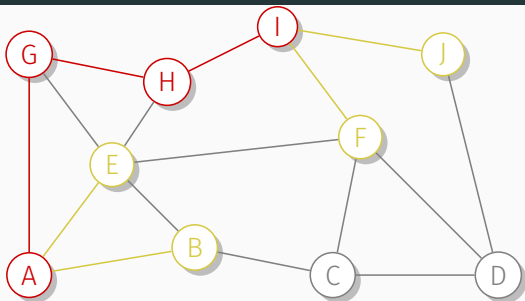


v	$d[v]$	$f[v]$	$\pi[v]$
A	1	∞	/
B	∞	∞	A
C	∞	∞	/
D	∞	∞	/
E	∞	∞	A

v	$d[v]$	$f[v]$	$\pi[v]$
F	∞	∞	I
G	2	∞	A
H	3	∞	G
I	4	∞	H
J	∞	∞	/

F
I
H
G
E
B
A
S

Průchod grafem do hloubky, krok 13

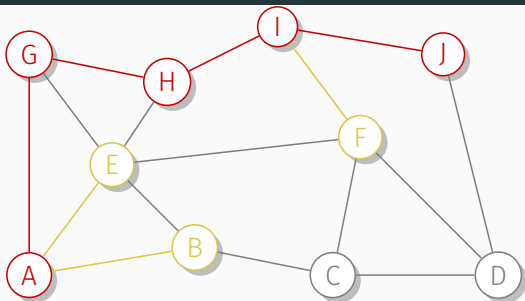


v	$d[v]$	$f[v]$	$\pi[v]$
A	1	∞	/
B	∞	∞	A
C	∞	∞	/
D	∞	∞	/
E	∞	∞	A

v	$d[v]$	$f[v]$	$\pi[v]$
F	∞	∞	I
G	2	∞	A
H	3	∞	G
I	4	∞	H
J	∞	∞	I

J
F
I
H
G
E
B
A
S

Průchod grafem do hloubky, krok 14

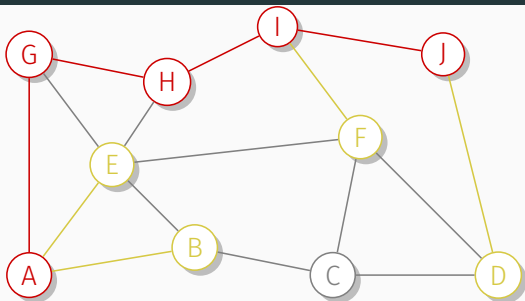


v	$d[v]$	$f[v]$	$\pi[v]$
A	1	∞	/
B	∞	∞	A
C	∞	∞	/
D	∞	∞	/
E	∞	∞	A

v	$d[v]$	$f[v]$	$\pi[v]$
F	∞	∞	I
G	2	∞	A
H	3	∞	G
I	4	∞	H
J	5	∞	I

J
F
I
H
G
E
B
A
S

Průchod grafem do hloubky, krok 15

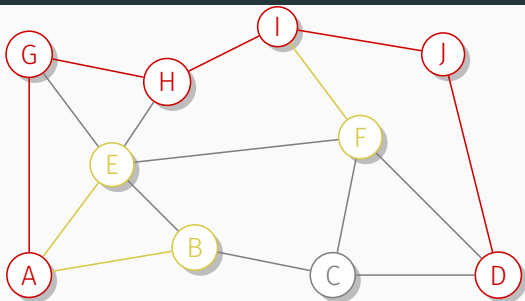


v	$d[v]$	$f[v]$	$\pi[v]$
A	1	∞	/
B	∞	∞	A
C	∞	∞	/
D	∞	∞	J
E	∞	∞	A

v	$d[v]$	$f[v]$	$\pi[v]$
F	∞	∞	I
G	2	∞	A
H	3	∞	G
I	4	∞	H
J	5	∞	I

D
J
F
I
H
G
E
B
A
S

Průchod grafem do hloubky, krok 16

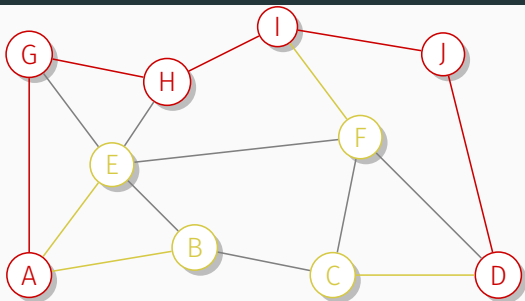


v	$d[v]$	$f[v]$	$\pi[v]$
A	1	∞	/
B	∞	∞	A
C	∞	∞	/
D	6	∞	J
E	∞	∞	A

v	$d[v]$	$f[v]$	$\pi[v]$
F	∞	∞	I
G	2	∞	A
H	3	∞	G
I	4	∞	H
J	5	∞	I

D
J
F
I
H
G
E
B
A
S

Průchod grafem do hloubky, krok 17

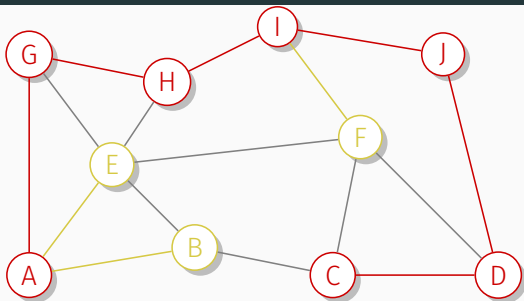


v	$d[v]$	$f[v]$	$\pi[v]$
A	1	∞	/
B	∞	∞	A
C	∞	∞	D
D	6	∞	J
E	∞	∞	A

v	$d[v]$	$f[v]$	$\pi[v]$
F	∞	∞	I
G	2	∞	A
H	3	∞	G
I	4	∞	H
J	5	∞	I

C
D
J
F
I
H
G
E
B
A
S

Průchod grafem do hloubky, krok 18

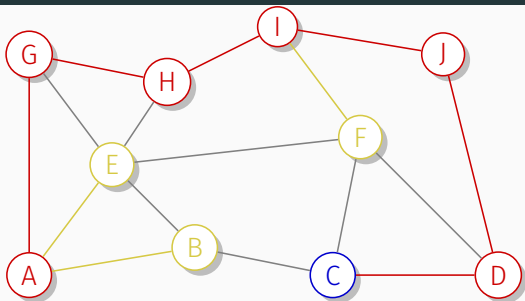


v	$d[v]$	$f[v]$	$\pi[v]$
A	1	∞	/
B	∞	∞	A
C	7	∞	D
D	6	∞	J
E	∞	∞	A

v	$d[v]$	$f[v]$	$\pi[v]$
F	∞	∞	I
G	2	∞	A
H	3	∞	G
I	4	∞	H
J	5	∞	I

C
D
J
F
I
H
G
E
B
A
S

Průchod grafem do hloubky, krok 19

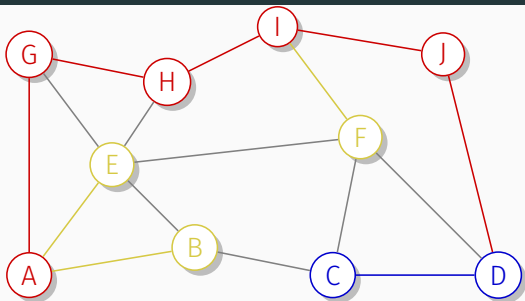


v	$d[v]$	$f[v]$	$\pi[v]$
A	1	∞	/
B	∞	∞	A
C	7	8	D
D	6	∞	J
E	∞	∞	A

v	$d[v]$	$f[v]$	$\pi[v]$
F	∞	∞	I
G	2	∞	A
H	3	∞	G
I	4	∞	H
J	5	∞	I

D
J
F
