SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 2856

OTKRIVANJE FIZIKALNIH IZRAZA GENETSKIM PROGRAMIRANJEM

Filip Prevendar

Zagreb, lipanj 2022.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 2856

OTKRIVANJE FIZIKALNIH IZRAZA GENETSKIM PROGRAMIRANJEM

Filip Prevendar

Zagreb, lipanj 2022.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 11. ožujka 2022.

DIPLOMSKI ZADATAK br. 2856

Pristupnik:	**Filip Prevendar (0036506769)**
Studij:	Računarstvo
Profil:	Računarska znanost
Mentor:	doc. dr. sc. Marko Đurasević

Zadatak: **Otkrivanje fizikalnih izraza genetskim programiranjem**

Opis zadatka:

Proučiti problem simboličke regresije. Pronaći odgovarajuće primjerke problema koji su izrađeni na temelju stvarnih fizikalnih zakona i koji će biti iskorišteni za ispitivanje razvijenih postupaka. Proučiti metodu genetskog programiranja kao i njezinu primjenu na problem simboličke regresije. Ostvariti programski okvir koji na temelju ulaznih podataka može rekonstruirati prikladni fizikalni izraz korištenjem genetskog programiranja. Proučiti i ostvariti pojedine prilagodbe genetskog programiranja s ciljem poboljšanje ostvarenih rezultata. Ocijeniti učinkovitost ostvarenog postupka. Radu priložiti izvorne tekstove programa, dobivene rezultate uz potrebna objašnjenja i korištenu literaturu.

Rok za predaju rada: 27. lipnja 2022.

SADRŽAJ

1. Uvod

U ovom radu se proučava problem simboličke regresije, i konkretna primjena na primjeru otkrivanja jednadžbi iz područja fizike iz podataka, koristeći se genetskim programiranjem.

Za podatke dobivene mjerenjem nekih fizikalnih pojava, računalnom simulacijom ili numeričkim proračunom, potrebno je dobiti jednadžbu ili izraz koja opisuje povezanost izmjerenih podataka. Poznavanje jednadžbe koja opisuje podatke donosi dublje razumijevanje tih podataka, te omogućava predviđanje novih podataka. U ovom radu se koristi skup podataka koji je dobiven numeričkim proračunima nekih poznatih jednadžbi iz područja fizike, i to 100 jednadžbi oblika $y = f(x)$ gdje je x skup točaka višedimenzionalnog ulaznog prostora, y pripadna vrijednost za dani x i f je nepoznata funkcija. Iz tog skupa podataka se primjenom genetskog programiranja pokušava se pronaći o kojoj se jednadžbi radi, odnosno kakav je oblik funkcije f. U radu su korištene već poznate jednadžbe, dok se u daljnjim primjenama algoritmi mogu primijeniti za otkrivanje novih jednadžbi.

2. Simbolička regresija

Simbolička regresija Udrescu i Tegmark (2020) je problem kod kojeg se pokušava pronaći model koji najbolje opisuje podatke. Za razliku od obične regresije, kod koje se za odgovarajuće podatke pokušavaju pronaći parametri modela koji najbolje opisuju podatke, kod simboličke regresije traži se model koji najbolje opisuje podatke, a ne parametri modela. U ovom radu se rješava problem simboličke regresije na skupu podataka zadanih kao niz ulaznih podataka i izlazni podatak, odnosno rješava se problem oblika $y = f(\mathbf{X})$ gdje je $\mathbf{X}$ skup ulaznih podataka, $f(\mathbf{X})$ nepoznata funkcija koju je potrebno odrediti, i y je skup izlaznih podataka. Podaci predstavljaju jednadžbe iz fizike u eksplicitnom obliku. Prednost simboličke regresije naspram obične regresije je taj da se kao rezultat dobije matematički izraz koji je lakše interpretirati, nego model koji se dobije postupcima rješavanja obične regresije.

Problem simboličke regresije se najčešće rješava genetskim programiranjem kod kojeg se počinje od praznog modela, te se model gradi od nekih gradivnih blokova koji predstavljaju jednostavnije funkcije, te se na taj način grade složeniji modeli.

3. Genetsko programiranje

Genetsko programiranje Koza (1993) je jedan od postupaka kojim se rješava problem simboličke regresije. Ono je proširenje genetskog algoritma pri kojem se umjesto jedinki fiksne veličine i strukture, koriste jedinke promjenjive veličine i strukture.

Genetski algoritam sastoji se od populacije jedinki koje predstavljaju moguće rješenje problema, i operatora za dobivanje novih jedinki. Svaka jedinka predstavlja moguće rješenje problema, tipično, kod genetskog algoritma je rješenje predstavljeno vektorom fiksne veličine koji sadrži parametre nekog konkretnog modela, dok je kod genetskog programiranja jedinka predstavljena stablom ili nekom prikladnijom strukturom koja predstavlja sami model. Za svaku jedinku je moguće odrediti dobrotu ili pogrešku te tako rangirati jedinke, cilj genetskog algoritma i programiranja je maksimizirati dobrotu ili minimizirati pogrešku.

Tipično, postoje dvije vrste operatora za dobivanje novih jedinki, to su operatori križanja i operatori mutacije. Operatori križanja uzimaju dvije jedinke i daju novu jedinku koja je kombinacija početne dvije, dok operatori mutacije uzimaju jednu jedinku i daju izmijenjenu početnu jedinku.

