MATEMATIČKI FAKULTET BEOGRAD

Master rad

Kolacov problem

Uros
Student: Drakulić Uroš 1116/2010

Beograd, Septembar 2015.

Sadržaj

1 UVOD 3

2 FORMULACIJA HIPOTEZE I BITNIH TERMINA 4
2.1 Grafovi 4
2.2 Formulacija hipoteze i osnovni pojmovi 4
2.3 Istorijat, težina i bitna pitanja u vezi sa Kolacovim problemom 10

3 PREDSTAVLJANJE PROBLEMA 12
3.1 Statistički model niza Tix za fiksirano x 12
3.2 Predstavljanje pomoću usmerenih grafova 14
3.3 Predstavljanje preko vektora parnosti i Tera-ova teorema 17
3.4 Predstavljanje preko stabla prethodnika 22
3.5 Predstavljanje Kolacovih putanja preko binarnog stabla prethodnika 27
3.5.1 Svođenje na dijagram stanja 28

4 ALGORITMI ZA EFIKASNU PROVERU KOLACOVE HIPOTEZE 30
4.1 Uvodna zapažanja i pojam rekordera 30
4.2 Algoritam pretrage i ubrzanje 33
4.2.1 A Priori prekidanje pretrage 33
4.2.2 A posteriori prekidanje pretrage 35
4.2.3 Ubrzavanje iteracija 37
4.3 Sužavanje prostora pretrage za maksimumima u odnosu na σ i $\sigma\infty$ 39
4.3.1 Postupak (strategija) pretrage 41

5 TEHNIČKA SPECIFIKACIJA SOFTVERSKOG REŠENJA 44
5.1 Razvojno okruženje i objektni model 44
5.2 Pregled bitnih klasa i metoda 46
5.3 Implementirane funkcionalnosti i moguća unapređenja 48
5.4 Kratko uputsvo za korišćenje aplikacije 49

DODATAK A 52
A.1 Otvoreni i zatvoreni čvorovi 52
A.2. Veza između otvorenih i zatvorenih čvorova i rekurentne formule 52

DODATAK B 54
B.1. Rezultati pretrage 54

LITERATURA 59

1 Uvod

Kolacov problem ili problem 3x+1 razmatra funkciju $f: \mathbb{Z}^+ \to \mathbb{Z}^+$, koja za proizvoljno izabrano n vraća $\frac{n}{2}$ ukoliko je n parno, a $3n + 1$ ukoliko je n neparno. Pod ovakvom pretpostavkom problem 3x+1 glasi:

Da li za svaki prirodan broj n postoji prirodan broj k, takav da je $f^k(n) = 1$? Kolacov (*Lothar Collatz* (1910-1990), nemački matematičar) problem kaže da je odgovor na ovo pitanje potvrdan.

U ovom radu je izložen pregled teorijskih i eksperimentalnih rezultata, kao i programska realizacija jednog od eksperimenata vezanih za problem 3x+1. U poglavlju 2 predstavljena je formulacija problema i termina bitnih za dalji rad. U poglavlju 3 je opisano nekoliko načina za predstavljanje problema 3x+1. Poglavlje 4 govori o algoritmima i heuristikama za efikasniju proveru tačnosti hipoteze za neki početni interval skupa prirodnih brojeva. Opis pratećeg softverskog rešenja sa uputsvom za upotrebu je dat u poglavlju 5, i na kraju kao dodatak, pregled rezultata rada programa uz dodatne teorijske napomene.

2 Formulacija hipoteze i bitnih termina

Kolacov problem polazi od kratke definicije i, na prvi pogled, krajnje jednostavne hipoteze. Iza toga se nalazi problem nerešiv do danas, iako širok naučni krug predlaže više pristupa rešavanju i predloga, kako bi se moglo rešiti. Ipak, nepredvidljivost odnosa parnih i neparnih koraka čine ovaj problem "nepristupačnim" za današnje metode i saznanja.

2.1 Grafovi

Neka je V konačan skup čvorova i obeležimo sa

$$E(V) = \{\{u, v\} \mid u, v \in V, u \neq v\} \tag{2.1}$$

skup podskupova veličine 2 sa različitim čvorovima.

Definicija 2.1 Par $G = (V, E)$ gde je $E \subseteq E(V)$ se naziva (neusmereni) *graf* nad V. Elementi iz V se nazivaju čvorovi grafa, a elementi iz E grane grafa.

Definicija 2.2 *Digraf* $G = (V, E)$ ili usmereni graf je uređeni par gde je V konačan skup čvorova i $E \subseteq E(V)$, sa tom razlikom da je $E(V)$ definisano kao

$$E(V) = \{\{u, v\} | u, v \in V, u \neq v\} \tag{2.2}$$

Definicija 2.3 *Put dužine m* između čvorova v_0 i v_m je niz čvorova $[v_0, v_1, v_2, \dots, v_m]$ u grafu G ako važi

$$(v_{i-1,} v_i) \in E \text{ , za } i = 1,2, \dots m$$

Definicija 2.4 *Ciklus* je put čiji se prvi i poslednji čvor poklapaju. Ciklus je prost ako se, osim prvog i poslednjeg čvora, niti jedan drugi čvor ne pojavljuje na putu dvaput.

Definicija 2.5 *Povezani graf* je neusmereni graf u kojem između bilo koja dva čvora postoji put koji ih spaja. S druge strane, za nepovezani graf postoji bar jedan par čvorova između kojih ne postoji put, tj. postoji više odvojenih povezanih delova koji se nazivaju *komponente povezanosti*.

Definicija 2.6 Graf je slabo povezan ako je povezan neusmeren graf, dobijen od datog grafa zamenom usmerenih grana neusmerenim.

2.2 Formulacija hipoteze i osnovni pojmovi

Definicija 2.7 (Kolacova funkcija): Funkcija $C(x)$ definisana na sledeći način:

$$C(x) = \begin{cases} 3x + 1 & \text{ako je } x \equiv 1(mod 2) \\ \dfrac{x}{2} & \text{ako je } x \equiv 0(mod 2) \end{cases} \tag{2.3}$$

se naziva Kolacova funkcija.

Kolacov problem (po nemačkom matematičaru Lotaru Kolacu), se odnosi na ponašanje niza $C(n), C^{(2)}(n) = C\big(C(n)\big), C^{(3)}(n), \ldots$. Drugi nazivi za ovaj problem su problem *3x+1, 3x+1 hipoteza*, *Ulamova*[1] *hipoteza* [1] *Kakutanijev*[2] *problem* i *Haseov*[3] *algoritam* ili *Sirakuza problem* [2].

Hipoteza 2.1 (hipoteza 3x+1): Za proizvoljno izabrano $\boldsymbol{n}$, postoji $\boldsymbol{k}$, takvo da je $\boldsymbol{C^{(k)}(n) = 1}$, posle čega se niz $\boldsymbol{C^{(k+1)}(n), C^{(k+2)}(n)}$ ... periodično ponavlja 4, 2, 1, 4, 2, 1...

Dati ciklus se naziva trivijalni ciklus Kolacove funkcije.

Primer 2.1: Višestrukom primenom funkcije $C(x)$ na broj 39 dobija se sledeći niz: 39, 118, 59, 178, 89, 268, 134, 67, 202, 101, 304, 152, 76, 38, 19, 58, 29, 88, 44, 22, 11, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1, 4, 2, 1.... $C^k(39) = 1$ prvi put za $k = 34$ i $C^k(39) < 39$ prvi put za $k = 13$.

Definicija 2.8 (putanja 3x+1): Niz brojeva $\big(n, C(n), C^{(2)}(n),\ C^{(3)}(n), \ldots.\big)$ predstavlja putanju $3x + 1$ (skraćeno putanju) za neko n. Za putanju se kaže da je:

i. *konvergentna*, ako za neko k važi $C^{(k)}(n) = 1$.
ii. *netrivijalna i ciklična*, ako je niz $C^{(k)}(n)$ periodičan i $C^{(k)}(n) \neq 1$, za svako $k \geq 1$.
iii. *divergentna*, ako $\lim\limits_{k\to\infty} C^{(k)}(n) = \infty$.

Hipoteza 2.2 (hipoteza 3x+1 - druga forma): Svako $n \geq 0$ ima konvergentnu putanju.

Definicija 2.9 (vreme zaustavljanja za funkciju $\boldsymbol{C(x)}$) Najmanje k za koje je ispunjeno $C^{(k)}(x) < x$, se naziva *vreme zaustavljanja* (eng. *glide*) za početno x, u oznaci $G(x)$. Ukoliko takvo k ne postoji, tada je $G(x) = \infty$.

Definicija 2.10 (potpuno vreme zaustavljanja za funkciju $\boldsymbol{C(x)}$) Za početno x, najmanje k za koje je ispunjeno $C^{(k)}(x) = 1$, se naziva *potpuno vreme zaustavljanja*(eng. *delay*), u oznaci $D(x)$. Ukoliko takvo k ne postoji, tada je $D(x) = \infty$.

Definicija 2.11 (parni i neparni koraci): Neka je x broj za koji je hipoteza 2.1 tačna. U nizu $x, C(x), C^2(x), \ldots., C^k(x) = 1, \ldots.(C^m(x)! = 1$, za $m < k)$, svaki prelaz na naredni član niza se naziva korak. U zavisnosti od tipa prelaza, razlikuju se parni i neparni koraci, za početno x. Ukupan broj parnih koraka do prve jedinice se obeležava sa $E(x)$, a ukupan broj neparnih koraka sa $O(x)$. Pored toga, važi $D(x) = E(x) + O(x)$.

[1] Stanislaw Ulam(1909-1984), poljski matematičar
[2] Shizuo Kakutani(1911-2004), japanski matematičar
[3] Helmut Hasse(1898-1979), nemački matematičar

Primer 2.2: Za $x = 39$ je $E(x) = 23$, $O(x) = 11$ i $D(x) = 34$.

Definicija 2.12 (minimum i maksimum putanje): Za početno x, maksimum i minimum putanje 3x+1 se definiše na sledeći način:

$$Mx(n) = \lim_{k\to\infty} Max\left(n, C(n), C^{(2)}(n),\ C^{(3)}(n), \dots.,\ C^{(k)}(n)\right)$$

$$Mn(n) = \lim_{k\to\infty} Min\left(n, C(n), C^{(2)}(n),\ C^{(3)}(n), \dots., C^{(k)}(n)\right)$$

Na osnovu toga, za svako n važi da je putanja:

i. *konvergentna*, ako je $Mn(n) = 1$
ii. *divergentna*, ako $Mx(n)$ ne postoji i
iii. *ciklična*, u ostalim slučajevima.

Ako se na putanji nađe već postojeća vrednost, putanja postaje ciklična pa nije divergentna. Za putanju koja nema ni maksimum ni minimum važi da ne dostiže vrednost koja već postoji na putanji. Zasad nije poznato n sa cikličnom putanjom (osim 1, 4, 2…). Odatle sledi:

Hipoteza 2.3 (hipoteza 3x+1 - treća forma): Ne postoji n, $n \geq 0$, sa divergentnom putanjom.

Definicija 2.13 (funkcija 3x+1): Funkcija $T(x)$ (*funkcija 3x+1*) definiše se izrazom:

$$T(x) = \begin{cases} \dfrac{3x+1}{2} & \text{ako je } x \equiv 1(mod2) \\ \dfrac{x}{2} & \text{ako je } x \equiv 0(mod2) \end{cases} \qquad 2.4$$

Lako se uočava veza funkcija $C(x)$ i $T(x)$:

$$T(x) = \begin{cases} C(C(x)) & \text{ako je } x \equiv 1(mod2) \\ C(x) & \text{ako je } x \equiv 0(mod2) \end{cases}$$ [4]

Primer 2.3: Primenom funkcije $T(x)$ na 39 dobija se niz 39, 59, 89, 134, 67, 101, 152, 76, 38, 19, 29, 44, 22, 11, 17, 26, 13, 20, 10, 5, 8, 4, 2, 1... Indeks prve jedinice u ovom nizu je manji odnosu na niz dobijen primenom funkcije $C(x)$.

Da bi za neko n, $n > 1$, važilo $T^{(i)}(n) = 1$, pri čemu je $i > 1$, mora da postoji neko $k < i$ takvo da je $T^{(k)}(n) < n$. U slučaju da su poznate putanje za sve brojeve manje od n, nakon k-te iteracije direktno sledi da hipoteza 3x + 1 važi i za broj n.

[4] Funkciju T(x) su, nezavisno jedan od drugog, uveli R. Terras i C.J.Everett [5]

Definicija 2.14 (vreme zaustavljanja za funkciju $\boldsymbol{T(x)}$) Najmanje k za koje je ispunjeno $T^{(k)}(x) < x$, se naziva *vreme zaustavljanja* (eng. *stopping time*) za početno x, u oznaci $\sigma(x)$. Ukoliko takvo k ne postoji, tada je $\sigma(x) = \infty$.

Definicija 2.15 (potpuno vreme zaustavljanja za funkciju $\boldsymbol{T(x)}$) Za početno x, najmanje k za koje je ispunjeno $T^{(k)}(x) = 1$, se naziva *potpuno vreme zaustavljanja* (eng. *total stopping time*), u oznaci $\sigma_\infty(x)$. Ukoliko takvo k ne postoji, tada je $\sigma_\infty(x) = \infty$.

Primer 2.4: Za 39, $\sigma(39) = 8$ i $\sigma_\infty(39) = 23$.

Zasad nije poznat broj koji nema konačno vreme zaustavljanja ili potpuno vreme zaustavljanja. Postojanje konačnog vremena zaustavljanja, za neko n, dokazuje i postojanje konvergentne putanje, sve dok se zna da svi brojevi manji od n imaju konvergetne putanje. Odatle:

Hipoteza 2.4 (problem 3x+1 - četvrta forma): Svako $n \geq 2$ ima konačno vreme zaustavljanja.

Potpuno vreme zaustavljanja se može predstaviti kao zbir parnih i neparnih koraka na putanji $\left(\sigma_\infty(x) = O(x) + E(x)\right)$. Pored vremena zaustavljanja, od značaja je informacija o brojevima koji imaju vreme zaustavljanja veće nego svi brojevi manji od njega.

Definicija 2.16 (rekorder): Rekorder (eng. *peak*) je svako x koje ima vrednost nekog svojstva veće u odnosu na svako y, $y < x$.

Definicija 2.17: Neka je $\sigma(x) = k$. Ako za svako $y < x$ važi $\sigma(y) < k$, tada x ima *rekordno vreme zaustavljanja* i kaže se da je x *rekorder u odnosu na vreme zaustavljanja* (eng. *stopping time peak*).

Primer 2.5: Ne računajući 1 i 2, nekoliko prvih rekordera i za funkciju $C(x)$ i $T(x)$ u odnosu na vreme zaustavljanja su:

x	$\sigma(x)$	$G(x)$
3	4	6
7	7	11
27	59	96
703	81	132
10 087	105	171

Lista svih rekordera u odnosu na vreme zaustavljanja do $5 \cdot 10^{11}$ se nalazi u dodatku B. Lista svih dosada poznatih rekordera se može naći na [3] i [4]. U tabeli se vidi da, sa porastom ulaznog x, rekorderi u odnosu na vreme zaustavljanja se sve ređe pojavljuju. Rekorderi su isti za obe funkcije, i to važi za dosada poznate rekordere.

Definicija 2.18: Neka je $\sigma_\infty(x) = k$. Ako za svako $y < x$ važi $\sigma_\infty(y) < k$, tada x ima *rekordno potpuno vreme zaustavljanja* i kažemo da je x *rekorder u odnosu na potpuno vreme zaustavljanja* (eng. *total stopping time peak*).

Primer 2.6: Ne računajući 1, prvih nekoliko rekordera i za funkciju $C(x)$ i $T(x)$ u odnosu na potpuno vreme zaustavljanja su:

x	$\sigma_\infty(x)$	$D(x)$
2	1	1
3	5	7
6	6	8
7	11	16
9	13	19
18	14	20

Lista svih rekordera u odnosu na potpuno vreme zaustavljanja do $5 \cdot 10^{10}$se nalazi u dodatku B. Lista svih dosada poznatih rekordera se može naći na [3] i [4]. Kao u prethodnom primeru, i ovde su rekorderi u odnosu na potpuno vreme zaustavljanja isti za obe funkcije, i to važi za dosada poznate rekordere.

Definicija 2.19 (klase u odnosu na potpuno vreme zaustavljanja): k-ta klasa R_k u odnosu na potpuno vreme zaustavljanja, je skup brojeva koji imaju potpuno vreme zaustavljanja k, tj.

$$R_k = \{x \in \mathbb{Z}^+, \sigma_\infty(x) = k\}, r_k = card(R_k).$$

Veruje se da je $r_k < \infty$ za svako k. Svaka klasa se razlaže na disjunktne podklase, u oznaci $R_{k,i}$ gde i označava broj neparnih koraka prilikom iteracija:

$$R_{k,i} = \{x \in R_k \ \wedge O(x) = i\},\ r_{k,i} = card(R_{k,i}) \text{ i}$$

$$R_k = \bigcup_{i\in\mathbb{N}} R_{k,i}\ , r_k = \sum_{i\in\mathbb{N}} r_{k,i}$$

Definicija 2.20 (rekord klase): Rekord k-te klasa R_k je najmanji element u toj klasi. Lista dosad poznatih rekorda klase je dostupna na [4].

Ukoliko bi hipoteza 3x+1 bila tačna, tada bi važilo $\mathbb{N} = \bigcup_{k=1}^{\infty} R_k$.

Definicija 2.21 (proširenje): *Proširenje*[5] broja x, u oznaci $s(x)$, se definiše izrazom:

$$s(x) = \frac{Mx(x)}{\ln x}\ ^{6} \qquad 2.5$$

[5] eng. expansion

[6] prirodni logaritam za neko x sa 3x+1 putanje se naziva i *visina*. Takve vrednosti za x se prikazuju na logaritamskoj skali.

