Prohlídkové okruhy

Daniel Skýpala

16. prosince 2021

1 Zamyšlení

"Je možné předpokládat, že počet podmnožin křižovatek, kde z každé křižovatky jedné podmnožiny je možné se dostat do všech ostatních křižovatek této podmnožiny a zpět, je tři."

Můžeme si všimnout, že definice podmonožin křižovatek je ekvivalentní s definicí silně souvilé komponenty. Každý vrchol leží v jedné právě jedné silně souvislé komponentě. Pokud hledáme cestu mezi vrcholy a, b pojďme rozebrat jednotlivé případy, jestli existuje cesta mezi nimi v závislosti na tom, v jakých jsou komponentách:

1. Pro libovolné dva vrcholy ve stejné silné komponentě cesta existuje (z definice).
2. A pokud jsou dva vrcholy v různých komponentách, tak musí existovat posloupnost hran mezi těmito dvěma vrcholy. V jedné komponentě se umíme dostat z každého vrcholu do každého, takže nás zajímají jen hrany mezi komponentami. Musí tedy existovat posloupnost hran mezi komponentami, taková že se dostaneme ze startovní komponenty do cílové.

2 Popis algoritmu

Samotný alogritmus má tři části:

1. Najití silně souvislývh komponent
2. Spočítaní dosažizelnosti mezi komponentami
3. Zodpovězení dotazů

2.1 Najití silně souvislých komponent

Na určení silně souvislých komponent existuje více algoritmů, z čehož jsem si já vybral Korasajův.

Algoritmus probíhá následovně:

1. Uděláme si inverzi G^T grafu G (pokud hrana v G vedla z a do b, tak v G^T vede z b do a).

2. Vyrobíme si zásobník.

3. Opakovaně spouštíme prohledávání do hloubky na G^T z každého nenavštíveného vrcholu (nenavštivujeme již navštívené). Pokaždé když vrchol opouštíme, přidáme si ho do zásobníku.

4. Postupně odstraňujeme vrcholy ze zásobníku a spouštíme na ně prohledávání do hloubky v G (a zase nenavštivujeme již navštívené). Všechny vrcholy, které navštívíme v jednom spuštění dfs, jsou v jedné komponentě.

Proč toto vlastně funguje? Předtím než se pustíme do objasňování složitějších částí, vyřešeme ty jednoduší:

1. *V zásobníku bude každý vrchol právě jednou.* Každý vrchol tam určitě je, protože jsme ho navštívili (jinak bychom na něj spustili dfs) a tedy jsme ho museli i opustit (a při tom přidat do zásobníku). Zároveň, aby byl v zásobníku víckrát než jednou, museli bychom ho opustit, a tedy i navštívit víckrát. Nicméně my nenavštěvujeme vrcholy opakovaně.

2. *Každý vrchol je v právě jedné komponentě.* Na všechny nenavštívené vrcholy spouštíme dfs, pro každý vrchol v zásobníku (to jsou všechny vrcholy). No a každý navštívený vrchol je v jedné komponentě. Navštívené vrcholy nenavštěvujeme znovu.

3. *Pokud existuje cesta z A do B i z B do A, jsou v jedné komponentě, jinak jsou v různých komponentách.* Pojďme rozebrat všechny případy:

   - $A \not\leftrightarrow B$ Pokud neexistuje cesta ani jedním směrem, tak dfs v G nemá šanci dojít ve stejném spuštění z A do B a tedy je zařadit do stejné komponenty.
   - $A \rightarrow B$ V zásobníku bude B po A. Buď jsme při spuštění dfs v G^T na B neměli navštívené ani A, ani B. Pak jsme z B došli do A (pozor - graf je invertovaný), a tedy jsme A opuštěli dříve, tedy bude v zásobníku dříve. Pokud bylo A již navštívené, už v zásobníku je. (V G^T nejde z A dojít do B, protože pak by cesta byla obousměrná, což porušuje náš původní předpoklad.)