Genetsko programiranje radi na principu iteracija, svaku iteraciju se trenutna populacija transformira u iduću populaciju pomoću operatora za dobivanje novih jedinki. Kod izgradnje nove populacije, razlikuju se dvije inačice genetskog programiranja, generacijski i eliminacijski algoritam. U generacijskoj inačici, svaku iteraciju se gradi nova populacija tako da se za svaku jedinku iz nove populacije bira neka jedinka iz stare populacije uz određenu vjerojatnost izbora, te se izabrana jedinka transformira uz operator mutacije, te se tako transformirana jedinka stavlja u novu populaciju. Također je moguće, uz određenu vjerojatnost odabrati dvije jedinke iz stare populacije, kombinirati ih operatorom križanja, te ubaciti u novu populaciju. Kod eliminacijskog algoritma, izabiru se K nasumične jedinke, dvije koje imaju najbolju dobrotu se kombiniraju operatorom križanja, rezultat se transformira operatorom mutacije, te konačna jedinka zamijenjuje onu s najgorom dobrotom od izabranih K jedinki. Ovakav postupak se još naziva i K-turnirska selekcija.

Elitizam je svojstvo genetskog programiranja da zadrži najbolju jedinku koju je pronašao. Kod eliminacijskog algoritma, elitizam je očuvan time što se pri izboru K jedinki mijenja ona najgora, tako da najbolja uvijek ostaje u populaciji. Kod generacijske inačice, elitizam nije očuvan, jer je moguće da najbolja ne bude izabrana u novu populaciju. Kako je elitizam poželjno svojstvo genetskog programiranja, u generacijskoj inačici, kako bi se zadržao elitizam, najbolja jedinka se iz trenutne populacije direktno kopira u novu populaciju.

Genetsko programiranje i genetski algoritam nije potpun niti optimalan algoritam, što znači da nema garanciju da će doći do najboljeg rješenja problema, već se izvodi ili određeni broj iteracija, ili dok mjera dobrote ili pogreške najboljeg rješenja ne dođe do zadovoljavajuće razine.

Kako se genetsko programiranje može koristiti za rješavanje raznih problema, postoji više mogućih oblika jedinki. Za rješavanje problema simboličke regresije, prikladno je koristiti stabla kao jedinke, gdje unutarnji čvorovi predstavljaju matematičke operacije, a listovi stabla predstavljaju konstante ili ulazne varijable.

Vrlo česta pojava kod genetskog programiranja je napuhivanje (eng. *bloat*), to je pojava kod koje jedinke rastu do veličina koje nisu prihvatljive, model koji pokušavamo pronaći postaje presložen za primjenu ili postaje sklon prenaučenosti na ulazne podatke. Kako bi se izbjegla pojava napuhivanja, često se u genetskom programiranju odbacuju jedinke koje su veće od zadane granice, u izračun dobrote jedinke se nekako uključi i funkcija kazne ovisna o veličini jedinke, ili se koriste operatori mutacije i selekcije koji ne uzrokuju rast jedinke.

4. Implementacija

Sustav za rješavanje problema simboličke regresije implementiran je u programskom jeziku java i u nastavku je detaljnije opisan.

Skup podataka korišten u radu je preuzet iz Udrescu i Tegmark (2020) i zadan je u obliku niza točaka $\mathbf{X}_i$ u višedimenzionalnom prostoru, te pripadnim vrijednostima tražene funkcije za zadanu točku prostora $y_i = f(\mathbf{X}_i)$. Cilj je otkriti oblik nepoznate funkcije f. Točke višedimenzionalnog prostora $\mathbf{X}_i$ su predstavljene vektorom, i u nastavku se pojedine komponente tog vektora nazivaju varijable x_0, x_1, ... Pripadne izlazne vrijednosti y su uvijek jednodimenzionalne, odnosno skalari.

Jedinke su prikazane u obliku stabla. U tom stablu nalaze se tri vrste čvorova, terminalni i funkcijski čvorovi. Terminalni čvorovi nemaju djece, dok funkcijski čvorovi imaju jedno ili dvoje djece. Terminalni čvorovi predstavljaj konstante ili varijable. Funkcijski čvorovi dijele se na unarne i binarne funkcijske čvorove, ovisno o tome koliko imaju djece, i predstavljaju matematičke operacije i funkcije. Pregled čvorova je dan u tablici 4.1 .

Vrijednost stabla za neku točku ulaznog prostora određuje se na način da se u stablo u terminalne čvorove koji predstavljaju varijable uvrste konkretne vrijednosti ulaznih varijabli, te se od dna prema vrhu stabla primjenjuju matematičke operacije koje su predstavljene čvorovima stabla. Na taj način se dobije izlaz stabla za neku točku ulaznog prostora, $y_{i,stablo} = f_{stablo}(\mathbf{X}_i)$.

Od funkcija dobrote, implementirane su dvije varijante, obično srednje kvadratno odstupanje i normalizirano srednje kvadratno odstupanje. Obično srednje kvadratno odstupanje za svaku od N točaka ulaznog prostora $\mathbf{X}_i$ računa pripadni izlaz za stablo $y_{i,stablo}$, te se srednje kvadratno odstupanje izračuna prema formuli $mse_{stablo} = \frac{1}{N}\sum_{i=1}^{N}(y_{i,stablo} - y_i)^2$. Normalizirano srednje kvadratno odstupanje se računa na način da se najprije od izlaznih podataka oduzme njihova srednja vrijednost, te se podijele sa svojom standardnom devijacijom. Na taj način se postiže da izlazni podaci imaju srednju vrijednost $\mu = 0$ i standardnu devijaciju $\sigma = 1$. To se radi i na stvarnim izlaznim podacima, i na izlaznim vrijednostima stabala, čime se eliminira

