Dijkstrův algoritmus (připomenutí)

Základní předpoklad $w : H \rightarrow R^{+}$ (nezáporné délky hran)

“Upravený“ algoritmus prohledávání do šířky

<u>Dijkstra(G,s,w)</u>

```
InitPaths(G,s)
S:=∅; InitQueue(Q)
for každý uzel u∈U do Enqueue(Q,u)              O(|U|)
while not EmptyQueue(Q)
      do  u:=ExtractMin(Q); S:=S∪{u}            O(|U| . lg |U|)
            for uzel v∈Adj[u]
                  do Relax(u,v,w)                O(|H| . lg |U|)
```

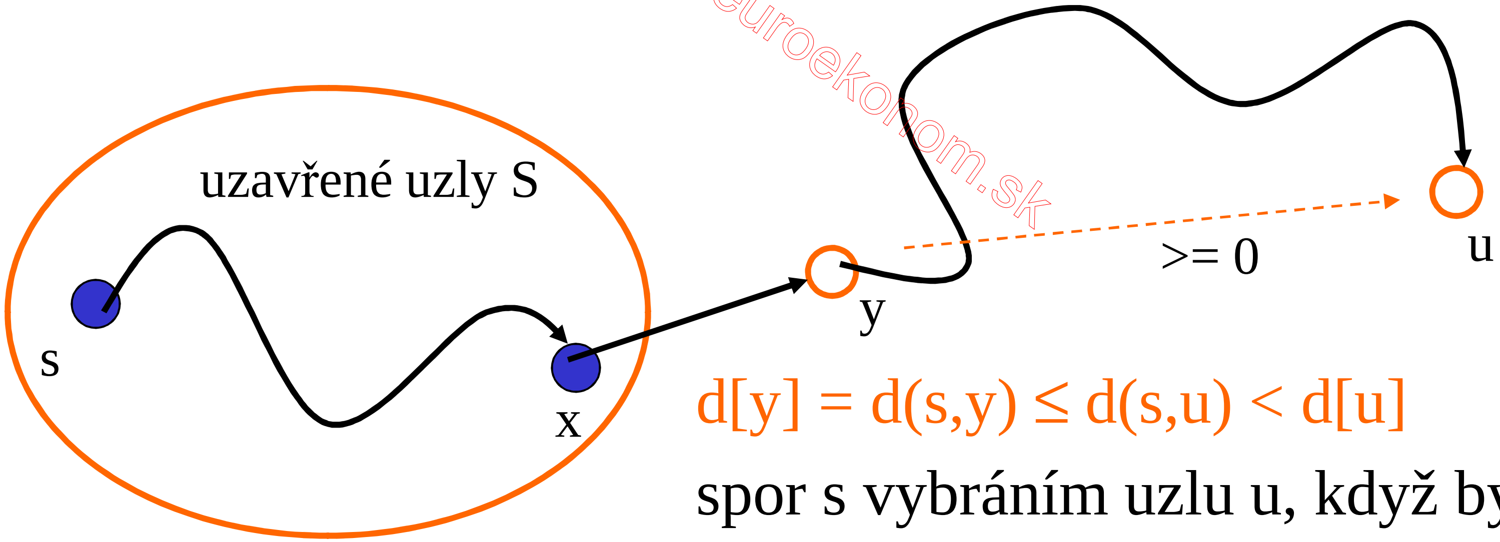
Možné ještě **O(|U| . lg |U| + |H|)** nebo **O(|U|**2)**

Důkaz správnosti Dijkstrova algoritmu

<u>Tvrzení:</u> Při uzavření uzlu u (řádka 5 ... $S := S \cup \{u\}$) platí

$$d[u] = d(s,u)$$

<u>D:</u> sporem - nechť je $d[u] > d(s,u)$ pro nějaký uzavřený uzel, mějme nejkratší cestu $s \rightarrow u$, x je poslední uzavřený



$d[y] = d(s,y) \leq d(s,u) < d[u]$

spor s vybráním uzlu u, když byl k dispozici uzel y s menší hodnotou

Bellmanův-Fordův algoritmus

? Co dělat v případě záporně ohodnocených hran ?

