Aktivnost 14

Piranski sladoledarji

Sladoledarji: ko ga najbolj potrebuješ, ga ni nikjer. Raziščimo in bomo videli, da je razporejanje sladoledarjev v resnici zelo težak problem. Računalnikarji vedo povedati celo, da gre praktično za najtežji problem, kar jih sploh je.

Namen

Učenci spoznajo še en problem iz teorije grafov.

Prvič vidijo primer problema, ki ga je zelo težko rešiti; ko poiščejo določeno rešitev, ne vedo, ali je optimalna ali ne.

Vidijo, da lahko sestavijo nalogo, ki jo sami znajo preprosto rešiti, za druge pa je težka. Na ta način spoznajo koncept enosmernih funkcij. Uporabili ga bomo kasneje, pri kriptografiji.

Trajanje

Ena ura

Potrebščine

Vsak otrok potrebuje

- polo z nalogo,
- žetone, figurice ali kaj podobnega za označevanje vozlišč.

Učitelj potrebuje

- prosojnico z rešitvijo in sestavljanjem naloge.

Piranski sladoledarji

Na listu je zemljevid Pirana (morda ni najbolj natančen, morda pa v resnici sploh ne gre za Piran ;). Črte pomenijo ulice in krogi križišča.

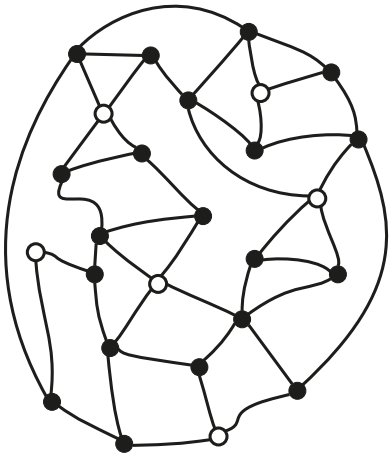
Poleti je v mestu veliko otrok, željnih sladoleda, zato bo potrebno po mestu razpostaviti sladoledarje. Radi bi jih imeli toliko, da bi bilo do najbližjega sladoleda vedno potrebno iti le do konca ulice in potem največ eno ulico naprej. Z drugimi besedami: če križišče nima sladoledarja, naj bo najbližji sladoledar vedno v vsaj enem sosednjem križišču.

Koliko sladoledarjev potrebujemo in kam jih je potrebno postaviti? Poskusi rešiti nalogo s čim manj sladoledarji.

1. Razdeli otroke v male skupine (lahko pa delajo tudi posamično), razdeli jim pole in oznake (žetone, gumbe, figurice...). Pojasni jim nalogo.

2. Pokaži otrokom, kako postavljajo oznake. Postavi oznako na eno od križišč in povej, katera križišča pokriva. Postavi še eno oznako dve križišči stran, razloži kaj si pokril in kaj je še nepokrito.

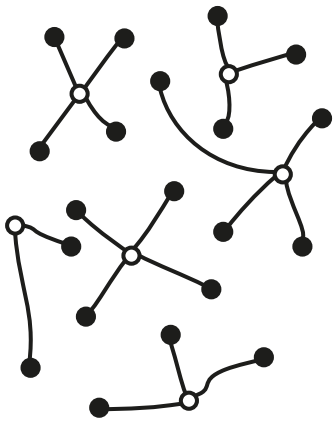
3. Otroci naj poskušajo različne razporede sladoledarjev. Ko bodo našli postavitve, ki pokrijejo vsa križišča (očitna rešitev je, da označiš vsa križišča...), jih opomni, da so sladoledarski vozički dragi in naj poskusijo zmanjšati njihovo število.



4. Čez nekaj časa lahko otrokom poveš, da je najmanjše potrebno število sladoledarjev za ta zemljevid je šest (rešitev je na desni) in jih izzovi, naj jih razporedijo. Izkaže se, da jo je še vedno težko najti in nekateri bodo obupali. Celo rešitev z osmimi ali devetimi sladoledarji je kar težko poiskati.

5. Pokaži otrokom, kako so sestavljalci sestavili nalogo.

    a. Narisali so nekaj križišč, v katerih bodo stali sladoledarji .
    b. Vsakemu križišču so dorisali nekaj križišč (pokaži sliko); očitno je, da za rešitev zadošča šest sladoledarjev.
    c. Nato so povezovali neizbrana, črna vozlišča, da so skrili pravo rešitev. Pazi, bela moraš pustiti pri miru in tudi novih ne smeš dodajati. Očitno je rešitev naloge – križišča, na katera je potrebno postaviti sladoledarje, enaka kot prej.



6. Naj si vsak otrok izmisli svoj zemljevid mesta. Nariše naj prazne kroge, doriše nove in jih pobarva, poveže

pobarvane kroge in nato pobarva še kroge, na katerih bodo morali stati sladoledarji. Nato naj si sošolci izmenjajo naloge.

7. Opozori jih, da jim je pravkar uspelo nekaj zanimivega: sestavili so naloge, katerih rešitev sami poznajo, sošolcem pa jih je težko najti. Povej, da se takšne reči pogosto uporabljajo za šifriranje besedil, ki ga bodo spoznavali čez nekaj tednov.

