C2142 Návrh algoritmů pro přírodovědce

8. Nejkratší vzdálenosti.

Tomáš Raček

Jaro 2015

Nejkratší vzdálenosti v grafu

Problém. Určete nejkratší vzdálenost mezi vrcholy u a v v grafu.

Pozorování. V tuto chvíli můžeme posuzovat vzdálenost pouze jako počet hran na cestě mezi u a v.

Takový výpočet realizuje prohledávání do šířky (BFS) v lineárním čase vůči velikosti grafu.

Rozšíření. Uvažme případ, kdy bychom chtěli přiřadit hranám na cestě různou váhu.

Graf $G = (V, E, w_e)$ nazveme hranově ohodnocený, w_e je funkce, která každé hraně přiřazuje její ohodnocení – reálné číslo.

Nejkratší vzdálenost mezi dvěma vrcholy v grafu je minimální součet ohodnocení hran některé cesty mezi těmito vrcholy.

Matice vzdáleností

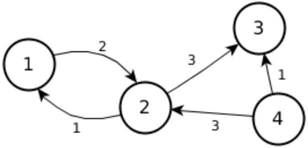
Poznámka. Na hranově neohodnocený graf se lze dívat jako na speciální případ ohodnoceného, kde $\forall (u, v) \in E : w_e(u, v) = 1$.

Matice vzdáleností W je rozšíření matice sousednosti:

$$W_{i,j} = \begin{cases} 0 & \text{pro } i = j \\ w_e(i, j) & \text{pro } (i, j) \in E \\ \infty & \text{pro } (i, j) \notin E \end{cases}$$

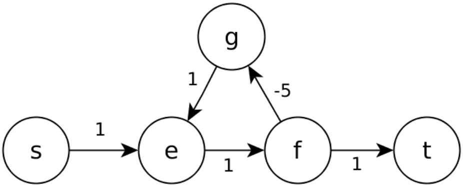
Příklad

$$W = \begin{pmatrix} 0 & 2 & \infty & \infty \\ 1 & 0 & 3 & \infty \\ \infty & \infty & 0 & \infty \\ \infty & 3 & 1 & 0 \end{pmatrix}$$



Záporný cyklus

Příklad. Určete nejkratší vzdálenost mezi vrcholy s a t v grafu:



Pozorování. Graf obsahuje cyklus záporné délky (vrcholy e, f, g), nejkratší vzdálenost mezi s a t není definována.

Poznámka. Uvažme graf bez cyklů záporné délky. Každá nejkratší cesta mezi dvěma vrcholy obsahuje libovolný vrchol nejvýše jednou.

Bellman-Fordův algoritmus

Pozorování. Z předchozí poznámky vyplývá, že nejkratší cesta mezi dvěma vrcholy v grafu obsahuje nejvýše $|V| - 1$ hran.

Relaxace hrany. Nechť (u, v) je hrana v grafu G s ohodnocením $w_e(u, v)$ a hodnoty $u.d$ a $v.d$ jsou v daném okamžiku nejkratší nalezené vzdálenosti do u, resp. do v z výchozího vrcholu. Zjevně pak platí, že:

$$v.d = \min(v.d, u.d + w_e(u, v))$$

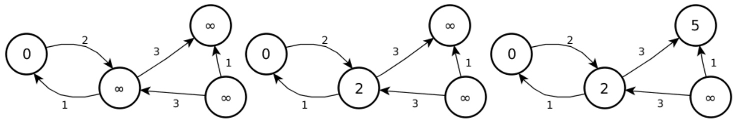
Tato (případná) změna ohodnocení $v.d$ se nazývá relaxací.

Bellman-Fordův algoritmus je založen na těchto dvou principech.

- počítá nejkratší vzdálenosti z výchozího vrcholu do všech ostatních (1:N)
- $(|V| - 1)$krát relaxuje všechny hrany

Bellman-Fordův algoritmus – poznámky

Složitost Bellman-Fordova algoritmu je $\mathcal{O}(|V| \cdot |E|)$, relaxace hrany má totiž zřejmě konstatní složitost.



Pozorování

- Pokud při některé z iterací nedojde ke změně hodnoty $v.d$ pro žádný vrchol v, pak je možné výpočet ukončit.
- Provedením jedné iterace výpočtu navíc lze v grafu detekovat záporné cykly.
- Pokud dojde k relaxaci hrany, lze u koncového vrcholu nastavit ukazatel na jeho předchůdce $\rightarrow$ rekonstrukce cesty.

Bellman-Fordův algoritmus – pseudokód

```
function Bellman-Ford(G = (V, E, w_e), s) is
    ∀v ∈ V : v.d ← ∞;  s.d ← 0
    for i ← 1 to |V| − 1 do
        for all (u, v) ∈ E do
            if v.d > u.d + w_e(u, v) then
                v.d ← u.d + w_e(u, v)
            fi
        done
    done
    for all (u, v) ∈ E do
        if v.d > u.d + w_e(u, v) then
            Error : Negative cycle detected
        fi
    done
end
```

Dijkstrův algoritmus

Dijkstrův algoritmus představuje odlišný způsob řešení problému nejkratších vzdáleností v grafu.

- Opět řeší problém typu 1:N.
- Vyžaduje graf s nezáporným ohodnocením všech hran.