Bellman-Ford(G,s,w)

```
InitPaths(G,s)
for i:=1 to |U|-1 do
      for každou hranu (u,v)∈H do Relax(u,v,w)
for každou hranu (u,v)∈H
   do if d[v] > d[u] + w(u,v) then return false
return true
```


Složitost **O(|U| . |H|)**

? Proč má nyní Relax konstantní časovou složitost ?

? Nelze B-F algoritmus nějak upravit / zrychlit ?

Co když **zavedeme frontu uzlů s úspěšným Relax** a bereme jen hrany vycházející z těchto uzlů?

- **ukončení** - při vyprázdnění fronty
- **problém** - co když se fronta nevyprázdní?
- v nejhorším případě zase **O(|U| . |H|)**

Nejkratší cesty pro acyklické grafy

```
Topologicky uspořádáme uzly grafu G
InitPaths(G,s)
for každý uzel u v pořadí podle topologického uspořádání
    do for každé v∈ Adj[u]
           do Relax(u,v,w)
```

?? Složitost ??

Nejkratší cesty mezi všemi páry uzlů

Označme |U| = n

n x Dijkstra .....
- **$O(n^2 \,.\, \lg n + n \,.\, |H|)$** Fib. halda
- **$O(|H| \,.\, n \,.\, \lg n)$** binární halda
- **$O(n^3)$** sekvenční fronta

Ale POZOR - jen pro nezáporné ohodnocení,

jinak n x Bellman-Ford ... **$O(n^2 \,.\, |H|) \sim O(n^4)$!?**

? Jde to lépe ? **ANO**

Nejkratší cesty a násobení matic

Graf G reprezentujeme **maticí w-délek** **W** (vychází z matice sousednosti **V** a zahrnuje současně délky hran w):

$$\mathbf{w_{ij}} = \begin{cases} \mathbf{0} & \text{pro } i = j \\ \mathbf{w(u_i, u_j)}, & i \neq j,\ (u_i,u_j) \in H \\ \infty & \text{jinak} \end{cases}$$

matice w-vzdáleností D = [d_{ij}] : $d_{ij} = d_w(u_i,u_j)$

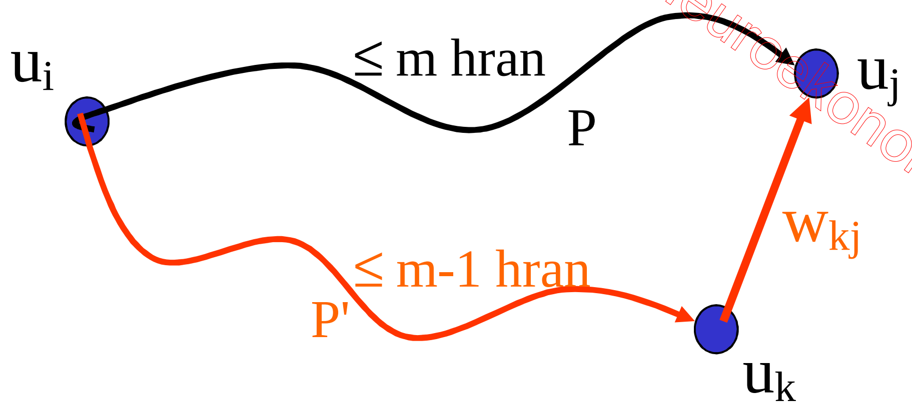
matice předchůdců P = [p_{ij}]

$$\mathbf{p_{ij}} = \begin{cases} \mathbf{nil} & \dots\ i=j \text{ nebo neexistuje cesta } u_i \rightarrow u_j, \\ \mathbf{u_k} & \dots\ u_k \text{ je předchůdce } u_j \text{ na cestě } u_i \rightarrow u_j \end{cases}$$

Cesty s omezeným počtem hran

Předpoklad: nemáme záporné cykly!

