Teorie grafů

19. přednáška z LGR

Obsah

Silně souvislé grafy

Definice

Orientovaný graf je *silně souvislý*, pokud pro každé dva jeho vrcholy u, v existuje orientovaná cesta z u do v (a tudíž i zpět).

Tvrzení

Orientovaný graf je silně souvislý, právě když je souvislý a každá jeho hrana leží v nějakém cyklu.

Silně souvislé grafy

Definice

Každý maximální podgraf grafu G, který je silně souvislý, se nazývá *komponenta silné souvislosti* grafu G (nebo také *silně souvislá komponenta*).

Poznámka

Komponenta silné souvislosti je jednoznačně určena množinou svých vrcholů, je to podgraf indukovaný danou množinou vrcholů. Budeme se silně souvislými komponentami pracovat podle potřeby jako s množinami vrcholů nebo jako s podgrafy.

Silně souvislé grafy

Definice

Nechť $G = (V, E)$ je orientovaný graf. Kondenzace grafu G je orientovaný graf $G_{kon} = (V_{kon}, E_{kon})$, kde

- V_{kon} je množina všech silně souvislých komponent grafu G,
- pro $K_1 \neq K_2$ je $(K_1, K_2) \in E_{kon}$ právě tehdy, když existují vrcholy $u, v \in V$, $u \in K_1$, $v \in K_2$ a $(u, v) \in E$.

Silně souvislé grafy

Tvrzení

Kondenzace grafu je vždy acyklický graf.

Poznámka

Zdrojová komponenta orientovaného grafu G je komponenta, která je v kondenzaci G_{kon} zdrojem ($d_{in}(K) = 0$) a *výlevková komponenta* je v kondenzaci G_{kon} výlevkou ($d_{out}(K') = 0$).
Každý orientovaný graf má zdrojovou i výlevkovou komponentu.

Prohledávání grafu

Vsuvka: Jak najít komponenty souvislosti neorientovaného grafu?

Pro hledání komponent souvislosti grafu můžeme použít prohledávání grafu do hloubky nebo do šířky.

- Prohledávání grafu do hloubky (DFS = depth first search) z vrcholu r - jde po větvi co nejdále, pak se vrací a hledá odbočky, navštívené vrcholy dává do zásobníku;
- Prohledávání grafu do šířky (BFS = breadth first search) z vrcholu r - najde nejdříve všechny sousedy vrcholu r, pak sousedy sousedů atd., navštívené vrcholy dává do fronty;

Strom prohledávání z vrcholu r je tvořen všemi nalezenými vrcholy spolu s hranami, které byly použity k jejich prvnímu nalezení.

Prohledávání grafu

Pozorování

Pro prohledávání grafu do hloubky nebo do šířky platí:

- Strom prohledávání s kořenem r obsahuje všechny vrcholy dostupné z vrcholu r. Vrcholy stromu prohledávání s kořenem r tvoří komponentu souvislosti grafu obsahující vrchol r.
- Pokud prohledávání opakujeme z dosud nenavštíveného vrcholu r', bude strom prohledávání opět obsahovat všechny vrcholy dostupné z vrcholu r', tedy komponentu souvislosti obsahující vrchol r'.
- Opakovaným prohledáváním nalezneme postupně všechny komponenty souvislosti.

Prohledávání grafu

Prohledání grafu do hloubky či do šířky z vrcholu r

Vstup: Neorientovaný graf $G = (V, E)$, vrchol $r \in V$.
Výstup: Strom $T_r = (V_r, E_r)$ cest z vrcholu r do všech dostupných vrcholů.

Datové struktury: Pole N délky $n = |V|$, kde $N(v) = \mathit{true}$, pokud byl vrchol v již navštíven. Pole P délky $m = |E|$, kde $P(e) = \mathit{true}$, pokud byla hrana e již použita. Zásobník S (pro DFS) či fronta Q (pro BFS).

Prohledávání grafu

Prohledání grafu do hloubky z vrcholu r

(inicializace)

- for all $v \in V$ do $N(v) \leftarrow \mathit{false}$ enddo
- for all $e \in E$ do $P(e) \leftarrow \mathit{false}$ enddo
- $S \leftarrow \emptyset$

(procedura) DFS(r)

- vlož r do zásobníku S, $N(r) \leftarrow \mathit{true}$
- $V_r \leftarrow \{r\}$, $E_r \leftarrow \emptyset$

Prohledávání grafu

- while $S \neq \emptyset$ do
  - $v \leftarrow top(S)$ (to nezmění S!)
  - if existuje nepoužitá hrana $e = \{v, w\}$ then
    - vyber hranu $e = \{v, w\}$, pro níž je $P(e) = false$
    - $P(e) \leftarrow true$
    - if $N(w) = false$ then
      - vlož w do zásobníku S, $N(w) \leftarrow true$
      - $V_r \leftarrow V_r \cup \{w\}$, $E_r \leftarrow E_r \cup \{e = \{v, w\}\}$ endif
  - else odstraň $v = top(S)$ ze zásobníku S endif
  - enddo
- output $T_r = (V_r, E_r)$

Prohledávání grafu

Prohledání grafu do šířky z vrcholu r

Procedura BFS(r) vypadá stejně, pouze navštívené vrcholy dává do fronty Q.
Strom prohledávání do šířky z vrcholu r obsahuje nejkratší cesty z r do dostupných vrcholů, co do počtu hran.

