# GENETSKI ALGORITMI – PREDAVANJE

## 1. UVOD

- Literatura:
  - skripta GA (1. dio) (http://www.zemris.fer.hr/~golub/ga/ga_skripta1.pdf)
  - stranica o GA (http://www.zemris.fer.hr/~golub/ga/ga.html)
  - skripta PIOA (http://java.zemris.fer.hr/nastava/pioa/)
- stohastička metoda optimiranja (*usporediti sa hill-climbing metodama*)
- opisao John H. Holland (1973.)
- uvjet primjene GA: postojanje neodređenog (velikog) broja rješenja optimizacijskog problema
- oponašanje evolucijskog procesa
- skup rješenja – skup jedinki – *populacija*
- svaka jedinka (kromosom) predstavlja jedno rješenje
- svaka jedinka ima svoju ocjenu kvalitete – dobrota (*fitness*)
- evolucija: 'dobre' jedinke preživljavaju i izmjenjuju svojstva, 'loše' izumiru
- struktura genetskog algoritma:

```
pocetak
   stvori populaciju P(0)
   ponavljaj
      odaberi P(t) iz P(t-1)
      primjeni genetske operatore na P(t)
   dok nije zadovoljen uvjet zaustavljanja
kraj
```

- elementi GA (moraju biti definirani):
  - funkcija dobrote (*fitness function*)
  - prikaz rješenja
  - početno generiranje rješenja
  - postupak odabira (selekcije)
  - genetski operatori: križanje i mutacija
  - uvjet zaustavljanja
  - izbor parametara GA
- pretpostavimo **primjer**: *tražimo minimum funkcije jedne varijable*

## 2. FUNKCIJA DOBROTE

- daje ocjenu dobrote (kvalitete) pojedinog rješenja
- nema dodatnih zahtjeva (derivabilnost, neprekinutost...)
- ponašanje funkcije cilja često je nepoznato (najveća i najmanja vrijednost, opseg...)
- definiramo: *dobrota* (*fitness*) pojedinog rješenja je mjera kvalitete rješenja - ne mora biti jednaka funkciji cilja!
- najčešće: za svaki problem definiramo preslikavanje funkcije cilja ($f$) u dobrotu jedinke (F)
- **primjer**: najveća $f$ u populaciji = $f_{MAX}$ (najlošija jedinka); tada je dobrota pojedine jedinke: $F_i = f_{MAX} - f_i$

- općenitiji pristup: preslikavanje u zadani interval (*windowing*)
- dobrota: $F_i = a + (b-a)\frac{(f_i - f_{worst})}{(f_{best} - f_{worst})}$ (valjano i za minimizaciju i maksimizaciju!)
- u oba slučaja dobrota je u intervalu $[a,b]$!

## 3. PRIKAZ RJEŠENJA

- konačan broj mogućih rješenja (određeni broj bitova)
- uvjet: definirana donja i gornja granica intervala [dg, gg] za moguća rješenja, za sve varijable
- višedimenzijske funkcije – više brojeva u jednom kromosomu (vektor)

### a. Binarni prikaz

- originalni GA: **binarni prikaz** (n bitova s $2^n$ vrijednosti)
- svaki kromosom: niz od *n* bitova – jedna vrijednost x iz [dg, gg]

| | | |
|---|---|---|
| **000...00** | b = 0 | **dg** |
| **111...11** | b = $2^n$-1 | **gg** |

- oznake: b - binarna vrijednost kromosoma; dd,gg - donja i gornja granica; n - broj bitova; x - realna vrijednost

***preslikavanje***

$$x = dg + \frac{b}{(2^n - 1)}(gg - dg) \qquad b = \frac{x - dg}{gg - dg}(2^n - 1)$$

- željena preciznost rješenja (p – broj znamenki iza decimalne točke) određuje duljinu kromosoma

$$n \geq \frac{\log\lfloor 1 + (gg - dg)\cdot 10^p \rfloor}{\log 2}$$

- npr. gg-dg = 100, p = 4 decimale, n >= 20

### b. Prikaz Grayevim kodom

- motivacija: udaljenost kromosoma je jednaka i u problemskoj i u algoritamskoj domeni!
- binarni kromosom: b = {b1 b2 ... bn}
- grayev kromosom: g = {g1 g2 ... gn}

```
binarni --> gray
g1 = b1;
za i = 2 do n
   gi = bi-1 XOR bi;
```

```
gray --> binarni
b1 = v = g1;
za i = 2 do n
   ako (gi == 1) v = NOT v;
   bi = v;
```

