Kombinatorická optimalizace
cvičení č.5
Nejkratší cesty v grafu

Přemysl Šůcha (suchap@fel.cvut.cz)

17. března 2010

1 Nejkratší cesty v grafu

Úloha hledání nejkratších cest patří mezi úlohy teorie grafů, které se v aplikacích vyskytují nejčastěji. Tyto úlohy bývají zadány orientovaným grafem G, jehož každá hrana $e \in E(G)$ je ohodnocena reálným číslem a(e), které nazýváme *délkou hrany* [1]. Potom *délka cesty* je součet délek jednotlivých hran tvořících cestu. Jsou-li x, y dva vrcholy grafu, pak vzdálenost $u(x, y)$ z vrcholu x do vrcholu y definujeme jako délku nejkratší cesty z x do y. Pokud taková cesta neexistuje, definujeme vzdálenost $u(x, y) = inf$. Velmi důležitou vlastnost těchto úloh vystihuje věta o *trojúhelníkové nerovnosti*.

Věta 1.1 *Jestliže graf neobsahuje cyklus se zápornou délkou, pak pro všechny trojice vrcholů i, j, k vzdálenost splňuje nerovnost*

$$u(i, j) \leq u(i, k) + u(k, j). \tag{1}$$

Jedním z algoritmů, který řeší problém nejkratších cest, je *Dijkstruv algoritmus*. Tento algoritmu pracuje správně pouze s grafy s nezápornými délkami hran. Algoritmus lze snadno upravit tak, aby se vypořádal i s hranami se zápornou délkou, ale má pak větší časovou náročnost.

Vstup: Graf G, výchozí vrchol r a ohodnocení hran $a : E(G) \rightarrow \mathbb{R}^+$
takové, že pro všechny hrany platí $a(e) \geq 0$.
Výstup: G' Pro každý vrchol v hodnoty $U(v)$, $ODKUD(v)$

$U(r) := 0$;
$U(v) :=$ inf pro $v \neq r$;
$D := \emptyset$;
$ODKUD(x) := Inf$ pro všechny v;
while *pro libovolný* $v \in (V \setminus D)$ *platí* $U(v) \neq$ inf **do**
    Z množiny $V \setminus D$ vyber vrchol x, který má nejnižší hodnotu $U(x)$.;
    $D = D \cup (x)$;
    forall $e \in E^+(x)$ **do**
        y = koncový vrchol e;
        if $U(x) + a(e) < U(y)$ **then**
            $U(y) := U(x) + a(e)$;
            $ODKUD(y) = x$
        end
    end
end

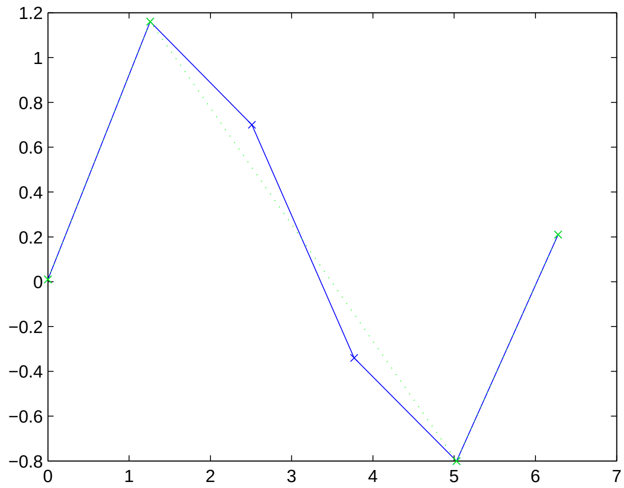
Algoritmus používá množinu D, ve které si uchovává seznam vrcholů, jejichž hodnota U je již definitivní. Proměnná $ODKUD$ slouží k rekonstrukci nejkratší cesty v grafu. Na pozici y je vždy uložen index vrcholu x, ze kterého se do vrcholu y „přijde“ po nejkratší cestě. Složitost základní verze Dijkstrova algoritmu je $O(n^2)$.

2 Aproximace funkce

Málo známou, ale velmi zajímavou aplikací hledání nejkratších cest v grafu je aproximace dat po částech spojitou funkcí [2]. Vstupem pro aproximaci dat může být například množina n naměřených vzorků dat $\{(x_i, f(x_i))\}$ (viz obrázek 1). Cílem je vybrat co nejmenší počet bodů naměřených dat $(x_i, f(x_i))$ tak, aby byla chyba aproximace pokud tyto body proložíme úsečkou co nejmenší (viz obrázek 2).

```
x = [ 0     1.26  2.51  3.77  5.03  6.28];
f = [ 0.01  1.16  0.70 -0.34 -0.80  0.2100];
```

Obrázek 1: Aproximovaná funkce.



Obrázek 2: Původní (modrá) a aproximovaná (zelená) funkce.