I
H
G
E
B
A
S

Průchod grafem do hloubky, krok 20

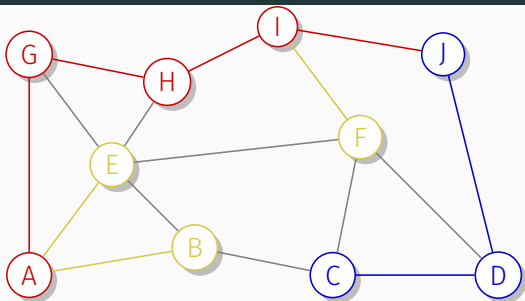


v	$d[v]$	$f[v]$	$\pi[v]$
A	1	∞	/
B	∞	∞	A
C	7	8	D
D	6	9	J
E	∞	∞	A

v	$d[v]$	$f[v]$	$\pi[v]$
F	∞	∞	I
G	2	∞	A
H	3	∞	G
I	4	∞	H
J	5	∞	I

J
F
I
H
G
E
B
A
S

Průchod grafem do hloubky, krok 21

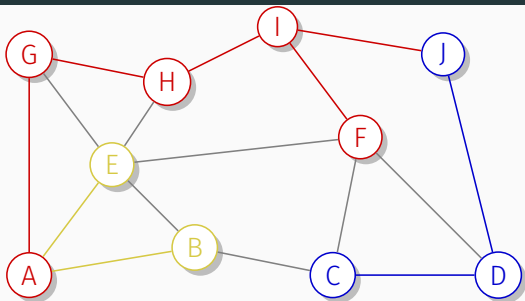


v	$d[v]$	$f[v]$	$\pi[v]$
A	1	∞	/
B	∞	∞	A
C	7	8	D
D	6	9	J
E	∞	∞	A

v	$d[v]$	$f[v]$	$\pi[v]$
F	∞	∞	I
G	2	∞	A
H	3	∞	G
I	4	∞	H
J	5	10	I

F
I
H
G
E
B
A
S

Průchod grafem do hloubky, krok 22

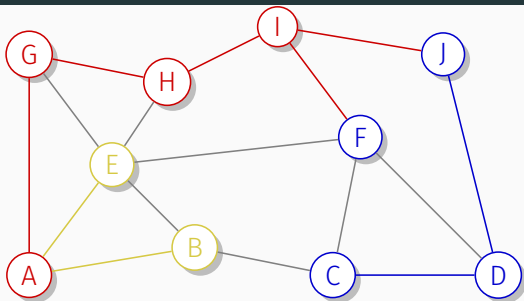


v	$d[v]$	$f[v]$	$\pi[v]$
A	1	∞	/
B	∞	∞	A
C	7	8	D
D	6	9	J
E	∞	∞	A

v	$d[v]$	$f[v]$	$\pi[v]$
F	11	∞	I
G	2	∞	A
H	3	∞	G
I	4	∞	H
J	5	10	I

F
I
H
G
E
B
A
S

Průchod grafem do hloubky, krok 23

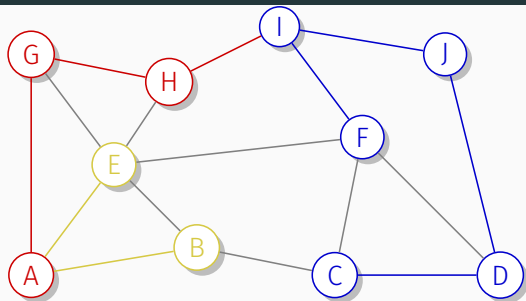


v	$d[v]$	$f[v]$	$\pi[v]$
A	1	∞	/
B	∞	∞	A
C	7	8	D
D	6	9	J
E	∞	∞	A

v	$d[v]$	$f[v]$	$\pi[v]$
F	11	12	I
G	2	∞	A
H	3	∞	G
I	4	∞	H
J	5	10	I

I
H
G
E
B
A
S

Průchod grafem do hloubky, krok 24

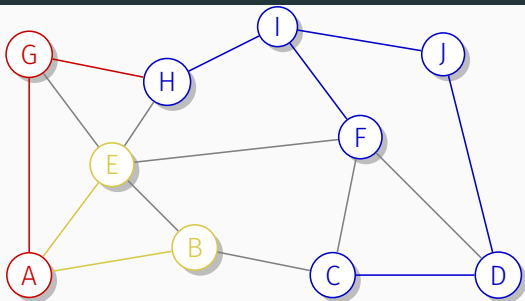


v	$d[v]$	$f[v]$	$\pi[v]$
A	1	∞	/
B	∞	∞	A
C	7	8	D
D	6	9	J
E	∞	∞	A

v	$d[v]$	$f[v]$	$\pi[v]$
F	11	12	I
G	2	∞	A
H	3	∞	G
I	4	13	H
J	5	10	I

H
G
E
B
A
S

Průchod grafem do hloubky, krok 25

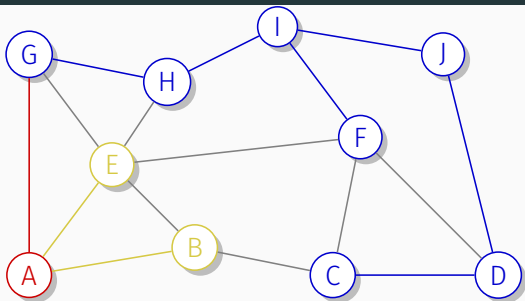


v	$d[v]$	$f[v]$	$\pi[v]$
A	1	∞	/
B	∞	∞	A
C	7	8	D
D	6	9	J
E	∞	∞	A

v	$d[v]$	$f[v]$	$\pi[v]$
F	11	12	I
G	2	∞	A
H	3	14	G
I	4	13	H
J	5	10	I

G
E
B
A
S

Průchod grafem do hloubky, krok 26

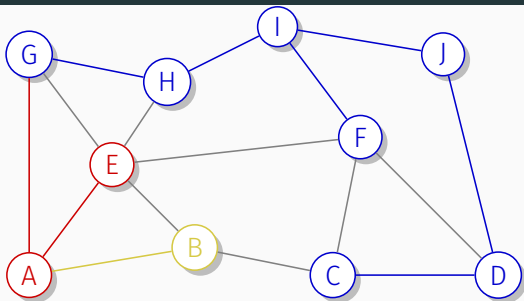


v	$d[v]$	$f[v]$	$\pi[v]$
A	1	∞	/
B	∞	∞	A
C	7	8	D
D	6	9	J
E	∞	∞	A

v	$d[v]$	$f[v]$	$\pi[v]$
F	11	12	I
G	2	15	A
H	3	14	G
I	4	13	H
J	5	10	I

E
B
A
S

Průchod grafem do hloubky, krok 27

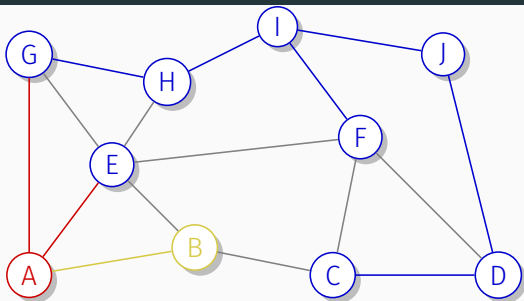


v	$d[v]$	$f[v]$	$\pi[v]$
A	1	∞	/
B	∞	∞	A
C	7	8	D
D	6	9	J
E	16	∞	A

v	$d[v]$	$f[v]$	$\pi[v]$
F	11	12	I
G	2	15	A
