6. Dijkstrův algoritmus a haldy

FIXME: Nerevidováno!

Na této přednášce budeme pokračovat v problému hledání nejkratších cest v grafech ohodnocených reálnými čísly. Již jsme potkali Bellmanův-Fordův algoritmus a jeho zobecnění v podobě průzkumnického algoritmu.

Situace: Máme orientovany graf G a funkce $l : E(G) \to \mathbb{R}$ přiřazující hranám jejich ohodnocení (délky). Pro vrcholy $u, v \in V(G)$ budeme chtít spočítat jejich vzdálenost $d(u, v)$, což bude délka nejkratší cesty z u do v nebo ∞, pokud žádná cesta neexistuje.

Aby se vzdálenosti chovaly „rozumně“ (tj. jako metrika), budeme chtít, aby platily následující vlastnosti:

- $d(u, u) = 0$,
- $d(u, v) \le d(u, w) + d(w, v)$ (trojúhelníková nerovnost).

To nemusí obecně platit (kazí nám to záporné cykly), proto budeme studovat pouze grafy, v nichž záporné cykly neexistují. Pak už budou obě vlastnosti splněny, jak plyne například z následujícího lemmatu:

Lemma: V grafu bez záporných cyklů existuje ke každému nejkratšímu sledu z u do v stejně dlouhá uv-cesta.

Důkaz: Máme-li nejkratší uv-sled, který není cestou, opakuje se v něm nějaký vrchol $w \in V(G)$. Délka cyklu $l(c) \ge 0 \Rightarrow l(u \ldots v \ldots w) \le l(původnísled)$. Tento postup můžeme opakovat a po konečném počtu kroků dostaneme cestu, tedy platí trojúhelníková nerovnost. ♡

Opakování: Dijkstrův algoritmus

- $D(v)$ …dočasná vzdálenost z s do v
- Značky $Z(v)$
  - :Neviděn
  - :Viděn
  - :Hotov

1. $D(*) \leftarrow +\infty, D(s) \leftarrow 0$
2. $Z(*) \leftarrow Neviden, Z(s) \leftarrow V$
3. while $\exists v : Z(v) = V$
4.   vybereme $v : Z(v) = V, D(v) = min$
5. $Z(v) = H$
6.   for $\forall w : (v, w) \in E(G)$
7. $D(w) \leftarrow min(D(w), D(w) + l(v, w))$
8.   if $Z(w) = N \Rightarrow Z(w) \leftarrow V$

Věta: Pokud G je nezáporně ohodnocený graf, pak se Dijkstrův algoritmus zastaví a vydá $\forall v : D(v) = d(s, v)$ (tedy správné hodnoty).

Důkaz: Následující posloupností lemmat. ♡

Lemma 1: Pokud $v_0, \ldots, v_k$ je nejkratší v_0v_k-cesta, pak $v_0, \ldots, v_{k-1}$ je nejkratší v_0v_{k-1}-cesta.

Důkaz: Pokud by tomu tak nebylo, můžeme $v_0, \dots, v_{k-1}$ vyměnit za kratší cestu, a tím získat kratší $v_0 v_k$-cestu. ♡

Lemma 2: Algoritmus se zastaví po $\leq n$ průchodech cyklem.

Důkaz: Zřejmé z toho, že každý vrchol uzavřeme nejvýše jednou. ♡

Lemma 3: Po zastavení jsou hotovy právě vrcholy dosažitelné z s.

Důkaz: Viz minulá přednáška. ♡

Lemma 4: $D(v)$ uzavíraných vrcholů tvoří neklesající posloupnost.

Důkaz: V okamžiku, kdy uzavíráme v platí $\forall w \notin H : D(w) \geq D(v)$, případně přepočítáme $D(w)$ na $D(v) - l(v,w) \geq D(v)$. ♡

Lemma 5: Pokud $v \in H$, pak $D(v)$ se už nezmění.

Důkaz: Indukcí podle běhu algoritmu. ♡

Lemma 6: Pro $\forall v D(v)$ je délka nejkratší sv-cesty, jejíž vnitřní vrcholy leží všechny v H.

Důkaz:

- po 1. průchodu OK
- uzavíráme-li další vrchol v:

  a) $D(w)$ pro $w \in H$ Podle Lemma 5 se $D(w)$ nemění. Musíme nahlédnout, že se opravdu změnit nemá.

  b) $D(w)$ pro $w \notin H$ $D(w) = min D(v) + l(v,w)$

♡

Existuje pomalejší algoritmus pro grafy se zápornými hranami bez záporných cyklů. **Bellman-Fordův algoritmus**

1. $D(*) \leftarrow \infty, D(s) \leftarrow 0$
2. Opakuji
3.   Pro $\forall v \in V$
4. $prozkoumej(v)$
5.   (koukne se, jestli nejde vylepšit cestu
6.   do sousedů v)
7. dokud se nějaké $D(\dots)$ mění

Lemma 1: Pokud $D(v) < \infty$, pak existuje sled z s do v délky $D(v)$. (speciálně z toho plyne $\forall v D(v) \geq d(s,v)$)

Lemma 2: $\forall v D(v)$ nikdy neroste. **Lemma 3:** Po k fázích je $\forall v D(v) \leq$ délka nejkratšího sv-sledu o $\leq k$ hranách. *Důkaz:* Indukcí…pro $k \geq 0$ OK Indukční krok: Doběhla $k-1$ fáze, pouštíme k-tou

♡

Věta: Pro graf bez záporných cyklů se Bellman-Fordův algoritmus zastaví po nejvýše m fázích a vydá $D(v) = d(s,v)$ pro všechna v. (zřejmé z lemmat)

Lemma 4: Pokud v grafu existuje záporný cyklus dosažitelný z s, algoritmus se nezastaví. (Zajímavý test na to, zda graf obsahuje záporný cyklus.)

Časová složitost Bellman-Fordova algoritmu : $O(n * m)$

Floyd-Warshallův algoritmus G je graf se záporným ohodnocením hran, bez záporných cyklů $D^k_{i,j}$ = délka nejkratší cesty z v_i do v_j přes $v_1 \dots v_k$ $D^0_{ij} = l(v_i, v_j)$, $D^0_{ii} = 0$ $D^n_{ij} = d(v_i, v_j)$ - skutečná vzdálenost v grafu

1. $D_{i,j} \to l(v_i, v_j), D_{i,i} \to 0 \forall i, j$
2. for $k = 1$ to n
3.   for $i = 1$ to n
4.   for $j = 1$ to n
5. $D_{i,j} = min(D_{i,j}, D_{i,k} + D_{k,j})$

Věta: Floyd-Warshallův algoritmus spočítá $D_{i,j} = d(v_i, v_j)$ v čase $O(n^3)$.
Shrnutí:
$s \to *$ Dijkstrův algoritmus $O(n^2) \dots$ s haldou $O((n+m)\log m) \dots$ s regulární haldou $O(n + m * logn) \dots$ s Fibbonaciho haldou $O(n * logn + m)$ Bellman-Ford $O(n * m)$
$* \to *$ Floyd-Warshall $O(n^3)$