Poznámka

Pro orientované grafy vypadají procedury DFS(r) a BFS(r) stejně, pouze hrany jsou orientované $e = (v, w)$.

Prohledávání grafu

Hledání komponent souvislosti

Vstup: Neorientovaný graf $G = (V, E)$.
Výstup: Komponenty souvislosti, resp. množiny jejich vrcholů.

(inicializace)

- for all $v \in V$ do $N(v) \leftarrow \mathit{false}$ enddo
- for all $e \in E$ do $P(e) \leftarrow \mathit{false}$ enddo
- $i \leftarrow 0$, $S \leftarrow \emptyset$ (resp. $Q \leftarrow \emptyset$)

(hledání komponent souvislosti)

- while 'existuje nenavštívený vrchol' do
  - $r \leftarrow v$, kde $N(v) = \mathit{false}$
  - DFS(r) (resp. BFS(r))
  - $i \leftarrow i + 1$, $K_i \leftarrow V_r$ enddo
- output $K_1, \ldots, K_i$

Prohledávání grafu

Časová náročnost

Označme opět m počet hran, n počet vrcholů grafu.
Prohledávání grafu do hloubky nebo do šířky z vrcholu r trvá čas $O(m)$, po každé hraně stromu prohledávání projdeme jednou směrem vpřed (a pomyslně podruhé při návratu u DFS).

Prohledání celého grafu, tudíž i nalezení komponent souvislosti vyžaduje čas $O(m+n)$.

Silně souvislé komponenty

Jak najít komponenty silné souvislosti orientovaného grafu?

Pozorování

Pro hledání silně souvislých komponent orientovaného grafu použijeme také prohledávání do hloubky nebo do šířky, ale situace bude komplikovanější.

- Strom prohledávání s kořenem r obsahuje všechny vrcholy orientovaně dostupné z vrcholu r, nemusí z nich však existovat orientované cesty zpět do r. Strom prohledávání se může rozpadnout na více silně souvislých komponent.
- Silně souvislá komponenta obsahující vrchol r je podmnožinou množiny vrcholů stromu prohledávání s kořenem r.

Silně souvislé komponenty

Pozorování

- Pokud prohledávání opakujeme z dosud nenavštíveného vrcholu r', nemusíme získat strom všech vrcholů orientovaně dostupných z vrcholu r' (některé už mohly být navštíveny z vrcholu r). Strom prohledávání bude obsahovat pouze vrcholy orientovaně dostupné z vrcholu r' v podgrafu ještě nenavštívených vrcholů.
- Silně souvislá komponenta obsahující vrchol r' je podmnožinou množiny vrcholů stromu prohledávání s kořenem r'.

Silně souvislé komponenty

Tvrzení

Pokud vrchol r patří do výlevkové komponenty orientovaného grafu, pak strom prohledávání do hloubky nebo do šířky obsahuje vrcholy jediné silně souvislé komponenty (té, v níž leží vrchol r).

První myšlenka algoritmu

Prohledat orientovaný graf z vrcholu ve výlevkové komponentě, najdeme tím právě tuto silně souvislou komponentu. Když ji z grafu vyhodíme, bude mít vzniklý podgraf opět výlevkovou komponentu, najdeme ji prohledáním z libovolného vrcholu v ní. Postup můžeme opakovat, dokud nenajdeme všechny silně souvislé komponenty.

Ovšem jak najít vrchol ve výlevkové komponentě???

Silně souvislé komponenty

Obohacené prohledávání do hloubky

Lze efektivně najít vrchol ve zdrojové komponentě pomocí DFS:

- Při prohledání celého orientovaného grafu do hloubky je kořen posledního stromu prohledávání ve zdrojové komponentě.
- Kořen posledního stromu je vrchol, který jsme zpracovali, opustili jako poslední. Obohatíme prohledávání do hloubky o číslování vrcholů ve chvíli, kdy je opouštíme, tj. když už z nich nevede nepoužitá hrana. První zpracovaný vrchol má nejnižší číslo, poslední zpracovaný vrchol má nejvyšší číslo. Případně můžeme vrcholy při opouštění vkládat do jiného zásobníku Z.

Silně souvislé komponenty

Tvrzení

Při prohledávání orientovaného grafu do hloubky s číslováním opouštěných vrcholů vedou všechny mezikomponentové hrany z vrcholu s vyšším číslem opouštění do vrcholu s nižším číslem opouštění (aneb v zásobníku Z směrem od vrcholu ke dnu).

Poznámka

Pro acyklický graf tak získáme inverzní očíslování k topologickému očíslování vrcholů.

Silně souvislé komponenty

Tvrzení

Nechť $G = (V, E)$ je orientovaný graf. Vytvořme orientovaný graf $G^{op} = (V, E^{op})$, který má opačně orientované hrany.