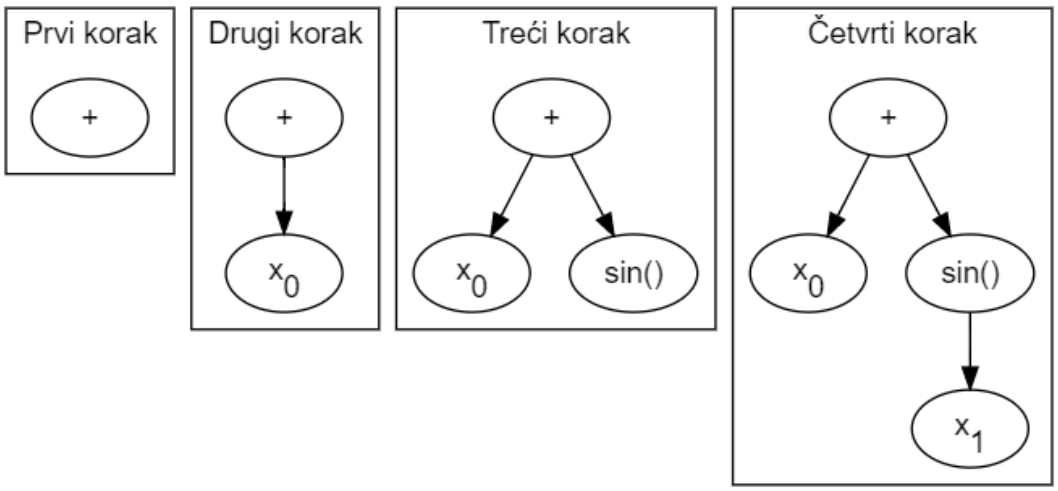
Tablica 4.1: Korišteni čvorovi. U prvom stupcu je prikazano ime čvora, u drugom stupcu je prikazan broj djece koji čvor ima, te je u trećem stupcu prikazana matematička operacija koju čvor predstavlja.

Ime čvora	Broj djece	Matematička operacija
Konstanta	0	c
Varijabla	0	x
Negacija	1	$-x$
Sinus	1	$\sin(x)$
Kosinus	1	$\cos(x)$
Inverzija	1	x^{-1}
Kvadriranje	1	x^2
Korijen	1	$\sqrt{x}$
Eksponenciranje	1	e^x
Logaritam	1	$\ln(x)$
Zbrajanje	2	$x + y$
Oduzimanje	2	$x - y$
Množenje	2	$x * y$
Dijeljenje	2	$\frac{x}{y}$

pomak i skaliranje nepoznate funkcije, već je daje veća važnost obliku funkcije. Potom se za normalizirane vrijednosti izlaza računa srednje kvadratno odstupanje kao i u prethodnoj varijanti. Odnosno, srednja vrijednost stvarnog izlaza je $\mu = \frac{1}{N}\sum_{i=1}^{N} y_i$, njegova varijanca je $\sigma^2 = \frac{1}{N-1}\sum_{i=1}^{N}(y_i - \mu)^2$, pa je normalizirani stvarni izlaz jednak $y_i' = \frac{y_i - \mu}{\sigma}$. Za pojedina stabla vrijede slične formule, $\mu_{stablo} = \frac{1}{N}\sum_{i=1}^{N} y_{i,stablo}$, $\sigma^2_{stablo} = \frac{1}{N-1}\sum_{i=1}^{N}(y_{i,stablo} - \mu_{stablo})^2$ i $y'_{i,stablo} = \frac{y_{i,stablo} - \mu_{stablo}}{\sigma_{stablo}}$. Srednje kvadratno odstupanje se potom računa prema formuli $nmse_{stablo} = \frac{1}{N}\sum_{i=1}^{N}(y'_{i,stablo} - y'_i)^2$.

Postupak dobivanja početnih stabala ostvaren je preko generatora stabala koji generira stabla tako da odabere jedan od tri tipa čvora, terminalni, unarni ili binarni, i ako odabere čvorove koji imaju djecu, generira drugo stablo i postavi ga kao dijete čvora. Postupak je rekurzivan i prikazan je na slici 4.1. Kako bi se izbjegao preveliki rast stabala, zadano je i ograničenje na visinu stabla, tako da kada se dosegne određena visina, sva generirana djeca na graničnoj visini su terminalni čvorovi.

Postupci križanja uzimaju dvije jedinke i stvaraju novu jedinku na temelju ulazne dvije. Od postupaka križanja ostvareno je križanje zamjenom podstabala i križanje



Slika 4.1: Generiranje stabala. Najprije se u prvom koraku izabire čvor koji predstavlja zbrajanje, potom se izabire varijabla x_0 kao njegovo dijete. U trećem koraku se izabire čvor koji predstavlja funkciju sinus, i konačno u četvrtom koraku se izabire čvor koji predstavlja varijablu x_1 čime završava izgradnja stabla.

