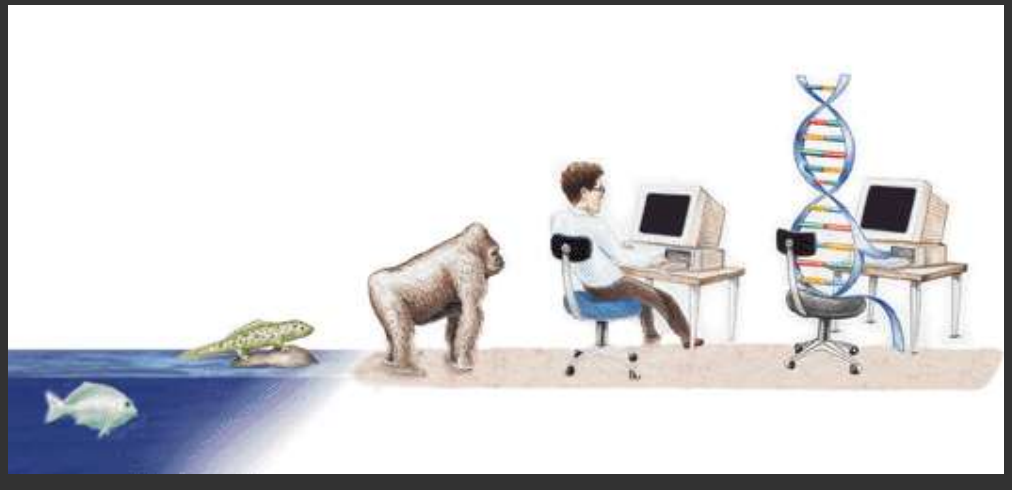
VEŠTAČKA INTELIGENCIJA

EVOLUCIONI PRISTUPI:
GENETSKI ALGORITMI

Sadržaj

- Definicija
- Pregled koraka GA
- Operatori GA



Prof. dr Leonid Stoimenov

Osnova

- Razvio ih je Džon Holand 70-tih
- Genetski algoritmi (GA) su deo **evolucione obrade** (eng. *evolutionary computing*).
  - Inspirisani su Darvinovom teorijom evolucije.
- Problemi se rešavaju korišćenjem evolucionog procesa koji na kraju daje **"najbolje" rešenje**
  - rešenje evoluira
  - Kombinacija informacija iz dobrih rešenja (crosover)
- Karakteristike:
  - Nije baš brz
  - Dobra heuristika za kombinatorne probleme
  - Mnogo varijacija, tj reprodukcionih modela, operatora

Primena genetskih algoritama

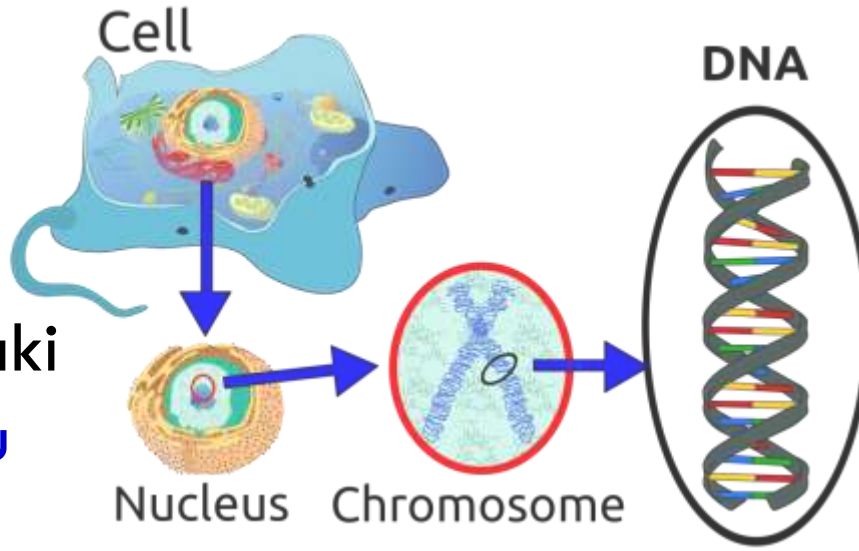
- Genetski algoritmi se primenjuju kada se problem ne može rešiti algoritamski, ali je moguće proveriti valjanost pojedinih rešenja.
- Upotrebom genetskih algoritama nije zagarantovano nalaženje najboljeg rešenja, tj. nije moguće dokazati da je dobijeno rešenje optimalno.
- Rešenje koje se dobija genetskim algoritmom se zato naziva **zadovoljavajuće rešenje**.