### c. Prikaz brojem s pomičnim zarezom

- jednostavnije za implementaciju, puno brže
- kvaliteta rješenja ovisi o algoritmu i problemu

## 4. POČETNO GENERIRANJE RJEŠENJA

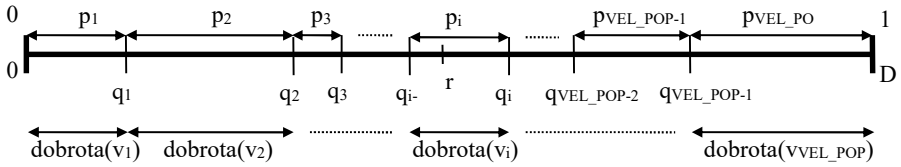
- najčešće: slučajno izabrana rješenja
- **primjer**: slučajne vrijednosti iz [dg, gg]
- problem: primjenjivost slučajno dobivenih rješenja (nevaljana rješenja)
- ako je skup rješenja teško načiniti – neka druga metoda

## 5. ODABIR (SELEKCIJA)

- preživljavanje 'dobrih' i odumiranje 'loših' jedinki
- različiti postupci odabira razlikuju se po načinu određivanja dobrih i loših jedinki
- ideja: svaka jedinka ima određenu *vjerojatnost preživljavanja* koja je (manje ili više) proporcionalna s dobrotom jedinke
- parametar: veličina populacije, N – u većini implementacija je konstantna
- vrste *postupka odabira*:
  - generacijski (*generational*) – nova populacija se stvara od kopija nekih jedinki stare populacije
  - eliminacijski (*steady-state*) – neke jedinke se eliminiraju a nove ih nadoknađuju
- deterministički odabir ne daje dobre rezultate! (npr. eliminacija najlošijih)
- u obje vrste postupka odabira potrebno je definirati **operator odabira**:
  - operator odabira vraća jednu jedinku iz (pod)skupa jedinki na temelju njihove dobrote

a. **Jednostavni generacijski odabir** (*roulette-wheel selection*)

- generacijskog tipa, uz korištenje kotača ruleta kao operatora odabira
- dobrota jedinke $D_i$, ukupna dobrota D (suma $D_i$)
- dobrote se slažu na pravac




  - generira se slučajni broj [0, D] i odabire jedan kromosom za sljedeću generaciju
  - ponavlja se N puta
- bolje jedinke – više primjeraka iste, vjerojatnost odabira proporcionalna s dobrotom
- nedostaci:
  - dobrota mora biti pozitivna
  - jako dobra jedinka dobiva puno kopija – zagušenje populacije
  - velik utjecaj iznosa dobrote – često potrebno skaliranje

**b. Jednostavni eliminacijski odabir** *(elimination selection)*

- eliminacijskog tipa, uz korištenje istog operatora odabira (kotač ruleta)
- određeni postotak populacije se eliminira, a nove jedinke se stvaraju uporabom genetskih operatora
- definira se mjera *nekvalitete*, npr. $D_i^{-1} = D_{MAX} - D_i$ (kazna)
  - opet se gradi pravac, ali s kaznama
  - generira se slučajni broj i odabire kromosom za eliminaciju
  - nakon uklanjanja jedne jedinke, pomiče se mjerilo pravca s kaznama
- parametar: postotak eliminacije – koliki broj jedinki se eliminira

c. **Turnirski odabir** (*tournament selection*)

- korištenje mehanizma turnira kao *operatora* odabira; potupak može biti eliminacijskog ili generacijskog tipa
- eliminacijski (*preporučeni*): slučajni odabir *k* jedinki, najlošija među njima se eliminira i zamjenjuje novom (uporabom genetskih operatora)
    - generacijski: slučajni odabir *k* jedinki, *najbolja* među njima se prenosi u sljedeću generaciju
- *k* - veličina turnira (npr. 3)
- pogodan za paralelno izvođenje, jednostavan za implementaciju

