Základy matematiky a statistiky pro humanitní obory II

Pavel Rychlý   Vojtěch Kovář

Fakulta informatiky, Masarykova univerzita
Botanická 68a, 602 00 Brno, Czech Republic
{pary, xkovar3}@fi.muni.cz

část 3

Obsah přednášky

Souvislé komponenty

- Souvislé komponenty
  - největší souvislé podgrafy
  - → mezi každými dvěma vrcholy existuje cesta
- Silně souvislé komponenty
  - v případě orientovaných grafů
  - mezi každými dvěma vrcholy existuje cesta tam i zpět

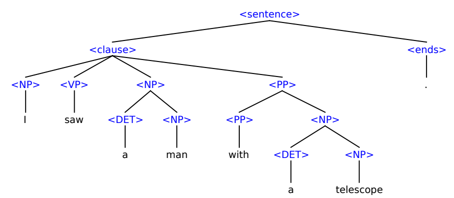
Vzdálenost v grafu

- Délka cesty
  - neohodnocený graf: počet hran v cestě
  - ohodnocený graf: součet ohodnocení jednotlivých hran v cestě
- Vzdálenost mezi dvěma vrcholy X a Y
  - je délka nejkratší cesty z X do Y

Kořen a listy stromu

- Kořen stromu
  - jeden vyznačený vrchol
  - kreslíme většinou nahoře :)
- Listy stromu
  - vrcholy stupně 1, které nejsou kořenem
  - kreslíme většinou dole

Příklad – syntaktický strom



Kostra grafu

- Podgraf, který
  - obsahuje všechny vrcholy původního grafu
  - je strom
  - → musíme odstranit všechny cykly
- Minimální kostra grafu
  - pro ohodnocený graf
  - kostra s nejmenším součtem ohodnocení hran
  - analogicky maximální kostra

Procházení grafu

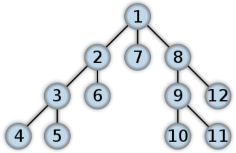
- Např. hledáme určitý vrchol, chceme projít všechny, ...
- Procházení do hloubky – depth-first search
  - začínáme z nějakého vrcholu, ten označíme
  - označíme libovolný sousední neoznačený vrchol a pokračujeme z něj
  - pokud to dál nejde (všechny sousední vrcholy jsou označené), vrátíme se k nejbližšímu vrcholu, ze kterého to ještě jde

Procházení grafu

- Procházení do šířky – breadth-first search
    - začínáme z nějakého vrcholu, ten označíme
    - vybereme všechny sousední neoznačené vrcholy a přidáme je do seznamu
    - postupně ze začátku seznamu odebíráme a provádíme předchozí kroky
    - končíme, když je seznam prázdný

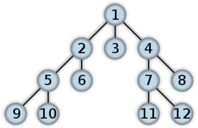
Procházení do hloubky z vrcholu u

- $DFS(G, u)$
- ============
- označ u
- for všechny hrany (u, v) vycházející z vrcholu u:
    - if v není označen:
        - $DFS(G, v)$



Procházení do šířky z vrcholu u

- $BFS(G, u)$
- ============
- $Q = [u]$
- while Q je neprázdný:
    - odstraň první prvek z Q a přiřaď jej do t
    - označ t
    - přidej všechny neoznačené sousedy t na konec Q



Kruskalův algoritmus

- Vstup
    - neorientovaný graf G
    - ohodnocení hran w
- Výstup
    - minimální kostra grafu G
- Algoritmus je tzv. hladový
    - v každém kroku vybírá lokálně optimální možnost
- Idea
    - setřídit hrany podle ohodnocení
    - v každém kroku přidat do kostry tu nejmenší, která nevytvoří cyklus
    - udržujeme si seznam souvislých komponent kostry

Kruskalův algoritmus (G, w)

- $K \leftarrow [\,]; comp \leftarrow \{\}$
- for u in $G(V)$:
    - $comp[u] \leftarrow set(u)$
- setřiď $G(E)$ podle w
- for (u, v) in $G(E)$:
    - if $comp[u] \neq comp[v]$:
        - $K.append((u, v))$
        - $newset = union(comp[u], comp[v])$
        - for x in $newset$: $comp[x] \leftarrow newset$
- K je minimální kostra grafu

Dijkstrův algoritmus

- Vstup
    - graf s hranami ohodnocenými funkcí w
    - ohodnocení hran musí být nezáporné
    - počáteční vrchol s
- Výstup
    - vzdálenosti z vrcholu s do všech dalších vrcholů grafu
- Idea
    - udržujeme si nejmenší známé vzdálenosti do všech vrcholů
    - na začátku nekonečno
    - procházíme postupně vrcholy a hodnoty upravujeme

Dijkstrův algoritmus (G, s)

- for u in $G(V)$:
    - $d[u] \leftarrow infinity$
- $d[s] \leftarrow 0$
- $N \leftarrow G(V)$
- $p \leftarrow \{\}$
- while $N \neq [\,]$:
    - $u \leftarrow$ vrchol z N s nejmenší hodnotou $d[u]$
    - for všechny hrany (u, x) vycházející z vrcholu u:
        - $alt \leftarrow d[u] + w((u, x))$
        - if $alt < d(x)$: $d[x] \leftarrow alt; p[x] \leftarrow u$
    - odstraň u z N
- d jsou vzdálenosti vrcholů z vrcholu s
- p obsahuje předchozí vrcholy na nejkratší cestě z s