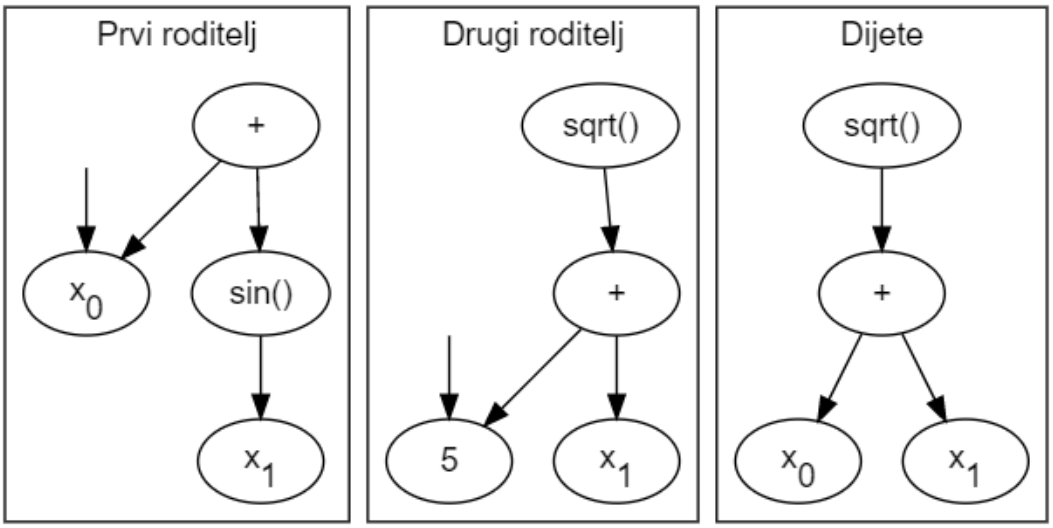
ujedinjenjem stabala.

Zamjena postabala izabire po jedan čvor na oba ulazna stabla, uzima podstablo od izabranog čvora na jednom stablu i postavlja ga na mjesto izabranog čvora drugog stabla, što je prikazano na slici 4.2. Križanje ujedinjenjem stabala izabire neki binarni čvor i postavlja ulazna stabala kao djecu tog binarnog čvora Ovo križanje je prikazano na slici 4.3.

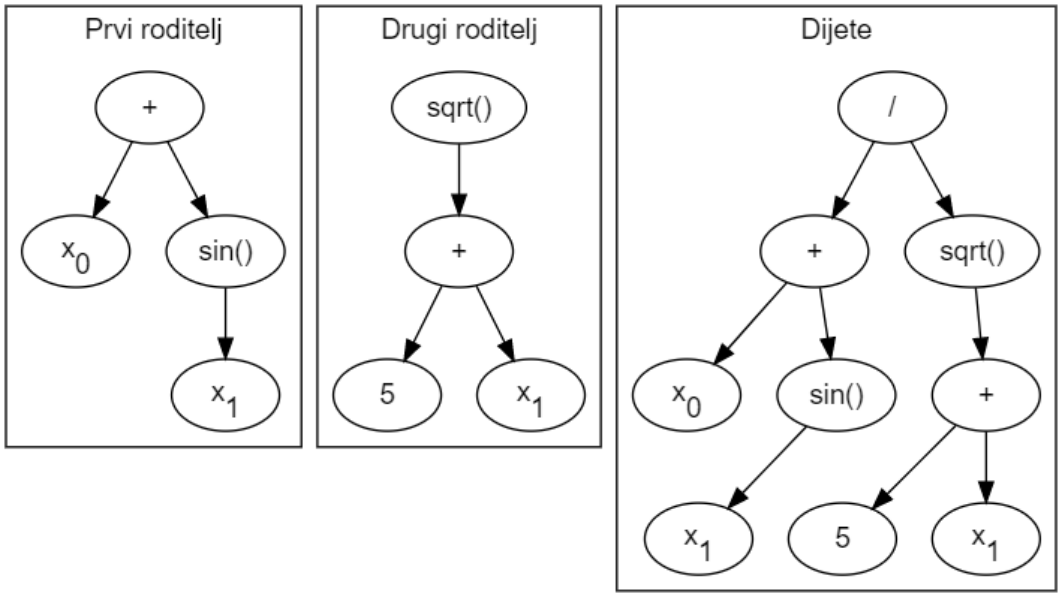
Postupci mutacije uzimaju jednu jedinku i stvaraju novu jedinku koja je izmijenjena početna jedinka. Mutacija potpunom zamjenom ignorira ulaznu jedinku i generira potpuno novu jedinku, ovime se povećava raznolikost jedinki u populaciji. Mutacija izrastom izabire nasumični čvor u jedinci i generira potpuno novo stablo, te izabrani čvor zamijeni novim stablom. Mutacija zamjenom čvora terminalnim izabire nasumični čvor u jedinci i taj čvor zamijenjuje s terminalnim čvorom. Mutacija uzimanjem podstabala izabire čvor u stablu, te vraća podstablo ispod izabranog čvora. Mutacija zamjenom tipa izabire čvor u stablu, te mu mijenja vrstu, ali ne i tip, odnosno terminalni čvorovi ostaju terminalni, binarni ostaju binarni i unarni ostaju unarni, ali im se promijeni vrsta. Mutacije su prikazane na slici 4.4.

Kod operatora mutacije i selekcije, koriste se nasumično izabrani čvorovi koji su izabrani na način da se generira slučajni broj $n \in [0, veličina(stablo)]$, te se vrši obilazak stabla dok se ne pronađe n-ti čvor.

