Optimalno iskanje racionalnih števil

Matej Cerar

May 22, 2024

1 Uvod

Na kratko predstavimo cilje. V prvem, drugem in tretjem poglavju podrobno razložimo algoritem za iskanje. V drugem poglavju si ogldamo iskanje za cela števila in podamo časovno zahtevnost takega algoritma. V tretjemm poglavju se osredotočimo na racinalni del. Podrobneje pogledamo podani algoritem, razložimo probleme pri implementaciji ter zopet podamo časovno zahtevnost. V tretjem poglavju opišemo splošno implementacijo ter spletno stran.

Želimo razložiiti in implementirati algoritem za optimalno iskanje racionalnih števil. Seveda je nesmislno iskati poljubno racionalno število, zato se omejimo na seznam oblike $\mathfrak{Q}_M = \{\frac{p}{q} \mid p, q \in \{1, \cdots, M\}\}$. Iskanje si lahko predstavljamo kot igro med dvema igralcema, prvi igralec, ki postavlja vprašanja in drugi igralec, ki na ta vprašanja odgovarja. Igra se začne tako, da prvi igralec izbere M (to omeji velikost $\mathfrak{Q}_M$) nato drugi igralec izbere x iz $\mathfrak{Q}_M$. Igra se nadaljuje tako da prvi igralec postavlja vprašanja oblike: ali je $x \geq y$? Na kar drugi igralec odgovarja le z ja ali ne. V spodnjem članku dokažejo, da prvi igralec lahko ugane x v največ $2\log_2(M) + O(1)$. Opmnimo še, da je M splošno znanje med tem, ko je celoten seznam $\mathfrak{Q}_M$ znan le drugemu igralcu.

2 Binarno iskanje celo-številskega dela

Vsako racionalno število lahko napišemo kot $x = \lfloor x \rfloor + \frac{a}{b}$, kjer sta a in b tuji števili ter $a < b$. Zanima nas le optimalno iskanje celega števila $\lfloor x \rfloor$. V našem primeru iščemo celo število iz seznama števil $1, \cdots, M$.
Uporabimo eksponentno iskanje, najprej primerjamo x z 2^k za $k = 0, 1, \cdots, \lceil \log_2(M) \rceil$, ko dobimo odgovor ja x je manjši kot 2^k, vrnemo interval $[2^{k-1}, 2^k]$. Na tem intervalu izvedemo binarno iskanje. Najslabši potrebovan čas za izvedbo teh korakov je $2k + O(1) = 2\log_2(\lfloor x \rfloor) + O(1)$.

3 Iskanje racionalnega dela

Z binarnim iskanjem smo našli interval $[\lfloor x \rfloor, \lfloor x \rfloor + 1]$. Preostane nam še poiskati ulomek $\frac{a}{b}$. Definirajmo še $\mathcal{M} = \lfloor M/\lfloor x \rfloor \rfloor$. Dokažimo naslednjo lemo.

Lema 3.1. *Za* $\frac{a}{b} \in \mathfrak{Q}_M$ *velja* $0 \leq a < b < \mathcal{M}$

Proof. Naj bo $x = \alpha/b$, kjer $\alpha = \lfloor x \rfloor b + a$. Potem velja $\alpha/b \geq \lfloor x \rfloor$ in zato $b \leq \alpha/\lfloor x \rfloor \leq M/\lfloor x \rfloor$. □

Z zgornjo lemo smo omejili a in b. Na intervalu $[\lfloor x \rfloor, \lfloor x \rfloor + 1]$ lahko izvajamo binarno iskanje: Problem nastane ob vprašanju kdaj se ustaviti. Ustaviti se želimo, ko bo interval tako majhen, da bo znotraj intervala zagotovo le ena številka iz $\mathfrak{Q}_M$. Izkaže se, da je dovolj da dobimo interval oblike $\left[\frac{\mu}{2\mathcal{M}^2}, \frac{\mu+1}{2\mathcal{M}^2}\right]$, za nek $\mu \in \{1, \cdots, 2\mathcal{M} - 1\}$. Naslednja lema nam pove, da je tak interval res dovolj majhen.

Lema 3.2. *Naj bosta* $\frac{a}{b}, \frac{c}{d} \in \mathfrak{Q}_M$*, taka da velja* $\frac{a}{b}, \frac{c}{d} \in \left[\frac{\mu}{2\mathcal{M}}, \frac{\mu+1}{2\mathcal{M}}\right]$*. Potem velja* $\frac{a}{b} = \frac{c}{d}$*.*

Proof. Dovolj pogledati kakšna je najmanjša možna razlika med $\frac{a}{b}, \frac{c}{d}$.

$$\left|\frac{a}{b} - \frac{c}{d}\right| = \left|\frac{ac - bd}{bd}\right| \geq \frac{1}{\mathcal{M}^2}$$