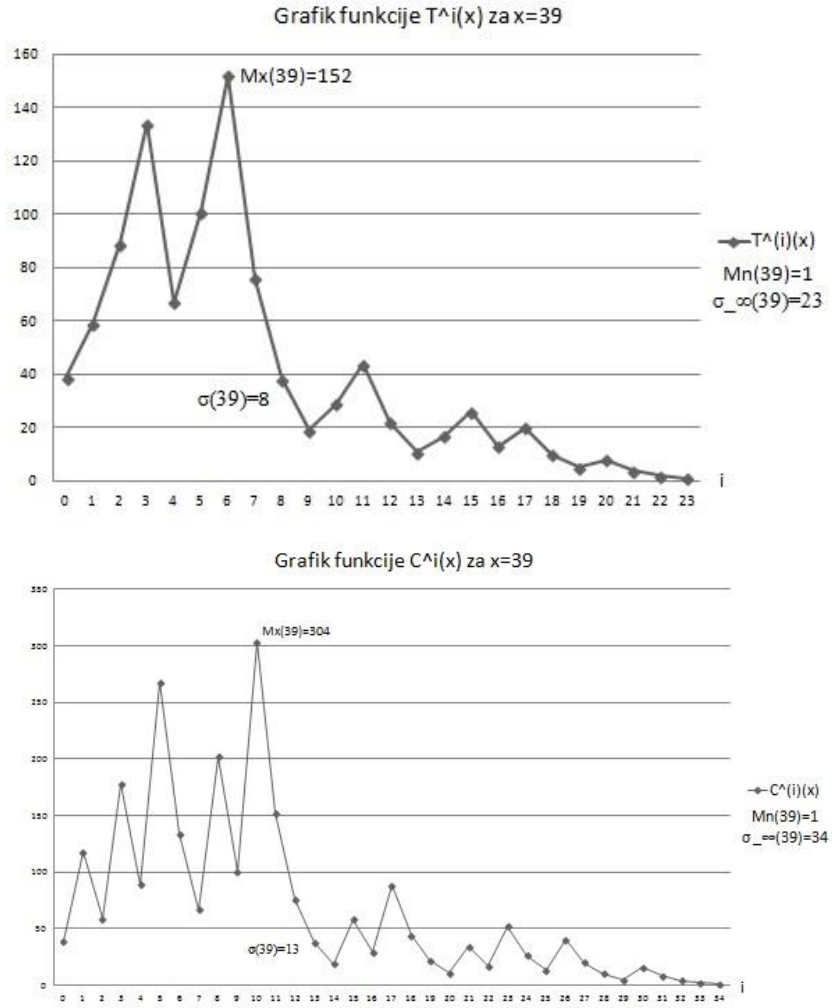
Definicija 2.22 (razgranatost): *Razgranatost*[7] broja x, u oznaci $f(x)$, je koliko različitih brojeva u nizu se pojavljuje na putanji i definiše se izrazom:

$$f(x) = \frac{\sigma_\infty(x)}{\ln x} \qquad 2.6$$

Definicija 2.23 (kompletnost): *Kompletnost*[8] broja x, u oznaci $c(x)$, predstavlja odnos između broja neparnih koraka tokom iteracija i potpunog vremena zaustavljanja, odnosno

$$c(x) = \frac{O(x)}{\sigma_\infty(x)}. \qquad 2.7$$

Primer 2.7: Na slici 1 su predstavljeni grafici fukcija $C^i(x)$ i $T^i(x)$, $i = 0,1,\ldots$ za $x = 39$. Na graficima se uočava nasumično ponašanje uzastopnih iteracija.



Slika 1 Grafici funkcija T^i (x) i C^i (x),i=0,1,… za x=39

[7] eng. finesse
[8] eng. completeness

2.3 Istorijat, težina i bitna pitanja u vezi sa Kolacovim problemom

Kolacov problem se pripisuje nemačkom matematičaru Lotaru Kolacu. Ranih tridesetih godina dvadesetog veka, kao student Univerziteta u Hamburgu, Kolac se zanimao za razne funkcije iz oblasti teorije brojeva. Pošto se podjednako interesovao za teoriju grafova, došao je na ideju da funkcije slične funkciji $C(x)$ predstavi preko usmerenih grafova, uvidevši vezu između strukture usmerenih grafova i ponašanja iteracija takvih funkcija. Jedna od tih funkcija je i $g(n)$, definisana kao:

$$g(n) = \begin{cases} \frac{2}{3}n & \text{ako je } n \equiv 0(mod3) \\ \frac{4}{3}n - \frac{1}{3} & \text{ako je } n \equiv 1(mod3) \\ \frac{4}{3}n + \frac{1}{3} & \text{ako je } n \equiv 2(mod3) \end{cases}$$

Sam problem odlučivanja da li ovako definisane funkcije uvek imaju konačnu putanju, potiče od ove funkcije, $g(n)$, i to za $n = 8$ (da li je $g^{(k)}(8)$ ograničeno). Na ovo pitanje se nikad nije došlo do odgovora, ali se veruje da je ciklus beskonačan. Smatra se da je problem koji se tiče $g^{(k)}(8)$ originalni Kolacov problem, iako nikad nije zvanično objavljen.

Što se tiče problema 3x+1, njegovo originalno poreklo je nepoznato, jer postoji više izvora.

Lotar Kolac je izneo problem 1937. godine, a u matematičkim krugovima se pojavio tek ranih pedesetih. Brajan Tvejts (*Bryan Thwaites*) je objavio problem 1952. godine i nagradu od 1000 funti za njegovo rešavanje. Čarls Trig (*Charles W.Trigg*) trvrdi da je problem 3x+1 predlozio Kenet Vilijams (*Kenneth S. Williams*) 1976. godine, i na to mu je paznju skrenuo jedan od njegovih studenata, u svom radu na temu istorije problema 3x+1 [3].

Vremenom, problem 3x+1 se pojavljivao pod drugim imenima. Kolacov kolega Hase (*H.Hasse)* se bavio samim problemom i njegovom generalizacijom, pa je po njemu nazvan Haseov algoritam (*Hasse's algorithm*). Ime problem Sirakuza (*Syracuse problem*) je takođe preložio Hase tokom posete Univerzitetu u Sirakuzi tokom pedesetih. *S.Kakutani* i *S.Ulam* su tokom zasebnog istraživanja i rada na problemu isti nazvali *Kakutanijev problem* (*Kakutani`s problem*), odnosno *Ulamov problem* (*Ulam`s problem*).

Problem 3x+1 se pojavljuje pod različitim imenima, nude se razne nagrade za njegovo rešavanje i zaokljuplja jako širok naučni krug. Dokaz za to je i veliki broj zvaničnih publikacija na njegovu temu koje predlažu različite pristupe i rešenja.

Sama formulacija problema 3x+1 na prvi pogled deluje prosto. Međutim, iza toga je problem nerešiv sve do danas. Sam problem pripada velikoj klasi problema, koji se tiču analize i ponašanja iteracija pri preslikavanjima, gde se deo domena neprestano proširuje, i u isto vreme "skuplja"

na drugom delu. Ova klasa problema je definitivno važna, i trenutno jako popularna u sferi matematičkih (i fizičkih) istraživanja. Napredak u rešavanju i nalaženje konkretnih rešenja za ovakve funkcije bi bio od izuzetnog značaja.

Teško je reći kolika je težina problema 3x+1, pošto predstavu o težini problema imamo tek kad (i ako) je problem rešen. Trenutni rezultati sugerišu da je problem 3x+1 ekstremno težak. Deo težine se ogleda u nemogućnosti analize uzastopnih iteracija funkcije $T(x)$ i njenoj pseudo-slucajnoj prirodi, odnosno "mešanju" parnih i neparnih brojeva tokom iteracija.

Ukoliko bi za određeni broj iteracija, u zavisnosti od početnog n, bilo strogo dokazano da odnos parnih i neparnih brojeva ne odstupa bitno od očekivanog odnosa 1:1, tada bi bilo moguće pokazati nepostojanje beskonačnih putanja (da se kroz iteracije ne dolazi do 1). Pored toga, ostaje i veliki problem iz teorije brojeva, odbacivanje pretpostavke o postojanju ogromnog netrivijalnog ciklusa za 3x+1 funkciju. Za današnje metode i saznanja iz oblasti teorije brojeva, ovaj problem je "nepristupačan" i po rečima Pal Erdeša (*Pàl Erdős*), "*Matematika nije još uvek spremna da se suoči sa ovakvim problemima"* [4]. Više informacija o problemu 3x+1 i sličnim problemima se može naći u radu [5].

3 Predstavljanje problema

Tokom istorije, pri pokušaju rešavanja problema 3x+1 korišćeni su razni pristupi i uključene različite grane matematike (statistika, algebra, diskretna matematika, teorija brojeva). Pristupi uvode različite strukture i modele za lakše predstavljanje problema 3x+1 Kolacov graf, vektor parnosti, stablo predhodnika). Sve to pomaže u konstrukciji efikasnih algoritama za pretragu za rekorderima i proveru validnosti hipoteze.

3.1 Statistički model niza $T^i(x)$ za fiksirano x

Ako se grafikom predstavi logaritamska vrednost iteracije u zavisnosti od rednog broja iteracije dobija se predstava kako se može proceniti ukupno vreme zaustavljanja.

Za parni korak, vrednost iteracije opada za faktor $\frac{1}{2}$, dok za neparni korak raste približno za faktor $\frac{3}{2}$ $(T^i(x) \approx \frac{3}{2}T^{i-1}(x))$. Uz pretpostavku da je broj parnih i neparnih koraka jednak, u i koraka imamo $\frac{i}{2}$ parnih i isto toliko nepranih koraka. Ukupan faktor za i koraka je:

$$\left(\frac{1}{2}\right)^{\frac{i}{2}} \cdot \left(\frac{3}{2}\right)^{\frac{i}{2}} = \left(\frac{3}{4}\right)^{\frac{i}{2}} = \sqrt{\frac{3}{4}}^{i} = \sqrt{\frac{4}{3}}^{-i} = k^{-i}$$

Odatle sledi da, u proseku, vrednost iteracija opada eksponencijalno u odnosu na početno x,

$$T^i(x) \approx x \cdot k^{-i}$$

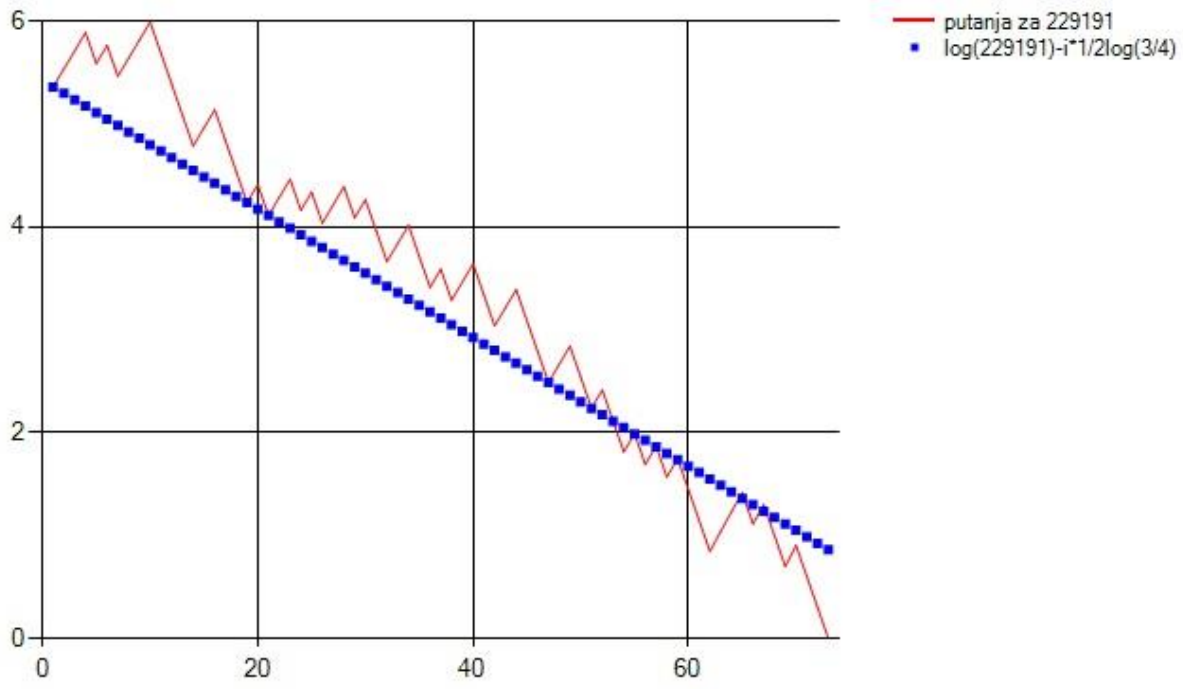
Logaritmovanjem prethodne aproksimacije, dobija se:

$$\log T^i(x) \approx \log\left(xk^{-i}\right) = \log x - i \log k.$$

Za $i = 1$, dobijamo faktor opadanja od $-\frac{1}{2}\log\frac{3}{4} \sim -0.14384$**.** Kada je $i \approx \sigma_\infty(x)$, $\log T^i(x) = 1$. Odatle, procena za $\sigma_\infty(x)$ je:

$$\sigma_\infty(x) \approx c \log x, \text{ gde je } c \approx 6.95212$$

za svako x treba oko $6.95212 \log x$ koraka da dodje do 1. Na slici 2 to prikazano na primeru putanje za 229191.



Slika 2 Statistički model niza T^i (x) za x=229191

3.2 Predstavljanje pomoću usmerenih grafova

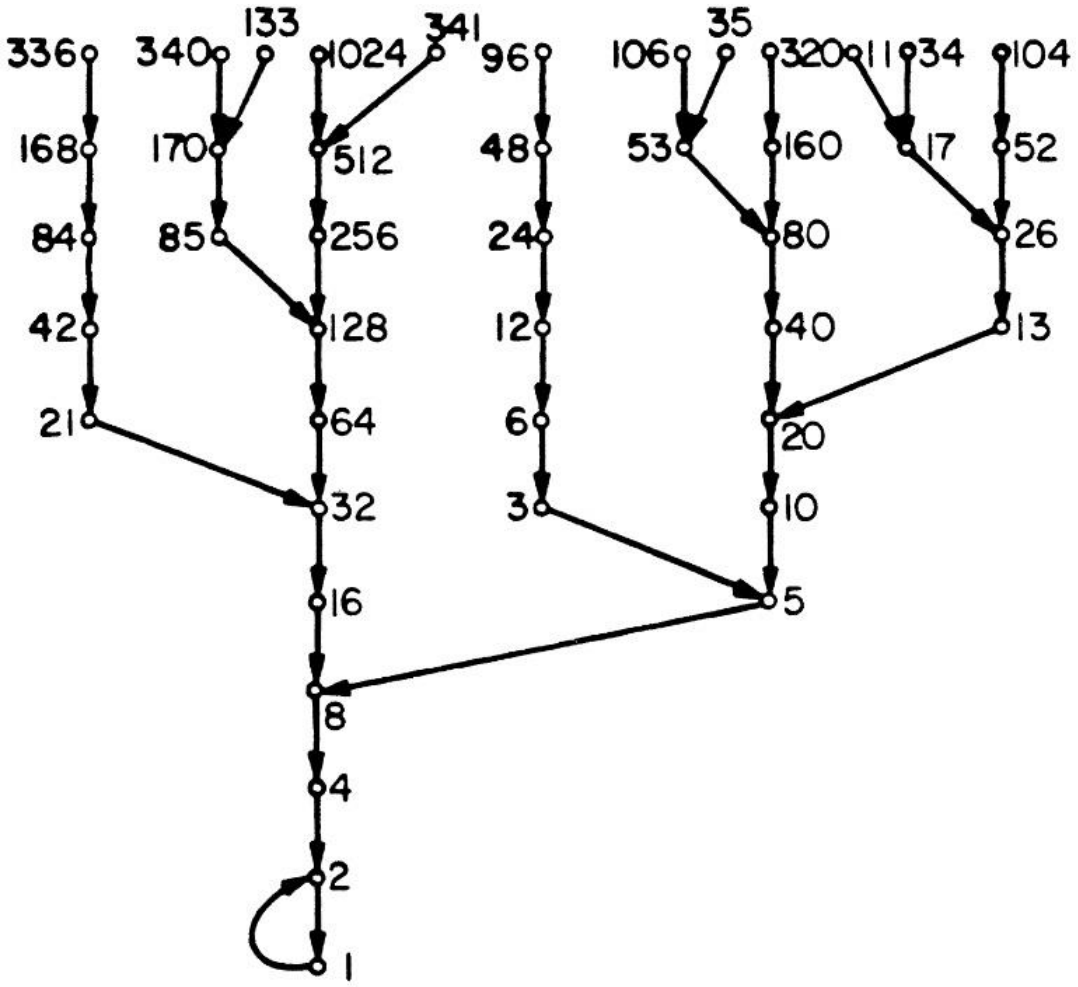
Jedan od pristupa problemu 3x+1 uključuje predstavljanje Kolacove funkcije preko usmerenog grafa koji se naziva *Kolacov graf*.

Definicija 3.1 (Kolacov graf): Kolacov graf je usmereni graf kod koga su čvorovi neparni celi brojevi, sa usmerenim granama od n ka $T(n)$.

Deo Kolacovog grafa je prikazan na slici 3. Problem 3x+1 se može preformulisati u terminima Kolacovog grafa na sledeći način:

Hipoteza 3.1 (problem 3x+1 - peta forma): Kolacov graf je slabo povezan.

Ova forma problema 3x+1 polazi od definicije problema u obrnutom smeru, tj. umesto pretpostavke da svaki broj nakon određenog broja iteracija dolazi do jedan, uzima se da se, polazeći od jedan, dolazi do svakog pozitivnog celog broja.



Slika 3 Iteracije funkcije T(n) predstavljene Kolacovim grafom

Ovim pristupom su obuhvaćeni i parni brojevi, pa graf zapravo prikazuje iteracije relacije $R(x)$, inverzne funkciji $C(x)$. Definiše se na sledeći način:

Teorema 3.1: Polazeći od funkcije $C(x)$, izvodi se njoj inverzna relacija $R(x)$ na sledeći način**:**

$$R(x) = \begin{cases} \{2n\} & \text{ako je } n = 0,1,2,3,5(mod6) \\ \left\{2n, \dfrac{n-1}{3}\right\} & \text{ako je } n = 4(mod6) \end{cases} \qquad 3.1$$

Dokaz: Pretpostavlja se da ovakav graf osim 1-2-4 nema drugih ciklusa. Za svako pozitivno celo n važi: $n \equiv 1(mod2)$ akko $3n + 1 = 4(mod6)$. Ekvivaletno, $\frac{n-1}{3} \equiv 1(mod2)$ akko $n = 4(mod6)$. ■

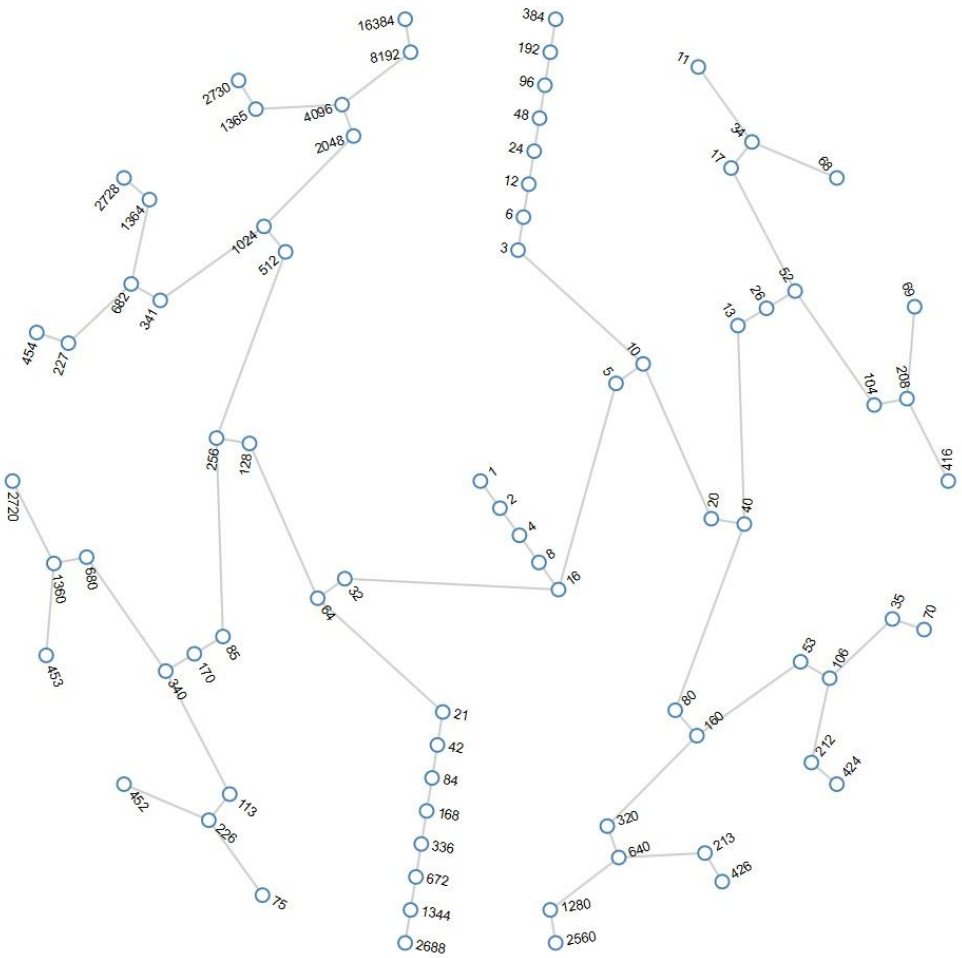
U nastavku teksta vitičaste zagrade se izostavljaju.

