Sveučilište u Zagrebu
Fakultet elektrotehnike i računarstva

Jednostavan genetski algoritam za problem dobivanja fraze

Jan Čapek

Mentor: Marko Đurasević

Zagreb, svibanj 2019.

1 Uvod

Genetski algoritmi su jedan od pokušaja oponašanja prirode odnosno evolucije.

Ideja iza genetskih algoritama je da će se neki problem početi rješavati tako da se postavi mnogo (generaciju) nasumičnih početnih rješenja koja mogu biti vođena nekim pretpostavkama, ali i ne moraju i često nisu. Takva rješenja će najvjerojatnije biti pogrešna, ali ako postoji neki sustav određivanja kvalitete rješenja tako da se odaberu ona bolja, može se ih povezati tako da se napravi neko još bolje rješenje. Odavde dolazi naziv "genetski algoritam" jer, iz neke generacije rješenja, križaju se ona bolja u nadi kako će sljedeća generacija biti još bolja.

Ovakvi algoritmi nisu namijenjeni da uvijek daju potpuno točno rješenje problema već daju dovoljno dobro rješenje s obzirom na problem te vrijeme računanja.

Genetski algoritmi se uglavnom koriste kod rješavanja, u prihvatljivom vremenu, teško riješivih problema. Takvi problemi ne mogu biti riješeni u prihvatljivom vremenu jer su računala jednostavno prespora da prođu kroz sve moguće kombinacije rješenja i na temelju njih odaberu najbolju. Kada bi se išlo takvom taktikom, program bi se izvodio mjesecima, godinama, a možda i tisućljećima. Jedan od takvih problema bi bio dobivanje neke fraze i taj problem će biti obrađen u sljedećem poglavlju.

2 Problem dobivanja fraze

Zamislimo da želimo dobiti neku frazu fiksne duljine npr. Aristotelovu izreku “Više se voli ono što se teže dobije.” koja ima 36 znakova. Hrvatska abeceda, ako se ne ubrajaju slova poput ‘lj’ koji se tvore s dva znaka, ima 27 slova koja također mogu biti velika i mala što u konačnici znači da imamo na raspolaganju 54 slova. Uz to ako ubrojimo i osnovne interpunkcijske znakove (točka, zarez, upitnik i uskličnik), za izgradnju fraze na raspolaganju imamo 58 različitih znakova. Kad bi se pokušalo doći do spomenute fraze silom, tako da se prolazi kroz sve moguće kombinacije, moralo bi se proći kroz 58^{36} kombinacija slova i interpunkcijskih znakova kako bi se sigurno došlo do navedene fraze i ako se račun pojednostavi pa se kaže da se mogu proći 10^8 različitih kombinacija u sekundi, trebalo bi $9,65 \cdot 10^{47}$ godina da se prođe kroz sve njih.

Iz navedenog se vidi da pristup silom nije prihvatljiv stoga će se koristiti genetski algoritam za dolazak do navedene rečenice.

Na umu treba imati sljedeće:

1. Genetski algoritmi bazirani su na biološkoj evoluciji stoga je potrebno smisliti način kako prenijeti “znanje” (genetski kod) u sljedeću generaciju.
2. Algoritam mora moći odrediti koja rješenja (jedinke) će proslijediti svoj genetski kod u sljedeću generaciju odnosno treba osmisliti sistem rangiranja rješenja tako da bolja imaju veće šanse razmnožavanja.
3. Potrebno je imati raznoliku populaciju, inače algoritam neće doći do željenih rezultata.

2.1 Kreiranje populacije

Za spomenuti problem, kreira se populacija “jedinki” (rješenja) od kojih je svaka jedinka jedan nasumično generirani niz znakova jednake duljine kao i fraza koja se traži.

Taj niz znakova jedinke će biti njen DNK, odnosno, to će biti podatak koji opisuje nju, ali i njene potomke. Važno je naglasiti da DNK jedinke ne mora direktno biti rješenje problema kao što je to slučaj u problemu dobivanja fraze, to mogu biti i parametri neke funkcije koja će onda biti korištena za rješavanje problema.