H	3	14	G
I	4	13	H
J	5	10	I

E
B
A
S

Průchod grafem do hloubky, krok 28

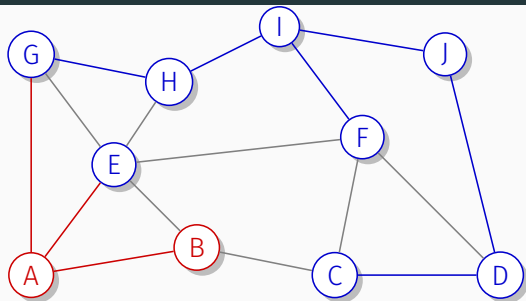


v	$d[v]$	$f[v]$	$\pi[v]$
A	1	∞	/
B	∞	∞	A
C	7	8	D
D	6	9	J
E	16	17	A

v	$d[v]$	$f[v]$	$\pi[v]$
F	11	12	I
G	2	15	A
H	3	14	G
I	4	13	H
J	5	10	I

B
A
S

Průchod grafem do hloubky, krok 29

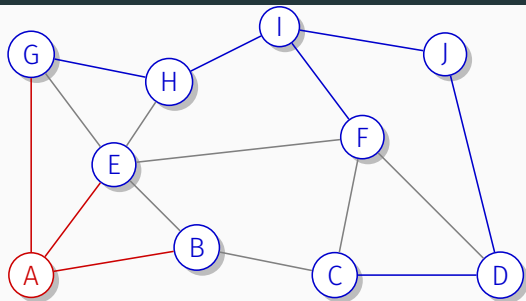


v	$d[v]$	$f[v]$	$\pi[v]$
A	1	∞	/
B	18	∞	A
C	7	8	D
D	6	9	J
E	16	17	A

v	$d[v]$	$f[v]$	$\pi[v]$
F	11	12	I
G	2	15	A
H	3	14	G
I	4	13	H
J	5	10	I

B
A
S

Průchod grafem do hloubky, krok 30

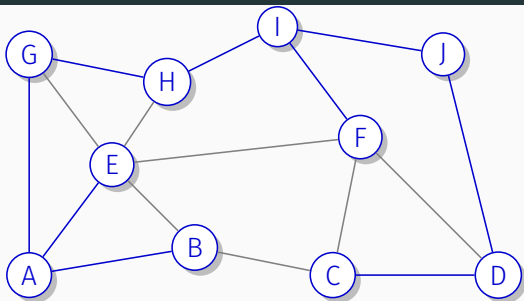


v	$d[v]$	$f[v]$	$\pi[v]$
A	1	∞	/
B	18	19	A
C	7	8	D
D	6	9	J
E	16	17	A

v	$d[v]$	$f[v]$	$\pi[v]$
F	11	12	I
G	2	15	A
H	3	14	G
I	4	13	H
J	5	10	I



Průchod grafem do hloubky, krok 31



v	$d[v]$	$f[v]$	$\pi[v]$
A	1	20	/
B	18	19	A
C	7	8	D
D	6	9	J
E	16	17	A

v	$d[v]$	$f[v]$	$\pi[v]$
F	11	12	I
G	2	15	A
H	3	14	G
I	4	13	H
J	5	10	I

S

Algoritmus průchodu grafem do šířky

Input : Graf $G(V, E)$, počáteční vrchol $s \in V$

Output: BF-strom

```
1 foreach  $u \in V/\{s\}$  do
2   | state [u]  $\leftarrow$  unknown;
3   | d [u]  $\leftarrow \infty$ ;
4   |  $\pi$  [u]  $\leftarrow$  nothing;
5 end
6  $Q \leftarrow \emptyset$ ;
7 state [s]  $\leftarrow$  discovered;
8 d [s]  $\leftarrow$  0;
9  $\pi$  [s]  $\leftarrow$  nothing;
10 Enqueue( $Q, s$ );
```


Algoritmus průchodu grafem do šířky (pokrač.)

```
11 while  $Q \neq \emptyset$  do
12      $u \leftarrow Dequeue(Q)$ ;
13     foreach  $v \in Adj(G, u)$  do
14         if state  $[v] = \text{unknown}$  then
15             state  $[v] \leftarrow \text{discovered}$ ;
16              $d[v] \leftarrow d[u] + 1$ ;
17              $\pi[v] \leftarrow u$ ;
18              $Enqueue(Q, v)$ ;
19         end
20     end
21     state  $[u] \leftarrow \text{finished}$ ;
22 end
```