     Když je tedy B v zásobníku po A, bude na B spuštěna konstrukce komponent (dfs v originálním grafu) dříve. A vzhledem k tomu, že z B nejde v G dojít do A, tak B bude v komponentě bez A. A při spuštění dfs na A už bude B navštívené, takže nebudou ve stejné komponentě.
   - $A \leftarrow B$ Toto je ekvivalentní $B \rightarrow A$, což je popsáno v předchozím bodě.

- $A \leftrightarrow B$ Nyní sice o pořadí v zásobníku nemůžeme nic říct, ale podívejme se na konstrukci komponent. Buď spustíme dfs nejdřív na A, v tom případě dojdeme z A do B a oba vrcholy bude ve stejné komponentě. Nebo spustíme dfs na B, v tom případě dojdeme do A a vrcholy budou také ve stejné komponentě.

Čímž jsme dokázali korekntonst hledání komponent.

2.2 Spočítaní dosažizelnosti mezi komponentami

Základní myšlenkou zde je sloučit si vrcholy v jedné komponentě na jeden (v rámci komponenty jde dojít z každého do každého).

Jak takový graf vytvoříme? Vytvoříme si vrcholů, kdy každý z nich repreznetuje jednu komponentu. (Budu jim říkat vrcholo-komponenty.) Poté pro každou hranu v originálním grafu G:

- Pokud vede v rámci jedné komponenty, tak ji ignorujeme. (Koncové vrcholy jsou ve stejných komponentách.)
- V opačném případě vytvoříme hranu mezi vrcholo-komponentami, které tato hrana spojuje. (Zachováváme původní orientaci hrany.)

Můžeme si rozmyslet, že pokud vede hrana mezi dvěma vrcholy, jde se z každého vrcholu v první komponentě dostat do počátečního vrcholu v hraně (jsou ve stejné komponentě), poté jde po hraně přejít, a nakonec dojít do jakéhokoliv vrcholu v druhé komponentě.

No a nyní, když máme graf vrcholo-komponent, můžeme z každé z nich spustit dfs a uložit si, do kterých vrcholo-komponent jsme se dostali. (Aby se v tom dobře hledalo, můžeme tohle ukládat třeba v tabulce boolů - jsme se schopni z komponenty v řádku dostat do komponenty ve sloupci?)

2.3 Odpovědění dotazů

Když načteme dotaz, jen se podíváme ve kterých komponentách jsou oba vrcholy, a poté se podíváme, jesli jde dojít z první komponenty do druhé. Podle toho odpovíme.

3 Časová složitost

(Tak mimohodem značím N počet vrcholů a M počet hran, protože síla zvyku.)

1. Najití silně souvislých komponent:
   - Výroba inverze grafu. (Při dobré reprezentaci tohle jsme schopni každou hranu jednou projít a obrátit ji.) $O(N + M)$
   - Dfs na G^T: Každý vrchol navštívíme jednou, po každé hraně přejdeme dvakrát. $O(N + M)$

- Dfs na G - znovu $O(N + M)$

Celkem $O(N + M)$.

2. Spočítaní dosažizelnosti mezi komponentami (K - počet komponent):

   - Výroba grafu vrcholo-komponent. $O(K+M)$ (Každou hranu musíme projít.)
   - Dfs na grafu: $O(K(K + K^2)) = O(K^3)$) (Pouštíme K dfs, každé z nich projde všechny vrcholy i hrany. Hran je nejvýše kvadraticky tolik, co vrcholů.)

   Celkem K^3.

3. Zodpovězení všech dotzů $O(O)$

Celkem $O(N + M + K^3 + O)$, ale vzhledem k tomu, že v zadání je slíbeno, že počet silně souvislých komponent je nejvýše 3, tak $O(N + M + O)$.

4 Časová složitost

Pamatujeme si různé seznamy - komponenty pro vrcholy, navštívenost vrcholů, invertovaný graf, ... Největší z toho je invertovaný graf ($O(N + M)$) a matice dosažitelnosti $O(K^2)$, takže celkem $O(N + M + K^2) = O(N + M)$.