Primena GA

Domain	Application Types
Control	gas pipeline, pole balancing, missile evasion, pursuit
Design	semiconductor layout, aircraft design, keyboard configuration, communication networks
Scheduling	manufacturing, facility scheduling, resource allocation
Robotics	trajectory planning
Machine Learning	designing neural networks, improving classification algorithms, classifier systems
Signal Processing	filter design
Game Playing	poker, checkers, prisoner's dilemma
Combinatorial Optimization	set covering, travelling salesman, routing, bin packing, graph colouring and partitioning

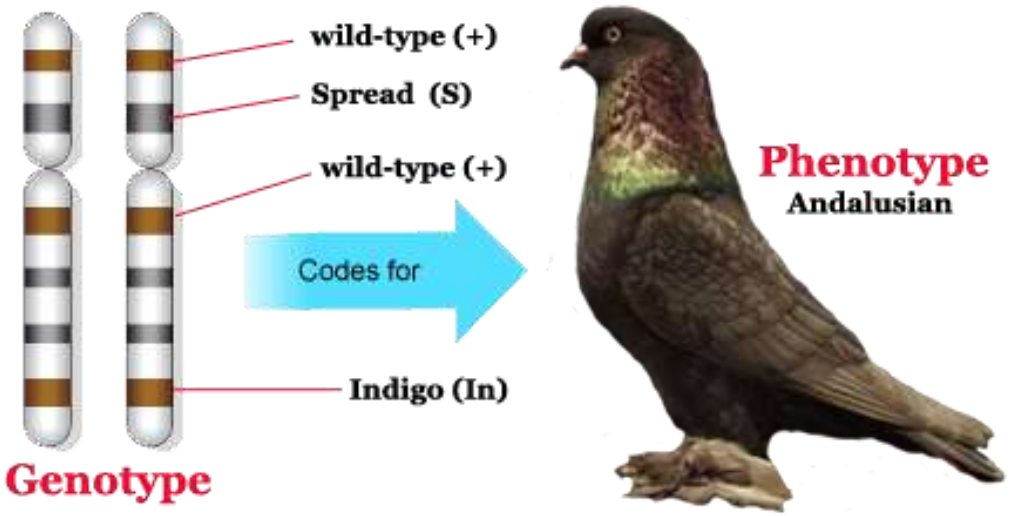
Biološka osnova

- Svi živi organizmi se sastoje iz ćelija.
- U svakoj ćeliji nalazi se isti skup **hromozoma**.
- Hromozomi sadrže DNK lance i predstavljaju "opis" celokupnog organizma.
- Hromozomi se sastoje iz **gena** koji predstavljaju blokove DNK.
- Svaki gen opisuje pojedini protein.
- U osnovi može se reći da svaki gen opisuje pojedinu osobinu organizma (npr. boja očiju).



Biološka osnova

- Svaki gen ima svoj položaj u hromozomu koji se naziva **lokus**.
- Ukupan skup genetskog materijala se naziva **genom**.
- Određeni skup gena u genomu se naziva **genotip**.



Biološka osnova (reprodukcija)

- U toku reprodukcije dolazi do **rekombinovanja** (**crossover**) gena roditelja čime se formiraju novi hromozomi.
- Novi potomak koji tako nastaje može da *mutira*.
- **Mutacija** dovodi do malih promena DNK elemenata.
  - Mutacije su najčešće izazvane greškama u "kopiranju" gena od roditelja.
- **Dobrota** (**fitness**) novonastalog organizma se meri njegovim uspehom u životu (preživljavanju).

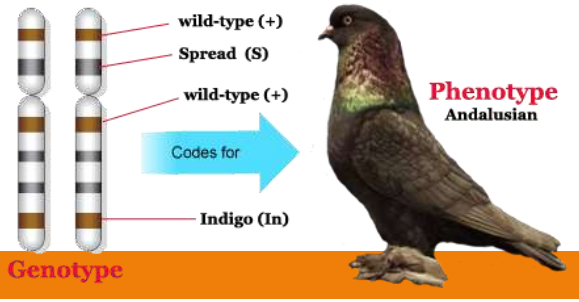
Genetski Algoritam (GA)

- Holland-ov originalni GA je poznat kao *Simple Genetic Algorithm* (SGA)
- Ostali GA danas koriste različite:
  - Reprezentacije
  - Mutacije
  - Crossover-e
  - Mehanizme selekcije