Veličina populacije ovisi o kompleksnosti problema kojeg se rješava. Populacija mora biti dovoljno velika da se maksimizira raznolikost jedinki jer, ako će faliti raznolikosti, slične jedinke će se međusobno razmnožavati, a posljedica toga će biti da je potomak ili vrlo sličan svojim roditeljima ili čak i isti kao neki od svojih roditelja. Takve stvari bi

algoritam držale u beskonačnoj petlji jer bi pomak k rješenju bio premali ili pomaka ne bi uopće ni bilo. Također, populacija jedinki ne smije biti prevelika jer će biti vrlo "skupo" rangirati te jedinke i onda ih križati u svrhu kreiranja sljedeće generacije. Biti će toliko skupo da će algoritam raditi sporije s većom raznolikosti nego sa znatno manjom.

Cilj je odrediti veličinu generacije tako da ona bude optimalna s obzirom na raznolikost, a opet da se ocjenjivanje jedinki i križanje istih ne izvodi predugo.

U slučaju navedenog problema potrage za Aristotelovom izrekom od 36 znakova, biti će dovoljno da broj jedinki u generaciji bude oko 300 do 400.

2.2 Funkcija dobrote

Kao što je već navedeno, da bi se križale najbolje jedinke, potrebno je prvo imati neki sustav ocjenjivanja jedinki. Upravo tome služi funkcija fitnesa.

Funkcija dobrote je jedna od najvažnijih elemenata kod genetskog algoritma. O njoj ovisi koje će se jedinke razmnožavati i koliko često

Funkcija će uzeti jedinkine gene te ih evaluirati i dati neku vrijednost. Ta vrijednost ne ovisi o DNK ostalih jedinki, bitno je samo da bolja jedinka ima veću vrijednost od one lošije.

Kod određivanja funkcije dobrote također je važno da ona jasno razdvaja bolja rješenja od lošijih. Npr. neka se koristi linearna funkcija dobrote za iznad naveden problem čija će vrijednost biti jednaka broju slova koja je jedinka pogodila (pravo slovo na pravom mjestu):

1. rješenje: *funkcija_dobrote(jedinka_A) = 30*
2. rješenje: *funkcija_dobrote(jedinka_B) = 32*

Ovakva će funkcija dobrote reći da je *jedinka_B* samo 6.67% bolja od *jedinka_A*. Rezultat toga će biti da *jedinka_A* i *jedinka_B* gotovo podjednako često rade potomke, a u stvari se želi dobiti to da *jedinka_B* koja je pogodila 2 slova više radi potomke dosta češće od *jedinka_A*.

Mnogo bolja funkcija bi kod ovog problema bila kvadratna funkcija, po njoj bi *jedinka_B* bila bolja za čak 13.78%. Naravno, kakva će funkcija dobrote biti ovisi o problemu koji se rješava. Npr. za vrlo mali broj slova, linearna funkcija bi bila dovoljno dobra.

2.3 Metode odabira roditelja sljedeće generacije

Još jedan u nizu problema kreiranja genetskog algoritma je odabir roditelja sljedeće generacije. Iako ovo zvuči trivijalno s obzirom na to da je funkcija dobrote već osmišljena, reći ću da ovaj problem i nije toliko trivijalan jer različiti algoritmi selekcije roditelja mogu dovesti do drastično različitih rezultata.
Kao što je navedeno u uvodu problema, želja je imati što raznolikiju populaciju, a pritom i najuspješniju. U nastavku biti će sagledane dva načina selekcije roditelja.

2.3.1 Kolo sreće

Kolo sreće je jednostavna metoda odabira roditelja gdje se nađe suma svih vrijednosti funkcije dobrote od svih jedinki te se za svaku jedinku računa postotak koji ona zauzima na kolu sreće.