**Neka svojstva postupaka selekcije**

- **Selekcijski pritisak** (SP) - omjer vjerojatnosti preživljavanja dobrih i loših jedinki
- za operator kotača ruleta, prikladan način definicije dobrote uz zadani SP:

$$F_i = 1 + (SP - 1)\frac{(f_i - f_{worst})}{(f_{best} - f_{worst})}$$

- uz turnirski odabir, veličina turnira utječe na selekcijski pritisak
- **Elitizam** – očuvanje najbolje jedinke
- pokazuje se korisnim (spriječava opadanje kvalitete trenutno najboljeg rješenja)
- inherentno ugrađeno u eliminacijske odabire – trenutno najbolja jedinka se nikada ne eliminira (*objasniti*)

```
pocetak (GA s k-turnirskim odabirom)
   stvori populaciju P(0)
   ponavljaj
      odaberi k jedinki iz populacije
      pronadi i obriši najlošiju od k odabranih
      generiraj novu jedinku pomoću genetskih operatora
   dok nije zadovoljen uvjet zaustavljanja
kraj
```

## 6. KRIŽANJE

- binarni operator
- koristi se za stvaranje novih jedinki (nakon eliminacije postojećih)
- dvije jedinke – roditelji, prenose svojstva na rezultat – dijete
- parametar: vjerojatnost križanja $p_c$ (samo kod generacijskog odabira, nakon stvaranja nove populacije)

**a. Binarni prikaz**

- **Križanje s jednom točkom prekida** (*single-point crossover*)
- **Križanje s dvije ili više točaka prekida** – analogno (*multi-point crossover*)
- **Jednoliko (uniformno) križanje**
    - promatra se svaki par bitova (*3 kombinacije*)
    - za svako križanje generira se slučajni niz bitova R (slučajni kromosom)
    - **DIJETE = AB + R(A⊕B).**
- **Segmentirano križanje** (*segmented*) - prepisuju se bitovi iz jednog roditelja; definira vjerojatnost promjene roditelja (*segment switch rate*) nakon svakog bita
- *half-uniform, shuffle, reduced surrogate, non geometric, random respectful...*

- u slučaju višedimenzijskih kromosoma, na pojedine elemente kromosoma mogu se primijeniti isti operatori
- dodatno: jednostavno križanje: zamjena dijelova višedim. kromosoma
- kada koje križanje? velike populacije, veliki kromosomi – 1 ili 2 točke prekida; male populacije – uniformno

**b. Broj s pomičnim zarezom**

- jedna (ili više) točaka prekida: zamjena dijelova vektora realnih vrijednosti
- **Aritmetičko križanje**: slučajni broj između vrijednosti 2 kromosoma roditelja
  - $D = a \cdot X_1 + (1-a) \cdot X_2,\ a \in [0,1]$
- varijante:
  - može se primijeniti na jednu varijablu (*single arithmetic crx*)
  - sve varijable s istim parametrom $a$ (*whole arithmetic crx*)
  - sve varijable s različitim parametrom $a$ za svaku varijablu (*local arithmetic crx*)
- **Heurističko križanje**: $D = a \cdot (X_2 - X_1) + X_2,\ a \in [0,1]$, gdje $X_2$ predstavlja bolje od dva rješenja!
  - potrebno provjeriti narušavanje ograničenja (po potrebi ponoviti)
- *BGA, BLX, SBX, discrete, flat, average...*

## 7. MUTACIJA

- unarni operator
- uloga mutacije:
  - izbjegavanje lokalnih optimuma
  - obnavljanje izgubljenog genetskog materijala (npr. sve jedinke imaju isti bit na nekom težinskom bitnom mjestu)