Pogovor

Spodnji pogovor je primeren za nekoliko starejše otroke. Za mlajše ga seveda ustrezno poenostavi.

S problemi, podobnimi temu, se pogosto srečujemo v vsakdanjem življenju: kako dobro razpostaviti poštne nabiralnike in bankomate, kje naj bodo avtobusne postaje in tako naprej. Resnični zemljevidi niso sestavljeni s trikom, zato rešitve ne poznamo vnaprej. Kako bi se lotili reševanja takšnega problema?

Na prvi pogled bi kdo rekel: pa poskusimo vse možne postavitve sladoledarjev in poglejmo, katere izmed pravilnih rešitev zahtevajo najmanj sladoledarjev. Pa poglejmo: ker imamo 26 križišč lahko postavimo prvega sladoledarja na 26 mest. Vendar nobena rešitev z enim sladoledarjem ni pravilna, torej bomo morali dodati še drugega. Tega damo na enega od preostalih 25 križišč, torej imamo že $26 \times 25 = 650$ različnih postavitev dveh sladoledarjev. No, ni tako hudo, vsako postavitev smo šteli dvakrat – enkrat smo dali prvega sladoledarja na križišče A in druge na križišče B, drugič pa obratno. V resnici je postavitev dveh sladoledarjev 325. Vendar tudi dva nista dovolj. Izkaže se, da je različnih postavitev treh sladoledarjev že 2600, štirih 14.950, petih 65.780, šestih pa že 230.230. Vseh postavitev z enim, dvem, tremi, štirimi, petimi ali šestimi sladoledarji skupaj je 313.912.

Vidimo, da številke hitro naraščajo. V mestu s stotimi križišči lahko dvajset sladoledarjev postavimo že na več kot 10^{20} načinov (to je 1 in dvajset ničel). Otrokom napiši to številko (z vsemi ničlami), da bodo videli, kako brezupna bi bila ta naloga.

Podobno hudih problemov, kot so sladoledarjevi, je še veliko. Postopkom, ki takšne, težke probleme rešujejo tako, da lepo po vrsti poskusijo vse rešitve, pravimo postopki po metodi *grobe sile*. Vidimo pa, da ti delujejo le, dokler so problemi dovolj majhni.

Recimo, da imamo računalnik, ki lahko v eni sekundi preskusi milijon razporeditev sladoledarjev (v resnici tako hitrih računalnikov sploh ni!). Če mesto s stotimi križišči potrebuje dvajset sladoledarjev, bi računalnik iskal pravilno razporeditev 16 bilijonov let, to je, 16 milijonov milijonov let. To je malo dlje časa, kot je staro vesolje!

Algoritmi z metodo grobe sile so očitno prepočasni. Kako se tedaj lotiti takšnih problemov? Se spomnimo blatnega mesta? Tam smo se naučili preprostega postopka: vedno smo asfaltirali najkrajšo pot. Če je bila nepotrebna, pa smo vzeli naslednjo najkrajšo in tako naprej, dokler niso bile povezane vse hiše. (Postopkom, ki delujejo na tak način, da v vsakem koraku pograbijo tisto, kar je v dani situaciji najboljše, pravimo *požrešni postopki*.) Ne bi bilo mogoče tu narediti kaj podobnega? Morda postaviti prvega sladoledarja na križišče, v katerem se stika največ ulic, drugega v naslednje križišče z

največ ulicami, razen če tam ne bi bilo potreben in tako naprej? Ne, to žal ne bi delovalo. Tudi v tem primeru ne: v najboljši rešitvi v križišče, v katerem se stika največ, pet ulic, sploh ni bilo potrebno postaviti sladoledarja!

Kako pa tedaj? Imamo kak drug postopek, ki nalogo reši v doglednem času. Odgovor je zanimiv: nimamo in ne vemo, ali ga samo še nismo odkrili, ali pa je v resnici nemogoče izumiti boljši postopek.

Za kaj gre?

Tako kot iskanje optimalnega barvanja grafov je tudi iskanje optimalnih položajev sladoledarjev (ali, kot se problemu reče v matematiki, problem dominantne množice vozlišč, *dominating set problem*) NP-poln problem.

Kaj so NP-polni problemi? Obstajajo problemih, ki jih je možno rešiti v linearnem času: če imamo dvakrat večji problem (recimo dvakrat več točk), za reševanje potrebujemo dvakrat toliko časa. Imamo probleme, ki zahtevajo kvadratni čas: kakorkoli zvito si izmislimo algoritem, bo za dvakrat večji problem potreboval vsaj štirikrat več časa, za, recimo, šestkrat večji problem pa šestintridesetkrat. Nekateri algoritmi zahtevajo kubični čas in za dvakrat večji problem potrebujejo osemkrat daljši čas. Lahko si predstavljamo algoritem, pri kateri bi bil čas izvajanja sorazmeren sedemnajsti potenci velikosti problema, n^{17}.