Deo grafa koji prikazuje rezultate iteracija funkcije $R(x)$ nalazi se na slici 4. Funkcija $C(x)$ je inverzna za relaciju $R(x)$, a koji se oblik te funkcije primenjuje zavisi od parnosti $R(x) = n$. Ukoliko se umesto funkcije $C(x)$ posmatra $T(x)$, dobija se njoj inverzna relacija $Q(x)$.

Teorema 3.2: Polazeći od funkcije $T(x)$, izvodi se njoj inverzna relacija $Q(x)$ na sledeći način**:**

$$Q(x) = \begin{cases} 2n & ako\ je\ n = 0,1(mod3) \\ 2n, \dfrac{n-1}{3} & ako\ je\ n = 2(mod3) \end{cases} \qquad 3.2$$

Dokaz: Slično kao u slučaju funkcije $R(x)$. ■



Slika 4 Graf iteracija Kolacove funkcije za putanje dužine 14[9]

Hipoteza 3.2 (problem 3x+1 – šesta forma): Za Kolacov graf, kod koga se čvorovi formiraju u odnosu na funkciju inverznu funkciji 3x+1, važi da osim 1-2 nema drugog ciklusa.

[9] Preuzeto sa http://www.jasondavies.com/collatz-graph

3.3 Predstavljanje preko vektora parnosti i Tera-ova teorema[10]

Definicija 3.2 (vektor parnosti): Vektor parnosti za neko N, u oznaci $v(N) = (v_0(N), v_1(N) \dots)$ je vektor oblika

$$v(N) = \left(v_i | v_i = T^{(i)}(N)(mod\ 2), \qquad i = \overline{0,n}, \qquad n = \sigma_\infty(N)\right) \qquad 3.3$$

Vektorom parnosti se u potpunosti opisuje ponašanje funkcije 3x+1 u zavisnosti od N.

Primer 3.1: Za $N = 25$, vektor parnosti je $v(25) = [1,0,1,1,0,0,1,1,0,1,0,0,1,0,0,0,1]$.

Lema 3.1: Vektor prvih k elemenata vektora se određuje kroz koje grane prolazi $T(n)$ u prvih k iteracija.

Dokaz: Indukcijom po k. Za $k = 1$ to je direktna posledica definicije $T(n)$. Ulazno n može se napisati u obliku: $n = n_0 = 2n_1 + r_0$, gde je r_0 poslednja cifra u binarnom zapisu broja n. Primenom T na n_0 dobija se:

$$T(n_0) = T(2n_1 + r_0) = \begin{cases} n_1 & za \ \ r_0 = 0, \\ 3n_1 + 1 & za \ \ r_0 = 1 \end{cases}$$

Dalje, za računanje $T^{(2)}(n_0)$, n_1 se predstavlja u obliku $n_1 = 2n_2 + r_1$, gde je r_1 poslednja cifra u binarnom zapisu n_1(i pretposlednja cifra u zapisu n_0). Dobija se

$$T^{(2)}(n_0) = \begin{cases} n_2 & za \ r_0 = 0 \ \ i \ \ r_1 = 0, \quad n_0 = 4n_2 + 0, \\ 3n_2 + 2 & za \ r_0 = 0 \ \ i \ \ r_1 = 1, \quad n_0 = 4n_2 + 1, \\ 3n_2 + 1 & za \ r_0 = 1 \ \ i \ \ r_1 = 0, \quad n_0 = 4n_2 + 2, \\ 9n_2 + 8 & za \ r_0 = 1 \ \ i \ \ r_1 = 1, \quad n_0 = 4n_2 + 3. \end{cases}$$

Ovaj process je moguće nastaviti i izračunati $T^{(k)}(n_0)$, za proizvoljno k. Svakom novom iteracijom dobija se još jedna cifra iz binarnog zapisa za početno n, odnosno n_0. ■

Kao što se može videti, vektorom parnosti je, za početno N, potpuno određeno za svaku iteraciju funkcije 3x+1 da li se primenjuje na parni ili neparan broj.

Ostatak teksta u ovoj tački napisan je na osnovu materijala sa sledećeg linka.

Lema 3.2: Neka je $N, N \in \mathbb{Z}^+$, oblika $a \cdot 2^k + b, (b < 2^k)$. Tada je $v_k(a \cdot 2^k + b) = v_k(b)$, odnosno prvih k elemenata vekora parnosti zavise samo od b.

Dokaz: Indukcijom po k. Za $k = 1$, $N = 2a + b$, gde b može biti 0 ili 1. Tvrđenje je tačno na osnovu definicije problema 3x+1, jer N može biti ili paran ili neparan broj. Pretpostavimo da tvrđenje važi za neko k.

[10] Riho Terras(1939-2005), matematicar iz Estonije

Neka je $N = a \cdot 2^{k+1} + b$, $(b < 2^{k+1})$. Ako je N paran, $T(N) = \frac{a \cdot 2^{k+1} + b}{2}$, i ako je N neparan $T(N) = 3a \cdot 2^k + \frac{3b+1}{2}$. U oba slučaja prvih k elemenata od $v(T(N))$ zavise od b i jednaki su $v(a)$, čime je tvrđenje dokazano. ■

Primer 3.2: Za $b = 25, k = 6$ i $a = 0,1,2,3$, vektori parnosti navedeni su u narednoj tabeli:

a	$N = a \cdot 2^k + b$	$v(N)$
0	25	$[1,0,1,1,0,0,1,1,0,1 \dots]$
1	89	$[1,0,1,1,0,0,0,1,1,0 \dots]$
2	153	$[1,0,1,1,0,0,1,0,1,0 \dots]$
3	217	$[1,0,1,1,0,0,0,0,1,1 \dots]$

Kao što se može videti, kod svakog od datih vektora prvih 6 članova je isto kao i za $a = 0$ odnosno $N = 25$.

Lema 3.3: Neka je $w_i (0 \leq i < k)$ proizvoljni binarni vektor dužine k. Tada, postoji N za koje važi $v_i(N) = w_i \ (0 \leq i < k)$.

Dokaz: Indukcijom po k. Za $k = 1$, $w_0 = 0$ ili $w_0 = 1$. Tada je N ili paran ili neparan broj. Pretpostavimo da tvrđenje važi za neko k.

Neka je $w_i (0 \leq i < k + 1)$ proizvoljni binarni vektor dužine $k + 1$. Tada postoji neko $N = a \cdot 2^{k+1} + b$, $(b < 2^{k+1})$ za koje je $v_i(N) = w_i (0 \leq i < k)$.

Dalje, uzmemo dva broja $M_1 = a \cdot 2^{k+1} + b$ i $M_2 = a \cdot 2^{k+1} + 2^k + b$. Za oba važi $v_i(M_1) = v_2(M_1) = w_i, (0 \leq i < k)$. Neka je $T^k(M_1) = x$. Iz definicije funkcije 3x+1 sledi $T^k(M_1) = y = x + 3^n$, gde je n broj jedinica vektora $w_i (0 \leq i < k)$.

Pošto je 3^n uvek neparno, sledi da su x i y različite parnosti. Odatle je $w_{k+1}(M_1) \neq w_{k+1}(M_2)$. Kako po pretpostavci postoji $v_i(M) = w_i, (0 \leq i < k + 1)$, sledi da je tvrđenje tačno. ■

Primer 3.3: Uzmimo vektor parnosti $[1,0,1,1,0,0, x \dots]$. Jasno je da x može biti 0 ili 1. Za $N = 25$ imamo da je $T^{(6)}(25) = 11$, pa će umesto x biti 1. Ako uzmemo $N = 25 + 2^6 = 89$, po lemi 3.1 , sedmi element vektora parnosti mora biti 0, i zaista $T^{(6)}(89) = 38$. Može se videti da vektor parnosti u prvih 6 elemenata, ima tri jedinice, pa je tako sedmi element za tekući broj moguće izračunati i preko prethodnog iz $T^{(6)}(89) = T^{(6)}(25) + 3^3 = 11 + 27 = 38$.

Definicija 3.3 (broj parnosti reda k): Za proizvoljno N, broj parnosti reda k, u oznaci P_k, definiše se kao $P_k = \sum_{i<k}(v_i \cdot 2^i)$.

P_k zapravo predstavlja vektor parnosti napisan u obrnutom smeru kao binarni broj. Pri tome važi $0 \leq P_k < 2^k$.

Definicija 3.4: Za proizvoljno N, $\sigma(N) = k$, broj parnosti reda k se definiše kao $P(N) = P_k(N)$.

Primer 3.4: Za $N = 25$ imamo:

$$P(0) = 1, P(1) = 1, P(2) = 5, P(3) = 13, P(4) = 13, P(5) = 13, P(6) = 77 \text{ itd.}$$

Lema 3.4: Za M i $N, M, N \in \mathbb{Z}^+$ važi $P_k(M) = P_k(N)$ akko $M = N(mod\ 2^k)$.

Dokaz: Dokaz leme sredi neposredno iz leme 3.1.

Direktna posledica leme 3.3 je da se svaka klasa kongruencije x po modulu 2^k preslikava u jedinstveni broj y, gde su x i y između 0 i $2^k - 1$. Prema tome, funkcija T je na skupu klasa ekvivalencije po modulu 2^k permutacija. ■

Primer 3.5: Za $k = 4$ imamo brojeve oblika $N = a \cdot 2^4 + r$ gde je $r = \overline{0,15}$. Za $a = 1$ se formira sledeća tabela:

r	$N = 2^4 + r$	$T^0(N), T^1(N), T^2(N), T^3(N)$	$v_4(N)$	$P_4(N)$
0	16	16, 8, 4, 2	[0,0,0,0]	0
1	17	17, 26, 13, 20	[1,0,1,0]	5
2	18	18, 9, 14, 7	[0,1,0,1]	10
3	19	19, 29, 44, 22	[1,1,0,0]	3
4	20	20, 10, 5, 8	[0,0,1,0]	4
5	21	21, 32, 16, 8	[1,0,0,0]	1
6	22	22, 11, 17, 26	[0,1,1,0]	6
7	23	23, 35,53, 80	[1,1,1,0]	7
8	24	24, 12, 6, 3	[0,0,0,1]	8
9	25	25, 38, 19, 29	[1,0,1,1]	13
10	26	26, 13, 20, 10	[0,1,0,0]	2
11	27	27, 41, 62, 31	[1,1,0,1]	11
12	28	28, 14, 7, 11	[0,0,1,1]	12
13	29	29, 44, 22, 11	[1,0,0,1]	9
14	30	30, 15, 23, 35	[0,1,1,1]	14
15	31	31, 47, 71, 107	[1,1,1,1]	15

Drugim rečima, za $k = 4$ imamo permutaciju $k = (1,5)(2,10)(9,13)$.

Ova definicija vektora parnosti daje interesantnu karakterizaciju 3x+1 putanja. Sledeća lema daje još jaču tvrdnju, da se na osnovu vektora parnosti, $v_i(N), (0 \leq i < k)$ može dobiti dobra aproksimacija za $T^k(N)$.

Definicija 3.5: Neka je za proizvoljno $N \in \mathbb{Z}^+$, njegov vektor parnosti v_k. Tada je $d(k)$ broj jedinica u vektoru v_k, tj. $d(k) = \sum_{i=0}^{k} v_i$, $(0 \leq i < k)$.

Lema 3.5: Za proizvoljno $N \in \mathbb{Z}^+$, njegov vektor parnosti $v(N)$ važi $T^{(k)} \leq S_k = N \cdot 3^{d(k)} \cdot 2^{-k}$, gde je $T^{(k)} = S_k + R_k$, $R_k = \sum_{i=0}^{k} v_i \cdot 3^{d(i+1,k)}/2^{k-i}$.

Datu aproksimaciju je moguće zapisati i u formi logaritma:

$\ln(S_k) \leq \ln(T_k) = \ln(S_0) + d(k) \cdot \ln 3 - k \cdot \ln 2$, gde $\lim_{k\to\infty} \frac{T^{(k)}(N)-T_k}{T^{(k)}(N)} = 0$ za $T^{(k)}(N) \to \infty$.

Primer 3.6: Za $N = 229191$, 3x+1 putanja je 229191, 343787, 515681, 773522, 386761, 580142,… , odnosno odgovarajući vektor parnosti $v(763) = [1,1,1,0,1,0, \dots]$. Primenom prethodne leme za $T^5(N)$ dobijamo:

$$T^5(229191) = 229191 \cdot 3^4 \cdot 2^{-5} = 580139.71875\ 78125$$

što predstavlja vrednost za oko 2,3 manju od stvarne, ili u procentima $0{,}0004\%$.

Polazeći od ove procene za $T^k(N)$ preko vektora parnosti, može se iskoristiti $v(N)$ za dobijanje dobre procene o konvergenciji za N. Zato se uvodi sledeća definicije:

Definicija 3.6 (konvergentni i divergentni vektori): Neka je v_i $(0 \leq i < k)$ jedna komponenta vektora parnosti v dužine k, za neko N. Najmanje j za koje je $d(j) < 0$ se naziva vreme konvergencije, a vektor v konvergentan. Ukoliko takvo j ne postoji, onda je v divergentan vektor.

Hipoteza 3.3: Neka je $c(j) = d(j) \cdot \ln 3 - j \cdot \ln 2$ za svako $j < k$. Tada je vektor parnosti v konvergentan ako je $c(j) < 0$ za neko $j < k$. Najmanje j za koje je $c(j) < 0$ se naziva vreme konvergencije za v, ili vreme konvergencije za N kom odgovara vektor parnosti v. Ukoliko takvo j ne postoji, onda je v, a samim tim i odgovarajuće N, divergentno.

Na osnovu leme 3.4, potreban ali ne i dovoljan uslov da N ima vreme zaustavljanja k je da vektor parnosti ima vreme konvergencije k.

Lema 3.6: Neka je v_i vektor parnosti dužine k sa vremenom konvergencije k i neka je M skup brojeva koje imaju vektor parnosti v. Tada, svi dovoljno veliki brojevi N (u odnosu na M) imaju vreme zaustavljanja k.

Dokaz: Dokaz je dostupan u [4]. ■

Primer 3.7: Svako $N \equiv 3(mod 16)$ ima vektor parnosti v koji počinje sa $[1, 1, 0, 0]$. Vreme konvergencije za ovakvo v je 4. Po lemi 3.5 , svako takvo dovoljno veliko N ima isto vreme zaustavljanja 4.

Na osnovu leme 3.5 sledi zaključak da samo konačno mnogo brojeva sa vremenom konvergencije k nema isto vreme zaustavljanja. Zapravo, ni za jedno N osim jedinice nije poznato

da ispunjava ovaj uslov. Pošto (na osnovu leme 3.5) takav broj treba da bude relativno mali, veruje se da ne postoji. To opravdava sledeću hipotezu:

Hipoteza 3.4: Svako $N > 1$ ima vreme konvergencije jednako vremenu zaustavljanja.

Lema 3.7: Neka je v_k skup svih vektora parnosti dužine k i w_k podskup od v, sačinjen od svih divergentnih vektora iz v. Tada važi:

$$\lim_{k\to\infty} \frac{|w_k|}{|v_k|} = 0.$$

Teraova (*Terras*) teorema govori preciznije o konvergenciji prethodnog izraza. Tačnije, teorema pokazuje da sa proširenjem intervala, udeo brojeva koji nemaju konačno vreme zaustavljanja teži nuli. Drugim rečima:

Teorema 3.1 (Teraova teorema[11]): Neka je dato $M \in \mathbb{Z}^+$ I $D(M)$ (u)deo brojeva manjih od M koji nemaju konačno vreme zaustavljanja. Tada važi:

$$\lim_{M\to\infty} D(M) = 0.$$

[11] http://www.ericr.nl/wondrous/terras.html

3.4 Predstavljanje preko stabla prethodnika

Struktura stabla je dobar izbor za predstavljanje nepredvidivog ponašanja 3x+1 putanja. Stablo prethodnika, kao jedna od formi stabla na dobar način ilustruje pomenutu problematiku.

Definicija 3.7 (stablo prethodnika): *Stablo prethodnika* (eng. *predecessor tree*) je varijanta stabla kod koga je u korenu 1, a naslednici nekog čvora x su svi brojevi n, takvi da je $T(n) = x$.

Primer 3.8: Ako želimo da prikazemo porodično stablo na ovaj način i u koren stavimo osobu A, direktni naslednici će biti njeni roditelji. Nivo ispod toga, baba i deda itd. Na taj način poreklo osobe A, počevši od nekog pretka definišemo tako što lociramo odgovarajućeg pretka i pratimo putanju nagore kroz stablo do korena. Slično, ako za koren postavimo 1 a kao direktne naslednike sve one neparne brojeve koji dolaze direktno do 1 kroz iteracije Kolacove funkcije (bez da se između njih i 1 na putanji nađe neki drugi neparan broj), možemo formirati putanju za proizvoljan broj. Dovoljno je da broj lociramo i idemo kroz stablo od njega prema korenu (1).

Stablo prethodnika služi da da jasniji uvid u strukturu iteracija Kolacove funkcije, odnosno da vizuelno pokaže neka svojstva vezana za Kolacove putanje i tako pomogne u objašnjenju njihove "nasumične" prirode.

Zbog jednostavnosti, stablo uključuje samo neparne cele brojeve. Svaki neparni broj x u stablu predstavlja i svaki parni broj $x \cdot 2^k, k = 1,2, \ldots$, jer se primenom funkcije $T^{(k)}$ taj broj preslikava u x.

Ovo stablo omogućuje da se hipoteza 3x+1 preformuliše na još jedan način.

Hipoteza 3.5: Konstrukcija stabla predaka koje sadrži sve neparne brojeve znači da svaki broj dolazi do 1 pri iteraciji funkcije $3x + 1$.