. Zgornja ocena sledi iz naslednjih opazk; največjo vrednost, ki jo imenovalca b in d zavzameta je $\mathcal{M}$, zato imenovalec omejimo s $\mathcal{M}^2$. V številcu pa seveda najmanše možne naravno število. □

Ni težko razmisliti, da je časovna zahtevnost za binarno iskanje največ $\lceil \log_2(2\mathcal{M}^2) \rceil = 2 \log M - 2 \log \lceil x \rceil + O(1)$.
Zdi se, da smo že skoraj končali. Prvi igralec je našel $\lfloor x \rfloor$ ter zožal racionalni del na interval $\left[\frac{\mu}{2\mathcal{M}}, \frac{\mu+1}{2\mathcal{M}}\right]$, za katerega ve, da vsebuje natanko eno številko iz $\mathfrak{Q}_M$. Težava je, da prvi igralec ne pozna seznama $\mathfrak{Q}_M$, zato izvede naslednji algoritem:

Algorithm 1 najdi-racionalni-del

Require: $\alpha, \beta, \gamma, \delta$
Ensure: a, b
1: **if** $\left\lfloor \frac{\beta}{\gamma} \right\rfloor = \left\lfloor \frac{\alpha}{\delta} \right\rfloor$ in $\frac{\beta}{\gamma} \notin \mathbb{Z}$ **then**
2: $(b, a') \leftarrow$ najdi-racionalni-del$(\delta, \gamma \mod \delta, \beta, \alpha \mod \beta)$ (Primer 1)
3: $a = \left\lfloor \frac{\beta}{\gamma} \right\rfloor b + a'$ (Enačba (1))
4: **return** a, b
5: **else**
6: **return** $a = \left\lfloor \frac{\alpha}{\beta} \right\rfloor, b = 1$ (Primer 2)
7: **end if**

Razložimo zgornji algoritem. Naj bo $I = \left[\frac{\mu}{2\mathcal{M}}, \frac{\mu+1}{2\mathcal{M}}\right]$, vemo, da znotraj intervala I leži natanko eno število iz $\mathfrak{Q}_M$. Torej vsi ulomki znotraj I, ki so različni od iskanega a/b, morajo imeti imenovalec večji od b. To je bistveno, saj nam pove, da je dovolj poiskati ulomek, ki ima najmanši imenovalec v I. V naslednji lemi ozirmo dokazu le te, razložimo zakaj naš algoritem najdi-racionalni-del deluje.

Lema 3.3. *Za dan interval* $I = \left[\frac{\alpha}{\beta}, \frac{\gamma}{\delta}\right]$ *obstaja ulomek* $a_{\min}(I)/b_{\min}(I) \in I$, *da velja: za vse* $\frac{a}{b} \in I$, $a_{\min}(I) < a$, $b_{\min}(I) < b$.

Proof. Želimo pokazati da obstajata $a_{min}(I)$ in $b_{min}(I)$, to naredimo z uporabo zgoraj opisnaega rekurzivnega algoritma. Ločimo dva primera.
Primer 1: Predpostavimo, da I vsebuje celo število, recimo, da I vsebuje cela števila $z_1 < \cdots < z_k$. Trdimo, da za vsak $\frac{a}{b} \in I$ velja $a \geq z_1$. To je očitno res, če je $z_1 = 1$ ali $b = 1$ ali $\frac{a}{b} > z_1$. Predpostavimo da velja, $z_1 - 1 < \frac{a}{b} < z_1$ in $b \neq 1, z_1 \neq 1$. Potem velja $a > b(z_1 - 1)$, kar pomeni $a \geq z_1$. Zato imamo $a_{\min}(I) = z_1$ in $b_{\min}(I) = 1$.
Primer 2: Zdaj predpostavimo, da I ne vsebuje celega števila, Naj bo $\frac{a}{b}$ poljuben ulomek v I. V tem primeru velja

$$\frac{\alpha}{\beta} \leq \frac{a}{b} \leq \frac{\alpha}{\beta} \quad \text{in} \quad \lfloor \frac{\alpha}{\beta} \rfloor = \left\lfloor \frac{a}{b} \right\rfloor = \left\lfloor \frac{\gamma}{\delta} \right\rfloor .$$

Torej a lahko rzpišemo kot

$$a = \left\lfloor \frac{a}{b} \right\rfloor b + a' = \left\lfloor \frac{\gamma}{\delta} \right\rfloor b + a',$$

kjer je $a' = a \mod b$. Naj bo $\alpha' = \alpha \mod \beta$ in $\gamma' = \gamma \mod \delta$. Potem velja tudi

$$\frac{\alpha'}{\beta'} \leq \frac{a'}{b} \leq \frac{\gamma'}{\delta} \text{ in zato } \frac{\delta}{\gamma'} \leq \frac{b}{a'} \leq \frac{\beta}{\alpha'}$$

.