Neka je sljedeća tablica stanje populacije:

Jedinka	Funkcija dobrote
A	15
B	35
C	23
D	5
Σ	78

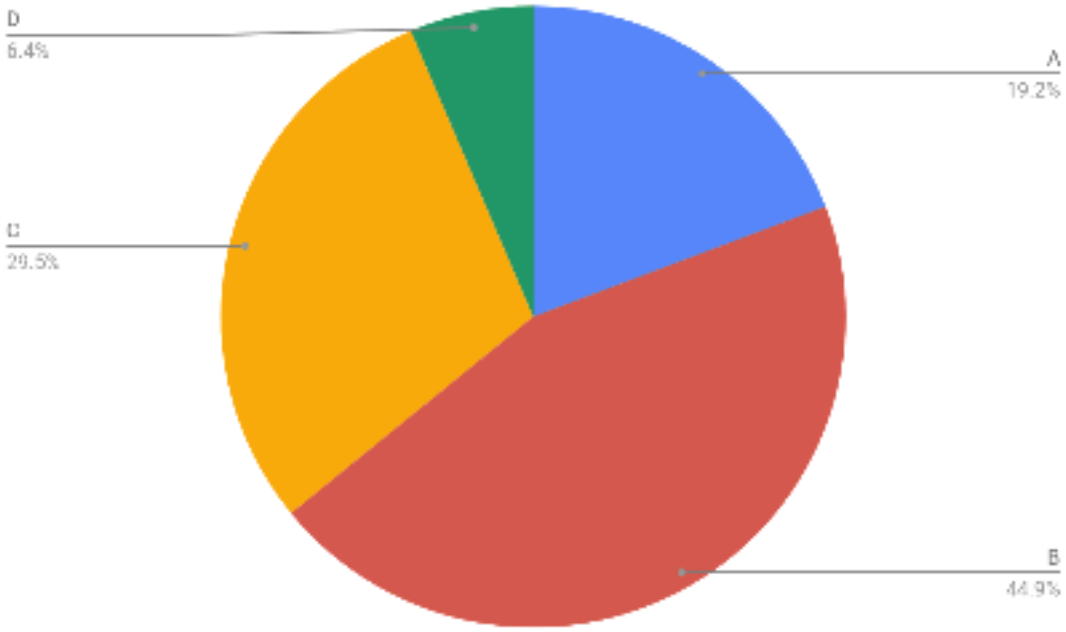
Tablica 1. Trenutno stanje populacije

Račun udjela svake jedinke u kolu sreće će biti:

Jedinka	Udio u kolu sreće
A	15 / 78 = 19.23%
B	35 / 78 = 44.87%
C	23 / 78 = 29.49%
D	5 / 78 = 6.41%

Tablica 2. Udio jedinki u kolu sreće

Konačno, kolo sreće će izgledati ovako:



Slika 1. Vizualizacija kola sreće

Iz grafa se jasno vidi da će se metodom kola sreće jedinka B razmnožavati u gotovo 45% slučajeva. Kod ovakvog stanja populacije selekcija metodom kola sreće je u redu, ali što se dogodi kad neka jedinka zauzme većinu kola sreće? Uzmimo sljedeću situaciju:

Jedinka	Udio u kolu sreće
A	85%
B	5%
C	5%
D	5%

Tablica 3. Problematičan slučaj kola sreće

U ovakvoj situaciji jedinka A će se razmnožavati gotovo svaki put, a posljedica toga će biti da će sljedećoj generaciji faliti raznolikosti.

2.3.2 Selekcija prvih nekoliko najboljih

Kao što i samo ime govori, ova metoda funkcionira na način da se odabere broj najboljih jedinki koje će se razmnožavati svakog ciklusa. Npr. u problemu dobivanja fraze reći ćemo da će taj broj biti 5. Nakon što se odredi broj jedinki koje će se razmnožavati, potrebno im je pridijeliti vjerojatnost razmnožavanja ovisno o rangu na kojem se jedinka nalazi po njenoj dobroti.

Za dobivanje fraze vrijedit će sljedeće:

Rang	Vjerojatnost razmnožavanja
1.	40%
2.	30%
3.	15%
4.	10%
5.	5%

Tablica 4. Vjerojatnosti razmnožavanja jedinki po rangu

Jasno se može vidjeti da za razliku od kola sreće, ova metoda selekcije će osigurati da niti jedna jedinka ne prevlada kod razmnožavanja jer smo pridijelili konstantne vjerojatnosti razmnožavanja.

Ovakva selekcija će vrlo lijepo poslužiti za dobivanje fraze.