Sinovi čvora $T(k-1)$ u stablu su čvorovi (njih beskonačno mnogo akko k nije deljivo sa 3) za koje važi rekurzivna formula:

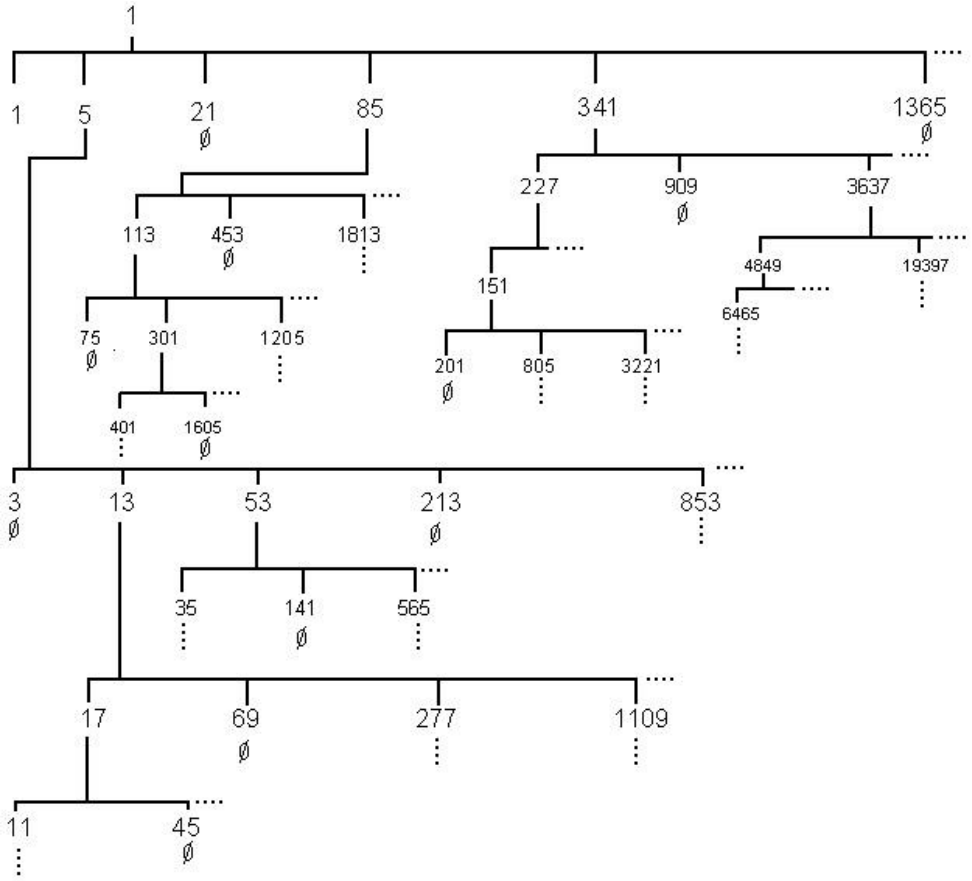
$$T(k) := \frac{-1 + T(k-1) \cdot 2^{p(k)}}{3} \qquad 3.4$$

gde je $p(k)$ parno za $T(k-1) \equiv 1(mod3)$, a neparno za $T(k-1) \equiv 2(mod3)$. U slučaju da je $T(k-1)$ deljivo sa 3, tada je taj čvor list u stablu predhodnika.

Stablo prethodnika opisano u definiciji 3.7 se naziva opšte stablo prethodnika. Druga varijanta je binarno stablo prethodnika i predstavlja transformaciju stabla prethodnika u opštem obliku po pravilu "prvi sin, naredni brat". Apstraktno stablo prethodnika na specifičan način opisuje strukturu podstabla čvora stabla prethodnika koji odgovara broju oblika $8k + 5$.

Opšte stablo prethodnika prikazano je na slici 5 . Ima dve prednosti u odnosu na ostale i to:

- listovi su neparni brojevi deljivi sa tri
- putanja nagore od bilo kog čvora do korena predstavlja njegovu Kolacovu putanju, npr. 11-17-13-5-1.



Slika 5 Deo opšteg stabla prethodnika u generalnom obliku

Kod prikaza stabla, sinovi za proizvoljan broj su poređani horizontalno u rastućem redosledu. Kako svi sinovi imaju istog sledbenika na Kolacovoj putanji (odnosno, istog prethodnika u stablu prethodnika) to drugi, treći i ostali se posmatraju kao "proširenje" (*extension*) prvog sina.

Binarno stablo prethodnika (slika 6) sadrži iste čvorove, ali drugačije povezane. Kod njega svaki čvor ima dva sina, prvog sina i narednog brata iz opšteg stabla prethodnika. I ovaj oblik sadrži čvorove, kao i opšte stablo, u nešto izmenjenoj strukturi, s tim što je ovde izbor čvorova malo drugačiji

- svaki čvor, čak i list ima svoje desno podstablo. Skup uzastopnih desnih potomaka čvora naziva proširenje tog čvora**.** Za svaki desni potomak važi da ima istog pretka, odnosno isti naredni element (sledbenika) na Kolacovoj putanji. Veza između svaka dva uzastopna potomka je $n : 4n + 1$.

- za razliku od opšteg stabla, brojevi deljivi sa tri nisu listovi, nego čvorovi koji imaju samo desnog sina.
- u binarnom stablu nema listova. Zbog toga, put naviše od čvora nije uvek Kolacova putanja. Na primer, putanja od 11 je 11-17-13-3-5-1, gde 3 predstavlja "višak"
- može se dokazati da se svi neparni brojevi mogu podeliti na dva disjunktna skupa: leve i desne sinove u stablu prethodnika, jer su desni sinovi oblika $8k + 5$, a levi sinovi oblika $8k + 1$, $8k + 3$ ili $8k + 7$. [6]

Kod levih potomaka je niz vrednosti uslovljen vrednošću roditeljskog čvora po modulu tri. Dužina tog niza (poslednji u nizu je čvor deljiv sa tri) je promenljiva.

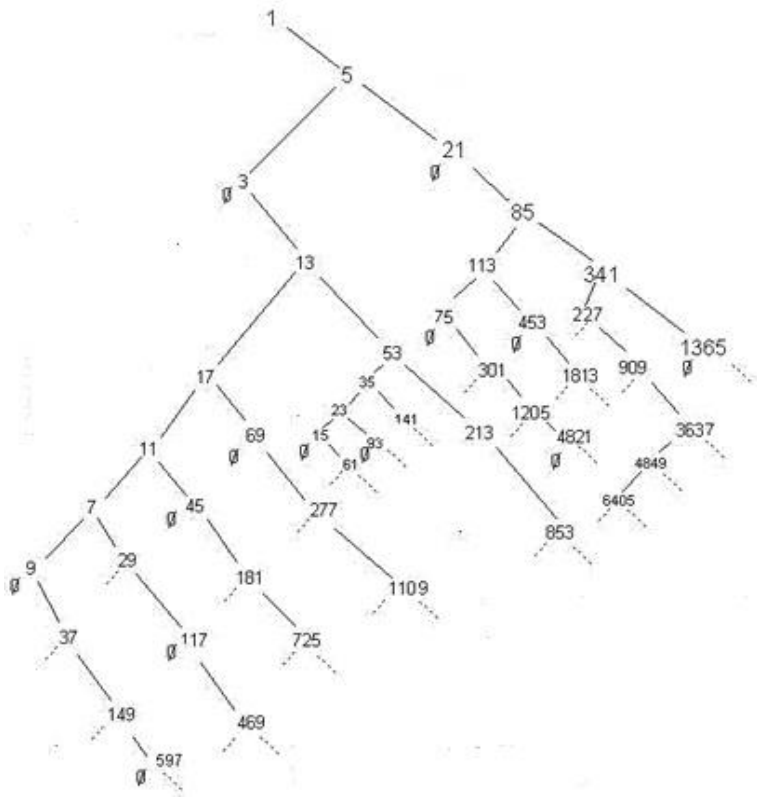
Za binarno stablo prethodnika je karakterističan niz čvorova dobijen silaskom ulevo od proizvoljnog čvora oblika $8n + 5$. Ti čvorovi su desni sinovi u stablu i generišu se na osnovu formule 3.4 . Takav niz se naziva levi niz (*left descent)*.

Definicija 3.8: Skup brojeva oblika $dn + c$, za neke konstante c i d označava se kraće $c[d]$.

Apstraktno stablo prethodnika zanemaruje eksplicitno predstavljanje proširenja čvorova, tako što u koren stavlja proizvoljan broj $5[8]$ (sva proširenja su tog oblika). Fokus je na levim sinovima u binarnom stablu. Čvor u korenu stabla se označava sa e. Građenje stabla se odvija rekurzivno, podelom vrednosti u nekom čvora na tri podskupa I to:

1. skup listova (brojevi oblika $24n + 5 + 2 \cdot 8$ ($21[24]$) (brojevi deljivi sa 3)
2. skup koji daje leve potomke manje od sebe (brojevi oblika $24n + 5$, odnosno 5[24]) (s-potomci)
3. skup koji daje leve potomke veće od sebe (brojevi oblika $24n + 5 + 1 \cdot 8$, odnosno $13[24]$) (b-potomci)

Dakle, svaki čvor se deli na tri podskupa $5[24]$, $13[24]$ i $21[24]$ (tri vrednosti od $5[8] mod 24$). Po modulu 3 su, redom, 2, 1 i 0. Sinovi prva dva su $(2(24n + 5) - 1)/3 = 16n + 3$, odnosno $(4(24n + 13) - 1)/3 = 32n + 17$, a treći nema sina.

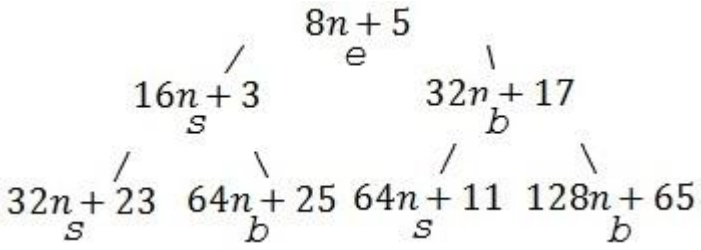


Slika 6 Deo binarnog stabla prethodnika

Sinovi se dobijaju primenom formula:

- $(2n-1)/3$ kao s-korak, za novo levo podstablo
- $(4n-1)/3$ kao b-korak, za novo desno podstablo

Proces se beskonačno ponavlja. Pošto se skup listova ne uzima u dalje razmatranje (akcenat je na levim potomcima), isključuje se iz stabla. Tako apstraktno stablo prethodnika dobija formu binarnog stabla. Prva tri nivoa apstraktnog stabla prethodnika, sa skupom vrednosti koje čine određene čvorove prikazan je na slici 7.



Slika 7 Vrednosti u prva dva nivoa apstraktnog stabla prethodnika

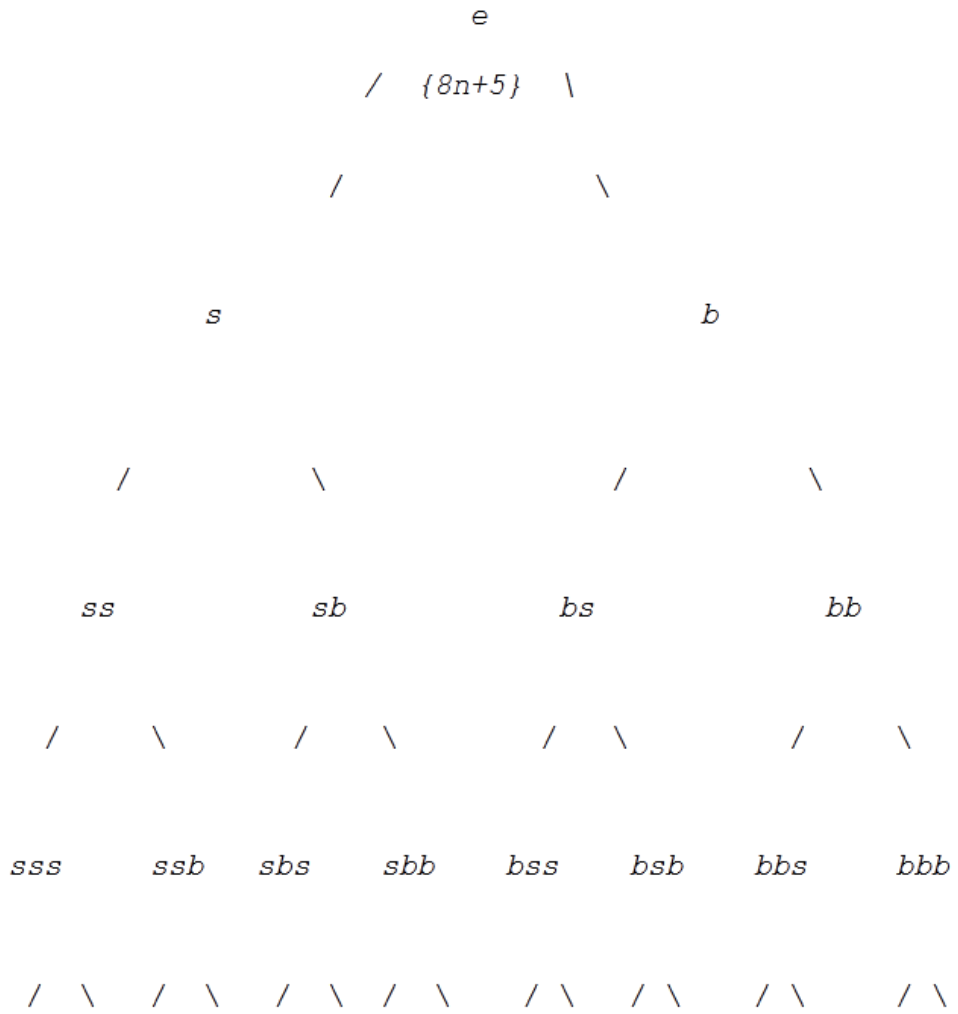
U stablu se levi, odnosno desni sin dobijaju primenom s, odnosno b koraka. Da bi se predstavila putanja od korena do bilo kog čvora (skupa vrednosti), koriste se nizovi sastavljeni od sledećih simbola:

- *e* – za koren i proširenja,
- *s* – za *s*-potomke,
- *b* – za *b*-potomke.

Tako, apstraktno stablo prethodnika generiše sve moguće kombinacije proizvoljne dužine od simbola b i s. Izvođenje sadržaja stabla za prve nivoe je prikazano na slici 8.

Definicija 3.9 (list-čvor): List-čvor (eng. *leaf-node*) predstavlja čvor u binarnom stablu prethodnika čija je vrednost 3[0]. Nakon list-čvora nema daljeg grananja (nema potrebe za daljim razmatranjem putanja). Odatle i "list" u nazivu.

Skup levih potomaka (eng. *left descent assembly,* skraćeno *l.d.a*) čine levi potomci u binarnom stablu prethodnika. Predstavlja put od korena stabla prethodnika u apstraknom obliku do nekog list-čvora. Uključuje koren, unutrašnje čvorove i list-čvor.



Slika 8 Prva tri nivoa građenja apstraktnog stabla prethodnika

3.5 Predstavljanje Kolacovih putanja preko binarnog stabla prethodnika

Prvi pristup se bazira na *vektoru sekvenci* i polazi od pretpostavke da na početku vektora mora biti neparan broj i da bar jedno deljenje sa 2 prethodi jedinici. Vektor je oblika $v = [m_1, m_2, \ldots\ldots m_l]$, gde m_i označava broj deljenja sa 2 nakon i-tog neparnog broja na putanji.

U prethodnom delu je rečeno da je akcenat na levim potomcima na nivou binarnog stabla prethodnika i da se oni dele na s-potomke i b-potomke na nivou apstraktne varijante stabla prethodnika u zavisnosti od ostatka pri deljenju sa tri.

Ispostavlja se su s-potomci oblika $2[3]$, i na *Kolacovoj putanji* između njih i sledećeg neparnog broja postoji jedno deljenje sa 2, što se naziva s-korak. Slično, b-potomak i naredni neparni član Kolacove putanje dele dva deljenja sa 2, i to se naziva b-korak. Nailazak na putu naviše u binarnom stablu na proširenje, list-čvor ili koren se naziva e-korak.

Prvi način za predstavljanje Kolacovih putanja polazi od činjenice da se svaka putanja sastoji od *s*, *b* i *e*-koraka. Ukoliko svaka putanja kreće iz korena, put kroz prethodnike (u okviru stabla) predstavljamo '01' za s-korak, '001' za b-korak i '00' za e-korak. Dakle, ovim nizovima se predstavljaju vektori parnosti na osnovu tri osnovna koraka.

Ipak, kod ovog načina se javlja problem, a to je pojava dodatnih vrednosti na putanjama. Dodatne vrednosti se nazivaju fantom vrednosti. Kod binarnog stabla prethodnika to se javlja svaki put kada levi sin prelazi u svoje proširenje na narednom nivou. Ovaj problem je ilustrovan na slici 9.

Slika 9 Pojava fantom vrednosti na Kolacovoj putanji

Silazak udesno, označen sa e u oba slučaja, pokazuje odnos $n: 4n + 1$ između oca i njegovog proširenja. Dva tipa silaska ulevo, označeni sa *s*, odnosno *b*, pokazuju načine prelaza između levog sina i oca. 3 dostiže 5 uz jedno deljenje sa 2 (putanja od 3 je 3, 10, 5), dok 9 dolazi do 7 nakon 2 deljenja sa 2 (putanja od 9 je 9, 28, 14, 7). Ipak 3 i 9 su ovde višak bez obzira što su prethodnici od 5 i 7, jer se 3 ne pojavljuje na putanji između 13 i 5, odnosno 9 između 37 i 7. Oni svakako treba da budu deo stabla prethodnika, ali i da se pojavljuju na putanjama svojih "proširenja", što ovde nije slučaj.

Kod predstavljanja putanja, moguće je ubaciti "više" ekstenzije elemenata koji su fantom vrednosti. Više o predstavljanju putanja je moguće naći na [7].

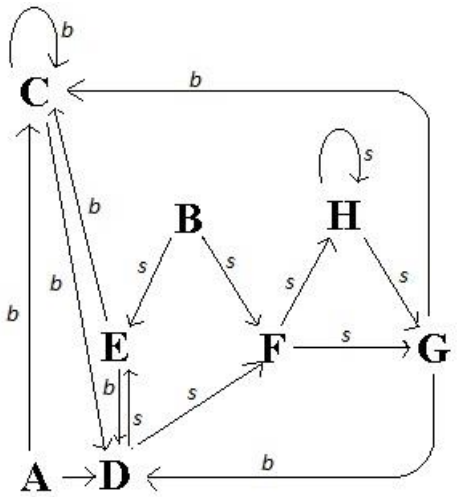
3.5.1 Svođenje na dijagram stanja

Dijagram stanja (*state transition diagram*) se odnosi na prelaze iz oca u sina u stablu prethodnika. Takvi prelazi se nazivaju stanja.

Definicija 3.10 (stanje): Stanje je uređen par, u oznaci (a, b), gde je a ostatak pri deljenju sa 8, a b ostatak pri deljenju sa 3.

Dijagram stanja pokazuje različita stanja za neparne brojeve, dok prelazi između stanja daju uvid u moguće promene kroz koje neparni brojevi prolaze tokom procesa generisanja sinova u stablu prethodnika. Stanja su podeljena u zavisnosti od ostatka pri deljenju sa 3 i 8. U dijagramu, stanja se označavaju određenim slovom u zavisnosti od vrednosti ostatka po modulu 8, a prelazi sa linijama koji označavaju *b*, odnosno *s*-korak. Dijagram stanja je prikazan na slici 10. Tabela prikazuje prelaz između stanja, u odnosu na vrednost brojeva pri deljenju sa 3, odnosno 8.

Za *b*-korak (brojevi oblika 1[3]), Kolacov prethodnik broja n je $(4n-1)/3$, dok je za *s*-korak (brojevi oblika 2[3]) prethodnik $(2n-1)/3$. Pošto se radi samo sa neparnim brojevima, u obzir dolaze samo oni koji daju neparan ostatak pri deljenju sa 8 (to su brojevi oblika 1[8], 3[8], 5[8], 7[8]). Zbog toga treba predstaviti samo 8 razlicitih stanja i prelaza između njih, dva ostatka pri deljenju sa 3 u kombinaciji sa 4 ostatka pri deljenju sa 8. Listovi (brojevi oblika 0[3]), se ne uzimaju u obzir jer se podrazumeva da je iz svakog l.d.a skupa moguće doći do lista u stablu prethodnika.



početno stanje	korak	naredno stanje
A 1[3] 5[8]	b	C 1[3] 1[8] i D 2[3] 1[8]
B 2[3] 5[8]	s	E 1[3] 3[8] i F 2[3] 3[8]
C 1[3] 1[8]	b	C 1[3] 1[8] i D 2[3] 1[8]
D 2[3] 1[8]	s	E 1[3] 3[8] i F 2[3] 3[8]
E 1[3] 3[8]	b	C 1[3] 1[8] i D 2[3] 1[8]
F 2[3] 3[8]	s	G 1[3] 7[8] i H 2[3] 7[8]
F 2[3] 1[8]	s	G 1[3] 7[8] i H 2[3] 7[8]
G 1[3] 7[8]	b	C 1[3] 1[8] i D 2[3] 1[8]
H 2[3] 7[8]	s	G 1[3] 7[8] i H 2[3] 7[8]

Slika 10 Dijagram stanja i tabela prelaza

Interesantno je da su u dijagramu sva stanja, osim *A* i *B*, dostupna iz nekog drugog. Jedino *A* i *B* ostaju kao nedostižna stanja.