Nechť P je cesta $u_i \rightarrow u_j$, která je tvořena max. **m** hranami a je nejkratší (podle w-délky)



$$w(P) = w(P') + w_{kj}$$

(cesty tvořené m hranami vzniknou prodloužením cest o m-1 hranách)

$\mathbf{d_{ij}^{(m)}}$ ... w-délka cesty P

$$d_{ij}^{(0)} = \begin{cases} 0 & \text{pro } i=j \\ \infty & \text{jinak} \end{cases}$$

Pro m=1, 2, ..., n-1 máme formuli

$$d_{ij}^{(m)} = \min\left(d_{ij}^{(m-1)}, \min\left(d_{ik}^{(m-1)} + w_{kj}\right)\right) \text{ přes všechna } k$$

$$= \mathbf{\min\left(d_{ik}^{(m-1)} + w_{kj}\right)} \text{ přes všechna } k \ (\mathbf{w_{jj}} = 0)$$

Operace "přinásobování" maticí W: D'' := D' . W

```
for i:=1 to n do
  for j:=1 to n do
      x:= ∞;
      for k:=1 to n do x := min (x, d'[i,k] + w[k,j])
      d"[i,j] := x
```

Nyní pronásobení použijeme

$\mathbf{D^{(1)} = D^{(0)}.W, \ \ D^{(2)} = D^{(1)}.W = W^2, \ldots, \ \ D^{(n-1)} = W^{n-1}}$

? Složitost ? **$O(n^4)$** **? Můžeme násobení zrychlit ?**

$\mathbf{D^{(1)} = W, \ \ D^{(2)} = W^2, D^{(4)} = W^4, \ldots, W^{2**k} D^{(2)} = W^2,}$

k= horní celá část lg(n-1), tedy složitost

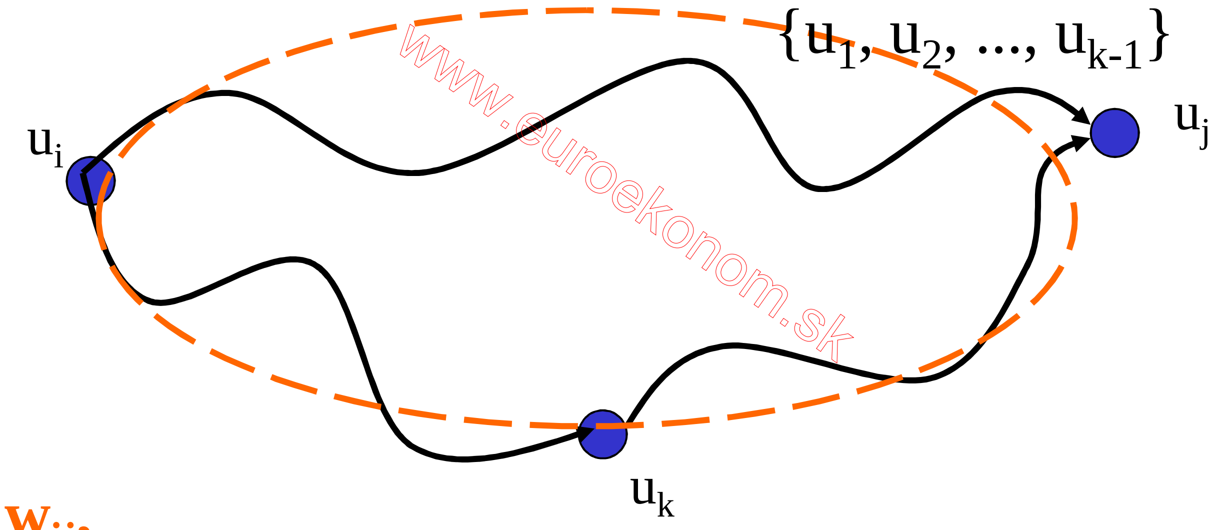
$O(n^3 . \lg n)$

?A co paměťová složitost?