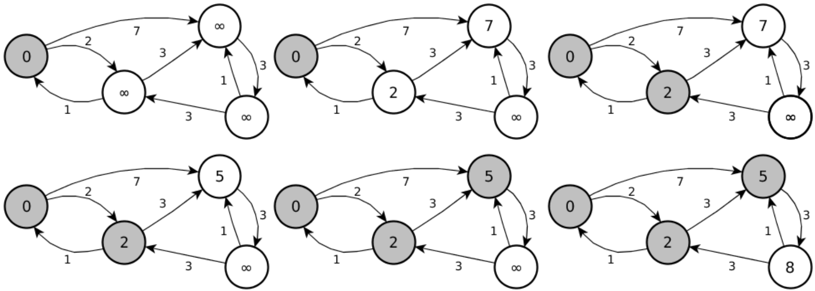
Myšlenka. Pokud je posloupnost $u, u_1, \ldots, u_n, v$ nejkratší cesta z u do v, pak posloupnost $u, u_1, \ldots u_k$, kde $k \leq n$ je nejkratší cesta z u do u_k.

Výpočet algoritmu. V každé iteraci rozšiřujeme množinu vrcholů, do kterých již známe nejkratší vzdálenost z výchozího.

Dijkstrův algoritmus

Příklad

1. Pokud je Q množina vrcholů s dosud neurčenou nejkratší vzdáleností, tak z ní nejprve vybereme vrchol u, jehož ohodnocení $u.d$ je nejnižší.
2. Poté přepočítáme všechny vzdálenosti do vrcholů z Q, do kterých vede z u hrana.



Dijkstrův algoritmus – pseudokód

```
function Dijkstra(G = (V, E, w_e), s) is
    ∀v ∈ V : v.d ← ∞
    s.d ← 0
    Q ← V
    while Q není prázdná do
        u ← t ∈ Q s minimální t.d
        Odstraň u z Q
        for all v : (u, v) ∈ E do
            if v.d > u.d + w_e(u, v) then
                v.d ← u.d + w_e(u, v)
            fi
        done
    done
end
```

Dijkstra – volba datové struktury

Složitost Dijkstrova algoritmu je dána volbou datové struktury pro Q. Zajímají nás dvě operace:

- f_{ext} – extrakce prvku s minimálním klíčem ($|V|$ operací)
- f_{dec} – snížení klíče $v.d$ (nejvýše $|E|$ operací)

Seznam vrcholů

- snížení klíče – $\mathcal{O}(1)$
- extrakce minimálního prvku – $\mathcal{O}(|V|)$
- celkem $\mathcal{O}(|E| + |V|^2) = \mathcal{O}(|V|^2)$

Binární halda

- snížení klíče i extrakce minima – $\mathcal{O}(\log |V|)$
- celkem $\mathcal{O}(\log |V| \cdot (|V| + |E|))$

Nejkratší vzdálenosti mezi všemi dvojicemi vrcholů

Pozorování. Určit nejkratší vzdálenost mezi dvěma vrcholy (1:1) má stejnou složitost jako určit nejkratší vzdálenost z jednoho vrcholu do všech ostatních (1:N).

Nejkratší mezi všemi dvojicemi vrcholů lze spočítat např. využitím $|V|$ volání Dijkstrova nebo Bellman-Fordova algoritmu, kdy v každém výpočtu volíme jiný výchozí vrchol. Jde to i lépe?

Floyd-Warshallův algoritmus počítá nejkratší vzdálenosti mezi všemi dvojicemi vrcholů v čase $\mathcal{O}(|V|^3)$. Navíc oproti Dijkstrovu algoritmu umí pracovat se zápornými hranami.

Floyd-Warshallův algoritmus

Myšlenka. Označme $d_{i,j}^{(k)}$ délku nejkratší cesty mezi vrcholy i a j, kde na této cestě jsou vrcholy pouze z množiny $\{1, \ldots, k\}$. Zjevně $d_{i,j}^{(0)} = w_e(i,j)$ a požadovaný výsledek odpovídá $d_{i,j}^{(|V|)}$.

Mohou nastat dvě možnosti pro $d_{i,j}^{(k)}$:

1. k není součástí nejkratší cesty $\rightarrow d_{i,j}^{(k)} = d_{i,j}^{(k-1)}$
2. k je součástí nejkratší cesty $\rightarrow d_{i,j}^{(k)} = d_{i,k}^{(k-1)} + d_{k,j}^{(k-1)}$

Odtud tedy:

$$d_{i,j}^{(k)} = \min\left\{d_{i,j}^{(k-1)}, d_{i,k}^{(k-1)} + d_{k,j}^{(k-1)}\right\}$$

Poznámka. Pokud $d_{i,i}^{(|V|)} < 0$ pro nějaké i, pak graf obsahuje cyklus záporné délky.

Floyd-Warshallův algoritmus – pseudokód

function Floyd-Warshall$(G = (V, E, w_e))$ **is**

  $\forall (u, v) \in \{1, \ldots, |V|\} \times \{1, \ldots, |V|\} : dist(u, v) \leftarrow \infty$

  $\forall v \in V : dist(v, v) \leftarrow 0$

  $\forall (u, v) \in E : dist(u, v) \leftarrow w_e(u, v)$

  for $k \leftarrow 1$ **to** $|V|$ **do**

    for $i \leftarrow 1$ **to** $|V|$ **do**

      for $j \leftarrow 1$ **to** $|V|$ **do**

        if $dist(i, j) > dist(i, k) + dist(k, j)$ **then**

          $dist(i, j) \leftarrow dist(i, k) + dist(k, j)$

        fi

      done

    done

  done

end