Primer 3.9: Za *l.d.a* vrednost 427 (1[3], 3[8]), na putanji se pojavljuju sledeći brojevi 445, 593, 263, 175, 233, 155, 103, 137, 91, 121, 161, 107, 71, 47, 31, 41, 27 (što je list u stablu prethodnika). Put kroz dijagram stanja I promena je:

A(445), b->D(593), s->F(395), s->H(263), s->G(175), b->D(233), s->F(155), s->G(103), b->D(137),

s->E(91), b->C(121), b->D(161), s->F(107), s->H(71), s->H(47), s->G(31), b->D(41), s->list(27).

Poslednji korak nije u dijagramu stanja jer je 27 list-čvor. Prateći tabelu prelaza, vidi se da dati primer sadrži sve prelaze iz tabele.

4 Algoritmi za efikasnu proveru Kolacove hipoteze

Iz definicija navedenih u poglavlju 2 se vidi da za svaki pozitivan ceo broj x postoje određena *svojstva* u odnosu na iteracije funkcije $T(x)$, odnosno $C(x)$. Najbitnija su vreme zaustavljanja, potpuno vreme zaustavljanja i maksimum. U ovom poglavlju su opisani neki algoritmi za efikasno nalaženje navedenih svojstava, tehnike za ubrzanje algoritama, veza između rezultata na C i T putanjama, kao i neke ideje za dalje ubrzanje izračunavanja. Ovaj deo teksta se zasniva na materijalu izloženim u [8] i [9].

4.1 Uvodna zapažanja i pojam rekordera

Efikasna provera validnosti hipoteze 3x+1 zahteva proveru velikog intervala brojeva, stoga je bitno dizajnirati brz i efikasan algoritam za računanje iteracija i bitnih svojstava za svaki od brojeva.

Iz definicija Kolacove funkcije $\big(C(x)\big)$ i funkcije 3x+1 $\big(T(x)\big)$, vidi se da između njih postoji veza. Za fiksirano x uočava se da skup brojeva na putanji dobijenih kroz iteracije funkcije 3x+1 (T-putanji) predstavlja podskup skupa brojeva na putanji dobijenih iteriranjem Kolacove funkcije (C-putanji). Takođe, svaki broj k na C-putanji, za početno x, ili se nalazi na T-putanji ili se nalazi na T-putanji kao $\frac{k}{2}$. Formalno:

Lema 4.1: *(i)* Za svako $i > 0$, svako $x > 0$ i svako $k > 0$, ako je $T^{(i)}(x) = k$, tada postoji neko $j \geq 0$ tako da je $C^{(j)}(x) = 2k$ i $C^{(j+1)}(x) = k$.

(ii) Za svako $j > 0$, svako $x > 0$ i svako $k > 0$, ako je $C^{(j)}(x) = k$, tada postoji neko $i \geq 0$ tako da je $T^{(i)}(x) = 2k$ ili $T^{(i)}(x) = \frac{k}{2}$.

Dokaz: Indukcijom po i, odnosno j, za *(i)* i za *(ii)*. ■

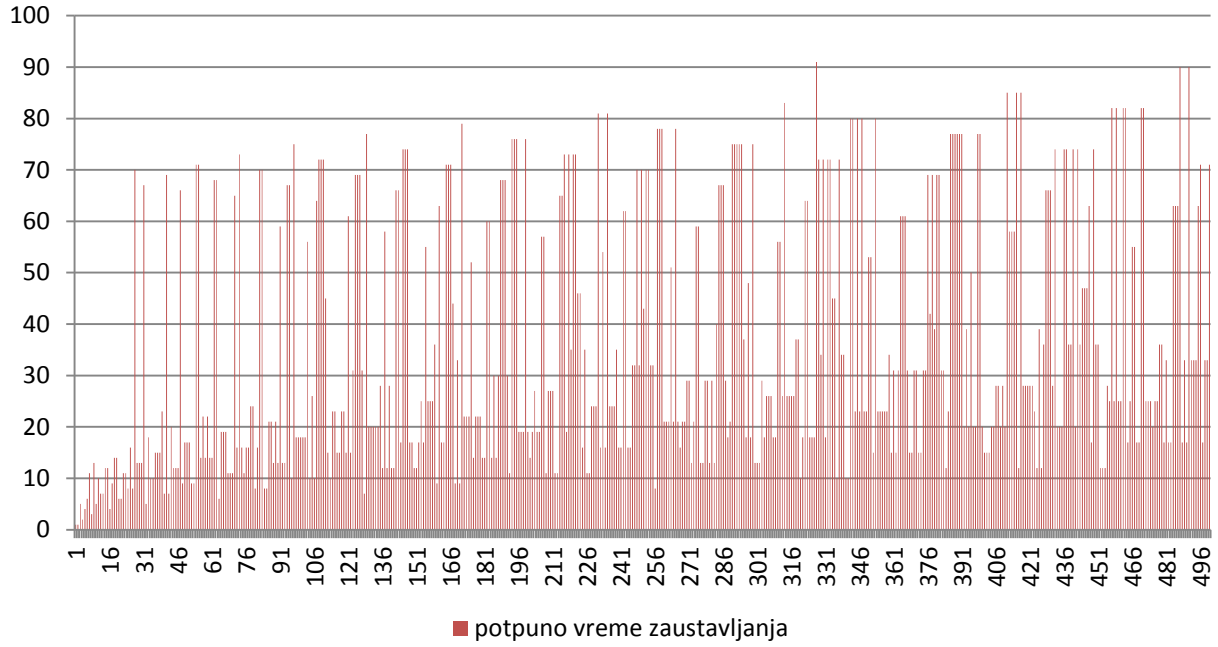
Primer 4.1: Za $x = 7$, primenom funkcije C putanja se sastoji od brojeva: 7, 22, 11, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1. Primenom funkcije T za isto x dobijamo putanju: 7, 11, 17, 26, 13, 20, 10, 5, 8, 4, 2, 1.

Na osnovu prethodne leme može se doći do još nekih zaključaka. Maksimum na C-putanji je dva puta veći broj od maksimuma na T-putanji, ukoliko je paran broj. Dalje, svaki korak funkcije $T(x)$ u sebi sadrži tačno jedan paran korak funkcije $C(x)$. Vreme zaustavljanja i potpuno vreme zaustavljanja, za fiksirano x u odnosu na funkciju koju primenjujemo, imaju različite vrednosti, osim u slučaju da je x stepen dvojke.

Proverom validnosti za brojeve iz nekog interval se uočava da, sa porastom gornje granice intervala vrednosti za vreme zaustavljanja, potpuno vreme zaustavljanja i maksimum takođe variraju. Međutim, ako se za svako od svojstava posmatra učestalost pojavljivanja maksimalne

vrednosti to se dešava sve ređe u odnosu na porast intervala. Na slici 11 se to vidi za potpuno vremena zaustavljanja u intervalu [1, 500].

Primer 4.2:



Slika 11 Grafik potpunog vremena zaustavlja u odnosu na početnu vrednost u intervalu [1,500] za funkciju T(x)

Na grafiku, rekorderi su oni brojevi čija je ordinata veća u odnosu na sve brojeve manje od posmatranog.

Primer 4.3: Rekorderi u odnosu na vreme zaustavljanja na intervalu $[1,30]$ za funkciju $C(x)$ su: 2, 3, 6, 7, 9, 18, 25, 27 ...[12].

Nakon ovih zapažanja, postavlja se pitanje kako su povezana određena svojstva (maksimum, vreme zaustavljanja, potpuno vreme zaustavljanja) za isti broj i različite putanje, tj. da li na osnovu izračunavanja za jednu putanju možemo dobiti ili bar oceniti vrednosti za drugu putanju. Da li je rekorder za neko svojstvo istovremeno rekorder za neko drugo svojstvo? Slično, da li je rekorder za C-funkciju istovremeno rekorder za T-funkciju zbog njihove povezanosti. O odnosu između maksimuma i postojanja vrednosti na putanji govori sledeća lema.

[12] Dosadašnji progres i tabela sa svim dosad otkrivenim maksimumima je data u dodatku na kraju rada.

Lema 4.2: *(i)* Za svako $x > 2$ i svako $k > 0$, ako postoji neko $i \geq 0$ tako da je $C^{(i)}(x) = k$, tada postoji neko $0 \leq j \leq G(x)$[13] tako da je $C^{(j)}(x) = k$; pored toga, ako postoji neko $i \geq 0$ tako da je $T^{(i)}(x) = k$, tada postoji neko $0 \leq j \leq \sigma_\infty(x)$ tako da je $T^{(j)}(x) = k$.
(ii) Za svako $x > 0$, ako je $Mx_T(x) \neq x$, tada je $Mx_C(x) = 2 \cdot Mx_T(x)$.[14]
Posledica: Broj $x > 0$ je rekorder za $Mx(x)_T$ akko je rekorder za $Mx(x)_C$.
Pitanje odnosa vremena zaustavljanja i potpunog vremena zaustavljanja je i dalje otvoreno [8]. Ipak, i tu postoje određena pravila. Dokaz naredne dve leme se može videti u [3].

Lema 4.3: Za svako $x > 0$, ako je $G(x) \neq 0$, tada je $G(x)/2 < \sigma_\infty(n) \leq G(x)$.

Lema 4.4: Za svako $x > 0$, $\sigma_\infty(x) = E_C(x)$[15].
Navedene leme ne dokazuju da je maksimum za $D(x)$ istovremeno i maksimum za $\sigma_\infty(x)$, mada do sada nije pronađen izuzetak. Odatle sledeća hipoteza:

Hipoteza 3.1: Broj $x > 0$ je rekorder za $G(x)$ akko je rekorder za $\sigma_\infty(x)$.

U nastavku poglavlja su navedene strategije kojima se u različitim fazama algoritma ispituje da li je neko x kandidat za rekordera, a priori i a posteriori strategije za prekidanje pretrage, algoritam za ubrzavanje iteracija i strategija za ubrzanje pretrage za rekorderima.

[13] G(x) je oznaka za potpuno vreme zaustavljanja u slučaju funkcije C(x).
[14] Mx_T je maksimum na putanji u slučaju funkcije T(x), Mx_C u slučaju funkcije C(x).
[15] $E_T(x)$ je broj pranih koraka u slučaju funkcije T(x), $E_C(x)$ u slučaju funkcije C(x).

4.2 Algoritam pretrage i ubrzanje

Analizom dosadašnjih rezultata u [3], [10] i [4] zapaža se da se sa porastom prostora pretrage broj rekordera smanjuje.

Primer 4.4: Za funkciju $C(x)$ postoji 49 rekordera u odnosu na maksimum na putanji i 78 rekordera u odnosu na vrema zaustavljanja na intervalu $[1,\ 5 \cdot 10^{10}]$. Od toga, u intervalu $[10^9, 5 \cdot 10^{10}]$ nalazi se svega 5 rekordera u odnosu na maksimum na putanji i 12 u odnosu na potpuno vreme zaustavljanja.

Kako se prostor pretrage širi, rekorderi se proređuju. Jedan od glavnih zadataka kod realizacije programa za traženje rekordera je da se efikasno i najbrže sazna da li neki broj može biti rekorder. Za to postoje tri strategije:

- Prekidanjem pretrage tako što se pre samog prolaska kroz iteracije funkcija C ili T otkrije da ulazni broj nije kandidat za rekorder.
- Ubrzati izvršavanje algoritma za iteracije
- Zbog rada sa velikim brojevima, koji se ne mogu deklarisati preko osnovnih tipova, koristiti algoritme i strukture podataka za rad sa njima ili već postojeće klase za rad sa velikim brojevima. Pri tome voditi računa i izvršavati računske operacije što efikasnije i brže.

4.2.1 A Priori prekidanje pretrage

Najbolji način da se za određeni broj prekine pretraga je dokazati da on ne može da bude rekorder. Takva strategija se naziva *a priori* strategija za prekidanje pretrage. Manje efikasan način prekida pretragu nakon što ispita neka svojstva na putanji za neko x. Ova strategija naziva *a posteriori* strategija za prekidanje pretrage.

Tako, pretraga za rekorderima u odnosu na maksimalnu vrednost na putanji može se lako ograničiti samo na neparne brojeve, jer prva iteracija funkcije $C(x)$ za paran broj podrazumeva deljenje sa dva. Iz datog zapažanja zaključuje se da važi sledeća lema:

Lema 4.5: Za svako $n > 0$, $Mx(2n) = max\{2n,\ Mx(n)\}$.

Posledica: 2 je jedini paran rekorder u odnosu na maksimum na putanji. Isto važi i za funkciju $T(x)$.

Kod izračunavanja vremena zaustavljanja za parne brojeve, važi da je ono jednako vremenu zaustavljanja za upola manji broj plus jedan.

Lema 4.6: Za svako $n > 0$, $D(2n) = D(n) + 1$.

Posledica: Ako je neko k rekorder u odnosu na potpuno vreme zaustavljanja, tada prvi parni broj koji može biti kandidat je $2k$. Slično važi i za $\sigma_\infty(n)$.

Dokaz: Neka je neko k rekorder u odnosu na potpuno vreme zaustavljanja. Po definiciji, za svako $0 < j < k$, $D(j) < D(k)$. Iz prethodne leme, $D(2j)$ je ograničeno i važi:

$$D(2j) = 1 + D(j) \leq D(k) = D(2k) - 1 < D(2k). \blacksquare$$

Na osnovu ovih zapažanja lako je predvideti sve parne brojeve koji su kandidati za rekordera u odnosu na potpuno vreme zaustavljanja i maksimum. U narednom primeru se to vidi u slučajevima (3, 6), (9, 18), (27, 54), (670617279, 1341234558).

Primer 4.5: Parni rekorderi u odnosu na potpuno vreme zaustavljanja za funkciju $C(x)$ navedeni su u narednoj tabeli:

n	**2**	**3**	**6**	**7**	**9**	**18**	**54**	**...**	**537099606**	**670617279**	**1341234558**
$D(n)$	**1**	**7**	**8**	**16**	**19**	**20**	**112**	**...**	**965**	**986**	**987**

Posledica: Lako je predvideti sve moguće parne rekordere u odnosu na potpuno vreme zaustavljanja i maksimum. Time se isključuju svi parne brojevi i dobija faktor ubrzanja 2. Dodatno, parni brojevi uvek imaju vreme zaustavljanja 1, zbog deljenja sa 2 u prvoj iteraciji.

Lema 4.7: Za svako $n > 0$, $\sigma(2n) = 1$.

Posledica: Ako je $n > 2$ rekorder u odnosu na vreme zaustavljanja, n je neparno.

Tako pretraga za rekorderima u odnosu na vreme zaustavljanja isključuje parne brojeve. Dalja ušteda se postiže preskakanjem svih brojeva oblika $n = 4k + 1$, jer svaki takav broj ima vreme zaustavljanja 3, u slučaju funkcije $C(x)$, i 2 u slučaju funkcije $T(x)$. Nakon prve iteracije $T(n) = T(4k + 1) = 3(4k + 1) + 1/2 = 6k + 2$, dobija se paran broj i u sledećem koraku $T(6k + 2) = 3k + 1 < 4k + 1$. Time je dokazana sledeća lema i posledica.

Lema 4.8: Za svako $n > 0$, ako je $n = 4k + 1$, tada je $T(n)$ parno.

Posledica: Za svako $n > 1$, ako je $n = 4k + 1$, tada je $D(n) = 3$ i $\sigma(n) = 2$.

Ovim se postiže faktor ubrzanja od 2 u odnosu na skup neparnih brojeva, a 4 u odnosu na skup svih prirodnih brojeva (proverava se svaki četvrti). Još jedno prekidanje pretrage je moguće posmatranjem vrednosti na putanji i to u slučaju da je početni broj manji od nekog broja nakon m iteracija. Lema pomaže kod prekidanja pretrage za potpuno vreme zaustavljanja i maksimum. Sa druge strane, prestavlja kontradikciju u odnosu na definiciju a posteriori strategije za prekidanje pretrage u odnosu na vreme zaustavljanja,

Lema 4.9: Neka su dati k i j, tako da je $0 < j < k$. Ako postoji neko m, tako da je $C^m(j) = k$, tada k ne može biti rekorder u odnosu na potpuno vreme zaustavljanja ili maksimum.

Dokaz: Pošto su vrednosti iteracija za k isti kao i za j nakon m iteracija sledi $D(j) = m + D(k)$ i $Mx(j) \geq Mx(k)$. ■

Lema 4.10: Neka su dati k i j, tako da je $0 < j < k$. Ako postoji neko m takvo da je $T^{(m)}(j) = k$, tada k ne može biti rekorder za bilo koje od tri svojstva.

Specijalno, *a priori* prekidanje pretrage je moguće za sve brojeve oblika $n = 6k + 5$.

Lema 4.11: Neka je $n > 0$. Ako je n oblika $6k + 5$, tada važi $T((2k-1)/3) = n$, $C^{(2)}((2k-1)/3) = n$, i n ne može biti rekorder za bilo koje od tri svojstva.

Dokaz: Pretpostavimo da je $n > 0$ i $n = 6k + 5$. Tada važi:

$n = 6k + 5$, pa odatle $\frac{2n-1}{3} = 4k + 3$, n je neparan ceo broj i $T\left(\frac{2n-1}{3}\right) = \left(3 \cdot \left(\frac{2n-1}{3}\right) + 1\right)/2 = 6k + 5 = n$. Slično, $C\left(\frac{2n-1}{3}\right) = 3 \cdot \left(\frac{2n-1}{3}\right) + 1 = 12k + 10, C^2\left(\frac{2n-1}{3}\right) = \frac{12k+10}{2} = 6k + 5 = n$.

Dakle, broj n se nalazi na putanji za manji broj, $\frac{2n-1}{3}$, pa na osnovu prethodne leme ne može da bude rekorder. ■

Po modulu 12 otpada 9 na osnovu prethodnog tvrđenja, a ostaju 3, 7 i 11. Na osnovu prethodne leme otpada 11, pa ostaju 3 i 7, dva od 12. Tako se postiže ukupan faktor ubrzanja 6, a u odnosu na prethodno $3/2$ (od tri preostala slučaja odbacuje se jedan).

4.2.2 A posteriori prekidanje pretrage

Kod pretrage za rekorderima u odnosu na maksimum za funkciju $C(x)$, potrebno je kontrolisati da li trenutna vrednost raste kroz iteracije i da li se dostiže vrednost veća od trenutnog rekordera. Ove provere se vrše nakon neparnih koraka. Problem je što su jako skupe za *velike* brojeve i tako usporavaju pretragu, pa se provere moraju ograničiti na deo neparnih koraka. Taj deo je sve dok tekući broj ne dostigne vreme zaustavljanja. Ovakav zaključak je posledica naredne leme.