**a. Binarni prikaz**

- parametar: vjerojatnost mutacije *jednoga bita*, $p_m$ (obično 0.001 - 0.01)
- vjerojatnost mutacije kromosoma: $p_M \approx 1 - (1 - p_m)^n$, (n je broj bitova u kromosomu)
- primjer: $p_m$ = 0.005 (mutira se 5 bitova na 1000), n = 32; tada $p_M$ = 0.148 (14.8%, tj. svaki 7. kromosom će biti mutiran)
- očekivani ukupni broj mutacija uz N novih članova populacije: $p_M$*N
- **Jednostavna mutacija** (*simple mutation*): slučajna promjena jednoga bita unutar kromosoma
- **Jednolika mutacija** (*uniform*) - svi bitovi poprimaju slučajne vrijednosti
- **Nejednolika mutacija** (*non-uniform*) - kod "boljih" jedinki može se promijeniti samo podskup bitova manje težinske vrijednosti
  - npr. broj bitova = $1 + n\left(1 - \frac{f_{AVG} - f_i}{f_{AVG} - f_{BEST}}\right)$
- **Granična mutacija** (*boundary*) - kromosom se postavlja na donju ili gornju granicu područja (00...00 ili 11...11)

**b. Broj s pomičnim zarezom**

- parametar: vjerojatnost mutacije *jedinke*
- **Jednolika mutacija** - slučajni broj u intervalu
- **Gaussolika mutacija** - slučajna vrijednost po zadanoj raspodjeli

- **Granična mutacija** - pomak na granicu područja

- *primjer*: GA s eliminacijskim turnirskim odabirom

```
pocetak (GA s k-turnirskim odabirom)
   stvori populaciju P(0)
   ponavljaj
      odaberi k jedinki iz populacije
      pronađi i obriši najlošiju od k odabranih
      nova jedinka = križanje (dvije slučajno odabrane)
      primijeni mutaciju na novu jedinku s vjerojatnošću pM
   dok nije zadovoljen uvjet zaustavljanja
Kraj
```

- usporedba: GA s generacijskim jednostavnim odabirom (*usporediti jednu iteraciju!*)

```
pocetak (GA s generacijskim jednostavnim odabirom)
   stvori populaciju P(0)
   ponavljaj
      provedi N odabira jedinki iz populacije P(t) – duplikati su mogući
      definiraj novu populaciju P(t+1)
      primijeni operator križanja s vjerojatnošću pc na P(t+1)
      primijeni operator mutacije s vjerojatnošću pm na P(t+1)
      P(t) = P(t+1)
   dok nije zadovoljen uvjet zaustavljanja
Kraj
```

## 8. UVJET ZAUSTAVLJANJA

- ne znamo prirodu dobivenog rješenja!
- neki mogući uvjeti zaustavljanja:
  - broj iteracija (generacija)
  - broj evaluacija fje cilja
  - dostignuta vrijednost fje cilja
  - broj iteracija bez poboljšanja
  - vremensko ograničenje
- korisno: omogućiti nastavak rada (spremanje trenutnog stanja GA)

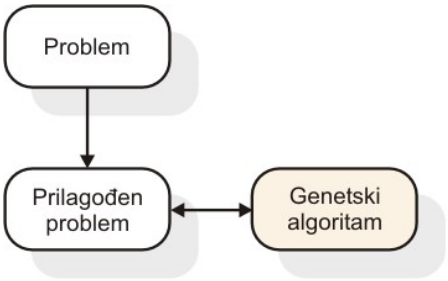
## 9. PARAMETRI GA

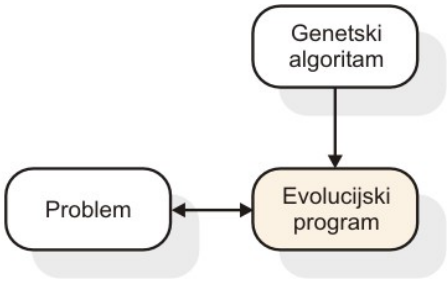
- veličina kromosoma (binarni prikaz), veličina populacije (20-500), postotak eliminacije / vjerojatnost križanja (0.2-0.8), veličina turnira (3-8), vj. mutacije jedinke (0.01-0.5) itd.
- drastični utjecaj na učinkovitost algoritma
- razvoj metoda prilagodljivih vrijednosti parametara