Tento problém se dá formulovat pomocí hledání nejkratší cesty v grafu G. Odpovídající graf G je tvořen n vrcholy, přičemž každý vrchol v_i odpovídá jednomu vzorku dat $(x_i, f(x_i))$. Graf G obsahuje hranu pro každou dvojici vrcholů i, j, takové že $i < j$. Příklad takového grafu je ukázán na obrázku 4. Hrana mezi vrcholy i, j určuje, že aproximujeme úsek mezi vzorky $(x_i, f(x_i))$ v $(x_j, f(x_j))$ úsečkou. Váha této hrany se skládá ze dvou komponent: penalizaci odpovídající počtu potřebných vzorků a penalizaci odpovídající chybě aproximace. První komponenta je vážena váhou α a druhá komponenta váhou β. Potom váha hrany $c_{i,j}$ může být vypočtena například

$$c_{i,j} = \alpha + \beta \left[\sum_{k=i}^{j} \left(f(x_k) - f'(x_k) \right)^2 \right], \tag{2}$$

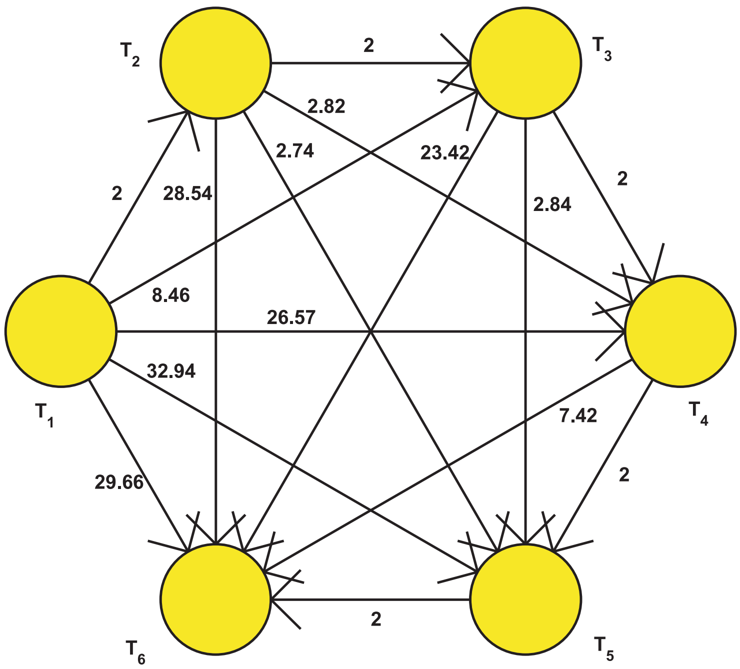
kde $f'(x_k)$ je funkční hodnota nahrazená aproximací. Funkční hodnotu $f'(x_k)$ můžeme vypočítat

$$f'(x) = f(x_i) + (x - x_i) \cdot \frac{f(x_j) - f(x_i)}{x_j - x_i} \tag{3}$$

Váhová matice, reprezentující data z obrázku 1, je zobrazena na obrázku 3. Tím odpovídající graf se nalézá na obrázku 4. Pomocí algoritmu na hledání nejkratších cest je možné najít optimální aproximaci vstupní funkce.

```
c = [ 0     2.00  8.46 26.57 32.94 29.66; ...
      0     0     2.00  2.82  2.74 28.54; ...
      0     0     0     2.00  2.84 23.42; ...
      0     0     0     0     2.00  7.42; ...
      0     0     0     0     0     2.00; ...
      0     0     0     0     0     0];
```

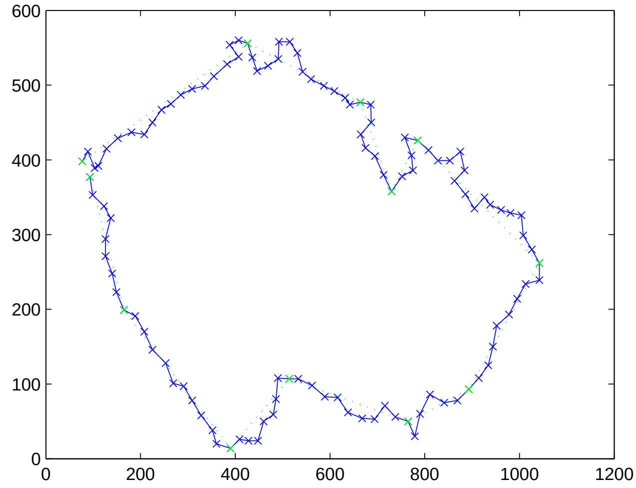
Obrázek 3: Matice vah $c_{i,j}$ pro $\alpha = 2$ a $\beta = 10$.



Obrázek 4: Graf odpovídající řešenému problému.

3 Aproximace vektorového obrázku

Stejným způsobem lze řešit aproximaci vektorového obrázku reprezentovaného jako posloupnost bodů (x_i, y_i). Jediný rozdíl je ve výpočtu váha hrany $c_{i,j}$. Tu je potřeba počítat například jako sumu kvadrátů vzdáleností bodů (x_k, y_k) od přímky z (x_i, y_i) do (x_j, y_j) kde $i < k < j$. Příklad aproximace bodů hranic České Repuliky je znázorněn na obrázku 5.



Obrázek 5: Původní body (modrá) a aproximované body (zelená).

Úkol: Naprogramujte Dijkstrův algoritmus. Ten použijte na určení bodů pro optimální aproximaci funkce z obrázku 1. Jako vstupní data použijte vektory z obrázku 1. Nakonec zobrazte výsledek, jak je ukázáno na obrázku 2.
Dobrovolně si můžete vyzkoušet variantu pro aproximaci vektorového obrázku. Postup je stejný, jen váhy hran $c_{i,j}$ je potřeba počítat jinak. Data obrázku naleznete na stránkách předmětu pod návodem k tomuto cvičení.

Rada: Na zobrazení výsledku použijte funkce `plot` a `hold on`. Pro množinové operace slouží funkce `union`, `intersect`, `setdiff`, a `ismember`. Pro zjištění indexu prvku vektoru použijte funkci `min`.

Reference

[1] J. Demel, *Grafy a jejich aplikace*. Academia, second ed., 2002.

[2] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall; United States Ed edition, 1993.

[3] B. H. Korte and J. Vygen, *Combinatorial Optimization: Theory and Algorithms*. Springer, third ed., 2006.