Lema 4.12: Neka je $n > 0$ rekorder u odnosu na maksimum. Ako za neko $m > 0, C^{(m)}(n) < n$, tada je $Mx(n) = max\{C^{(i)}(k) | 0 \leq i \leq m\}$.

Dokaz: Neka je $m > 0$ takvo da je $C^{(m)}(n) = k < n$. Iz $k < n$ sledi $Mx(j) < Mx(n)$, jer je n rekorder. Putanja za n je nakon ovog koraka ista kao za j, pa u narednim iteracijama se ne dostiže novi rekorder. ■

Pretpostavka za prethodnu lemu je da li je početno n rekorder. Ukoliko to nije slučaj je moguće da se maksimum na putanji dobije nakon pada ispod početne vrednosti.

Primer 4.6: Broj 55 dostiže vrednost 376 pre nego što prvi put padne ispod 55 (na 47). Nakon toga dostiže maksimum (9232). Dakle, neophodan uslov da bi lema važila je da n bude rekorder.

Nakon što vrednost iteracija padne ispod inicijalnog n, moguće je primeniti a posteriori prekidanje pretrage za traženja rekordera u odnosu na potpuno vreme zaustavljanjaje na osnovu leme koja procenjuje maksimalni broj iteracija potreban da se izvrši provera (prvi put padne ispod početne vrednosti ili dostigne 1).

Lema 4.13: Za svako $n > 0$, ako je $C^{(m)}(n) = k \leq j$, gde je j rekorder u odnosu na potpuno vreme zaustavljanja, tada je $D(n) \leq m + D(j)$.

Dokaz: Ako je $n = j$, tada je $D(n) = m + D(j)$. U slučaju $n < j$ sledi $D(n) < D(j)$, na osnovu tvrđenja da je j rekorder. ■

Lema 4.13 važi i za funkciju T i σ_∞.

U programskoj realizaciji, prethodna lema se primenjuje na sledeći način. Za neko n, nakon parnog koraka, nalazi se prvi rekorder j veći od trenutne vrednosti iteracije (ako postoji) i koristi lema za procenu broja koraka. Ukoliko je uslov tačan (broj trenutne iteracije + $D(j) < D(max_recorder$[16]$)$), prekida se pretraga i isključuje n kao potencijalni rekorder.

Prilikom iteracija $T(x)$, polazeći od nekog n, ako se dođe do parnog koraka koji daje vrednost manju od tekućeg rekordera u odnosu na σ_∞, tada se koristi navedena lema da se ograniči maksimalna vrednost za $\sigma_\infty(n)$. Važnost leme se ogleda u mogućnosti a posteriori prekida pretrage za rekorderima u odnosu na potpuno vreme zaustavljanja nakon prosečno konstantnog broja koraka. Ukoliko su poznati svi rekorderi do nekog j, tada se u intervalu $[j + 1, 2j]$ pretragu možemo prekinuti nakon prosečno 15 koraka, nezavisno od j.

Imajući u vidu činjenicu da vrednost $\sigma_\infty(n)$ raste logaritamski u zavisnosti od n, ova strategija ima efekta samo kada je za n potrebno više od 15 koraka za proveru. Osim toga, kako sa povećanjem n pretraga ide u beskonačnost, povećava se i prosečna vrednost koraka.

I uz sve navedene strategije za smanjenje pretrage za rekorderima po bilo kojoj od statistika, ostaje mnogo putanja koje (bar delimično) treba ispitati. Stoga je potreban efikasan algoritam za proveru datih putanja. Na slici 12 je preudo-kod algoritma za izračunavanje vrednosti iteracija funkcije $C(x)$, koji ne pamti vrednosti na putanji, već samo vrednosti potrebne za statistiku. U narednom delu biće reči o ekvivalentnom algoritmu koji se izvršava u kraćem vremenu.

[16] Misli se na potpuno vreme zaustavljanja za trenutno najvećeg rekordera.

```
//ulaz: BigInteger n > 0
//izlaz: maksimalna vrednost putanje
//i potpuno vreme zaustavljanja za n
int potpunoVremeZaustavljanja = 0;
BigInteger max = n;
while(n!=1)
{
    if(n.IsEven)
        n/=2;
    else
    {
        n=3*n+1;
        max = BigInteger.Max(max, n);
    }
    potpunoVremeZaustavljanja ++;
```

Slika 12 Algoritam za nalaženje maksimuma na putanji i potpunog vremena zaustavljanja za ulazno n

4.2.3 Ubrzavanje iteracija

Deljenje sa 2 se najbolje implementira kao pomeranje binarne reprezentacije broja za jedno mesto udesno. Kako je pomeranje za više pozicija približno brzo kao i za jednu poziciju, ideja je da se umesto parnog koraka broj pomera za onoliko mesta koliko je potrebno da se dobije neparan broj kao tekuća vrednost iteracije. Prednost primene pomeranja je što se ne mora proveravati da li je tekuća vrednost iteracije paran ili neparan broj. Pseudo-kod ovakvog algoritma je dat na slici 13. Provera parnosti je realizovana u *shiftNum* metodu (slika 14).

```
//ulaz: neparno n > 0
while(n!=1)
{
    n = 3*n + 1;
    //proveravamo maksimum
    max = BigInteger.Max(max, n);
    //šiftujemo
    n = shiftNum(n, out i);
    //računamo vreme zaustavljanja
    potpunoVremeZaustavljanja += i+1;
    i = 0;
}
```

Slika 13 Algoritam koji koristi pomoćnu funkciju za šiftovanje broja određeni broj mesta

```
public static BigInteger shiftNum(BigInteger n, out int k)
{
    k=0;
    while(n.IsEven)
    {
        n = n >> 1;
        k++;
    }
    return n;
}
```

Slika 14 Pomoćna funkcija za pomeranje

Prednost ove varijante algoritma je što se nakon poslednje naredbe može primeniti a posteriori strategija. Provere se vrše kada je n najniže moguće u odnosu na datu iteraciju, pre nego što ponovo prođe neparni korak, što znači da se u proseku potroši manje vremena na provere.

Još efikasniji algoritam uzima veće korake i obavlja više iteracija odjednom. U odnosu na algoritam sa slike 12 je nekoliko puta brži. Umesto da se gleda poslednji bit tekućeg broja i izvršava odgovarajući korak, ovde je pristup drugačiji. Za određeno n posmatra se poslednjih m bitova i na osnovu njih zna se raspored parnih i neparnih koraka u narednih m iteracija. Umesto $k+m$ iteracija funkcije $C(x)$ izračunava se vrednost polinoma jednaka rezultatu posle $k+m$ iteracija. Za neko x formira *standardni polinom*, tj. polinom oblika $\frac{3^k x+z}{2^m}$, koji prestavlja vrednost $k+m$ iteracija (k neparnih i m parnih koraka).

Primer 3.6: Za $x = 11$, prvih 5 vrednosti iteracija su: $34, 17, 52, 26, 13$, pošto su parnosti redom $1,0,1,0,1$. Rezultat ovih iteracija dobija se primenom na x polinoma:

$$\frac{1}{2}\left(\frac{1}{2}\left(3\left(\frac{1}{2}(3x+1)\right)+1\right)\right) = \frac{3^2x+5}{2^3}$$

Polinom odgovara vrednosti izvršavanja dva neparna i tri parna koraka.

4.3 Sužavanje prostora pretrage za maksimumima u odnosu na σ i σ_∞

U prethodnoj tački je pokazano kako se može ubrzati osnovni algoritam za traženje rekordera. Provera validnost hipoteze 3x+1 zahteva pokazivanje da je svaka od iteracija $T^{(k)}(n)$ ograničena za svako $n > 0$. Jedan od pristupa je pokazati da za svako n postoji k tako da je $T^{(k)}(n) < n$, što je ekvivalentno nalaženju vremena zaustavljanja.

Hipoteza 2.4 govori da bi za neko n važila pretpostavka, dovoljno je iterirati sve dok se ne dođe do vremena zaustavljanja. U ovom delu se pokazuje kako suziti i ubrzati pretragu za pronalaženje rekordera u odnosu na vreme zaustavljanja i potpuno vreme zaustavljanja. Zbog veze između funkcija $C(x)$ i $T(x)$, ovde se iznose činjenice vezane za funkciju $T(x)$, ali se podrazumeva da slična tvrđenja važe i za funkciju $C(x)$.

Ono od čega se kreće jeste lema 3.1, tj. da je na osnovu zadnjih k cifara iz binarne reprezentacije ulaznog n moguće pretpostaviti kroz koje grane će $T(n)$ prolaziti u toku izračunavanja $T^{(k)}(n)$.

Dalje, neka je $n_0 = 2^k n_k + m_k$, gde je $n_k = \left\lfloor \frac{n_0}{2^k} \right\rfloor$, $m_k = n_0 mod 2^k$. Tada važi

Svojstvo 1. Opšta forma za $T^{(k)}(n)$ je

$$T^{(k)}(n_0) = 3^{O(n_0)_k} n_k + T^{(k)}(m_k), \quad k \geq 0 \qquad \text{S 1}$$

gde je $O(n_0)_k$ broj neparnih koraka tokom k iteracija.

Dokaz: Kao što je pokazano, tvrđenje je tačno za $k = 0, 1$ i 2. Dokaz u opštem slučaju se izvodi indukcijom, uvođenjem n_{k+1}, m_{k+1} i formula $n_k = 2n_{k+1} + r_k$, $m_{k+1} = r_k 2^k + m_k$, gde je r_k poslednja cifra u binarnom zapisu n_k. Zamenom u izrazu S 1 dobija se

$$T^{(k)}(n_0) = 2 \cdot 3^{O(n_0)_k} n_{k+1} + T^{(k)}(m_{k+1}), \quad k \geq 0 \qquad 4.1$$

gde je

$$O(n_0)_{k+1} = O(m_{k+1})_{k+1} = \begin{cases} O(n_0)_k & za\ parno\ T^{(k)}(m_{k+1}), \\ O(n_0)_k + 1 & za\ neparno\ T^{(k)}(m_{k+1}). \end{cases} \blacksquare$$

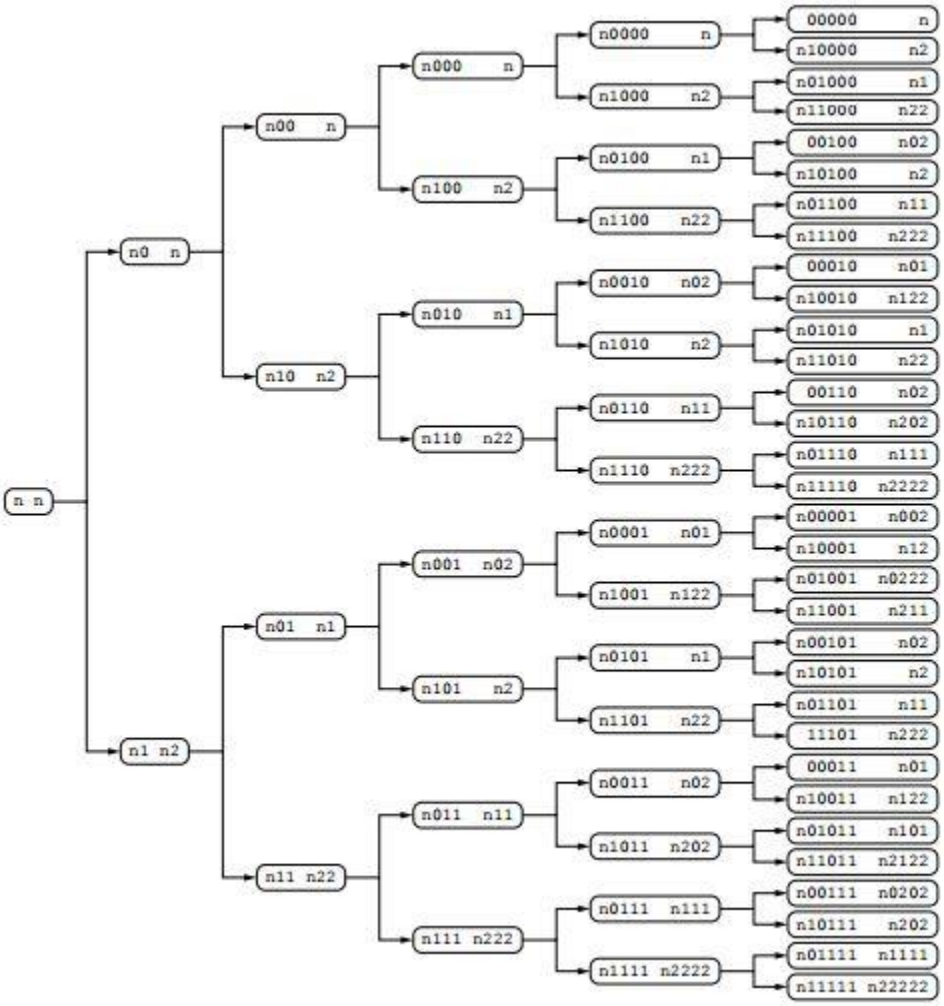
Svojstvo 2. Za $k \geq 0$ važi

$$T^{(k)}(m_k) < O(m_k)_k. \qquad \text{S 2}$$

Ovo nam govori da $T^{(k)}(m_k)$ nema više od $O(m_k)_k$ cifara kada se predstavi u sistemu sa osnovom 3. Dokaz u opštem slučaju se izvodi indukcijom. Na osnovu S 2 za svako k važi

$$T^{(k)}(m_{k+1}) = 3^{O(m_k)_k} r_k + T^{(k)}(m_k) < 2 \cdot 3^{O(m_k)_k}$$

Ovim, uz S 1 i S 2 se dobija da je predstavljanje $T^{(k)}(n_0)$ u sistemu sa osnovom 3 u potpunosti određeno sa: zadnjih $O(m_k)_k$ cifara je dato sa $T^{(k)}(m_k)$, a ostatak sa n_k. Izračunavanje $T^{(k)}(n_0)$ za n_0 može se prikazati u obliku binarnog stabla. Deo izračunavanja, za $k \leq 5$ je dato na Slika 15. Iako redosled vrednosti za $T^{(k)}(n_0)$ deluje nasumično, mogu se uočiti neka pravila. Naredna svojstva ukazuju da za strukturu iteracija funkcije $T(x)$ postoje određene pravilnosti koje uvek važe i pomažu kod ubrzanja pretrage. Tako, na osnovu svojstva 1 i svojstva 2 se dolazi do pravila za predstavljanje bilo koje iteracije u intervalu od 0 do k.



Slika 15 Rezultat iteracija T(n) za prvih k iteracija, 0 ≤ k ≤ 5. k predstavlja dubinu čvora. Unutar svakog čvora (ovalno polje) broj sa leve strane je binarna reprezentacija za n0, a desno vrednost k-te iteracije u sistemu sa osnovom 3. Zbog preglednosti, nk je predstavljeno kao n

Svojstvo 3. Za $0 \leq p \leq k$

$$T^{(p)}(n_0) = 2^{k-p} \cdot 3^{O(m_k)_k} n_k + T^{(p)}(m_k) \qquad \text{S 3}$$

Svojstvo 4. Za $1 \leq k \leq 40$ i $1 < m_k < 2^k$, ako se u rastućem poretku sortiraju brojevi

$$2^{k-p} \cdot 3^{O(m_k)_k}, \quad 0 \le p \le min\{k, \sigma(m_k)\},$$

tada su u rastućem poretku sortitani i brojevi

$$T^{(p)}(m_k), 0 \le p \le min\{k, \sigma(m_k)\} \qquad \text{S 4}$$

Svojstvo 4 je provereno u [9] i ima dve bitne posledice:

Svojstvo 5. Ako je $1 \le k \le 40$, $1 < m_k < 2^k$, i $n_k \ge 0$, tada je uređenje brojeva $T^{(p)}(m_k)$ isto kao i uređenje brojeva $T^{(p)}(2^p \cdot n_k + m_k)$ za $0 \le p \le min\{k, \sigma(m_k)\}$.

Odavde dalje sledi da se maksimum (minimum) na putanji dešava u istoj iteraciji za svako $1 < m_k < 2^k$ i $n_k \ge 0$. Posledica svojstva 5 ima ulogu kod ubrzanja pretrage za rekorderima u odnosu na vreme zaustavljanja.

Svojstvo 6. Neka je $\sigma(m_k) = p \le k \le 40$ i $1 < m_k < 2^k$. Tada je vreme zaustavljanja na putanji $2^k \cdot n_k + m_k$, za svako $n_k \ge 0$ jednako p.

Posledica: Za svako $n > 0$ važi $\sigma(2n) = 1, \sigma(4n + 1) = 2$.[17]

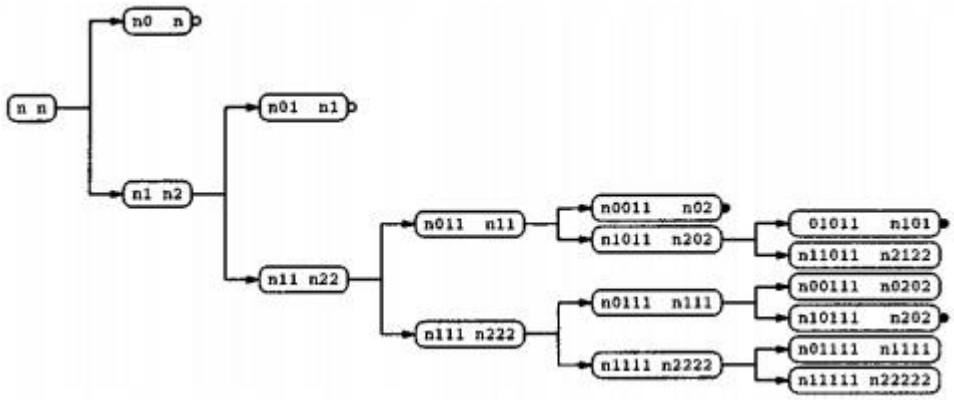
Na osnovu toga sledi da će pretraga za rekorderima u odnosu na vreme zaustavljanja ići kroz niz brojeva oblika $4k + 3$.

4.3.1 Postupak (strategija) pretrage

Svojstvo 6 i njegova posledica su ključni za postupak pretrage jer omogućavaju odsecanje čvorova stabla sa slike 15 u toku pretrage. Stablo se razgranava, i ima dubinu k. Mogu se zanemariti potomci svih čvorova koji zadovoljavaju uslov $T^{(k)}(n_k) \le n_k$. Svi čvorovi rezultujućeg, potkresanog stabla prikazanog na slici 16 , pripadaju jednom od tri slučaja:

i. u poslednjoj iteraciji dobija se vrednost manja od inicijalne, tj. dolazi se do vremena zaustavljanja. Ovi čvorovi su predstavljeni zatamnjenim krugovima i ne moraju se dalje deliti.
ii. u poslednjoj iteraciji na dubini k, za $n_k = 0$, dobija se vrednost jednaka inicijalnoj. U ovom slučaju dolazi do ciklusa. Ovi čvorovi su označeni praznim krugom i za m_k ponašaju se kao čvorovi u slučaju (i). Stoga, oni se ne moraju dalje deliti.
iii. u toku pretrage je dostignuta maksimalna dubina. Vreme zaustavljanja za sve vrednosti na putanji inicijalne vrednosti je veće od dubine drveta.

[17] Ovo sledi iz leme 4.8.



