Cvičenie 2

Príklad 2.1.

Máme úplný graf s n vrcholmi. O každej hrane vieme jej cenu, čo môže byť kladné aj záporné číslo. Chceme vybrať niekoľko hrán tak aby sme mali čo najnižšiu cenu a zároveň boli všetky vrcholy spojené pomocou vybratých hrán.

Náčrt riešenia 2.1.

Uvedomme si, že ak chceme dosiahnuť najnižšiu možnú cenu, určite sa nám oplatí vybrať všetky hrany so zápornou dĺžkou. To nám vytvorí nejaké komponenty, ktoré sú už navzájom prepojené a pospájame ich pomocou algoritmu na hľadanie najlacnejšej kostry.

Príklad 2.2.

Máme obdĺžnikovú mapu veľkosti $n \times m$. Na každom políčku je jedno kladné číslo. Táto mapa sa nám postupne zatápa. k minút od začiatku sú pod vodou všetky políčka, ktorých číslo je menšie alebo rovné k. Takisto máte zadané časy $t_1, t_2 \dots t_l$. Pre každý zadaný čas zistite koľko súvislých ostrovov sa nachádza nad hladinou (Ak je na ostrove jazero a na ňom je ďalší ostrov, rátame to ako dva rôzne ostrovy).

Náčrt riešenia 2.2.

Začíname s jedným súvislým komponentom, ktorý sa nám postupne rozpadáva na menšie časti a v každom momente musíme vedieť povedať, koľko ostrovov sa aktuálne na mape nachádza. Rozpadávanie sa však nie je operácia, ktorú by sme vedeli riešiť. To čo poznáme je spájanie komponentov, teda union find.

Otočíme si teda čas. Budeme predpokladať, že všetky políčka sú už zaplavené a postupne od najvyššieho sa budú vynárať spod vody. V tomto momente sa už dá použiť klasický union find. Vždy keď sa vynorí nové políčko, pozrieme sa, či ho máme pripojiť k nejakým iným komponentom, alebo je to samostatné z vody vytŕčajúce políčko, teda nový komponent.

Príklad 2.3.

Máme n vecí, o ktorých chceme zisťovať relatívnu váhu jednej voči druhej. Postupne nastane k udalostí:

- ! a b w – dozvedeli sme sa, že vec a je o w ťažšia ako vec b.
- ? a b – z toho, čo sme sa zatiaľ dozvedeli máme povedať, o koľko je vec a ťažšia ako vec b, poprípade usúdiť, že sa to určiť nedá

Navrhnite algoritmus, ktorý postupne spracuje všetky udalosti a správne odpovie na tie označené znakom "?".

Náčrt riešenia 2.3.

Prvým krokom je správna reprezentácia problému. Predstavme si celú situáciu ako graf. Veci tvoria vrcholy. Vždy keď sa dozviem relatívnu váhu dvoch vecí, pribudne mi medzi príslušné dva vrcholy hrana danej váhy.

Uvedomme si, že ak vieme, že predmet a je o 3 ťažší ako predmet b a ten je o 4 ťažší ako predmet c, tak si vieme vypočítať, že predmet a je o 7 ťažší ako predmet c. Vo všeobecnosti sa relatívna váha dvoch vecí dá vypočítať ako dĺžka cesty medzi vrcholmi, ktoré im prislúchajú.

Ako prvé sa však môžeme zamyslieť iba nad tým, či vieme danú odpoveď vypočítať alebo nie. Z predchádzajúceho pozorovania je jasné, že dve veci sú porovnateľné, ak v našom grafe ležia v tom istom komponente. A v podstate celé pridávanie hrán je úplne totožné s problémom union-findu.

Ak teda ideme použiť union-find, možno by sme sa mohli zamyslieť, či nevieme tento algoritmus upraviť tak, aby nám rovno vedel vypočítať aj relatívne váhy dvoch vrcholov v jednom komponente. To vôbec nebude také ťažké.

Zoberme si implementáciu pomocou stromov. To čo sa budeme snažiť počítať bude relatívna váha vrcholov komponentu, oproti koreňu, ktorý tento komponent reprezentuje. Vďaka tomu vieme porovnávať ľubovoľné dva vrcholy v tomto komponente. Nech x je koreň. Ak je a o w_a ťažší ako x a b o w_b ťažší ako x, tak a je od b ťažšie o $w_a - w_b$.

V našej implementácii si teda budeme pre každú hranu do otca pamätať, o koľko je syn ťažší ako otec. Ak potom chceme pre nejaký vrchol vypočítať o koľko je ťažší od koreňa, stačí sčítať všetky váhy hrán na ceste od tohto vrchola do koreňa. A to vieme spraviť veľmi jednoduchou úpravou funkcie `find()`.

Ostáva upraviť funkciu `union()`, ale to je v tomto momente ľahké. Keď pripájame jeden koreň pod druhý, vypočítame si, o koľko je ťažší a takúto váhu nastavíme novej hrane. Tu treba ešte doladiť nejaké detaily, lebo mi ako novú informáciu dostaneme relatívnu váhu dvoch náhodných vrcholov v týchto komponentoch, pomocou nich a operácie `find()` však vieme vypočítať aj potrebnú hodnotu. Skúste si to nakresliť a zamyslieť sa :)