Algoritmus Floyd-Warshall

Cesta P : $u_i \rightarrow u_j$ s vnitřními uzly z $\{u_1, u_2, ..., u_k\}$

$\mathbf{d_{ij}^{(k)}}$ - délka minimální cesty P



$\mathbf{d_{ij}^{(0)} = w_{ij},}$

$\mathbf{d_{ij}^{(k)} = \min (d_{ij}^{(k-1)} , d_{ik}^{(k-1)} + d_{kj}^{(k-1)})}$ $\mathbf{d_{ij}^{(n)} = d_{ij}}$

<u>Algoritmus Floyda-Warshala</u>

```
D^(0) := W
for k:=1 to n do
  for i:=1 to n do
    for j:=1 to n do
      d_ij^(k) := min (d_ij^(k-1) , d_ik^(k-1) + d_kj^(k-1))
return D^(n)
```

$$\mathbf{D}^{(0)} := \mathbf{W}$$

$$d_{ij}^{(k)} := \min\left(d_{ij}^{(k-1)},\ d_{ik}^{(k-1)} + d_{kj}^{(k-1)}\right)$$

!!! Složitost $O(n^3)$!!! ? A co paměť ?

? A co, když chceme znát cesty, ne jen vzdálenosti ?

Výpočet matice předchůdců:

$$p_{ij}^{(0)} = \begin{cases} \text{nil} & \text{pro } i=j \text{ nebo } w_{ij} = \infty \\ i & \text{jinak (tedy } (u_i, u_j) \in H) \end{cases}$$

$$p_{ij}^{(k)} = \begin{cases} p_{ij}^{(k-1)} & d_{ij}^{(k-1)} \leq d_{ik}^{(k-1)} + d_{kj}^{(k-1)} \\ p_{kj}^{(k-1)} & d_{ij}^{(k-1)} > d_{ik}^{(k-1)} + d_{kj}^{(k-1)} \end{cases}$$

Reflexivně-tranzitivní uzávěr grafu (relace)

$$W = V + E$$

$$d_{ij}^{(k)} = d_{ij}^{(k-1)} \vee (d_{ik}^{(k-1)} \wedge d_{kj}^{(k-1)})$$

Johnsonův aloritmus

Pro řídké grafy ($|H| < O(n^2)$) je $O(|H|.n)$ lepší než $O(n^3)$

Idea: n x Dijkstra, **ale co s w(h) < 0 ??**

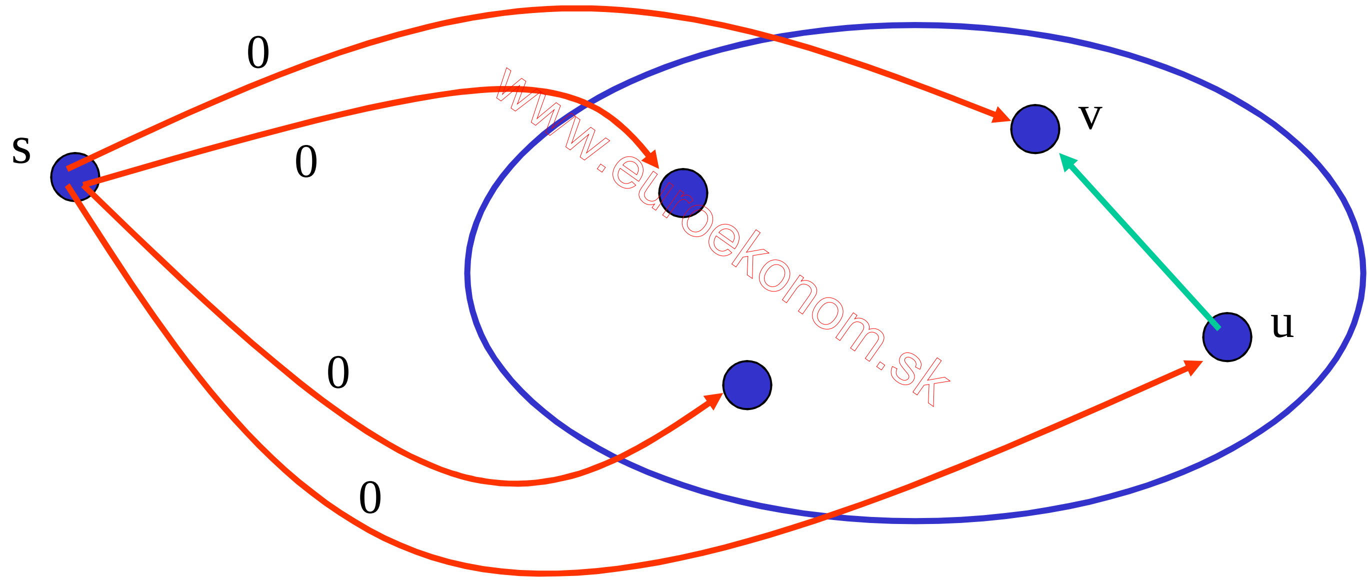
Přehodnocení w → w' takové, že

- pro každé u,v je minimální cesta podle w' minimální také podle w
- $w'(u,v) \geq 0$

