The Postal Owl

Logged in: Zdeněk Tomis (home) Logout

Algoritmy a datové struktury 1

Back to the course

DÚ 5: kritické hrany

Deadline: 2022-04-05 09:00 (27 days ago)

Pavel Veselý — modified 2022-03-18 11:09 (45 days ago) — reply

Kritická hrana budiž taková, která leží na všech nejkratších cestách mezi zadanými vrcholy s a t. Jinými slovy jde o hranu, jejíž prodloužení by ovlivnilo vzdálenost mezi s a t. Navrhněte algoritmus, který najde všechny kritické hrany v orientovaném grafu s nezápornými délkami hran.

Pavel Veselý — 2022-03-21 10:20 (42 days ago) — reply

PS: Pro jednoduchost můžete počítat i s exponenciálně velkými čísly v čase $O(1)$.

Zdeněk Tomis — modified 2022-03-25 22:16 (37 days ago) — edit — reply

Problém rozdělíme na dvě části:

1. Nalezení všech nejkratších cest
2. Nalezení mostů v grafu tvořeném nejkratšími cestami

Nalezení všech nejkratších cest

Jedná se o problém, který jsme řešili na cvičení. Použijeme upravený Dijkstrův algorimus, který ke každému vrcholu přiřadí nejen hodnotu - délku nejkratší cesty - ale také seznam vrcholů, které jsou v nejkratších cestách předchůdci.

Konkrétně se v při každé relaxaci podíváme na aktuální hodnotu každého souseda a součet hodnoty relaxovaného vrcholu a hodnotu hrany vedoucí do toho souseda.

- Pokud je součet větší, už známe kratší cestu do vrcholu a nic neděláme.
- Pokud je součet roven aktuální hodnotě souseda, přidáme sousedovi do seznamu přechůdců na nejkratších cestách aktuálně relaxovaný vrchol.
- Pokud je součet menší, přiřadíme sousedovi tento součet a jeho seznam nejdříve vymažeme a pak tam přidáme relaxovaný vrchol.

Až algoritmus skončí, máme v každém vrcholu uloženy všechny předchůdce na některé nejkratší cestě.

Časovou složitost algoritmu jsme asymptoticky neovlivnili, protože v každém kroku děláme konstantně více kroků.

Výsledná reprezentace se nám navíc bude hodit pro další krok hledání mostů.

Nalezení mostů

Problém hledání mostů v orientovaném grafu jsme řešili na přednášce, spustíme DFS z koncového vrcholu, jako seznam sousedů každého vrcholu využijime dříve uložený seznam předchůdců na nějaké nejkratší cestě.

V průběhu DFS ukládáme také pro každý vrchol v hodnotu $low(v)$, což intuitivně reprezentuje nejvyšší bod, do kterého se můžeme zpětnou hranou vrátit z v nebo z vrcholu v podstromu v. Můžeme to spočítat bez zpomalení algoritmu postupně při zavírání vrcholu, a to takto: u listu je to minimum z $in(x)$ všech zpětných hran vx. U nelistů je to minimum této hodnoty a ještě hodnot všech potomků. Všichni potomci jsou při zavírání vrcholu už zavřeni, takže výpočet bude fungovat. Při výpočtu minima zvážím každou hranu maximálně jednou, takže to asymptotickou časovou složitost opravdu neovlivní.

Pro každou hranu xy pak okamžitě můžu zjistit, jestli je mostem: Mostem není právě tehdy, když $low(y) < in(y)$, protože pak se můžu nějakou zpětnou hranou vrátit nad y a tudíž vytvořím kružnici s hranou xy. Jednoduše pak hrana je mostem, pokud $low(y) \geq in(y)$.

Toto můžu zvážit při zpětném průchodu hranou přímo v průběhu dfs, opět ho asymptoticky nezpomalím, protože udělám jedno porovnání.

Vlastnosti algoritmu

Správnost

Za přepdpokladu, že dílčí algoritmy jsou správně, musí být algoritmus správný, protože se jedná přesně o problém hledání mostů v grafu nejkratších cest.

Kdyby existovala hrana, která je na nějaké nejkratší cestě, ale není kritická, existuje i nejkratší cesta, která tuto hranu neobsahuje, tudíž leží na kružnici a není mostem.

Konečnost

Algoritmus se zastaví, jelikož jeho dílčí podalgoritmy se zastaví. Konečnost Dijkstrova algoritmu a DFS jsme již ověřovali.

Časová složitost

Jak jsem se již snažil nahlédnout, použili jsme dva standardní algoritmy a jejich asymptotickou časovou složitost jsme neovlivnili.

DFS běží v $O(m + n)$, Dijkstra lze pomocí Fibbonaciho haldy implementovat v $O(n \log n + m)$, součet patří do asymptotické třídy $O(n \log n + m)$.

Časová složitost problému je zdola omezena problémem hledání nejkratší cesty, takže rychlejší algoritmus nenajdeme.

Prostorová složitost

Prostorovou složitost jsme zvětšili u obou algoritmů, u druhého pouze konstanta-krát (samotné hrany už jsou výstup). U prvního algoritmu si netriviálně u každého vrcholu ukládáme seznam vrcholů, takže prostorová složitost se zvětší maximálně z $O(n)$ na $O(n^2)$. Kdybychom měli úplný graf, kde má každá hrana hodnotu nula, uložili bychom si opravdu n^2 vrcholů, takže při aktuálním algoritmu nelze prostorovou složitost snížit, lepší řešení by musela předejít ukládání nejkratších cest, od pohledu se ale zdá nemožné počítat kritičnost hran, když ještě nevíme, jestli budou ležet na nejkratší cestě (pokud nejkratší cesty nenalezneme nejdříve, nemůžeme si být u žádné hrany jisti, jestli na nejkratší cestě leží).

Pavel Veselý — 2022-03-30 16:41 (33 days ago) — reply

PPS: vzhledem k tomu, že mám do pondělí spoustu jiné práce, jsem prodloužil termín na úkol do dalšího pondělí a řešení budu opravovat až poté (s možností opravy, pokud mi nebude jasný algoritmus). Omlouvám se těm, kdo řešení odevzdali s předstihem.

Pavel Veselý — 2022-04-05 11:51 (27 days ago) — reply

Výborně, to je správné řešení. Chybí jen pár pozorování: proč mosty v grafu nejkratších cest mezi s a t musí být kritické hrany? Jaká je celková velikost seznamu předchůdců a jaká je celková časová složitost práce s těmito seznamy?

Co se týče časové a prostorové složitosti, prostorová nemůže být větší než časová (v běžném výpočetním modelu).

Points: 10.00

Return

New post (You can use Markdown with KaTeX math here)

Attachment (PDF or UTF-8 text): Durchsuchen... Keine Datei ausgewählt.

Submit Preview

---

mj@ucw.cz.