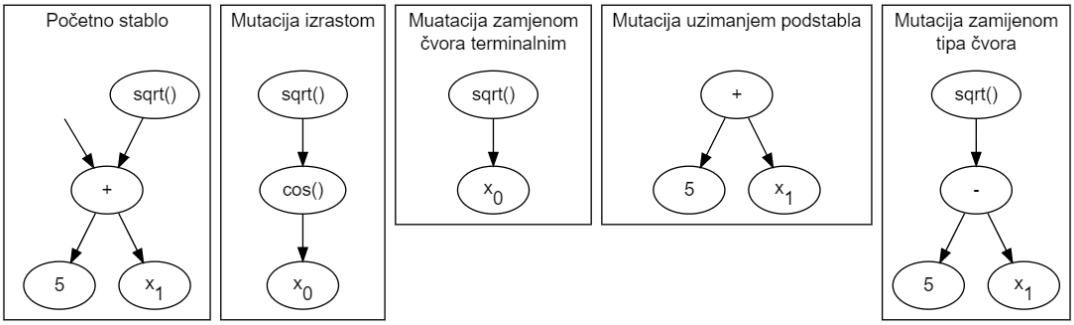
Implementirane su dvije inačice genetskog algoritma, 3-turnirski eliminacijski, i



Slika 4.2: Zamjena podstabala. Izabrani čvorovi na roditeljima su označeni strelicama. Konkretno, ovdje se uzima podstablo od čvora x_0 na prvom stablu, te se njime zamijenjuje podstablo 5 na drugom roditelju.



Slika 4.3: Ujedinjenje stabala. Izabrani čvor je čvor za dijeljenje, te su stabla spojena preko njega.



Slika 4.4: Mutacije. Na početnom stablu je izabrani čvor označen sa strelicom. Kod mutacije izrastom, umjesto plus čvora, izraslo je novo stablo $\cos(x_0)$. Kod Mutacije zamjenom čvora terminalnim, čvor je zamijenjen čvorom x_0. Kod mutacije uzimanjem podstabla, uzeto je podstablo $5 + x_0$. Kod mutacije zamjenom tipa čvora, čvor za zbrajanje je zamijenjen čvorom za oduzimanje. Mutacija potpunom zamjenom bi generirala stablo koje je neovisno o početnom stablu, stoga ovdje nije prikazana.

generacijski algoritam.

U 3-turnirskoj inačici, pri pokretanju algoritma generira se populacija željene veličine, te se u svakoj iteraciji nasumično odabiru tri jedinke, od te tri jedinke se dvije bolje kombiniraju operatorom križanja, te se uz određenu vjerojatnost primijeni ili ne primjeni neki od operatora mutacije. Nakon toga se jedinka ubacuje u populaciju na mjesto najgore od izabrane tri. Dodatno, prije ubacivanja u populaciju, provjerava se ima li jedinka visinu manju od najveće dopuštene visine, ukoliko nema generira se nova koja ima, te se nova ubacuje u populaciju. Ovime se ograničava neograničeni rast jedinke. Ovaj algoritam ima svojstvo elitizma, jer ako najbolja jedinka bude izabrana, nikada neće biti izbačena iz populacije. Iteracije se ponavljaju proizvoljan broj puta.

Kod generacijske inačice, na isti način se generira inicijalna populacija željene veličine. Zatim se u svakoj iteraciji konstruira nova populacija, i to na način da se iz trenutne populacije jedinke ubacuju u novu koja tada postaje trenutna. Ubacivanje se provodi tako da se pomoću proporcionalne selekcije odabire jedinka iz stare populacije, uz neku vjerojatnost se mutira, te se ubaci u novu populaciju. Alternativno, uz određenu vjerojatnost odabiru se dvije jedinke, kombiniraju se operatorom križanja, te se dobiveni rezultat mutira uz određenu vjerojatnost, i ubacuje u populaciju. I ovdje se provjerava visina dobivene jedine prije umetanja u populaciju, te se po potrebi generira nova. Ovaj algoritam nema ugrađeno svojstvo elitizma, pa se dodatno, pri formaciji nove populacije uzima najbolja jedinka, te se direktno kopira u novu populaciju.

U generacijskoj inačici, proporcionalna selekcija se vrši na način da se za svaku jedinku populacije izračuna dobrota jedinke $d_{jedinka}$, odredi se jedinka s najgorom

dobrotom, te se nova dobrota računa na način da se od stare dobrote oduzme najgora dobrota $d'_{jedinka} = d_{najgora} - d_{jedinka}$. Zatim se dobrote normaliziraju kako bi im suma bila jednaka jedinici, po formuli $d''_{jedinka} = \frac{d'_{jedinka}}{\sum d'}$. Potom se svakoj dobroti dodaje neka mala vrijednost kako bi i najgora jedinka dobila priliku da bude izabrana, te se dobrote ponovno normaliziraju da im suma bude 1. Konačno, računaju se pragovi za svaku jedinku kao zbroj dobrote jedinke i kumulativna suma dobrota prethodnih jedinki $p_i = \sum_{j=1}^{i} d''_j$. Tako dobiveni pragovi se tada koriste za izbor jedinke, tako da se generira nasumični realni broj $x \in [0, 1]$ te bude izabrana jedinka za koju je broj x manji od njenog praga, a veći od prethodnog praga.

5. Rezultati

Opisani algoritmi provedeni su na skupu podataka, na svim jednadžbama iz skupa podataka. Ukupno ima 100 nepoznatih jednadžbi, i za svaku postoji 100000 točaka iz ulaznog prostora i pripadni izlazi. Za svaki problem je algoritam proveden na smanjenom skupu podataka od 1000 točaka, koji je dobiven uzorkovanjem bez ponavljanja iz početnog skupa, kako bi se smanjilo vrijeme izvođenja algoritama. Algoritmi su provedeni 10 puta za svaki problem. Ograničenje na najveću dopuštenu visinu stabla u populaciji je postavljeno na 10. Provedeno je 1000 iteracija obje inačice algoritma. U oba algoritma je korištena veličina populacije od 50 jedinki. Vjerojatnost mutacije je postavljanja na 0.9, te je kod generacijskog algoritma korištena vjerojatnost križanja od 0.2.