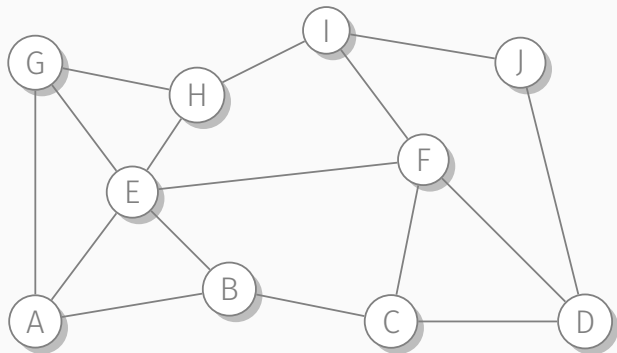
Vrcholy grafu

- šedá vrchol ve stavu unknown
- žlutá vrchol ve stavu discovered
- červená právě zpracovávaný vrchol
- modrá vrchol ve stavu finished

Hrany grafu

- šedá hrana mezi vrcholy ve stavu unknown nebo hrana nepatřící do BF-stromu
- žlutá hrana incidentní s vrcholy ve stavu discovered
- červená hrana incidentní s právě zpracovávaným vrcholem
- modrá hrana mezi vrcholy ve stavu finished

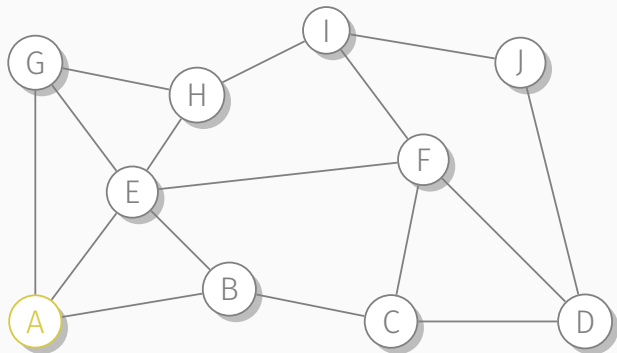
Průchod grafem do šířky, krok 1



Q

v	$d[v]$	$\pi[v]$
A	∞	/
B	∞	/
C	∞	/
D	∞	/
E	∞	/
F	∞	/
G	∞	/
H	∞	/
I	∞	/
J	∞	/

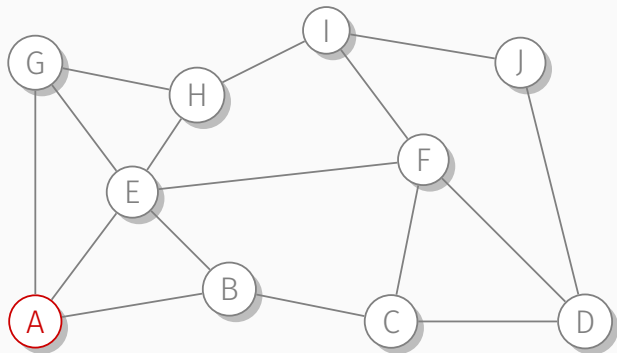
Průchod grafem do šířky, krok 2



Q [A |]

v	$d[v]$	$\pi[v]$
A	0	/
B	∞	/
C	∞	/
D	∞	/
E	∞	/
F	∞	/
G	∞	/
H	∞	/
I	∞	/
J	∞	/

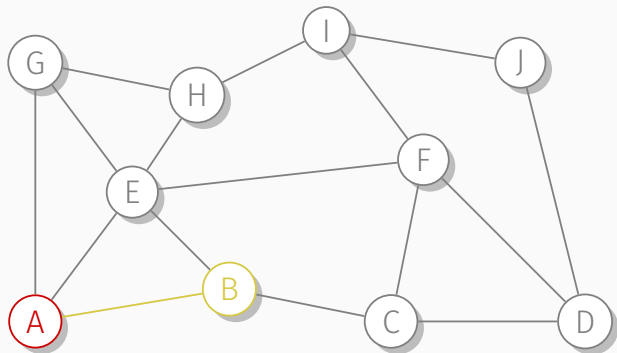
Průchod grafem do šířky, krok 3



Q

v	$d[v]$	$\pi[v]$
A	0	/
B	∞	/
C	∞	/
D	∞	/
E	∞	/
F	∞	/
G	∞	/
H	∞	/
I	∞	/
J	∞	/

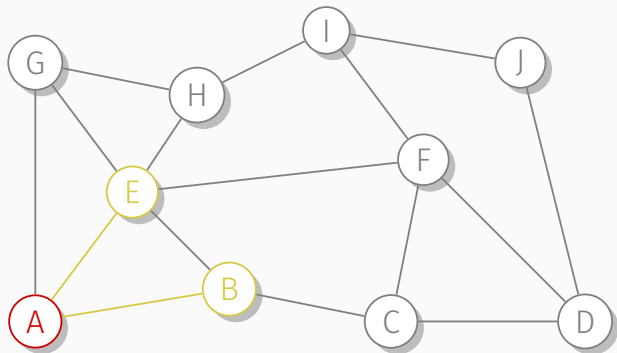
Průchod grafem do šířky, krok 4



Q [B]

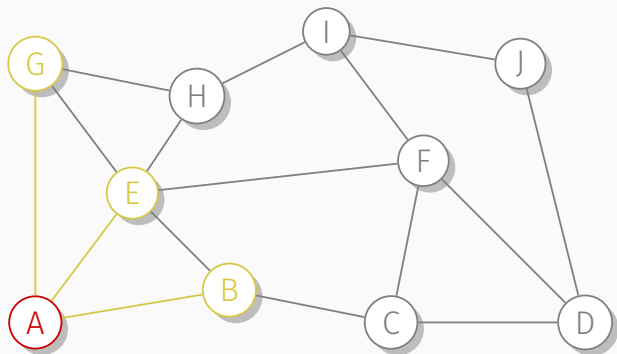
v	$d[v]$	$\pi[v]$
A	0	/
B	1	A
C	∞	/
D	∞	/
E	∞	/
F	∞	/
G	∞	/
H	∞	/
I	∞	/
J	∞	/