Izazovi za implementaciju GA

- Izbor kod osnovih elemenata implementacije:
  - Reprezentacija problema za GA
  - Veličina populacije, mutation rate, ...
  - selekcija, politika brisanja …
  - crossover, operacije mutacije …
- Kriterijum završetka
- Performanse, skalabilnost
- Rešenje je dobro koliko i funkcija evaluacije (često najteži deo)

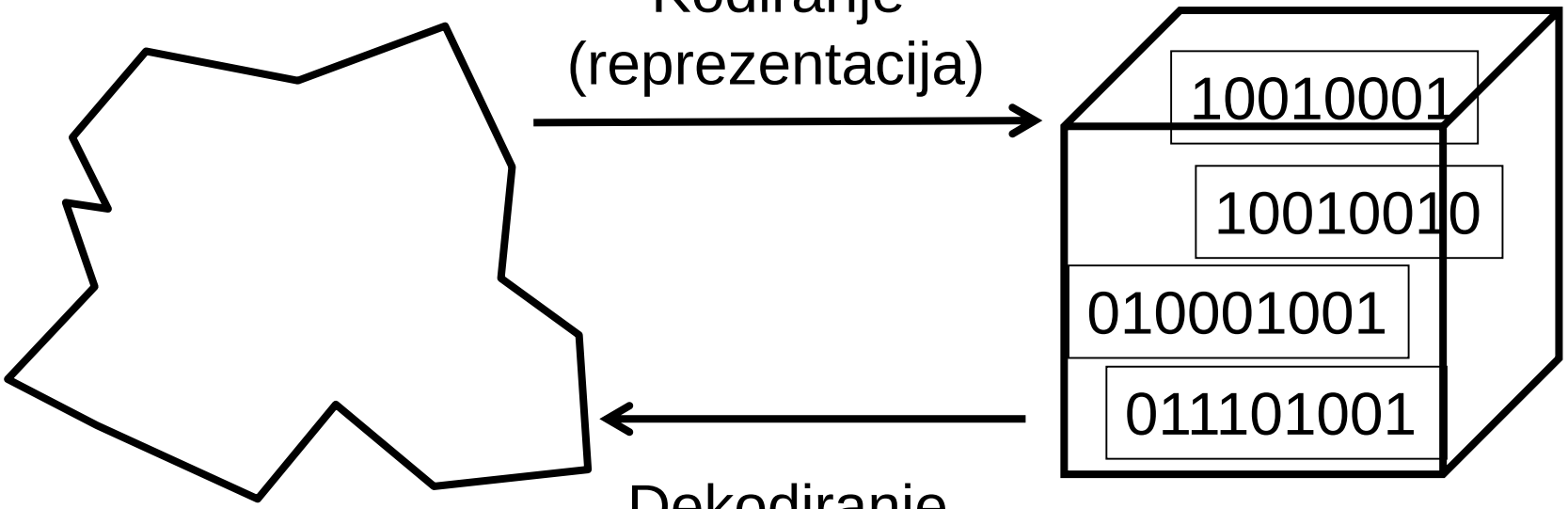
Reprezentacija GA



Prostor fenotipa

Genotip = $\{0,1\}^L$

Kodiranje
(reprezentacija)



Dekodiranje
(inverzna reprezentacija
- interpretacija)

Prostor traženja

- Pri rešavanju nekog problema traži se rešenje koje će biti najbolje od svih.
- Prostor svih mogućih rešenja, tj. skup rešenja u kome se traži najbolje rešenje se naziva **prostor traženja**.
- Svaka tačka u prostoru traženja predstavlja po jedno rešenje problema.
- Za svako moguće rešenje se može odrediti dobrota (fitness) za zadati problem.

Prostor traženja

- Traženje rešenja je ekvivalentno traženju ekstremnih vrednosti (minimum ili maksimum) u prostoru traženja.
- U određenim slučajevima prostor traženja može biti jasno definisan, ali najčešće su poznate samo određene tačke (rešenja).
- Korišćenjem genetskih algoritama, proces traženja (evolucija) generiše druge tačke u prostoru traženja (moguća rešenja).

Opis genetskog algoritma

- Algoritam zahteva polazni skup rešenja predstavljenih odgovarajućim hromozomima
  - **populacija**.
- Rešenja iz jedne populacije se koriste da formiraju novu populaciju u nadi da će ona biti bolja
  - Izbor rešenja za novu populaciju bira se na osnovu **dobrote** (**fitness**) rešenja.
- Ovaj proces se ponavlja sve dok se ne zadovolji određeni uslov (broj populacija, poboljšanje najboljeg rešenja).