## 10. POGLED IZVANA

- GA – podvrsta *evolucijskih algoritama*

- neke druge stohastičke metode: genetsko programiranje (*genetic programming*), simulirano kaljenje (*simulated annealing*), evolucijske strategije, tabu pretraživanje (*taboo search*), optimizacija kolonijom mrava (*ant colony optimization*), optimizacija rojem čestica (*particle swarm*), ...
- dva pristupa:

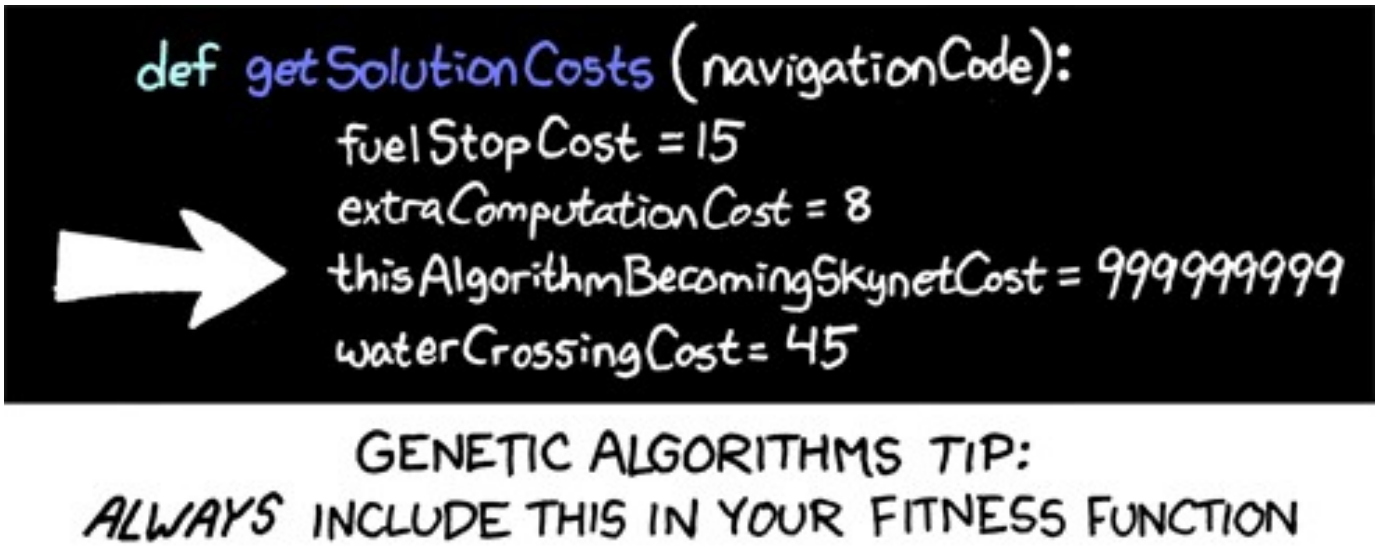



- prilagođavanje problemu obuhvaća: definicija prikaza rješenja te genetskih operatora za taj prikaz
  - npr. posebni skup operatora za TSP (*travelling salesman*) problem, raspoređivanje itd.
- drugi načini prikaza rješenja:
  - permutacijski vektor, vektor bitova, cijelih brojeva, matrice, stabla, ...
- srodna metoda: **genetsko programiranje** (*genetic programming, GP*) – pretraživanje prostora *računalnih programa* (algoritama) za rješavanje određenog problema
- rezultati mjerljivi (ili čak bolji) od ljudskih: analogni sklopovi i filteri, pravila za stanične automate, algoritmi raspoređivanja, upravljački algoritmi...

## 11. PREDNOSTI I NEDOSTACI

| ***Prednosti*** | ***Nedostaci*** |
|---|---|
| proizvoljni optimizacijski problem<br>puno mogućnosti nadogradnje i povećanja učinkovitosti<br>ponavljanje postupka rješavanja<br>daje skup rješenja<br>višemodalni problemi<br>jednostavnost izvedbe | često potrebno prilagoditi problem ili algoritam<br>velik utjecaj parametara<br>priroda rješenja je nepoznata<br>nema 100% učinkovitosti<br>sporost izvođenja |




(*Just make sure you don't have it maximize instead of minimize.*)