Průchod grafem do šířky, krok 5



v	$d[v]$	$\pi[v]$
A	0	/
B	1	A
C	∞	/
D	∞	/
E	1	A
F	∞	/
G	∞	/
H	∞	/
I	∞	/
J	∞	/

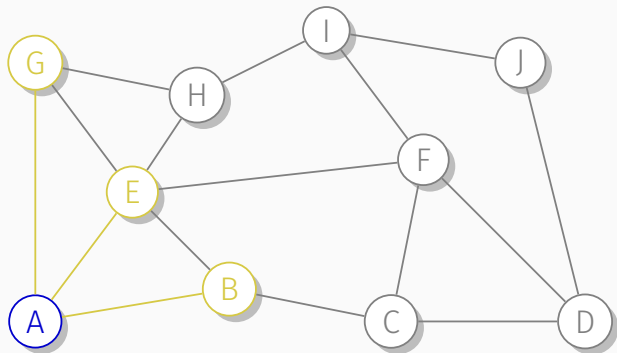
Průchod grafem do šířky, krok 6



Q [B | E | G |]

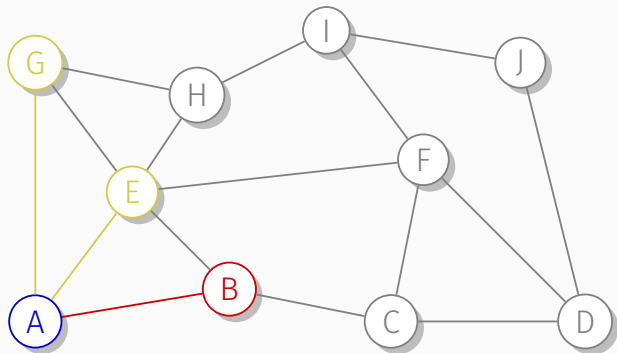
v	$d[v]$	$\pi[v]$
A	0	/
B	1	A
C	∞	/
D	∞	/
E	1	A
F	∞	/
G	1	A
H	∞	/
I	∞	/
J	∞	/

Průchod grafem do šířky, krok 7



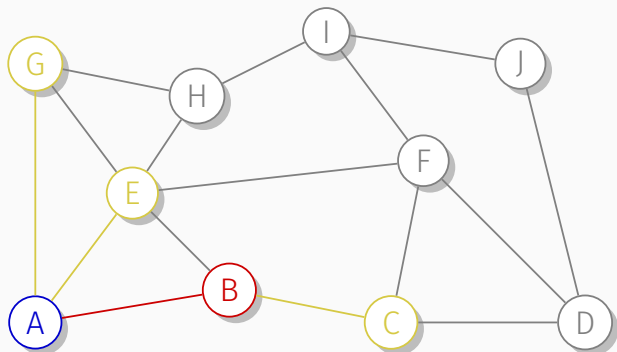
v	$d[v]$	$\pi[v]$
A	0	/
B	1	A
C	∞	/
D	∞	/
E	1	A
F	∞	/
G	1	A
H	∞	/
I	∞	/
J	∞	/

Průchod grafem do šířky, krok 8



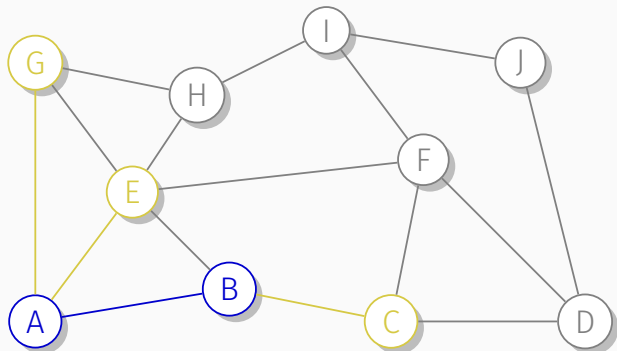
v	$d[v]$	$\pi[v]$
A	0	/
B	1	A
C	∞	/
D	∞	/
E	1	A
F	∞	/
G	1	A
H	∞	/
I	∞	/
J	∞	/

Průchod grafem do šířky, krok 9



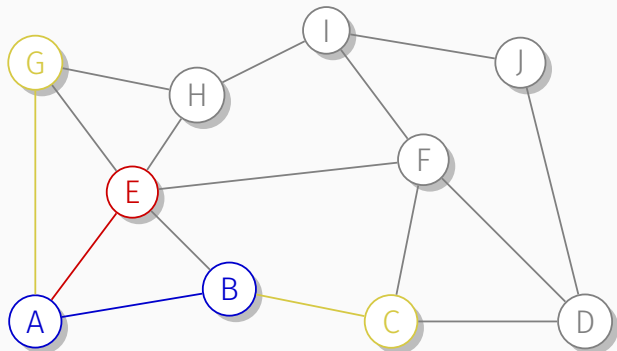
v	$d[v]$	$\pi[v]$
A	0	/
B	1	A
C	2	B
D	∞	/
E	1	A
F	∞	/
G	1	A
H	∞	/
I	∞	/
J	∞	/

Průchod grafem do šířky, krok 10



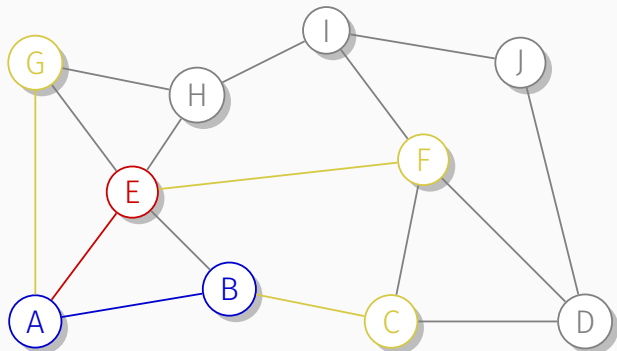
v	$d[v]$	$\pi[v]$
A	0	/
B	1	A
C	2	B
D	∞	/
E	1	A
F	∞	/
G	1	A
H	∞	/
I	∞	/
J	∞	/

Průchod grafem do šířky, krok 11



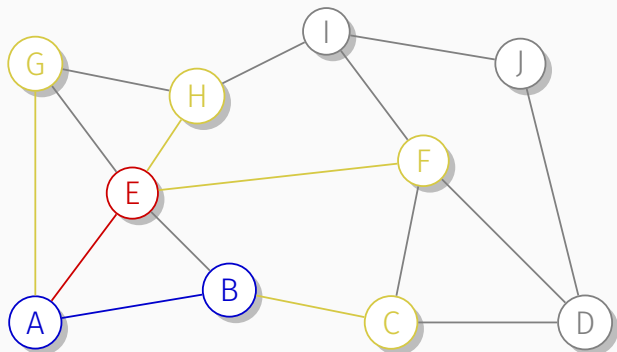
v	$d[v]$	$\pi[v]$
A	0	/
B	1	A
C	2	B
D	∞	/
E	1	A
F	∞	/
G	1	A
H	∞	/
I	∞	/
J	∞	/