2.4 Dobivanje sljedeće generacije

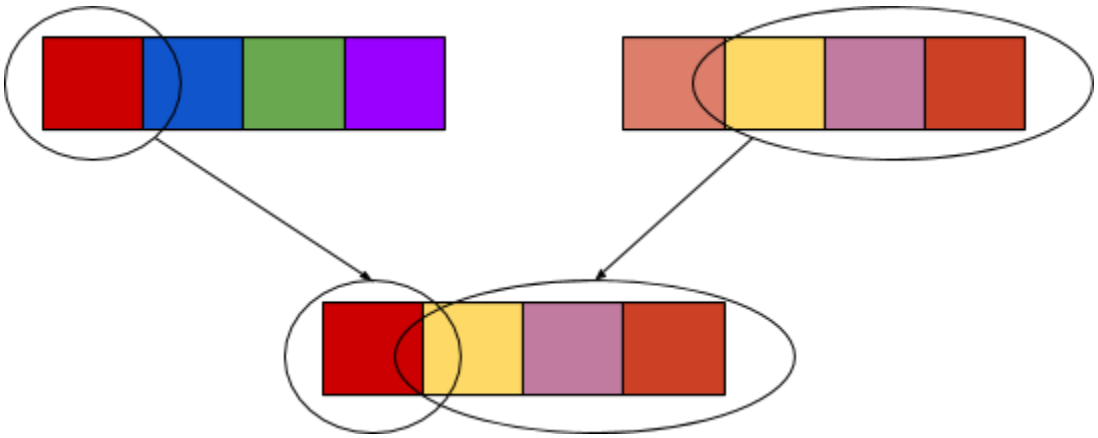
Nakon što se odaberu roditelji, potrebno je nekako napraviti potomka koji mora primiti genetski kod svojih roditelja. Postoje dva standardna načina kreiranja potomka uzeta iz prirode, a to su križanje i mutacija.

Za maksimalne rezultate, najbolje je upotrijebiti oba načina, prvo križati, a potom i mutirati. Kada ne bi bilo mutacije, velika je vjerojatnost da će ponestati raznolikosti u sustavu, te se rezultati efektivno neće više poboljšavati. Isto bi se dogodilo i kada bi se koristila samo mutacija bez križanja. U takvoj situaciji potomak bi trebao samo jednog roditelja i s malom vjerojatnošću mutacije također bi se izgubilo na raznolikosti i to znatno brže nego samo s križanjem. A kada bi vjerojatnost mutacije bila velika, onda bi izgubili mnogo dobrih informacija koje je roditelj prenio.

2.4.1 Križanje

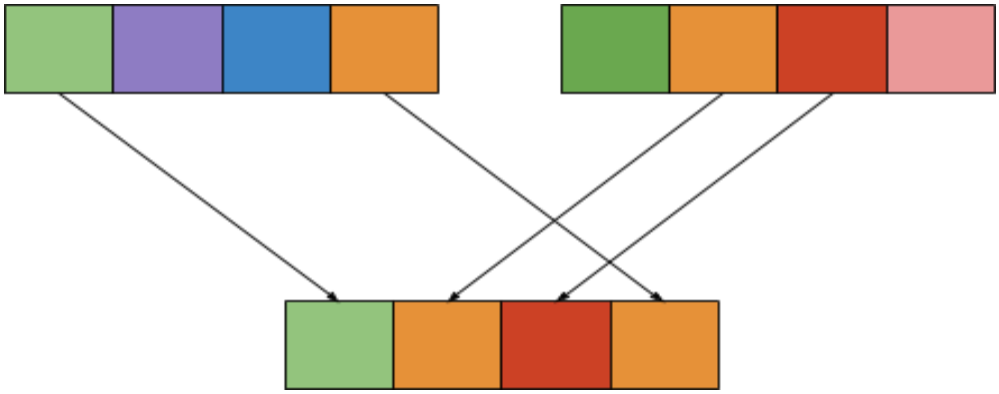
Ideja križanja kod jednostavnog genetskog algoritma je jednostavna i direktno je uzeta iz prirode. Kad imamo dva roditelja, uzmemo nekoliko gena od prvog roditelja, a ostatak gena od drugoga. Time smo ostvarili na najjednostavniji način prijenos informacija roditelja na potomka.