Slika 16 Varijanta podkresanog stabla sa slike 8 nakon primene svojstava 6 i 7

Čvorovi u slučaju (i) i (ii) se nazivaju zatvoreni čvorovi, a ostali se nazivaju otvoreni čvorovi. Broj otvorenih i zatvorenih čvorova na određenoj dubini k označavamo sa $n_o(k)$, odnosno $n_c(k)$. U narednoj tabeli je predstavljen broj otvorenih i zatvorenih čvorova, kao i njihov procenat u odnosu na ukupan broj čvorova za $k \leq 10$.

k	$n_o(k)$	$n_c(k)$	$n_o(k)/2^k$	$n_c(k)/2^k$
0	1	0	1,000000	0,000000
1	1	1	0,500000	0,500000
2	1	1	0,250000	0,250000
3	2	0	0,250000	0,000000
4	3	1	0,187500	0,062500
5	4	2	0,125000	0,062500
6	8	0	0,125000	0,000000
7	13	3	0,101563	0,023438
8	19	7	0,074219	0,027344
9	38	0	0,074219	0,000000
10	64	12	0,062500	0,011719

U dodatku A su dati rezultati za $k \leq 38$ dobijeni na osnovu metode *OpenCloseNodes(int k)* u okviru pratećeg programskog rešenja. Rezultati se poklapaju sa rezultatima iz rada [9]. Uz broj i udeo otvorenih/zatvorenih čvorova date su i rekurentne formule za izračunavanje broja čvorova u odnosu na dubinu k.

Ideja ubrzanja pretrage se bazira na strategiji da se u obzir uzimaju samo čvorovi koji inicijalno pripadaju slučaju (iii). Da bi se smanjio broj čvorova i ubrzala pretraga, treba smanjiti njihov udeo, što znači da maksimalna dubina drveta treba da bude što veća. Programsko rešenje koristi dubinu drveta 38**.** To je maksimum za konstrukciju potkresanog drveta sa slike 16 , bez da se dobije izuzetak o prekoračenju memorijskog prostora. Zbog broja čvorova na maksimalnoj dubini stabla

(1 934 757 182) nepraktično je čuvati ih i sortirati u odnosu na klasu kongruencije kojoj pripadaju[18]. To čini sekvencijalnu pretragu neizvodljivom.

Za efektivnu nesekvencijalnu pretragu se primenjuje sledeći pristup: program održava dve liste, kandidate za rekordere u odnosu na maksimum i vreme zaustavljanja, u odnosu na početnu vrednost putanje. Ušteda u održavanju navedenih listi se postiže analiziranjem 2^{10} uzastopnih inicijalnih vrednosti koje pripadaju svakoj od klasa kongruencije za čvorove tipa *(iii)*. Pošto se 2^{10} uzastopnih vrednosti testira u rastućem poretku, jedan prolaz kroz liste je dovoljan za njihovo ažuriranje.

Svaki čvor tipa *(iii)* ima tri pridružena broja: m_k; $T^{(k)}(m_k) > m_k$; i $y(k; m_k)$. Na osnovu svojstva 1 pridružene brojeve možemo iskoristiti i naći vrednost k-te iteracije, a odatle i izračunati izračunati $T^{(k)}(n_k \cdot 2^k + m_k)$ i tako preskočiti prvih k iteracija. Pošto su 2^{10} vrednosti za n_k uzastopne, svojstvo 1 primenjujemo na prvu, a za ostalih $2^{10} - 1$ vrednosti potrebno je manje zahtevno izračunavanje.

Opisan pristup ima dve mane: prva, rekorderi u odnosu na vreme zaustavljanja za koje je $\sigma(n) < 41$ nisu obuhvaćeni. Kako je $\sigma(27) = 59 > 40$, izostavljene rekordere je lako naći pretragom za inicijalne vrednosti između 2 i 27. Drugi nedostatak se tiče rekordera u odnosu na maksimum. Nalaženje maksimuma u prvih 40 iteracija za svaki od čvorova tipa (iii) je skupo. Ali, pošto je ta informacija relevantna samo za male vrednosti, može se preskočiti. Time je moguće dobiti netačne rekordere u odnosu na maksimum, ukoliko se maksimum javi pre četrdesete iteracije. Dobijanje izostavljenih i uklanjanje pogrešnih rekordera u odnosu na maksimum je moguće proverama i tehnikama opisanim u radu [9]. Dodatnim proverama se ispostavlja da je najveći broj kod kog se maksimum javlja pre četrdesete iteracije 704 511, i na osnovu toga vrši se provera u okviru metoda za pretragu rekordera u odnosu na maksimum.

[18] Za dubinu k klasa kongruencije je 2^k

5 Tehnička specifikacija softverskog rešenja

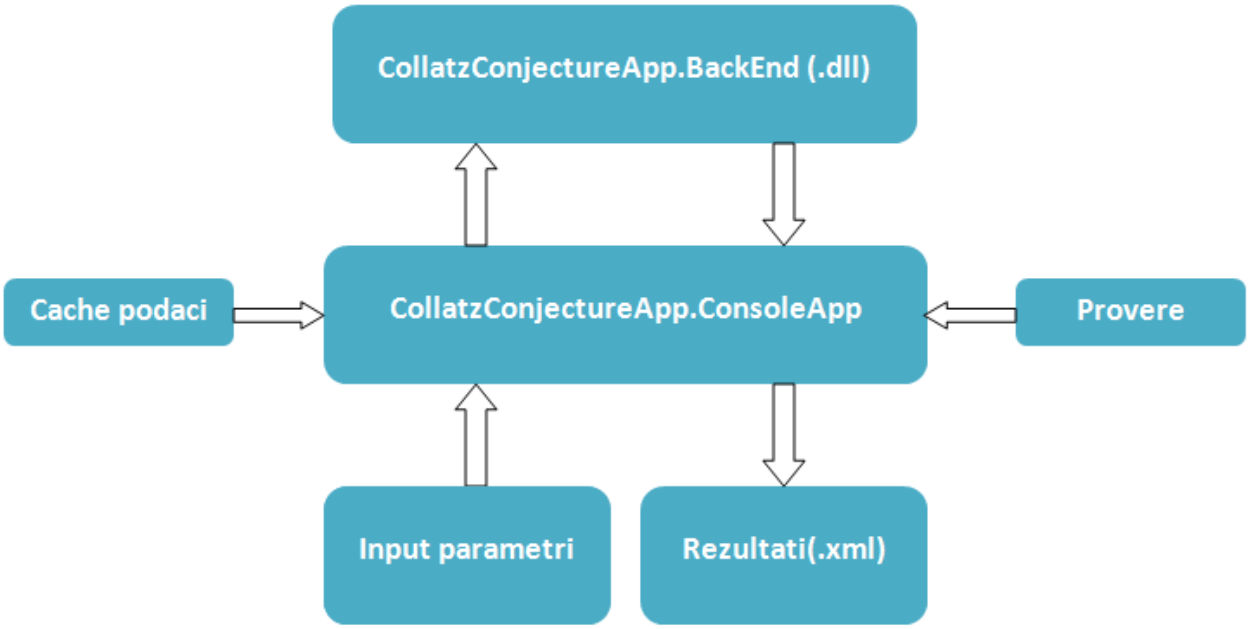
Pretraga za rekorderima u odnosu na bitna svojstva i testiranje validnosti Kolacovog problema su realizovani kroz prateće softversko rešenje. Na kraju rada, u dodatku B date su tabele sa rezultatima rada programa.

5.1 Razvojno okruženje i objektni model

Projekat *CollatzConjectureApp* je napisan pod razvojnim okruženjem Microsoft Visual Studio[19] 2012 i implementiran u programskom jeziku *C#*. Realizovan je kao projekat koji se sastoji iz dva dela i to:

- *CollatzConjectureApp.BackEnd* – pozadinska logika i klase koje kontrolišu izračunavanja, pretrage, snimanje i učitavanje podataka, proveru rezultata i ispis na standardni izlaz. Urađen je kao dinamička biblioteka klasa i poziva se preko reference u konzolnoj aplikaciji.
- *CollatzConjectureApp.ConsoleClient* – konzolna aplikacija uz pomoć koje korisnik pokreće i podešava dostupne funkcije implementirane u *CollatzConjectureApp.BackEnd* projektu.

Nakon provere i izračunavanja izlazni podaci se eksportuju u okviru XML fajla sa predefinisanim tagovima. Podaci predstavljaju kolekciju objekata, gde je svaki član kolekcije jedan od rezultata. Razlog za to je kasnija lakša obrada podataka za slučaj ponovnog korišćenja ili potrebe za pretvaranjem[20] u nove tipove ili strukture. Dodatno, postoji dosta biblioteka za deserijalizaciju xml fajlova. Model arhitekture projekta je dat na slici 17.



[19] https://www.visualstudio.com/
[20] kastovanje(eng. cast)

Slika 17 Model arhitekture softverskog rešenja

U nastavku je dat pregled najbitnijih klasa i metoda uz objašnjenje realizacije pretrage i provere validnosti, popis realizovanih i mogućih funkcionalnosti i uputstvo za korišćenje aplikacije.

5.2 Pregled bitnih klasa i metoda

Zbog jednostavnosti, sve provere i pretrage se odvijaju nad iteracijama funkcije $T(x)$. U [3] se mogu naći liste rekordera ako se koristi funkcija $C(x)$, pri čemu se krajnji rezultat u smislu validnosti i liste rekordera poklapaju. Zbog pojave "velikih" brojeva, i ograničenosti intervala za osnovne brojevne tipove podataka, koristi se klasa *BigInteger*[21]. Objekat *BigInteger* klase predstavlja ceo broj čija vrednost nema gornju ili donji granicu, a ima podršku za sve potrebne operacije za potrebe izračunavanja. Klase koje implemetiraju izračunavanja i strategije pretrage se nalaze u klasama projekta *CollatzConjectureApp.BackEnd*. To su:

- *CollatzNumber* – provera validnosti problema 3x+1 za pozitivan ceo broj x. Pored toga, u ovoj klasi je implementirano izračunavanje svih bitnijih veličina vezanih za x u kontekstu problema 3x+1. Pored standardnih, vreme zaustavljanja, potpuno vreme zaustavljanja, maksimum i klasa pripadnosti, dobija se informacija o elementima na putanji, odnos broja parnih i neparnih koraka, i vektor parnosti. Kompletno izračunavanje prati predefinisan prikaz rezultata na standardni izlaz izmenom[22] baznog metoda *ToString()*[23]

- *HelpManager* – ispis bitnih informacija o korišćenju programa i objašnjenje formata unosa za svaku od implementiranih funkcionalnosti.

- *Peak* – klasa čiji objekat predstavlja jedan rekord podatak[24]. Objekat se sastoji iz sledećih polja:

- *PeakN*(*BigInteger*)– rekorder (početna vrednost od koje kreće provera za rekordera)
- *PeakValBigInt*(*BigInteger*)– vrednost koju rekorder dostiže kod pretrage u kategoriji maximum. Podrazumevana[25] vrednost je 0.
- *PeakValInt*(*int*)– vrednost koju rekorder dostiže kod pretrage u kategoriji vreme zaustavljanja, potpuno vreme zaustavljanja, rekord klase. Podrazumevana vrednost je 0.
- *PeakNString*, *PeakValString* – string reprezentacije rekordera i vrednosti za rekordera. Razlog za uvođenje ova dva polja je izostanak podrške za serijalizaciju objekata *BigInteger* klase. Kod deserijalizacije, vrše se dodatne provere pri parsiranju vrednosti iz string reprezentacije u *BigInteger* ili *int*, zavisno od slučaja.

Pored polja, u klasi postoje i dva konstruktora, bilo da se radi o rekorderima čija vrednost je tipa *BigInteger* ili *int*.

[21] https://msdn.microsoft.com/en-us/library/system.numerics.biginteger(v=vs.110).aspx
[22] overidde
[23] https://msdn.microsoft.com/en-us/library/system.object.tostring(v=vs.110).aspx
[24] U literaturi *peak*, označava termin za rekordera, otuda i naziv klase.
[25] eng. default

- *CacheDataManager* – inicijalizacija, čuvanje i pristup međurezultatima ide preko ove klase. Međurezultati su predstavljeni preko niza slogova. Za ulaz k generiše se 2^k slogova. Svaki slog predstavlja strukturu *TYPair*. Struktura se sastoji od dva polja:

- TkVal – vrednost nakon k iteracija za pozitivan ceo broj x (x je indeks sloga u nizu)
- ykVal – broj neparnih koraka u prvih k iteracija.

Ideja koja se koristi kod formiranja međurezultata opisana u svojstvu 1. Međurezultati se nakon inicijalizacije koriste kod pretrage i služe da se preskoči prvih k iteracija ukoliko su ispunjeni određeni uslovi. U konkretnom slučaju to znači da ako je pretraga prešla određenu granicu, sigurno je da se neki od rekordera neće pojaviti pre k-te iteracije.

- *PeakDataManager* – kontrola liste rekordera, ispis na standarni izlaz za neko n, record podatak ili listu rekordera. Pored toga, tu je učitavanje i snimanje rezultata pretrage i kontrola unosa za direktorijum u koji se snimaju rezultati. U ovoj klasi se nalazi polje čija vrednost je konstantna, ne može se menjati (eng. *read-only*), *defaultPath* i predstavlja podrazumevanu putanju za snimanje ukoliko korisnik to ne uradi.

- *PeakSearchManager* – klasa u kojoj su smeštene pretrage, provera uslova izvođenja pretrage i pomoćne metode koje se koriste kod svake od pretraga. U okviru polja nalaze se objekti klasa *PeakCacheManager* i *PeakDataManager* uz pomoć kojih se rezultati u toku pretrage povezuju sa međurezultatima i metodama za učitavanje, snimanje i prikaz rekordera. Ova verzija programa pokriva tri vrste pretrage za rekorderima i to:

- pretraga u odnosu na vreme zaustavljanja. Pored standardnih ubrzanja (upotreba bitskih operacija umesto množenja i deljenja, primena leme 4.11 i 17). Od pomoćnih metoda tu su metoda za izračunavanje vremena zaustavljanja (*StoppingTimeEval*) i provera da li je neko x rekorder (*StoppingTime_CheckN*)
- pretraga u odnosu na potpuno vreme zaustavljanja. Pretraga koristi međurezultate i preskače prvih k iteracija. U kombinaciji sa pomoćnim metodom *TotalStoppingTimeEval* izračunava se potpuno vreme zaustavljanja, a u metodu *TotalStoppingTime_CheckN* vrši se provera da li je tekući broj x rekorder.
- pretraga u odnosu na maksimum. Slično kao kod prethodne pretrage, i ovde se koriste međurezultati preko objekta klase *PeakCacheManager* i nakon izračunavanja maksimuma vrše se provere da li je neko x rekorder. Izračunavanje maksimuma prati optimizovan postupak opisan u radu [9].

Rezultati za svaku od navedenih pretraga dati su u dodatku B. Za verifikaciju rezultata korišćeni su rezultati iz radova [8], [9] i iz arhive stranice [10].

5.3 Implementirane funkcionalnosti i moguća unapređenja

Cilj projekta *CollatzConjectureApp* je provera validnosti problema 3x+1 i rezultata pretrage. Rezultate pretrage je moguće verifikovati upoređivanjem rezultata dobijenih drugim softverskim implementacijama (kao u ovom slučaju) ili pozivanjem na prateće matematičke hipoteze i leme, što izlazi iz okvira ovog rada. Za više informacija mogu se pogledati radovi [11], [9] i [7]. Validnost problema je moguće dokazati na nivou provere za neko x, računanjem putanje ili određenih svojstava. Na višem nivou apstrakcije problema moguće je analizirati klase brojeva (npr. znamo da su za pretragu rekordera u odnosu na vreme zaustavljanja "zanimljivi" samo brojevi oblika $4k + 3$). Jedan od radova koji se bavi ovom problematikom je [12].

Kompleksnost problema 3x+1 i brojni pristupi rešavanju daju povod za implementaciju najrazličitijih algoritama. Kao što je izloženo u poglavlju 3, problem 3x+1 se može posmatrati preko grafovskih algoritama, pretragom slabla prethodnika, posmatranjem binarnih vektora ili dijagrama stanja prelaza. Za ovu verziju projekta, implemetirane su sledeće funkcionalnosti:

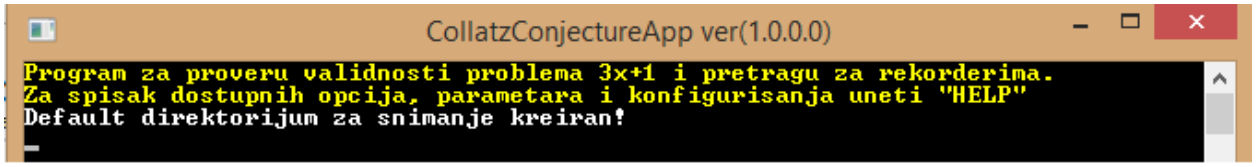
- provera validnosti problema 3x+1 za neko x
- optimizovana pretraga za rekorderima u odnosu na vreme zaustavljanja
- pretraga za rekorderima u odnosu na potpuno vreme zaustavljanja korišćenjem međurezultata
- pretraga za rekorderima u odnosu na maksimum korišćenjem međurezultata
- računanje broja otvorenih i zatvorenih čvorova do dubine 38
- prikaz, učitavanje i snimanje rekordera
- help sistem za rad sa programom
- podešavanja za putanju snimanja rezultata
- provere i sistem kontrole greške za sve navedene funkcionalnosti.

U daljem razvoju projekat bi mogao da uključi sledeće dodatke i funkcionalnosti:

- pretraga za rekorderima u odnosu na klasu kojoj pripadaju
- primena daljih optimizacija radi ubrzanja pretrage
- provera validnosti rekordera u toku same pretrage
- multi-thread programski režim (mogućnost pauziranja i nastavljanja pretrage, puštanje više pretraga odjednom)
- provera validnosti hipoteze za klasu brojeva
- GUI klijent sa user-friendly interfejsom (WinForms ili WPF aplikacija)
- unapređenje algoritma za otvorene i zatvorene čvorove, korišćenjem algoritma pretrage stabla u dubinu umesto u širinu.

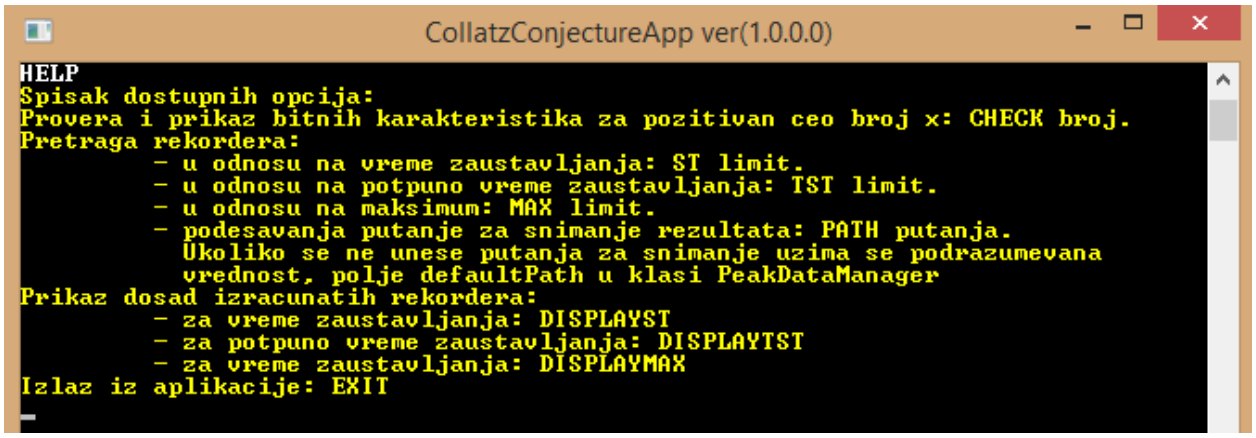
5.4 Kratko uputsvo za korišćenje aplikacije

Po pokretanju aplikacije korisniku se prikazuje početni ekran sa obaveštenjem da li su početna podešavanja ispravna. Sva obaveštenja imaju određenu boju kod prikaza zavisno od toga da li su informacija (plava), greška (crvena), tok programa (bela), pomoćna naredba (žuta) ili upozorenje (zelena). U slucaju da se neki od parametara ne unese u dobrom formatu dobija se odgovarajuća poruka o grešci.