Průchod grafem do šířky, krok 12



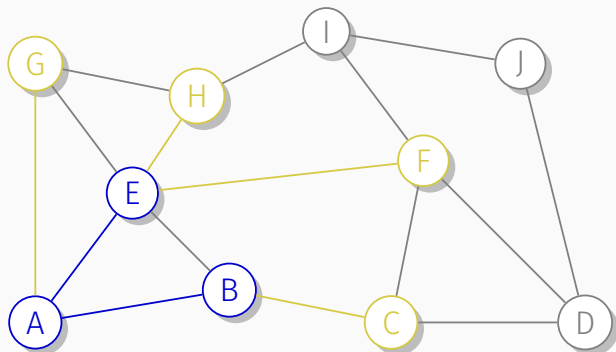
v	$d[v]$	$\pi[v]$
A	0	/
B	1	A
C	2	B
D	∞	/
E	1	A
F	2	E
G	1	A
H	∞	/
I	∞	/
J	∞	/

Průchod grafem do šířky, krok 13



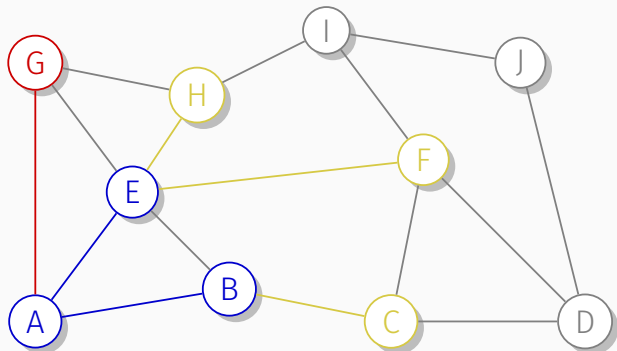
v	$d[v]$	$\pi[v]$
A	0	/
B	1	A
C	2	B
D	∞	/
E	1	A
F	2	E
G	1	A
H	2	E
I	∞	/
J	∞	/

Průchod grafem do šířky, krok 14



v	$d[v]$	$\pi[v]$
A	0	/
B	1	A
C	2	B
D	∞	/
E	1	A
F	2	E
G	1	A
H	2	E
I	∞	/
J	∞	/

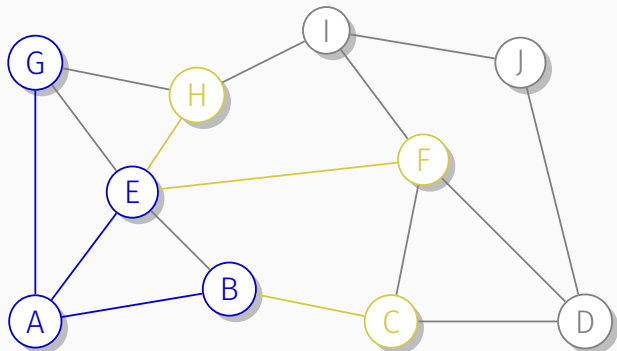
Průchod grafem do šířky, krok 15



Q [C | F | H |]

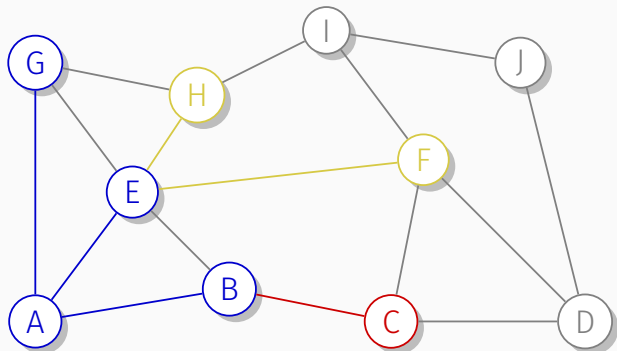
v	$d[v]$	$\pi[v]$
A	0	/
B	1	A
C	2	B
D	∞	/
E	1	A
F	2	E
G	1	A
H	2	E
I	∞	/
J	∞	/

Průchod grafem do šířky, krok 16



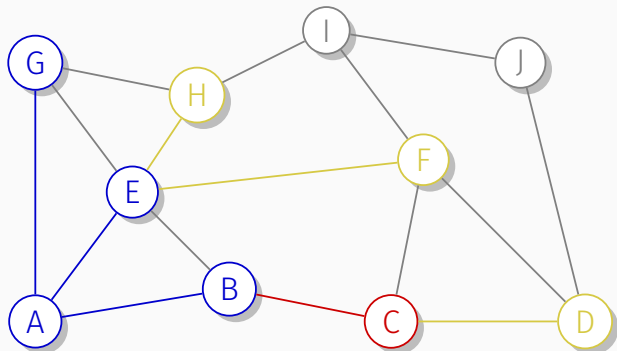
v	$d[v]$	$\pi[v]$
A	0	/
B	1	A
C	2	B
D	∞	/
E	1	A
F	2	E
G	1	A
H	2	E
I	∞	/
J	∞	/

Průchod grafem do šířky, krok 17



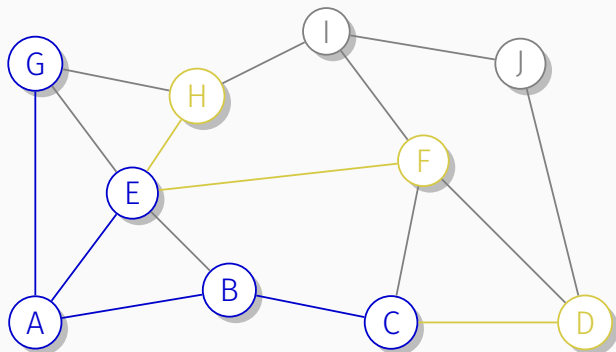
v	$d[v]$	$\pi[v]$
A	0	/
B	1	A
C	2	B
D	∞	/
E	1	A
F	2	E
G	1	A
H	2	E
I	∞	/
J	∞	/

Průchod grafem do šířky, krok 18



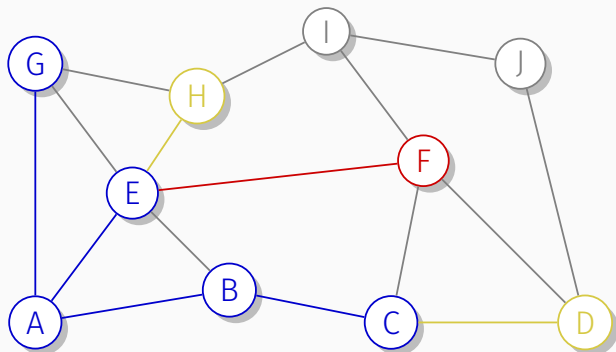
v	$d[v]$	$\pi[v]$
A	0	/
B	1	A
C	2	B
D	3	C
E	1	A
F	2	E
G	1	A
H	2	E
I	∞	/
J	∞	/