- G a G^{op} mají stejné komponenty silné souvislosti.
- Zdrojová komponenta v G je výlevkovou komponentou v G^{op} a naopak.

Aneb silně souvislé komponenty můžeme místo v G hledat v G^{op}. Vrcholy výlevkových komponent grafy G^{op} budou vrcholy zdrojových komponent grafu G a najdeme je přes obohacené DFS.

Silně souvislé komponenty

Kosarajův - Sharirův algoritmus

Vstup: Orientovaný graf G.
Výstup: Silně souvislé komponenty grafu G.

1. Prohledáme graf G do hloubky a zapamatujeme si pořadí, ve kterém jsme je opouštěli. Např. je dáváme při opouštění do zásobníku Z, výstupem první části je zásobník Z.
2. V grafu G obrátíme hrany, dostaneme graf G^{op}.
3. Prohledáme graf G^{op} do hloubky (nebo do šířky) a za kořeny stromů prohledávání bereme vždy nenavštívený vrchol z topu zásobníku Z (od nejvyššího čísla opouštění).
4. Vrcholy stromů druhého prohledávání jsou pak vrcholy jednotlivých silně souvislých komponent.

Silně souvislé komponenty

Korektnost algoritmu

- Terminace - prohledávání do hloubky (či do šířky) skončí
- Parciální korektnost - při druhém prohledávání platí invariant: "Kořen každého stromu prohledávání je ve výlevkové komponentě podgrafu grafu G^{op} indukovaného dosud nenavštívenými vrcholy."
  Tudíž vrcholy stromu prohledávání tvoří silně souvislou komponentu grafu G. (Důkaz plyne z předchozích tvrzení.)

Silně souvislé komponenty

Kosarajův - Sharirův algoritmus

Vstup: Orientovaný graf $G = (V, E)$, kde pro každý vrchol v je zadán seznam $A(v)$ hran s počátečním vrcholem v.
Výstup: Komponenty silné souvislosti, resp. množiny jejich vrcholů.

Datové struktury: Pole N délky $n = |V|$, kde $N(v) = \mathit{true}$, pokud byl vrchol v již navštíven. Pole P délky $m = |E|$, kde $P(e) = \mathit{true}$, pokud byla hrana e již použita. Zásobník S (pro DFS) či fronta Q (pro BFS), zásobník Z.

Silně souvislé komponenty

Kosarajův - Sharirův algoritmus

(Prohledej G do hloubky a zapamatuj si časy opouštění vrcholů.)

- for all $v \in V$ do $N(v) \leftarrow \mathit{false}$ enddo
- for all $e \in E$ do $P(e) \leftarrow \mathit{false}$ enddo
- $S \leftarrow \emptyset$, $Z \leftarrow \emptyset$
- while existuje nenavštívený vrchol do
  - $r \leftarrow v$, kde $N(v) = \mathit{false}$
  - DFS(r), který je v else-větvi (tj. když už neexistuje nepoužitá hrana z vrcholu v) obohacený o příkaz "vlož vrchol v do zásobníku Z"
  - enddo (výstupem této části je zásobník Z)

(Otoč hrany grafu G a vytvoř tak G^{op}, zadaný seznamy $B(v)$.)

- for all $e = (v, w) \in E$ do $B(w) \leftarrow B(w) \cup \{e' = (w, v)\}$ enddo

Silně souvislé komponenty

(Prohledej $G^{op} = (V, E^{op})$ do hloubky, resp. do šířky, přitom kořeny stromů ber z topu zásobníku Z.)

- for all $v \in V$ do $N(v) \leftarrow \mathit{false}$ enddo
- for all $e \in E^{op}$ do $P(e) \leftarrow \mathit{false}$ enddo
- $i \leftarrow 0$, $S \leftarrow \emptyset$ (resp. $Q \leftarrow \emptyset$)
- while $Z \neq \emptyset$ do
  - if $N(\mathit{top}(Z)) = \mathit{false}$ then
    - $r \leftarrow \mathit{top}(Z)$
    - DFS(r) (resp. BFS(r))
    - $i \leftarrow i + 1$, $K_i \leftarrow V_r$ endif
  - odstraň $\mathit{top}(Z)$ ze zásobníku Z enddo

(Vrcholy stromů prohledávání jsou silně souvislé komponenty.)

- output $K_1, \ldots, K_i$

Silně souvislé komponenty

Časová náročnost

Kosarajův - Sharirův algoritmus na hledání silně souvislých komponent v podstatě udělá dvakrát prohledávání grafu (poprvé nutně do hloubky). Potřebný čas je $O(m + n)$, kde m je počet hran, n je počet vrcholů grafu.

Orientované grafy

Literatura

- J. Demel: Grafy a jejich aplikace, Academia, 2015.
- J. Matoušek, J. Nešetřil: Kapitoly z diskrétní matematiky, Nakladatelství Karolinum, 2000.
- M. Dostál: Cvičení k přednášce LGR (najdete v nich důkazy některých tvrzení z přednášky a mnoho dalších příkladů).