Iako je postupak jednostavan, načina ima mnogo. Npr. može se uzeti manji dio od jednog roditelja, a onda veći dio od drugog roditelja. Ili se pak može nasumično birati koliko će se gena uzeti od prvog roditelja, a onda naravno ostatak od drugog. Takav pristup pridonosi raznolikosti sustava. Na sljedećoj slici je prikazano križanje s jednom točkom prekida, ali može ih naravno biti više.



Slika 2. Vizualizacija križanja s jednom točkom prekida

Još jedan način križanja je da se za svaki gen nasumično odabire roditelj od kojeg će se uzeti. Također, i ovaj način dozvoljava mnogo slobode, može se prvo nasumično izračunati vjerojatnost roditelja da prenese svoj gen i potom još nasumično birati samog roditelja. To sve opet pridonosi raznolikosti populacije. Sljedeća slika prikazuje takav način križanja.



Slika 3. Nasumično križanje

2.4.2 Mutacija

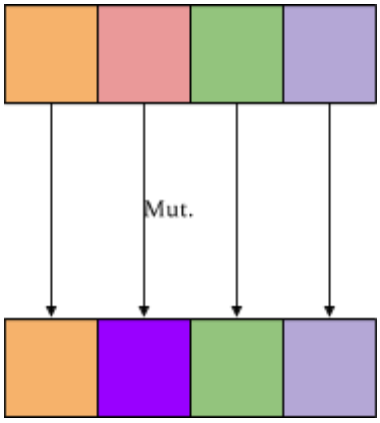
Kad je DNK potomka pomoću križanja roditeljskih gena napravljen, sad je potrebno, da maksimiziramo raznolikost, mutirati gene. Kao što je u uvodu potpoglavlja navedeno, ovaj korak nije nužan i u redu ga je izostaviti ako se zna da neće nedostajati raznolikosti.

Kako bi se odredilo hoće li gen mutirati ili ne, prvo je potrebno reći kolika će biti vjerojatnost mutacije. To je bitno jer veliki postotak mutacije može znatno unazaditi rješenje dok vrlo mali postotak moguće da neće ni utjecati na raznolikost.

Kao i kod gotovo svih stvari o kojima se pričalo u ovom radu, potrebno je naći balans za koji će se dobiti poželjni rezultati. Ta vjerojatnost mutacije isto tako ovisi o problemu koji se rješava, koliko jedinka ima gena itd.

Implementacija je vrlo jednostavna. Potrebno je proći kroz sve gene djeteta i za svaki odrediti hoće li mutirati. To se može tako da se kaže npr. vjerojatnost $p \in [0, 1]$ te se za svaki gen bira nasumični broj $x \in [0, 1]$ te ako je $x \leq p$ gen će mutirati.