Průchod grafem do šířky, krok 19



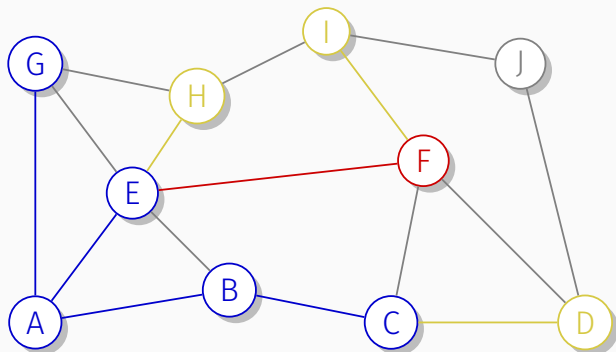
v	$d[v]$	$\pi[v]$
A	0	/
B	1	A
C	2	B
D	3	C
E	1	A
F	2	E
G	1	A
H	2	E
I	∞	/
J	∞	/

Průchod grafem do šířky, krok 20



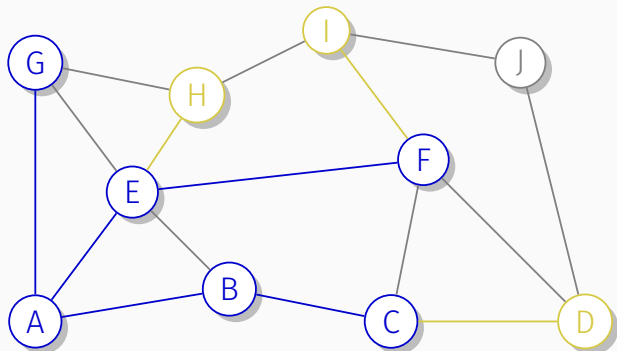
v	$d[v]$	$\pi[v]$
A	0	/
B	1	A
C	2	B
D	3	C
E	1	A
F	2	E
G	1	A
H	2	E
I	∞	/
J	∞	/

Průchod grafem do šířky, krok 21



v	$d[v]$	$\pi[v]$
A	0	/
B	1	A
C	2	B
D	3	C
E	1	A
F	2	E
G	1	A
H	2	E
I	3	F
J	∞	/

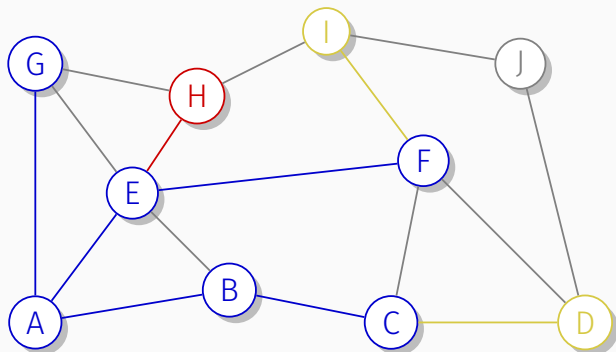
Průchod grafem do šířky, krok 22



Q [H | D | I |]

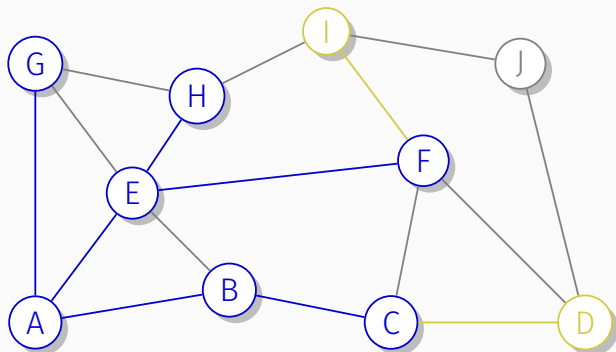
v	$d[v]$	$\pi[v]$
A	0	/
B	1	A
C	2	B
D	3	C
E	1	A
F	2	E
G	1	A
H	2	E
I	3	F
J	∞	/

Průchod grafem do šířky, krok 23



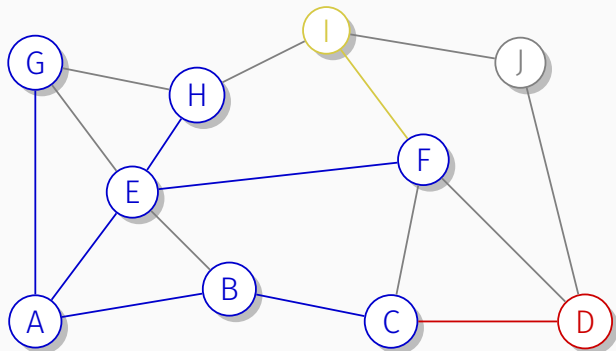
v	$d[v]$	$\pi[v]$
A	0	/
B	1	A
C	2	B
D	3	C
E	1	A
F	2	E
G	1	A
H	2	E
I	3	F
J	∞	/