Uspješnosti algoritama su prikazani u tablici 5.1 u nastavku. Prvi stupac predstavlja jednadžbu koja se traži, u drugom stupcu je uspješnost algoritma pri korištenju običnog srednjeg kvadratnog odstupanja, u trećem stupcu je uspješnost algoritma pri korištenju normaliziranog srednjeg kvadratnog odstupanja. Uspješnim pronalaskom se smatra izvođenje algoritma gdje u konačnoj populaciji jedinka s najboljom dobrotom predstavlja točni ili ekvivalentni izraz traženom, do na pomak i množenje s konstantom.

Tablica 5.1: Tablica rezultata. U prvom stupcu je prikazana jednadžba koja se traži, u drugom stupcu je uspješnost pronalaska jednadžbe korištenjem običnog mse kao funkcija dobrote u obliku eliminacijski/generacijski. U trećem stupcu je prikazana uspješnost korištenjem normaliziranog mse kao funkcije dobrote.

Jednadžba	mse	normalizirani mse
$\frac{1}{\sqrt{2\pi}}e^{-\frac{x_0^2}{2}}$	Ne/Ne	Ne/Da
$\frac{1}{\sqrt{2\pi x_1}}e^{-\frac{x_0^2}{2x_1^2}}$	Ne/Ne	Ne/Ne
$\frac{1}{\sqrt{2\pi x_1}}e^{-\frac{(x_0-x_2)^2}{2x_1^2}}$	Ne/Ne	Ne/Ne
$\sqrt{(x_1-x_0)^2+(x_3-x_2)^2}$	Ne/Ne	Ne/Ne

$\frac{x_0 x_1 x_2}{\sqrt{(x_4-x_3)^2+(x_6-x_5)^2+(x_8-x_7)^2}}$	Ne/Ne	Ne/Ne
$\frac{x_0}{\sqrt{1-\frac{x_1^2}{x_2^2}}}$	Ne/Ne	Ne/Ne
$x_0 x_3 + x_1 x_4 + x_2 x_5$	Ne/Ne	Ne/Ne
$x_0 x_1$	Da/Da	Da/Da
$\frac{x_0 x_1}{4\pi x_2 x_3^2}$	Ne/Ne	Da/Da
$\frac{x_0}{4\pi x_1 x_2^2}$	Ne/Ne	Da/Da
$x_0 x_1$	Da/Da	Da/Da
$x_0(x_1 + x_2 x_3 \sin(x_4))$	Ne/Ne	Ne/Ne
$\frac{1}{2} x_0 (x_1^2 + x_2^2 + x_3^2)$	Ne/Ne	Ne/Ne
$x_4 x_0 x_1 (\frac{1}{x_3} - \frac{1}{x_2})$	Ne/Ne	Ne/Ne
$x_0 x_1 x_2$	Da/Da	Da/Da
$\frac{1}{2} x_0 x_1^2$	Ne/Da	Da/Da
$\frac{x_0 - x_1 x_3}{\sqrt{1-\frac{x_1^2}{x_2^2}}}$	Ne/Ne	Ne/Ne
$\frac{x_3 - \frac{x_0 x_3}{x_2^2}}{\sqrt{1-\frac{x_1^2}{x_2^2}}}$	Ne/Ne	Ne/Ne
$\frac{x_0 x_1}{\sqrt{1-\frac{x_1^2}{x_2^2}}}$	Ne/Ne	Ne/Ne
$\frac{x_1 + x_2}{\sqrt{1+\frac{x_1 x_2}{x_0^2}}}$	Ne/Ne	Ne/Ne
$\frac{x_0 x_2 + x_1 x_3}{x_0 + x_1}$	Ne/Ne	Ne/Ne
$x_0 x_1 \sin(x_2)$	Ne/Da	Da/Da
$x_0 x_1 x_2 \sin(x_3)$	Ne/Da	Da/Da
$\frac{1}{4} x_0 (x_1^2 + x_2^2) x_3^2$	Ne/Ne	Ne/Ne
$\frac{x_0}{x_1}$	Da/Da	Da/Da
$\arcsin(x_0 \sin(x_1))$	Ne/Ne	Ne/Ne
$\frac{1}{\frac{1}{x_0} + \frac{x_2}{x_1}}$	Ne/Da	Ne/Da
$\frac{x_0}{x_1}$	Da/Da	Da/Da
$\sqrt{x_0^2 + x_1^2 - 2 x_0 x_1 \cos(x_2 - x_3)}$	Ne/Ne	Ne/Ne
$x_0 \frac{\sin(\frac{x_2 x_1}{2})^2}{\sin(\frac{x_1}{2})^2}$	Ne/Ne	Ne/Ne
$\arcsin(\frac{x_0}{x_1 x_2})$	Ne/Ne	Ne/Ne
$\frac{x_0^2 x_1^2}{6\pi x_2 x_3^3}$	Ne/Ne	Da/Ne
$(\frac{1}{2} x_0 x_1 x_2^2) \frac{8\pi x_3^2}{3} \frac{x_4^4}{x_4^2 - x_5^2}$	Ne/Ne	Ne/Ne