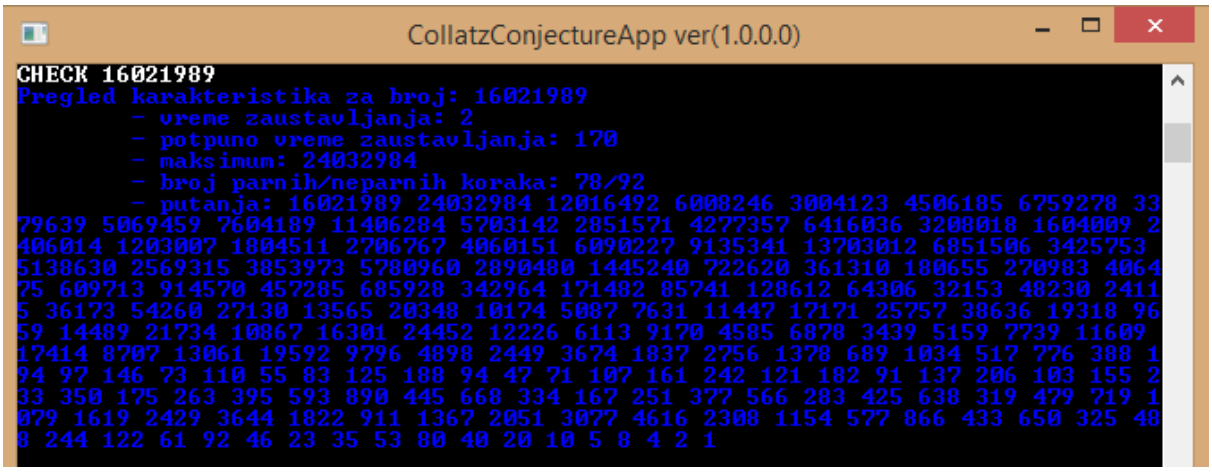
Slika 18 Početni prozor

Spisak svih komandi se dobija unosom komande **HELP**.



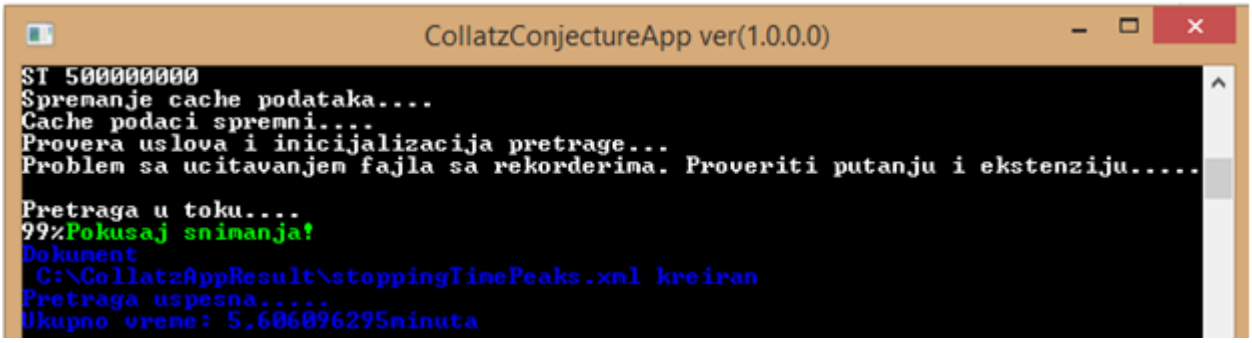
Slika 19 HELP komanda

Provera validnosti problema 3x+1 za neko x se vrši unosom komande **CHECK** i argumenta x.

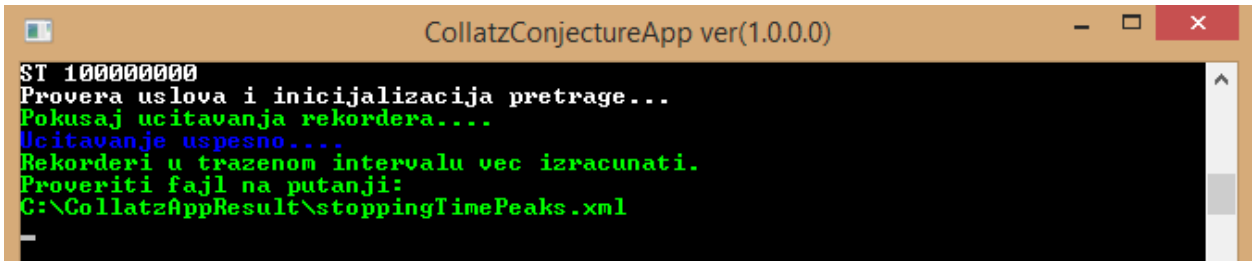


Slika 20 Provera validnosti problema za neko x

Startovanje neke od pretrage zahteva unos određene komande i granice intervala za pretragu. Ukoliko je pretraga u datom intervalu već obavljena, korisnik dobija obaveštenje. Tok pretrage je prikazan procentom pokrivenosti intervala pretrage.

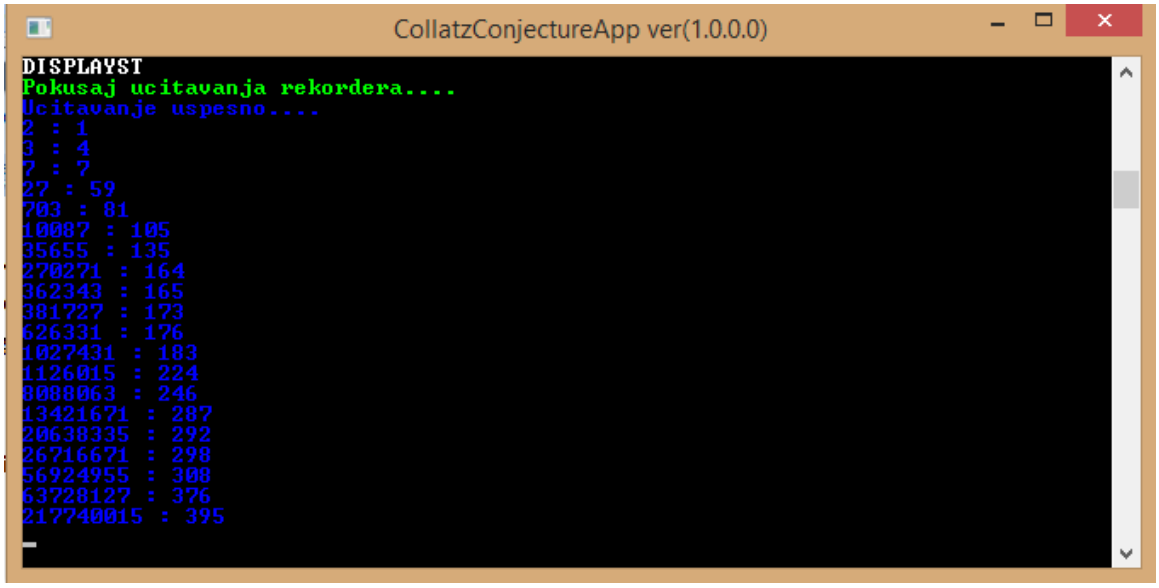


Slika 21 Proces pretrage



Slika 22 Obaveštenje ukoliko je pretraga već obavljena

Prikaz rezultata za neku od obavljenih pretraga se realizuje komandom za prikaz u format **DISPLAY**kriterijum, gde je kriterijum kategorija nad kojom je pretraga obavljena. Ukoliko nije bilo obavljenih pretraga dobija se odgovarajuća poruka o nedostatku rezultata.



Slika 23 Prikaz rezultata pretrage

Za izlaz iz aplikacije se koristi komanda **EXIT** uz standardnu close opciju.

Dodatak A

A.1 Otvoreni i zatvoreni čvorovi

Otvoreni čvor, u oznaci n_o je čvor na maksimalnoj dubini stabla pretrage koji ima naslednika(e) i nakon što je maksimalna dubina drveta dostignuta. Zatvoreni čvor, u oznaci n_c je čvor na nekoj dubini stabla pretrage u kom je pretraga prekinuta (nema potomke). I otvoreni i zatvoreni čvorovi na dubini k predstavljaju klasu kongruencije po modulu 2^k. Između njih postoji veza opisana formulama u narednom delu.

A.2. Veza između otvorenih i zatvorenih čvorova i rekurentne formule

Neka je $n_o(k,p), 0 \leq p \leq k$ broj otvorenih čvorova za koje je $O(m_k)_k = p$. Jasno je da je $n_o(0,0) = 1$, i $n_o(k) = \sum_{p=0}^{k} n_o(k,p)$. Kada dođe do podele otvorenog čvora na dubini $k-1, k > 0$, jedan od novodobijenih čvorova zadovoljava jednakost $O(m_k)_k = O(m_{k-1})_{k-1}$, a drugi $O(m_k)_k = O(m_{k-1})_{k-1} + 1$. Da je svaki čvor dobijen ovim postupkom otvoren, tada bi $n_o(k,p) = n_o(k-1,p-1) + n_o(k-1,p)$, sa pretpostavkom $n_o(k-1,-1) = n_o(k-1,k) = 0$, što bi značilo da $n_o(k,p)$ ima binomnu raspodelu. Međutim, kako svi generisani čvorovi nisu otvoreni, neki su i zatvoreni, i imajući u vidu svojstvo 4 (poglavlje 4.3), važi da se otvoreni čvorovi generišu u slučaju kada je $3^p > 2^k$. Tako za $k > 0$ imamo,

$$n_o(k,p) = \begin{cases} 0 & ako\ je\ 3^p < 2^k \\ n_o(k-1,p-1) + n_o(k-1,p) & ako\ je\ 3^p > 2^k. \end{cases}$$

Primenom istog postupak za generisanje zatvorenih čvorova dobija se:

$$n_c(k,p) = \begin{cases} n_o(k-1,p) & ako\ je\ 3^p < 2^k \\ 0 & ako\ je\ 3^p > 2^k. \end{cases}$$

Ako se zna $n_o(k,p)$ i $n_c(k,p)$, lako je izračunati $n_o(k)$ i $n_c(k)$. U tabeli 1 je dat broj otvorenih/zatvorenih čvorova za $k \leq 38$ i njihov udeo na određenoj dubini. Podaci su dobijeni kao izlazni rezultat metode *openClosedNodes* u okviru klase *TFunction* i poklapaju se sa rezultatima iz rada [9]**.** Za $k > 38$, pretraga se prekida usled memorijskog ograničenja intervala za *int* tip podataka.

k	$n_o(k)$	$n_c(k)$	$n_o(k)/2^k$	$n_c(k)/2^k$
0	1	0	1,000000	0,000000
1	1	1	0,500000	0,500000
2	1	1	0,250000	0,250000
3	2	0	0,250000	0,000000
4	3	1	0,187500	0,062500
5	4	2	0,125000	0,062500
6	8	0	0,125000	0,000000
7	13	3	0,101563	0,023438
8	19	7	0,074219	0,027344
9	38	0	0,074219	0,000000
10	64	12	0,062500	0,011719
11	128	0	0,062500	0,000000
12	226	30	0,055176	0,007324
13	367	85	0,044800	0,010376
14	734	0	0,044800	0,000000
15	1295	173	0,039520	0,005280
16	2114	476	0,032257	0,007263
17	4228	0	0,032257	0,000000
18	7495	961	0,028591	0,003666
19	14990	0	0,028591	0,000000
20	27328	2652	0,026062	0,002529
21	46611	8045	0,022226	0,003836
22	93222	0	0,022226	0,000000
23	168807	17637	0,020123	0,002102
24	286581	51033	0,017082	0,003042
25	573162	0	0,017082	0,000000
26	1037374	108950	0,015458	0,001623
27	1762293	312455	0,013130	0,002328
28	3524586	0	0,013130	0,000000
29	6385637	663535	0,011894	0,001236
30	12771274	0	0,011894	0,000000
31	23642078	1900470	0,011009	0,000885
32	41347483	5936673	0,009627	0,001382
33	82694966	0	0,009627	0,000000
34	151917636	13472296	0,008843	0,000784
35	263841377	39993895	0,007679	0,001164
36	527682754	0	0,007679	0,000000
37	967378591	87986917	0,007039	0,000640
38	1934757182	0	0,007039	0,000000

Tabela 1 Broj otvorenih i zatvorenih čvorova na dubini k

Dodatak B

B.1. Rezultati pretrage

U tabelama 2, 3 i 4 su prikazani rekorderi u odnosu na vreme zaustavljanja i maksimum u intervalu [1, $5*10^{11}$] i rekorderi u odnosu na potpuno vreme zaustavljanja u interval [1, $5*10^{10}$]. Dobijeni rezultati su rezultat rada aplikacije *CollatzConjectureApp.ConsoleApp* i identični su rezultatima iz radova [9], [8] i [4]. U navedenim radovima pokriven je veći interval (za [9] [1, $3*2^{53}$], za [8] [1, $5{,}6*10^{13}$], za [4] [1, 2^{60}]). Za [4] izračunavanja i dalje traju.

n	$\sigma(n)$
2	1
3	4
7	7
703	81
10 087	105
35 655	135
270 271	164
362 343	165
381 727	173
626 331	176
1 027 431	183
1 126 015	224
8 088 063	246
13 421 671	287
20 638 335	292
26 716 671	298
56 924 955	308
63 728 127	376
217 740 015	395
1 200 991 791	398
1 827 397 567	433
2 788 008 987	447
12 235 060 455	547

Tabela 2 Lista rekordera u odnosu na vreme zaustavljanja

n	$Mx(n)$
2	2
3	8
7	26
15	80
27	4 616
255	6 560
447	19 682
639	20 762
703	125 252
1 819	638 468
4 255	3 405 068
4 591	4 076 810
9 663	13 557 212
20 895	25 071 632
26 623	53 179 010
31 911	60 506 432
60 975	296 639 576
77 671	785 412 368
113 383	1 241 055 674
159 487	8 601 188 876
270 271	12 324 038 948
665 215	26 241 642 656
704 511	28 495 741 760
1 042 431	45 119 577 824
1 212 415	69 823 368 404
1 441 407	75 814 787 186
1 875 711	77 952 174 848
1 988 859	78 457 189 112
2 643 183	95 229 909 242
2 684 647	176 308 906 472
3 041 127	311 358 950 810
3 873 535	429 277 584 788
4 637 973	659 401 147 466
5 656 191	1 206 246 808 304
6 416 623	2 399 998 472 684
6 631 675	30 171 305 459 816
19 638 399	153 148 462 601 876
38 595 583	237 318 849 425 546
80 049 391	1 092 571 914 585 050
120 080 895	1 638 950 788 059 290
210 964 383	3 202 398 580 560 632
319 804 831	707 118 223 359 971 240
1 410 123 943	3 562 942 561 397 226 080
8 528 817 511	9 072 297 468 678 299 012

Tabela 3 Lista rekordera u odnosu na maksimum

12 327 819 503	10 361 199 457 202 525 864
23 035 537 407	34 419 078 320 774 113 520
45 871 962 271	41 170 824 451 011 417 002
51 739 336 447	57 319 808 570 806 999 220
59 152 641 055	75 749 082 531 195 100 772
59 436 135 663	102 808 194 685 920 926 084
70 141 259 775	210 483 556 894 194 914 852
77 566 362 559	458 306 514 538 433 899 928
110 243 094 271	686 226 824 783 134 190 180
204 430 613 247	707 630 396 504 827 495 544
231 913 730 799	1 095 171 911 941 437 256 778
272 025 660 543	10 974 241 817 835 208 981 874
446 559 217 279	19 766 638 455 389 030 190 536

Tabela 3 Nastavak

n	$\sigma_\infty(n)$
2	1
3	5
6	6
7	11
9	13
18	14
25	16
27	70
54	71
73	73
97	75
129	77
171	79
231	81
313	83
327	91
649	92
703	108
871	113
1 161	115
2 223	116
2 463	132
2 919	137
3 711	150
6 171	165
10 971	169
13 255	174
17 647	176
23 529	178
26 623	194
34 239	196
35 655	204
52 527	214
77 031	221
106 239	223
142 587	236
156 159	241
216 367	243
230 631	278
410 011	282
511 935	295
626 331	319
837 799	329
1 117 065	331

Tabela 4 Lista rekordera u odnosu na potpuno vreme zaustavljanja

1 501 353	333
1 723 519	349
2 298 025	351
3 064 033	353
3 542 887	366
3 732 423	374
5 649 499	384
6 649 279	416
8 400 511	429
11 200 681	431
14 934 241	433
15 733 191	441
31 466 382	442
36 791 535	466
63 728 127	592
127 456 254	593
169 941 673	595
226 588 897	597
268 549 803	602
537 099 606	603
670 617 279	616
1 341 234 558	617
1 412 987 847	625
1 647 652 263	630
2 610 744 987	656
4 578 853 915	679
4 890 328 815	706
9 780 657 630	707
12 212 032 815	720
12 235 060 455	739
13 371 194 527	755
17 828 259 369	757
31 694 683 323	761

Tabela 4 Nastavak

Literatura

[1] S. Ulam, Problems in Modern Mathematics, New York: John Wiley and Sons, 1964.

[2] Matthews, K. R and A. M. Watts, "A generalization of Hasse`s generalization of the Syracuse algorithm," *ACTA ARITHMETICA,* vol. XLIII, pp. 167-175, 1984.

[3] G. T. Leavens and M. Vermeulen, "3X+1 Search Programs," 1992.

[4] E. Roosendal, "On The 3x + 1 Problem - Eric Roosendaal home page," 2015. [Online]. Available: http://www.ericr.nl/wondrous/.

[5] J. C. Lagarias, The Ultimate Challenge: The 3x+1 Problem, 2011.

[6] P. Schorer, *Are We Near a Solution to the 3x+1 Problem?: A Discussion of a Promising,* Berkeley, CA, 2014.

[7] J. C. Lagarias, "The 3x+1 Problem: An Overview," *American Mathematical Society,* 2010.

[8] K. Conrow, "Proofs of binary and abstract predecessor tree structure," [Online]. Available: http://www-personal.ksu.edu/~kconrow/proofs.html.

[9] K. Conrow, "Representation of trajectories," [Online]. Available: http://www-personal.ksu.edu/~kconrow/cafeinst.html.

[10] T. O. E. Silva, "Maximum Excursion And Stopping Time Record-Holders For 3X+1 Problem: Computational Results," *Mathematics Of Computation,* vol. 68, pp. 371-384, 1999.

[11] J. C. Lagarias, "The 3x+1 Problem and its Generalizations," *The American Mathematical Montly,* pp. 3-23, 1985.

[12] E. Tzanis, "Collatz Conjecture - Properties and Algorithms," Computation And Reasoning Laboratory(Corelab), School of Electrical and Computer Engineering, National Tehnical University of Athens, Athens.

[13] L. Collatz, On the motivation and origin of the (3n + 1)- problem, vol. 3, Natural Science Edition, 1986, pp. 9-11.

[14] C. Everett, "Iteration of the number theoretic function f(2n) = n, f(2n + 1) = 3n + 2," *Advances In Math,* pp. 42-45, 1977.

[15] H. Hasse, *Unsolved Problems in Elementary Number Theory,* 1975.