Úprava G na G':

- přidáme uzel $U' = U \cup \{s\}$

  a hrany $H = H \cup \{(s,u): u \in U\}$, $w(h) = 0$ pro nové hrany



- položíme $h(u)=d(s,u)$ ... platí $h(v) \leq h(u) + w(u,v)$, tedy

  $$\mathbf{w'(u,v) = w(u,v) + h(u) - h(v) \geq 0 \ !!!}$$

Johnsonův algoritmus

$G \rightarrow G'$

Bellman-Ford(G',w,s) pokud false STOP ... **O(n . |H|)**

for každý uzel $u \in U$ **do** h(u):=d(s,u)

for každou hranu $(u,v) \in H$ **do** w'(u,v):=w(u,v)+h(u)-h(v)

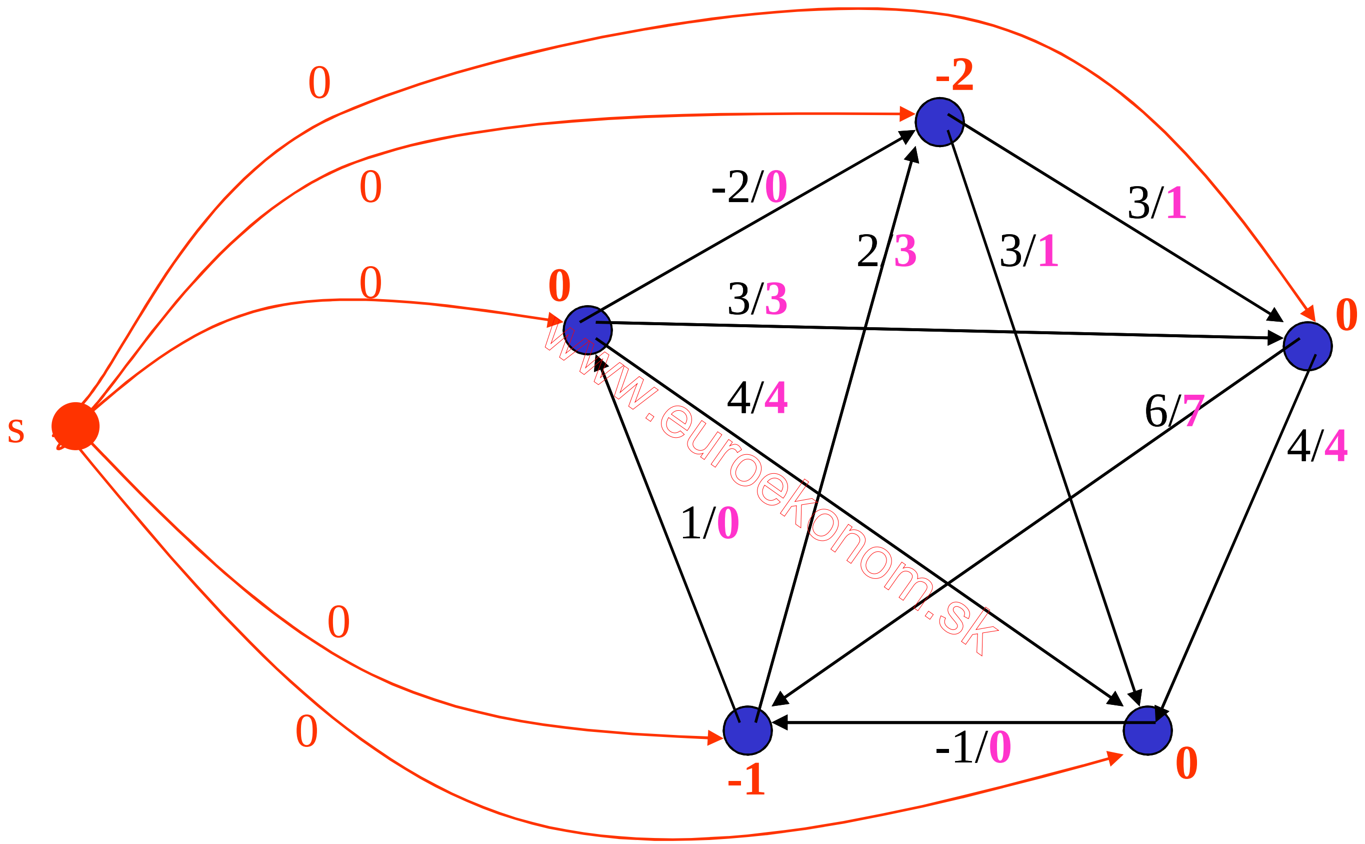
for každý uzel $u \in U$ **n . O(n . lg n + |H|)** Fib. heap