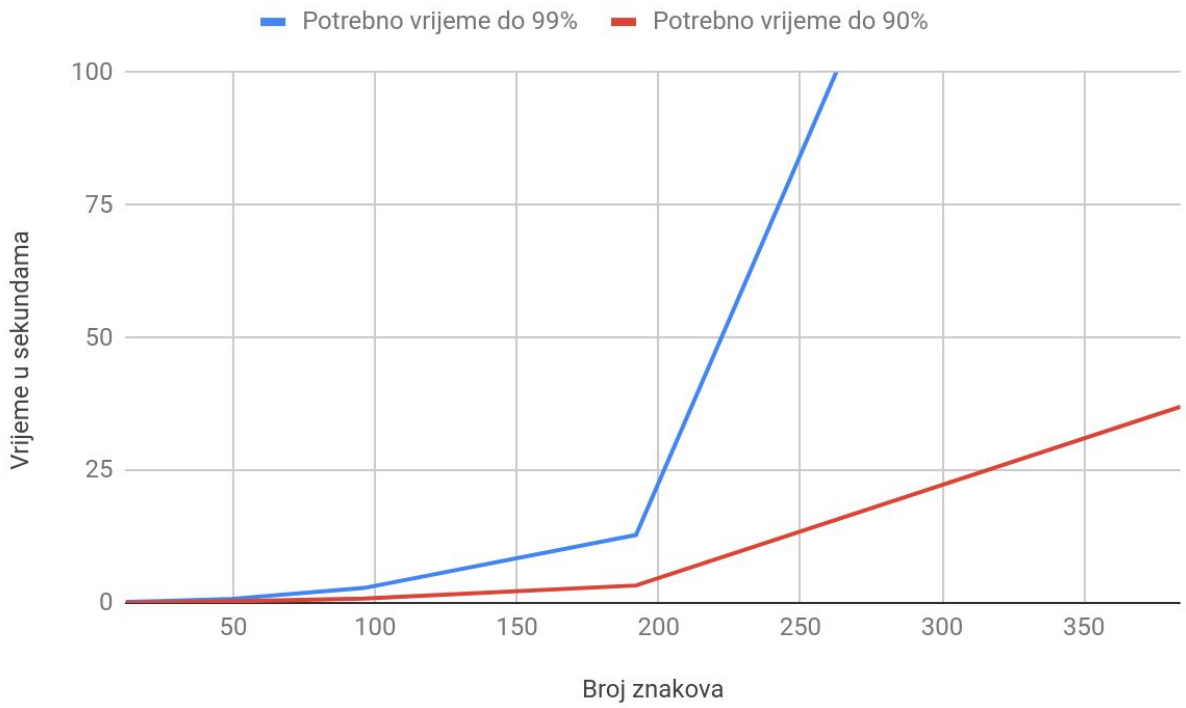
Primjer mutacije vidi se na sljedećoj slici.



Slika 4. Vizualizacija mutacije

2.4.3 Rezultati algoritma za dobivanje fraze

Algoritam opisan u poglavljima prije valja i testirati. Na sljedećoj slici je prikazan graf koji prikazuje koliko algoritmu treba da prijeđe stupanj prihvatljivosti od 99% točnosti i koliko mu treba da prijeđe stupanj prihvatljivosti od 90% točnosti.



Slika 5. Vrijeme potrebno da se dođe do određenog postotka točnosti fraze s obzirom na broj znakova

Kao što se iz grafa može vidjeti, potrebno vrijeme drastično raste s porastom broja znakova fraze, ali isto tako se može vidjeti da za manji postotak prihvatljivosti, potrebno vrijeme mnogo blaže raste. Iz tog podatka može se zaključiti kako najviše vremena treba za fino usavršavanje rješenja.

Očito je kako negdje oko 90% točnosti rješenja ponestane raznolikosti u populaciji. To bi se možda moglo riješiti povećanjem vjerojatnosti mutacije kod ulaska u prostor gdje više nema pretjeranog napretka.

Ovaj problem bi se također mogao riješiti uvođenjem križanja s više roditelja umjesto samo s dva.

3 Zaključak

Genetski algoritam kao takav je vrlo jednostavan u samoj svojoj ideji, ali postoje razne njegove implementacije. Čak i najjednostavniji algoritam, koji je bio obrađen u poglavlju prije, ima vrlo mnogo prostora za fino podešavanje.

Genetskim algoritmima rješavaju se teški problemi za koje se ne traži nužno potpuno točno rješenje već ono dovoljno dobro.

Također se može vidjeti kako je raznolikost populacije vrlo bitan faktor u dobivanju rješenja te kako se svi dijelovi algoritma grade oko toga da se maksimizira raznolikost, a istovremeno poboljša rješenje. Važnost raznolikosti se mogla vidjeti u poglavlju 2.4.3 gdje se jasno vidi kako je potrebno vrijeme za rješavanje problema drastično veće kada se ide od 90% do 100% jer u tom intervalu ponestane raznolikosti u odnosu na interval od 0% do 90% za velike fraze.

Još jedna stvar koja je bitna je ta da se genetski algoritam gradi oko problema koji se želi riješiti. Svi njegovi dijelovi od kreiranja populacije, osmišljanja funkcije dobrote, selekcije roditelja do reprodukcije roditelja mogu i hoće znatno utjecati na pronalazak rješenja, a dovoljno je da samo jedna od tih stvari ne valja da rješenje ne bude optimalno. Zbog toga je uvijek potrebno imati na umu prirodu problema čije rješenje se traži.