GENETSKI ALGORITMI – PREDAVANJE

UVOD

- Literatura:
  - skripta GA (1. dio) (http://www.zemris.fer.hr/~golub/ga/skripta1/ga_skripta1.doc)
  - stranica o GA (http://www.zemris.fer.hr/~golub/ga/ga.html)
- 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 cilja, 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*

FUNKCIJA CILJA

- 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*)

- za minimizaciju $F_i = a + (b-a)\frac{(f_{MAX} - f_i)}{(f_{MAX} - f_{MIN})}$, za maksimizaciju $F_i = a + (b-a)\frac{(f_i - f_{MIN})}{(f_{MAX} - f_{MIN})}$
- u oba slučaja dobrota je u intervalu $[a,b]$!

PRIKAZ RJEŠENJA

- konačan broj mogućih rješenja (određeni broj bitova)
- primjer: definiramo donju i gornju granicu intervala [dg, gg] za moguća rješenja
- najčešće: binarni prikaz (n bitova sa 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**

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

$$n \geq \frac{\log\left[(gg - dg) * 10^p + 1\right]}{\log 2}$$

- npr. gg-dg = 100, p = 4 decimale, n >= 20
- višedimenzijske funkcije – više brojeva u jednom kromosomu (vektor)

POČETNO GENERIRANJE RJEŠENJA

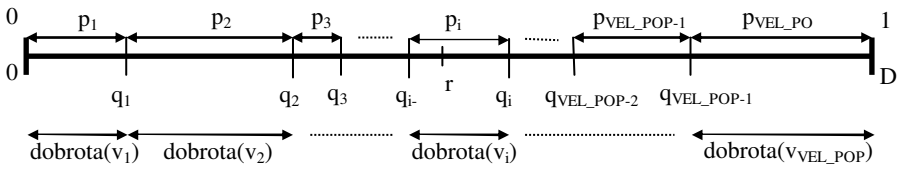
- najčešće: slučajno izabrana rješenja
- **primjer**: slučajne vrijednosti iz [dg, gg]
- problem: primjenjivost slučajno dobivenih rješenja (nevaljana rješenja)
- ako je skup rješenja teško načiniti – neka druga metoda

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 (VEL_POP) – u većini implementacija je konstantna
- vrste 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 odabira potrebno je definirati metodu izbora jedne jedinke iz (pod)skupa jedinki na temelju njihove dobrote – taj postupak obavlja se uporabom nekog **operatora odabira**
- **Generacijski jednostavni 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 poslažu 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 dobre jedinka – puno kopija – zagušenje populacije
  - velik utjecaj iznosa dobrote – često potrebno skaliranje
- **Eliminacijski jednostavni odabir**
- 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 (*nacrtati*)
  - 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
- **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

- Svojstvo (bilo koje vrste) odabira: **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
      pronađi i obriši najlošiju od k odabranih
      generiraj novu jedinku pomoću genetskih operatora
   dok nije zadovoljen uvjet zaustavljanja
kraj
```

KRIŽANJE

- binarni operator
- koristi se za stvaranje novih jedinki (nakon eliminacije postojećih)
- dvije jedinke - roditelji, prenose svojstva na rezultat - dijete
- **Križanje s jednom točkom prekida** (*nacrtati*)
- **Križanje s dvije ili više točaka prekida** - analogno
- **Jednoliko (uniformno) križanje**
- promatra se svaki par bitova (*nacrtati 3 kombinacije*)
- svaki put se generira slučajni niz bitova R (slučajni kromosom)
- **DIJETE = AB + R(A⊕B).**
- kada koje križanje? velike populacije, veliki kromosomi - 1 ili 2 točke prekida; male populacije - uniformno
- parametar: vjerojatnost križanja p_c (samo kod generacijskog odabira, nakon stvaranja nove populacije)

MUTACIJA

- unarni operator
- **Jednostavna mutacija**: slučajna promjena jednoga bita unutar kromosoma
- 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
- uloga mutacije:
  - izbjegavanje lokalnih optimuma
  - obnavljanje izgubljenog genetskog materijala (npr. sve jedinke imaju isti bit na nekom težinskom bitnom mjestu)

```
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
```

UVJET ZAUSTAVLJANJA

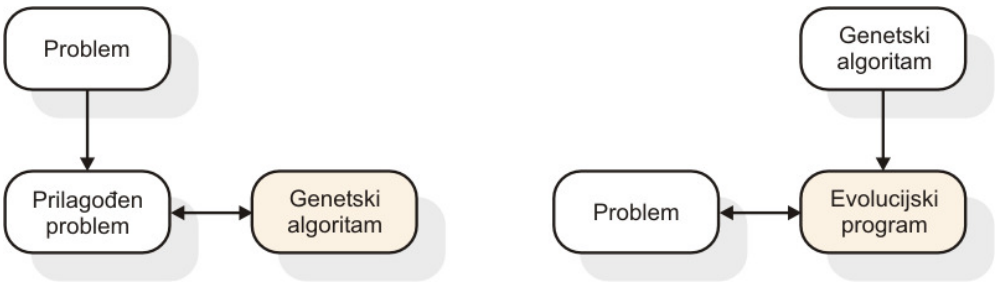
- ne znamo prirodu dobivenog rješenja
- neki mogući uvjeti zaustavljanja:
  - broj iteracija (generacija)
  - broj evaluacija fje cilja
  - dostignuta vrijednost fje cilja
  - broj iteracija bez poboljšanja
  - vremensko ograničenje
- korisno: omogućiti nastavak rada (spremanje trenutnog stanja GA)

PARAMETRI GA

- veličina kromosoma, veličina populacije (20-500), postotak eliminacije (0.2-0.8), veličina turnira (3-8), vjerojatnost križanja (0.2-0.8), vj. mutacije (0.001-0.02) itd.
- drastični utjecaj na učinkovitost algoritma
- razvoj metoda prilagodljivih vrijednosti parametara

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.

- srodna metoda: **genetsko programiranje** (*genetic programming, GP*) – pretraživanje prostora *računalnih programa* (algoritama) za rješavanje određenog problema
- najnoviji rezultati mjerljivi (ili čak bolji) od ljudskih: kvantni algoritmi, analogni sklopovi i filteri, pravila za stanične automate, algoritmi raspoređivanja...

PREDNOSTI I NEDOSTACI

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