$\frac{x_0 x_1 x_2}{x_3}$	Ne/Da	Da/Da
$\frac{x_2}{1-\frac{x_1}{x_0}}$	Ne/Ne	Ne/Da
$\frac{1+\frac{x_1}{x_0}}{\sqrt{1-\frac{x_1^2}{x_0^2}}} x_2$	Ne/Ne	Ne/Ne
$\frac{x_1}{2\pi} x_0$	Ne/Da	Da/Da
$x_0 + x_1 + 2\sqrt{x_0 x_1}\cos(x_2)$	Ne/Ne	Ne/Ne
$\frac{4\pi x_2 x_3^2}{x_0 x_1^2}$	Ne/Ne	Da/Da
$\frac{3}{2} x_0 x_1$	Ne/Da	Da/Da
$\frac{1}{1-x_0} x_1 x_2$	Ne/Ne	Ne/Da
$\frac{x_0 x_1 x_3}{x_2}$	Ne/Da	Da/Da
$x_0 e^{-\frac{x_1 x_2 x_4}{x_3 x_5}}$	Ne/Ne	Ne/Ne
$\frac{x_2 x_0^3}{\pi^2 x_4^2 (e^{\frac{x_2 x_0}{x_3 x_1}} - 1)}$	Ne/Ne	Ne/Ne
$\frac{x_0 x_1 x_2}{x_3}$	Da/Da	Da/Da
$x_0 x_1 x_2$	Da/Da	Da/Da
$\frac{1}{x_0 - 1} \frac{x_1 x_3}{x_2}$	Ne/Ne	Ne/Ne
$x_0 x_1 x_2 \ln(\frac{x_4}{x_3})$	Ne/Ne	Ne/Ne
$\sqrt{\frac{x_0 x_1}{x_2}}$	Ne/Ne	Da/Da
$\frac{x_0 x_2^2}{\sqrt{1-\frac{x_1^2}{x_2^2}}}$	Ne/Ne	Ne/Ne
$x_0[\cos(x_1 x_2) + x_3 \cos(x_1 x_2)^2]$	Ne/Ne	Ne/Ne
$\frac{x_0 (x_2 - x_1) x_3}{x_4}$	Ne/Ne	Ne/Da
$\frac{x_0}{4\pi x_2^2}$	Ne/Ne	Da/Da
$\frac{x_0}{4\pi x_1 x_2}$	Ne/Ne	Da/Da
$\frac{1}{4\pi x_0} \frac{x_1 \cos(x_2)}{x_3^2}$	Ne/Ne	Ne/Da
$\frac{3}{4\pi x_0} \frac{x_1 x_5}{x_2^5} \sqrt{x_3^2 + x_4^2}$	Ne/Ne	Ne/Ne
$\frac{3}{4\pi x_0} \frac{x_1}{x_3^3} \cos(x_2) \sin(x_2)$	Ne/Ne	Ne/Da
$\frac{3}{5} \frac{x_0^2}{4\pi x_1 x_2}$	Ne/Ne	Da/Da
$\frac{x_0 x_1^2}{2}$	Ne/Da	Da/Da
$\frac{x_0}{x_1} \frac{1}{1+x_2}$	Ne/Da	Ne/Da
$\frac{x_0 x_1}{x_2 (x_3^2 - x_4^2)}$	Ne/Ne	Ne/Ne
$x_0 (1 + \frac{x_4 x_5 \cos(x_3)}{x_1 x_2})$	Ne/Ne	Ne/Ne
$\frac{x_0 x_1^2 x_2}{3 x_3 x_4}$	Ne/Ne	Da/Da
$\frac{x_0 x_1}{1 - \frac{x_0 x_1}{3}} x_2 x_3$	Ne/Ne	Ne/Ne

$1 + \frac{x_0 x_1}{1 - \frac{x_0 x_1}{3}}$	Ne/Ne	Ne/Ne
$\frac{1}{4\pi x_0 x_1^2} \frac{2x_2}{x_3}$	Ne/Da	Ne/Da
$\frac{x_0}{\sqrt{1 - \frac{x_1^2}{x_2^2}}}$	Ne/Ne	Ne/Ne
$\frac{x_0 x_1}{\sqrt{1 - \frac{x_1^2}{x_2^2}}}$	Ne/Ne	Ne/Ne
$-x_0 x_1 \cos(x_2)$	Ne/Da	Da/Da
$-x_0 x_1 \cos(x_2)$	Ne/Da	Ne/Da
$\frac{x_0}{4\pi x_1 x_2 (1 - \frac{x_3}{x_4})}$	Ne/Ne	Ne/Ne
$\sqrt{\frac{x_0^2}{x_1^2} - \frac{\pi^2}{x_2^2}}$	Ne/Ne	Ne/Ne
$x_0 x_1 x_2^2$	Ne/Da	Da/Da
$x_0 x_1^2$	Da/Da	Da/Da
$\frac{x_0 x_1}{2\pi x_2}$	Ne/Ne	Da/Da
$\frac{x_0 x_1 x_2}{2}$	Ne/Da	Da/Da
$\frac{x_0 x_1 x_2}{2x_3}$	Ne/Da	Da/Da
$\frac{x_0 x_1}{4\pi x_2}$	Ne/Ne	Da/Da
$\frac{x_0 x_2 x_3 x_4}{\frac{x_1}{2\pi}}$	Ne/Ne	Da/Da
$\frac{x_0}{e^{\frac{x_3 x_4}{x_1 x_2}} + e^{-\frac{x_3 x_4}{x_1 x_2}}}$	Ne/Ne	Ne/Ne
$x_0 x_1 \tanh(\frac{x_1 x_2}{x_3 x_4})$	Ne/Ne	Ne/Ne
$\frac{x_0 x_1}{x_2 x_3} + \frac{x_0 x_4 x_7}{x_5 x_6^2 x_2 x_3}$	Ne/Ne	Ne/Ne
$x_0(1 + x_2)x_1$	Da/Da	Da/Da
$\frac{x_0 x_1 x_3}{x_2}$	Ne/Da	Da/Da
$\frac{x_0}{2(1+x_1)}$	Ne/Da	Ne/Da
$\frac{1}{e^{\frac{x_0}{2\pi} \frac{x_1}{x_2 x_3}} - 1}$	Ne/Ne	Ne/Ne
$\frac{\frac{x_0}{2\pi} x_1}{e^{\frac{x_0}{2\pi} \frac{x_1}{x_2 x_3}} - 1}$	Ne/Ne	Ne/Ne
$\frac{2x_0 x_1}{\frac{x_2}{2\pi}}$	Ne/Ne	Da/Da
$\sin(\frac{x_0 x_1}{\frac{x_2}{2\pi}})^2$	Ne/Ne	Ne/Ne
$\frac{x_0 x_1 x_2}{\frac{x_3}{2\pi}} \frac{\sin((x_4 - x_5)\frac{x_2}{2})^2}{((x_4 - x_5)\frac{x_2}{2})^2}$	Ne/Ne	Ne/Ne
$x_0 \sqrt{x_1^2 + x_2^2 + x_3^2}$	Ne/Ne	Ne/Ne
$x_0 \frac{x_1}{2\pi}$	Da/Da	Da/Da
$\frac{2x_0 x_1^2 x_2}{\frac{x_3}{2\pi}}$	Ne/Ne	Da/Da
$x_0(e^{\frac{x_1 x_2}{x_3 x_4}} - 1)$	Ne/Ne	Ne/Ne