Potem velja $\frac{b}{a'} \in I'$, kjer $I' = \left[\frac{\delta}{\gamma'}, \frac{\beta}{\alpha}\right]$. Pomembno je, da če obstajata $\hat{b}, \hat{a} \in I'$, taka da za vse $\frac{b}{a'} \in I'$, velja $b > \hat{b}$ in $a > \hat{a}$, potem nam po zamenjavi $\hat{b}$ za b in $\hat{a}$ za a' Enačba(1), da najmanjši a med vsemi možnim b-ji in a', tako da $\frac{b}{a'} \in I$.

Torej, da dokažemo obstoj $a_{\min}(I)$ in $b_{\min}(I)$, zadostuje dokazati obstoj $a_{\min}(I')$ in $b_{\min}(I')$. Oziroma, da določimo $a_{\min}(I')/b_{\min}(I')$, zadostuje rešiti problem v intervalu I'.

Če I' vsebuje celo število, se problem prevede na Primer 1. Zato predpostavimo, da I' ne vsebuje celega števila. Seveda velja, da $\gamma' \leq \gamma'$ in $\alpha' \leq \alpha$. Predpostavimo, da $\gamma' = \gamma'$ in $\alpha' = \alpha$, potem lahko ponovimo zgornji argument in "popravimo" imenovalce tako, da vzamemo imenovalec po modulu števca.

$$\frac{\alpha'}{\beta'} \leq \frac{a'}{b'} < \frac{\gamma'}{\delta'},$$

kjer je $b' = b \mod a'$, $\beta' = \beta \mod \alpha'$ in $\delta' = \delta \mod \gamma'$. Torej, problem se prevede na iskanje $a_{\min}(I'')$ in $b_{\min}(I'')$, kjer je $I'' = \left[\frac{\alpha'}{\beta'}, \frac{\gamma'}{\delta'}\right]$. Vemo tudi $\gamma = \gamma' = \gamma \mod \delta$ velja tudi $\gamma' < \delta$, kar pomeni $\delta' = (\delta \mod \gamma') < \delta$. Podobno $\gamma' < \gamma$.

Povzemimo kaj smo naredili, problem smo reurzivno prevedli na iskanje v novih intervalih. Za intervale vemo da se vsaj ena številka od imenovalcev in števcev zmanjša. Torej po dovolj iteracijah mora interval vsebovati celo število,

nato lahko uporabimo primer 1. V najslabšem primeru bomo dobili interval oblike $\left[\frac{1}{1}, \frac{1}{1}\right]$.

□

Bralec z ostrim očesom lahko opazi, da je algoritem opisan v primeru 2, zgornjega dokaza zelo podoben evklidovemu algoritmu za za iskanje največjega skupnega delitelja dveh števil. Zato ima tudi časovno zahtevnost enako le temu. Časovna zahtevnost je torej enaka $O(\log_2(\max(\alpha, \beta, \gamma, \delta)))$.

4 Implementacija

Kot v dokazu je tudi koda zgrajena najprej za binarno iskanje, nato pa za iskanje racionalnega dela. Algoritem sem implementiral kot "igro", kjer si mi izberemo M, nato nam računalnik poda seznam oblike $\mathfrak{Q}_M$. Mi si iz podanega seznam izberemo poljubono številko x, ki jo podamo računalniku. Računalnik nam vrne optimalno igro za te podatke, Kar pomeni najprej nam vrne intervale ki jih je zmanjševal za iskanje $\lfloor x \rfloor$.

Na primer v seznamu celih števil od ena do deset, iščemo celo številko 2. Program nam bo vrnil v oblik (število potebnih iteracij, seznami), za zgornji primer dobimo

$$(2, [[1, 2, 3, 4, 5], [1, 2, 3], [2]]).$$

To bo prvi korak, naslednji bo iskanje racionalnega dela. Najprej želimo najti interval velikosti $\left[\frac{\mu}{2\mathcal{M}}, \frac{\mu+1}{2\mathcal{M}}\right]$. Tudi tukaj nam računalnik vrne v obliki (število potrebnih iteracij, željeni interval zgornje oblike). Zopet primer na primeru kjer $M = 10$ si izberimo število $0,125 = 1/8$, računalnik nam bo vrnil

$$(2, (Fraction(31, 256), Fraction(1, 8))).$$

Opmnimo, da implementiramo v Pythonu in uporabljamo knjižnico Fractions.

Zadnj korak, ki nam preostane je najti število x, to naredimo z implementacijo algoritma opisanega zgoraj najdi-racionalni-del. Tudi tukaj izpišemo število potrebnih iteracij ter vse intervale oblike $\left[\frac{\alpha}{\beta}, \frac{\gamma}{\delta}\right]$ in seveda najdeno številko x.

Na koncu računalnik sešteje vse potrebne iteracije, ki so bile potrebne za vsakega od zgornjih korakov.

Da bo uporabniku prijetna izkušnja sem s pomočjo knjižnice Flask, naredil spletno stran. Na tej spletni strani je uporabnik prvi igalec, ki najprej vnese poljuben M in pozneje x, računalnik vrne vse zgoraj navedeno.