Toky v sítích

Petr Ryšavý

27. listopadu 2025

Katedra počítačů, FEL, ČVUT

TOKY V SÍTÍCH

Síť

Definice ***Síť*** *je uspořádaná šestice* (V, E, s, t, l, c)*, kde*

- $G = (V, E)$ *je orientovaný graf,*
- $c : E \mapsto \mathbb{Z}$ *je funkce přiřazující hranám jejich kapacity,*
- $l : E \mapsto \mathbb{Z}$ *je funkce přiřazující hranám jejich minimální povolené toky,*
- s, t *jsou dva různé vrcholy sloužící jako zdroj a stoka či spotřebič.*

- Ve většině implementací se setkáváme s $\forall E : l(e) = 0$. Rozdíl v algoritmu je ale minimální. Pro jednoduchost budeme tedy l místy vynechávat.
- Kapacity uvažujeme pouze celočíselné! Důvod je, že s iracionálními kapacitami mohou algoritmy konvergovat k optimu až po nekonečném počtu kroků.

Tok v síti

Definice ***Tok v síti*** *je funkce* $f : E \mapsto \mathbb{Z}$*, pro niž platí, že*

- *tok je omezený kapacitou pro každou hranu, tj.*

$$\forall e \in E : l(e) \leq f(e) \leq c(e);$$

- *do každého vrcholu mimo* s *a* t *stejně přiteče, jako odteče, tj.*

$$\forall v \in V \setminus \{s, t\} : \sum_{u:(u,v)\in E} f(u, v) = \sum_{u:(v,u)\in E} f(v, u).$$

Druhému bodu říkáme Kirchhoffovy zákony. Pro jednoduchost je často zapisujeme jako

$$f^+(v) = f^-(v).$$

Kirchhoffovy zákony neplatí pro zdroj a stok.

Velikost toku

Definice ***Velikost toku*** *f je rovna přebytku stoku, tj.*

$$f = f^+(t) - f^-(t).$$

Lemma

$$f = f^-(s) - f^+(s).$$

Velikost toku

Definice ***Velikost toku*** *f je rovna přebytku stoku, tj.*

$$f = f^+(t) - f^-(t).$$

Lemma

$$f = f^-(s) - f^+(s).$$

Důkaz Důsledek Kirchhoffových zákonů. ∎

Hledání maximálního toku

Definice *Pro síť* (V, E, s, t, c, l)*, že tok* f *je maximální možné.*

Příklad

FORD-FULKERSONŮV ALGORITMUS

Zlepšující cesta

- Co když najdeme cestu z s do t, kde není zcela vyčerpaná kapacita?
- Můžeme na této cestě přidat nějaký tok a zvýšit tím f.

Příklad

Protisměrná rezerva

- Počítat chybějící kapacitu pouze ve směru hran nestačí.
- Potřebujeme umět poslat i něco proti směru hrany.
- Pro hranu $(u, v) \in E$ máme volnou kapacitu $r^+ : e \mapsto \mathbb{Z}$ a $r^- : e \mapsto \mathbb{Z}$:

$$r^+(u, v) = c(u, v) - f(u, v),$$
$$r^-(u, v) = f(u, v) - l(u, v).$$

- Pak platí, že rezerva r hrany (u, v) je rovna

$$r(u, v) = r^+(u, v) + r^-(v, u) = c(u, v) - f(u, v) + f(v, u) - l(v, u).$$

- Hrana s nulovou rezervou je *nasycená*.

Příklad

Ford-Fulkersonův algoritmus

function FORD-FULKERSON(V, E, s, t, c, l) **returns** shortest path to each v

    $f \leftarrow$ libovolný tok, např. $f = l$
    while existuje cesta P, že všechny hrany $e \in P$ jsou nenasycené **do**
        $\varepsilon \leftarrow \min\{r(e) \mid e \in P\}$ ▷ Rezerva celé cesty
        for each $(u, v) \in P$ **do**
            $\delta \leftarrow \min\{f(v, u) - l(v, u), \varepsilon\}$ ▷ Co lze odečíst v protisměru
            $f(v, u) \leftarrow f(v, u) - \delta$
            $f(u, v) \leftarrow f(u, v) + \varepsilon - \delta$
        end for
    end while
end function

Souvislost s řezy

- Řez A, B rozděluje vrcholy do dvou podmnožin, že $A \cup B = V$.
- Zde volíme navíc $s \in A$ a $t \in B$.
- Definujeme $W^+(A, B)$ jako tok z A do B a $W^-(A, B)$ jako tok z B do A (značené $f(A, B)$, resp. $f(B, A)$).

