PRESEK
List za mlade matematike, fizike, astronome in računalnikarje

ISSN 0351-6652
Letnik **28** (2000/2001)
Številka 1
Strani 29–31

Tim Vidmar in Andrej Likar:

STAVEK SE PREDSTAVI

Ključne besede: računalništvo, zgradba besedila, štetje črk, simulirano kaljenje, programiranje.

Elektronska verzija:
http://www.presek.si/28/1430-Vidmar-Likar.pdf

STAVEK SE PREDSTAVI

Denimo, da beremo stavek, ki se glasi: "V tem stavku se pojavi a štiriintridesetkrat, b enkrat, c enkrat, č enkrat, d trinajstkrat, e sedemintridesetkrat, f enkrat, g enkrat, h enkrat, i dvajsetkrat, j sedemkrat, k sedemindvajsetkrat, l enkrat, m osemkrat, n devetnajstkrat, o petkrat, p štirikrat, r štiriintridesetkrat, s osemkrat, š petkrat, t devetinštiridesetkrat, u dvakrat, v osemkrat, z enkrat in ž enkrat."

Resnici na ljubo je to prav dolgočasen stavek. Je pa zanimiv, saj povsem drži. Vsaka črka se res pojavi tolikokrat, kot je v stavku zapisano. Poskusite zapisati kak drug stavek, ki bo imel podobno lastnost. Staviva, da ne bo prav preprosto.

Kako je uspelo nama? Najprej sva izbrala preprosto obliko stavka, kot jo vidimo zapisano zgoraj – kratek uvodni del in nato naštevanje, kolikokrat se pojavi posamezna črka. Nato sva uporabila računalnik, saj s poizkušanjem s svinčnikom in papirjem ne pridemo nikamor. Vendar tudi z računalnikom ne gre kar na slepo pregledovati različnih možnosti, saj jih je res zelo veliko. Poskusimo jih prešteti. Sistematično bi lahko iskali prave besede za črkami tako, da najprej za vse privzamemo, da se pojavijo "enkrat", potem pa začnemo za črko a spreminjati besedo na "dvakrat", "trikrat" in tako dalje. Vedno preverimo, ali smo uspeli sestaviti resnično izjavo, in če ne, nadaljujemo. Nato spremenimo besedo za črko b na "dvakrat" ter ponovimo tudi ves postopek s črko a. Tako nadaljujemo tudi z drugimi črkami. Vseh preizkusov bi bilo tako za vseh 25 črk 100^{25}, če privzamemo, da se nobena črka ne pojavi več kot stokrat. To je zelo veliko število. Če bi v eni sekundi preverili sto milijonov možnosti, bi vse pregledali šele v 3×10^{34} letih, kar je nepojmljivo dolg čas. Vedeti moramo, da obstaja vesolje "le" kakih 10^{10} let. Na pravo rešitev bi seveda naleteli po manjšem številu preizkusov, a še vedno prevelikem za še tako hiter superračunalnik. Število smiselnih možnosti lahko s premislekom krepko zmanjšamo. Hitro se da ugotoviti, da se bodo nekatere črke pojavile le enkrat. Ker se pri vsaki črki pojavi besedica "krat", začnemo pri črkah, ki jih ta besedica vsebuje, poskušanje z besedo "petindvajsetkrat", saj število pojavitev teh črk ne bo manjše od 25. S tem se število možnosti zelo zmanjša, a je še vedno preveliko, da bi stavek iskali na opisani način.

Naloge se lotimo drugače. Najprej potrebujemo mero, ki pove, kako dobro je stavek usklajen s samim seboj. Kar sam se ponuja kvadrat razlike med tem, kolikokrat se posamezna črka zares pojavi v stavku in kolikokrat je zapisano, da se pojavi. Kvadrate razlik seveda seštejemo po vseh petindvajsetih črkah. Če uspemo našo mero zmanjšati na nič, bo stavek samousklajen. Poskusimo s postopnim, a usmerjenim približevanjem

rešitvi: naključno izberemo eno izmed črk, preštejemo, kolikokrat se zares pojavlja v stavku, in to vanj tudi zapišemo. Seveda lahko s tem ustvarimo novo neskladje, vendar upamo, da bomo s ponavljanjem takih potez nazadnje uspeli. Potezo sprejmemo le, če se naša mera uspešnosti po njej zmanjša, sicer pa vzpostavimo stanje pred njo in izberemo novo potezo. Izkaže se, da postopek deluje tako, da se mera uspešnosti sprva hitro zmanjšuje, nato pa obstane pri vrednosti mere uspešnosti, ki je sicer majhna, a ni enaka nič – naš postopek se je ujel v zanko brez konca. Tedaj je smiselno vsaj začasno sprejeti tudi poteze, ki mere uspešnosti sicer ne zmanjšajo, a vsaj prekinejo vrtenje v krogu. Nato pa se znova spustimo v zmanjševanje razlik med resničnim in zapisanim stanjem.