Za takšne algoritme pravimo, da zahtevajo polinomski čas. In to smo še pripravljeni požreti. Takšnim problemom pravimo, da spadajo v razred P, to je razred problemov, ki so rešljivi v polinomskem času. Problematični pa so algoritmi, ki zahtevajo eksponentni čas, npr. 2^n. Pri teh se čas reševanja podvoji z vsako dodatno točko.

Problem sladoledarjev je že takšen. Če ga rešujemo z grobo silo, se (z malo zaokrožanja) čas reševanja pomnoži z vsakim dodatnim križiščem oziroma sladoledarjem.

Skočimo (a ne pregloboko!) v teorijo. Recimo, da od nekod dobimo domnevno rešitev problema sladoledarjev. Ali je rešitev pravilna ali ne, lahko hitro preverimo: gremo po križiščih in za vsakega preverimo, ali ima svojega sladoledarja ali pa sladoledarja na koncu ene od ulic. Čas reševanje narašča linearno z velikostjo problema – dvakrat več križišč bi nam vzelo samo dvakrat več časa. Čas *preverjanja* rešitve je torej polinomski.

Alan Turing, eden začetnikov teorije izračunljivosti, ki se ukvarja s tem, kakšne probleme je mogoče rešiti in v kakšnem času, si je izmislil stroj, ki ga danes imenujemo nedeterministični Turingov stroj. Z malenkost (a ne veliko) poenostavljanja si lahko predstavljamo, da nedeterministični Turingov stroj najde rešitev vsakega problema v polinomskem času, vendar le, če znamo v polinomskem času preveriti, ali je rešitev pravilna. *Deterministični Turingovi stroji* znajo narediti vse, kar znajo običajni računalniki (in obratno, vendar le, če bi imeli računalniki neskončen pomnilnik). Nedeterminističnih pa, žal, ni in si jih je Turing kratkomalo izmislil. Nedeterministični Turingov stroj bi namreč zahteval "preroka" (angl. *oracle*), ki bi v nekaterih trenutkih stroju prišepnil, po kateri poti naj se izvaja program.

Problemom, ki jih je mogoče z nedeterminističnim Turingovim strojem rešiti v polinomskem času, pravimo NP problemi (nedeterministočno polinomski problemi). Med njimi so tudi vsi problemi, ki jih je mogoče rešiti v polinomskem času tudi z determinističnimi Turingovimi stroji ali, po domače, običajnimi računalniki. Za nekatere probleme pa ne poznamo nobenega postopka, ki bi zahteval le polinomski čas (spomnimo se še enkrat: takšnih, pri katerih čas reševanja narašča samo linearno ali sorazmerno s kvadratom, kubom, petnajsto, sedemnajsto ali katerokoliže potenco velikosti problema). Takšnim problemom pravimo NP-polni problemi (*NP-complete problems*).

Točneje, za NP-polne probleme velja tole: problem je NP-poln, če lahko vanj (v polinomskem času) prevedemo druge probleme. Oba težka problema, ki smo ju spoznali, barvanje zemljevidov in razpostavljanje sladoledarjev, sta NP-polna problema. To pomeni, da lahko problem sladoledarjev rešimo tako, da pobarvamo ustrezno sestavljen zemljevid; barve držav nam bodo povedale, kam postaviti sladoledarje. (Pretvarjanje med problemoma ni trivialno, zemljevid, ki ga dobimo, je morda ogromen in barve je morda potrebno brati na kak zapleten način, a pomembno je, da pretvorba obstaja in zahteva samo polinomski čas.) In obratno, barvanje zemljevidov lahko spremenimo v zemljevid mesta, v katerem iz postavitve sladoledarjev uganemo barve držav na prvotnem zemljevidu.

Če tole malo premislimo, smo prišli do nečesa jako zanimivega. Vsi NP-polni problemi se prevedejo en na drugega. Zveni imenitno. *Če* bi nekdo odkril postopek, ki v polinomskem času razporedi sladoledarje, bi taisti postopek lahko uporabili tudi za tisoče drugih NP-polnih problemov. Ali pa, če bi znal nekdo v polinomskem času barvati grafe. Ali pa, če bi kdo rešil katerega drugega – enega, enega samega izmed tisočih NP-polnih problemov v polinomskem času! Takoj bi postali v polinomskem času rešljivi vsi. Vsi problemi, ki so NP-polni, bi postali P-problemi. Množici NP in P bi bili enaki, NP = P.

In veste kaj? Še nihče ni našel nobenega takšnega algoritma. Za niti enega izmed tisočih problemov. Dobro, potem se to najbrž ne da? Saj to je tisto: ne vemo. Tudi tega ni uspel dokazati še nihče. To je sveti gral teoretičnega računalništva. Računalnikarji verjamemo, da velja

$$NP \neq P$$

Preprosto zato, ker se nam zdi, da smo dovolj pametni, da bi nekdo že našel kak hiter algoritem za kak NP-poln problem, ko bi ta le obstajal. A dokler ne dokažemo, da je res tako, ne moremo biti prepričani.