Průchod grafem do šířky, krok 24



v	$d[v]$	$\pi[v]$
A	0	/
B	1	A
C	2	B
D	3	C
E	1	A
F	2	E
G	1	A
H	2	E
I	3	F
J	∞	/

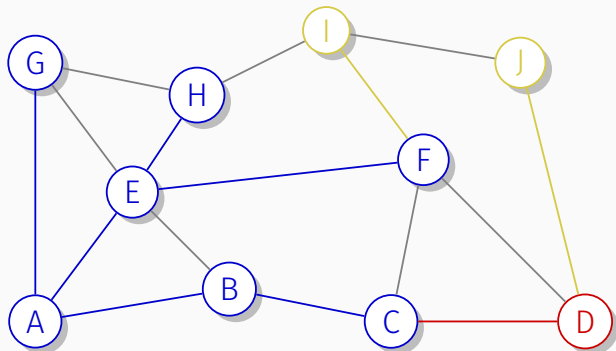
Průchod grafem do šířky, krok 25



Q | I |

v	$d[v]$	$\pi[v]$
A	0	/
B	1	A
C	2	B
D	3	C
E	1	A
F	2	E
G	1	A
H	2	E
I	3	F
J	∞	/

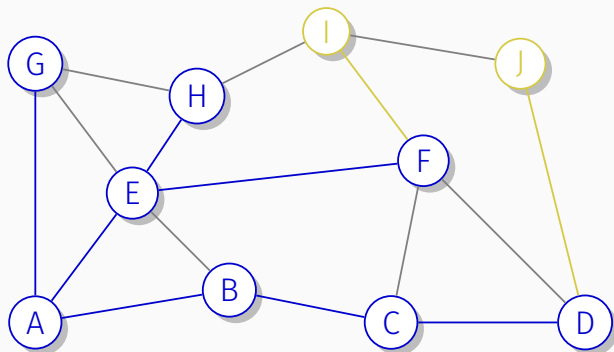
Průchod grafem do šířky, krok 26



Q | I | J |

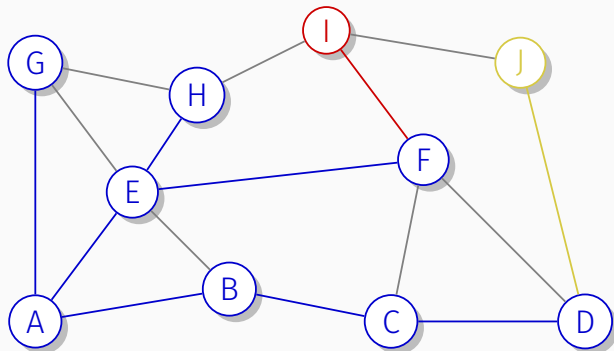
v	$d[v]$	$\pi[v]$
A	0	/
B	1	A
C	2	B
D	3	C
E	1	A
F	2	E
G	1	A
H	2	E
I	3	F
J	4	D

Průchod grafem do šířky, krok 27



v	$d[v]$	$\pi[v]$
A	0	/
B	1	A
C	2	B
D	3	C
E	1	A
F	2	E
G	1	A
H	2	E
I	3	F
J	4	D

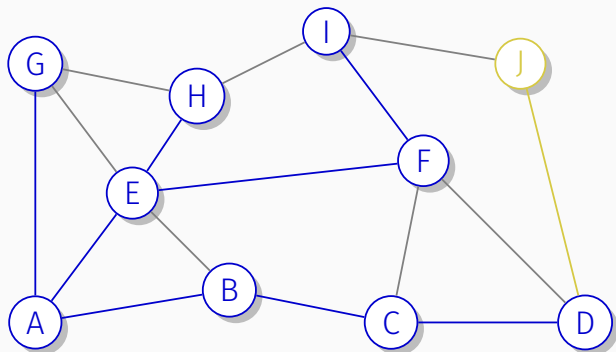
Průchod grafem do šířky, krok 28



Q [J |]

v	$d[v]$	$\pi[v]$
A	0	/
B	1	A
C	2	B
D	3	C
E	1	A
F	2	E
G	1	A
H	2	E
I	3	F
J	4	D

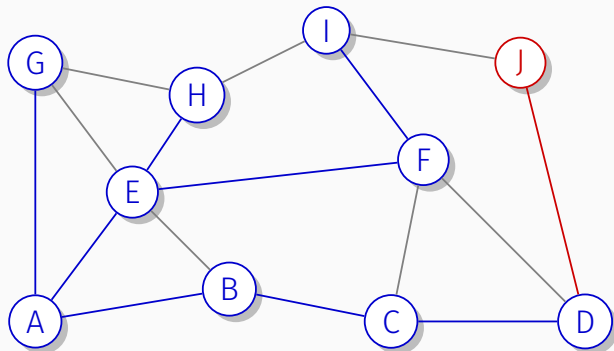
Průchod grafem do šířky, krok 29



Q [J |]

v	$d[v]$	$\pi[v]$
A	0	/
B	1	A
C	2	B
D	3	C
E	1	A
F	2	E
G	1	A
H	2	E
I	3	F
J	4	D

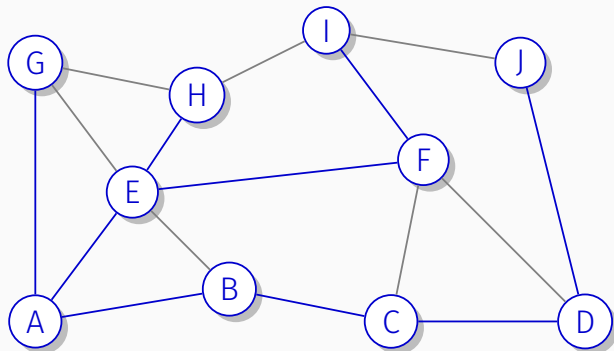
Průchod grafem do šířky, krok 30



Q

v	$d[v]$	$\pi[v]$
A	0	/
B	1	A
C	2	B
D	3	C
E	1	A
F	2	E
G	1	A
H	2	E
I	3	F
J	4	D

Průchod grafem do šířky, krok 31



Q

v	$d[v]$	$\pi[v]$
A	0	/
B	1	A
C	2	B
D	3	C
E	1	A
F	2	E
G	1	A
H	2	E
I	3	F
J	4	D

Animace

K oběma algoritmům průchodu grafem jsou dostupné dvě animace:

- vyhledání cesty v bludišti a
- aplikace v počítačové grafice – vyplňování oblasti.

Animace jsou k dispozici animace v samostatných souborech.

- **Strategie řešení problémů hrubou silou**
 - kniha [2], kapitola 3, strany 97 – 98
- **Třídění výběrem**
 - kniha [2], kapitola 3.1, strany 98 – 100
- **Bublinové třídění**
 - kniha [2], kapitola 3.1, strany 100 – 101
- **Třídění přetřásáním**
 - kniha [4], kapitola 4, strany 78 – 79
- **Sekvenční vyhledávání**
 - kniha [2], kapitola 3.2, strany 104 – 104
- **Vyhledávání podřetězce hrubou silou**
 - kniha [2], kapitola 3.2, strany 105 – 106

Zdroje pro samostatné studium (pokrač.)

- **Problém nejbližší dvojice bodů**
 - kniha [2], kapitola 3.3, strany 108 – 109
- **Konvexní obal množiny**
 - kniha [2], kapitola 3.3, strany 109 – 113
- **Úplné prohledávání**
 - kniha [2], kapitola 3.4, strany 115
- **Problém obchodního cestujícího**
 - kniha [2], kapitola 3.4, strany 116
- **Problém batohu**
 - kniha [2], kapitola 3.4, strany 116 – 117
- **Průchod grafem do hloubky**
 - kniha [2], kapitola 3.5, strany 122 – 125
- **Průchod grafem do šířky**
 - kniha [2], kapitola 3.5, strany 125 – 128

Děkuji za pozornost