do Dijkstra(G, w', u)

for každý uzel $v \in U$ **do** d(u,v) := d'(u,v)-h(u)+h(v)

O(n . |H| . lg n) pro binární heap

Příklad:



h(u) = min (0, délka nejkratší cesty s → u)   Dále

w'(u,v) = w(u,v) + h(u) - h(v)

Algebraické souvislosti

Floyd-Marshall:

$$\mathbf{d_{ij}^{(k)} = \min\left(d_{ij}^{(k-1)},\ d_{ik}^{(k-1)} + d_{kj}^{(k-1)}\right)}$$

dvojoperace **min, +**

označíme ⊕, ⊗

⊗ efekt složení dvou spojení $u \rightarrow v \otimes v \rightarrow x$

⊕ efekt kombinování alternativních spojení $u \rightrightarrows v$ ⊕

?Jaké vlastnosti musí tyto operace mít, aby to fungovalo?

Polookruh

$\boldsymbol{P} = \langle P, \oplus, \otimes, 0, 1\rangle$:

$$\left.\begin{array}{l} a \oplus (b \oplus c) = (a \oplus b) \oplus c \\ a \oplus 0 = 0 \oplus a = a \\ a \oplus b = b \oplus a \end{array}\right\}$$

$\langle P, \oplus, 0\rangle$ **je komutativní monoid**

$a \oplus a = a$ **idempotence**

$$\left.\begin{array}{l} a \otimes (b \otimes c) = (a \otimes b) \otimes c \\ a \otimes 1 = 1 \otimes a = a \end{array}\right\}$$

$\langle P, \otimes, 1\rangle$ **je monoid**

$a \otimes 0 = 0 \otimes a = 0$ **s nulovým prvkem**

$a \otimes (b \oplus c) = (a \otimes b) \oplus (a \otimes c)$ **distributivnost**

$(b \oplus c) \otimes a = (b \otimes a) \oplus (c \otimes a)$ **zleva a zprava**

Uzavřený polookruh:

$a_1 \oplus a_2 \oplus a_3 \oplus \ldots$ je definováno pro lib. $a_1, a_2, a_3, \ldots$ a je

- **asociativní, komutativní, idempotentní a navíc**
- **distributivní vůči $\otimes$**

Kdy máme ∞ mnoho spojení? Když máme cykly!

Uzávěr $\mathbf{c^* = 1 \oplus c \oplus (c \otimes c) \oplus (c \otimes c \otimes c) \ldots}$

$\mathbf{0^* = 1, \qquad c \otimes c^* = c^* \otimes c, \qquad c^* = 1 \oplus (c^* \otimes c), \ \ldots}$

<u>Příklady polookruhů</u>

$\langle R^+ \cup \{\infty\}, \min, \mathbf{+}, \infty, \mathbf{0} \rangle$

$a^* = \min(0, a, a+a, a+a+a, \ldots) = 0$

$\langle R \cup \{\infty, -\infty\}, \min, \mathbf{+}, \infty, \mathbf{0} \rangle \quad a^* = 0$ nebo $-\infty$