$2x_0(1-\cos(x_1x_2))$	Ne/Ne	Ne/Da
$\frac{(\frac{x_0}{2\pi})^2}{2x_1x_2^2}$	Ne/Ne	Da/Da
$\frac{2\pi x_0}{x_1x_2}$	Da/Da	Da/Da
$x_0(1+x_1\cos(x_2))$	Ne/Da	Da/Da
$\frac{-x_0x_1^4}{2(4\pi x_4)^2(\frac{x_2}{x_\pi})^2}\frac{1}{x_3^2}$	Ne/Ne	Ne/Ne
$\frac{-x_0x_1x_2}{x_3}$	Ne/Ne	Da/Da

Prema tablici 5.1, algoritmi koji koriste normalizirani mse kao funkciju dobrote su uspješniji, to je zato što normalizirani mse ne uzima u obzir pomak i skaliranje podataka, što olakšava pretragu, jer se treba pronaći izraz koji odgovara podacima oblikom, a ne još dodatno skaliranjem i pomakom podataka. Generacijski algoritam je uspješniji od eliminacijskog, upravo zato što generacijski u istom broju iteracija pretraži više jedinki, ali to dolazi uz cijenu duljeg vremena izvođenja. Oba algoritma pate od napuhivanja te su jedinke u konačnoj populaciji prilično velike, i vrlo često pri neuspješnom pronalasku, najbolja jedinka ima graničnu visinu.

6. Zaključak

U ovom radu je implementiran sustav za rješavanje problema simboličke regresije. Za to se koriste generacijski i eliminacijski genetski algoritmi, te su primijenjeni na skup podataka od 100 jednadžbi iz područja fizike. Korištene su dvije funkcije dobrote, obični mse i normalizirani mse. Generacijski genetski algoritam se pokazao uspješnijim od eliminacijskog algoritma, zbog toga što u istom broju iteracija uspije pretražiti više jedinki. Normalizirani mse se također pokazao uspješniji naspram običnog mse, upravo zato što ne prati pomak i skaliranje podataka, nego promatra samo oblik nepoznate jednadžbe.

Kako se radi o jednadžbama iz fizike, daljnji rad bi uključivao korištenje informacije o mjernim jedinicama varijabli pri formiranju izraza. Dodatno, daljnji rad bi mogao uključivati i operatore za preslagivanje stabala koji su slični operatorima mutacije, ali umjesto da promijene izraz koji je predstavljen stablom, generiraju stablo koje predstavlja ekvivalentni izraz, ali drugačije zapisan, ili pojednostavljen. I konačno, umjesto korištenja stroge granice na visinu stabala, moguće je informaciju o visini stabla uključiti u funkciju dobrote i na taj način kažnjavati prevelika stabla, umjesto odbacivanja prevelikih stabala.

LITERATURA

John R. Koza. Genetic programming - on the programming of computers by means of natural selection. U *Complex adaptive systems*, 1993.

Riccardo Poli, William B. Langdon, i Nicholas Freitag McPhee. *A field guide to genetic programming*. Published via `http://lulu.com` and freely available at `http://www.gp-field-guide.org.uk`, 2008. (With contributions by J. R. Koza).

Silviu-Marian Udrescu i Max Tegmark. Ai feynman: a physics-inspired method for symbolic regression, 2020.

Otkrivanje fizikalnih izraza genetskim programiranjem

Sažetak

U ovom radu pokušava se otkriti 100 jednadžbi iz fizike u obliku y = f(x) koje su zadane kao skup točaka x i pripadne vrijednosti y. Takav problem naziva se simbolička regresija i u ovom radu se rješava pomoću genetskog programiranja, posebnog oblika genetskog algoritma gdje su jedinke stabla umjesto vektora vrijednosti fiksne veličine.

Ključne riječi: Simbolička regresija, genetsko programiranje

Discovering physical expressions using genetic programming

Abstract

In this thesis 100 equations from physics are attempted to be guessed. The equations are in a form y = f(x) where x is given as a set of points, and y is associated value with the given point. Such problem is called symbolic regression and in this thesis, the problem is tackled with genetic programming, a special case of genetic algorithm where instead of using vectors with fixed length containing values, trees are used as a solution.

Keywords: Symbolic regression, genetic programming