Souvislost řezu a toku

Lemma *Pro libovolný řez A, B platí*

$$f = f(A, B) - f(B, A).$$

Souvislost řezu a toku

Lemma *Pro libovolný řez A, B platí*

$$f = f(A, B) - f(B, A).$$

Důkaz Indukcí přes velikost A s využitím Kirchhoffových zákonů. ∎

Korektnost Ford-Fulkersona

- Pro libovolný řez (A, B) platí z definice

$$f = f(A, B) - f(B, A) \leq c(A, B) - l(B, A),$$

  kde $c(A, B)$ je součet $c(a, b)$ pro všechny hrany jdoucí přes řez, že $a \in A$ a $b \in B$. Podobně $l(B, A)$.
- Cestu P můžeme hledat např. BFS/DFS na grafu nenasycených hran.
- Pokud P neexistuje, pak s a t jsou v různých komponentách.
- Zvolme A rovné komponentě s, a $B = V \setminus A$.
- Má-li řez všechny hrany nasycené, pak rezervy všech hran jsou nulové, což znamená, že výše platí rovnost.
- Ford-Fulkerson hledá nejenom maximální tok, hledá i řez s minimální kapacitou.

EDMOND-KARP

Problémy Ford-Fulkersona

- Nikdo nám neříká, jak vybíráme cestu P
- Při špatné volbě může F-F trvat velmi dlouho, např. skákat po jedné až do maximálního toku
- Při iracionálních vahách může dojít ke konvergenci až v nekonečnu, ε se pak mění např. 1, $\frac{1}{2}$, $\frac{1}{4}$, ...

Edmond-Karpův algoritmus

- Zařídíme, že hledáme zlepšující cesty v pořadí počtu hran
- Díky tomu každá hrana je maximálně $|V|$-krát kritická
- Díky tomu je čas běhu $\mathcal{O}(|E| \cdot |V|)$.
- Pouštíme BFS na grafu rezerv (*residual graph*).

Edmond-Karp (S. Halim)

```
int res[MAX_V][MAX_V], mf, f, s, t; // global variables
vi p; // p stores the BFS spanning tree from s
void augment(int v, int minEdge) { // traverse BFS spanning tree from s->t
    if (v == s) { f = minEdge; return; } // record minEdge in a global var f
    else if (p[v] != -1) { augment(p[v], min(minEdge, res[p[v]][v]));
                           res[p[v]][v] -= f; res[v][p[v]] += f; } }
// inside int main(): set up res, s, and t with appropriate values
mf = 0; // mf stands for max_flow
while (1) { // O(VE^2) (actually O(V^3 E) Edmonds Karp's algorithm
    f = 0;
    // run BFS, compare with the original BFS shown in Section 4.2.2
    vi dist(MAX_V, INF); dist[s] = 0; queue<int> q; q.push(s);
    p.assign(MAX_V, -1); // record the BFS spanning tree, from s to t!
    while (!q.empty()) {
        int u = q.front(); q.pop();
        if (u == t) break; // immediately stop BFS if we already reach sink t
        for (int v = 0; v < MAX_V; v++) // note: this part is slow
        if (res[u][v] > 0 && dist[v] == INF)
        dist[v] = dist[u] + 1, q.push(v), p[v] = u; // 3 lines in 1!
    }
    augment(t, INF); // find the min edge weight f in this path, if any
    if (f == 0) break; // we cannot send any more flow (f = 0), terminate
    mf += f; // we can still send a flow, increase the max flow!
}
printf("%d\n", mf); // this is the max flow value
```

Příklad

Příklad

- 11045 - My T-shirt suits me

References

- Halim, S., Halim, F., Skiena, S. S., & Revilla, M. A. (2013). Competitive Programming 3. Lulu Independent Publish.
- Martin Mareš, Tomáš Valla (2022). Průvodce labyrintem algoritmů. `https://pruvodce.ucw.cz/`

Děkuji za pozornost.

Čas na otázky!