V resnici se lotimo stvari tako, da vedno dovolimo tudi poteze, ki mero povečajo, vendar le z zelo majhno verjetnostjo, če je razlika med stanjem na papirju in v resnici velika. Bolj ko se ta manjša, bolj dopuščamo možnost, da za trenutek napravimo tudi korak v napačno smer, da se le izognemo stopicanju na mestu. Naš algoritem izgleda takole:

Preberi stavek v začetni obliki.
Ponavljaj:
    izberi eno izmed črk;
    preštej število pojavljanj te črke v stavku;
    zapiši število pojavljanj te črke v stavek;
    za vsako od petindvajsetih črk:
        preštej število pojavljanj te črke v stavku;
    izračunaj mero ujemanja med dejanskim in zapisanim stanjem;
    preveri, ali se je ujemanje izboljšalo:
        da – skoči na konec zanke (na pogoj "dokler");
        ne – izračunaj verjetnost, da spremembo vseeno odobrimo:
            sprememba odobrena – skoči na konec zanke;
            sprememba zavrnjena – vzpostavi stanje pred spremembo;
dokler ni popolnega ujemanja.
Zapiši stavek na izhodno enoto.

Za tiste, ki bi algoritem želeli zapisati tudi v kakem programskem jeziku, naj povemo, da korak, pri katerem se je mera uspešnosti spremenila za Δs, sprejmemo z verjetnostjo $e^{-\Delta s/T}$. Tu je T pozitivno število, ki je bilo v našem primeru enako 10, sicer pa pove nekaj o tem, kako zlahka dovolimo poteze, ki stanje začasno poslabšajo. Če je T zelo velik, so dovoljene prav vse poteze, če pa je zelo majhen, ni mogoča prav nobena poteza, ki stanja ne popravi. Vidimo tudi, da je vrednost zgornjega izraza

za vsako potezo, ki stanje izboljša, večja od 1, saj je tedaj Δs manjši od nič. To interpretiramo tako, da tako potezo vedno tudi sprejmemo.

Računalniški postopek, ki smo ga s tem opisali, je znan kot "simulirano kaljenje", saj spominja na postopek obdelave kovine, ki jo najprej segrejemo in nato ohladimo, da dosežemo kar najbolj stabilno in urejeno strukturo snovi. Nižanju temperature (naš parameter T) ustreza zmanjševanje verjetnosti za odobritev sprememb, ki ne vodijo v bolj urejeno stanje. V številnih primerih uporabe takega postopka je ključno prav pravilno postopno zmanjševanje te verjetnosti, sicer se lahko vseeno zgodi, da program ne najde najboljše rešitve, ampak konča v neskončni zanki. Korenine postopka so v termodinamiki, kjer sistem lahko začasno preide v stanje z višjo energijo, verjetnost za to pa je povezana z njegovo temperaturo. Po njegovem avtorju ga imenujemo tudi Metropolisov postopek. Njegov opis najdemo na primer v knjigi "Numerical Recipes", razdelek 10.9 (`http://www.nr.com/`).

Za konec preštejmo črke še enkrat. Če upoštevamo tudi naslov, imeni in priimka avtorjev ter opis algoritma, se v tem sestavku pojavlja a petstošestinštiridesetkrat, b petinšestdesetkrat, c dvajsetkrat, č dvainsedemdestkrat, d stoenainsedemdestkrat, e šeststopetinpetdesetkrat, f dvakrat, g sedemintridesetkrat, h petintridesetkrat, i štiristoštirikrat, j dvestodevetindvajsetkrat, k dvestotriinšestdesetkrat, l stotriinštiridesetkrat, m dvestotrikrat, n tristošestintridesetkrat, o štiristoenainpetdesetkrat, p dvestopetnajstkrat, r tristosedemindvajsetkrat, s tristoštiriintridesetkrat, š dvainosemdesetkrat, t štiristoenkrat, u petindevetdesetkrat, v dvestošestindvajsetkrat, z stošestnajstkrat, in ž trinajstkrat. Boste preverili?

Tim Vidmar, Andrej Likar

ŠTEVILSKA IZPOLNJEVANKA S SIMETRIJO – Rešitev iz XXVII, P–6, str. 322

7	×	8	:		2	+		2	=	3	0
8	:	8	×	1	5	+	1	4	=	2	9
9	+	8	−	1	0	+	2	2	=	2	9
6	×	5	×		2	−	1	1	=	4	9

3	0	+	2	9	+	2	9	+	4	9	=	1	3	7

Marija Vencelj