Koraci u genetskom algoritmu

1. [**Početak**] Generiše se slučajna populacija od **n** hromozoma.
2. [**Dobrota**] Računa se dobrota **f(x)** za svaki hromozom iz populacije.
3. [**Nova populacija**] se kreira ponavljanjem sledećih koraka:
    1. [**Izbor roditelja**] Biraju se dva hromozoma iz tekuće populacije prema dobroti (veća dobrota, veća šansa za izbor).
    2. [**Rekombinacija**] Novi hromozom se formira kombinovanjem dva izabrana horomozoma.
    3. [**Mutacija**] Novi hromozom mutira sa određenom verovatnoćom.
    4. [**Dodavanje potomka**] Novi hromozom se dodaje u novu populaciju.
4. [**Zamena populacija**] Novo generisana populacija postaje tekuća populacija.
5. [**Uslov za kraj**] Proverava se uslov za završetak algoritma $\rightarrow$ ukoliko je zadovoljen vraća se najbolje rešenje iz tekuće populacije.
6. [**Novi ciklus**] Ako uslov nije ispunjen ponavlja se algoritam od 2. koraka.

Prototip genetskog algoritma

Ulazne veličine

GA(*dobrota, granica dobrote, p, r, m*)

dobrota: *Funkcija koja pridružuje ocenu datoj hipotezi.*

granica dobrote: *Granica koja određuje uslov za zaustavljanje.*

p: *Broj hipoteza koje su uključene u populaciju.*

r : *Deo populacije koji će biti zamenjen u svakom koraku operatorom ukrštanja.*

m: *Procenat onih koje će da mutiraju.*

Prototip genetskog algoritma
Inicijalizacija

- ***Inicijalizacija populacije:***
  - **P** ← **p** slučajno generisanih hipoteza

- ***Procena:***
  - *Za svako h iz P izračunati* ***dobrota(h)***

Prototip genetskog algoritma
Glavna petlja

Dok je [$\mathbf{max}_h$ ***dobrota(h)***] < ***granica_dobrote***

Radi:

Kreiranje nove generacije, Ps:

1. ***Izbor:***

   Izabrati (primenom verovatnoće)
   (1- *r*)*p* članova iz P da budu članovi P_s.

   Verovatnoća $Pr(h_i)$ da hipoteza h_i bude izabrana iz P data je sa:

$$\mathbf{Pr}(h_i) = \frac{dobrota(h_i)}{\sum_{j=1..p} dobrota(h_j)}$$

Prototip genetskog algoritma
Glavna petlja

2. *Rekombinacija:*

- Probabilistički izabrati ***rp*/2** parova hipoteza iz *P*, u skladu sa $Pr(h_i)$.
- Za svaki par (h_1, h_2) proizvesti dva potomka koristeći operator rekombinacije.
- Uključiti sve potomke u P_s.

Prototip genetskog algoritma
Glavna petlja

3. ***Mutacija:***

   Izabrati *m* procenata od članova P_s sa uniformnom verovatnoćom.

   U svakom od njih invertovati jedan, slučajno izabran, bit njegove reprezentacije.

4. ***Izmena:***

   $P \leftarrow P_s$

5. ***Evaluacija:***

   za svaki *h* iz *P* izračunati ***dobrota(h)***.

Vratiti one hipoteze iz P koje imaju najveću dobrotu.

Komentar algoritma

- Dati opis GA je generalan
  - različiti parametri i podešavanja mogu biti implementirani drugačije za različite probleme.
- Pitanja su:
  - Kako kreirati hromozome (koje kodiranje upotrebiti)?
  - Kako birati roditelje?
  - Kako obavljati rekombinaciju i mutaciju?

Operatori GA

- Performanse GA u osnovi zavise od **rekombinacije** (crossover) i **mutacije** hromozoma.
- Operacija rekombinacije se koristi za pravljenje hromozoma potomka od hromozoma roditelja
  - Hromozom potomka neke gene uzima od jednog, a neke od drugog roditelja.
- Mutacija ima za cilj da spreči da sva rešenja iz populacije upadnu u lokalni optimum rešenja.
  - Mutacija nasumično menja pojedine karakteristike kreiranog hromozoma potomka.

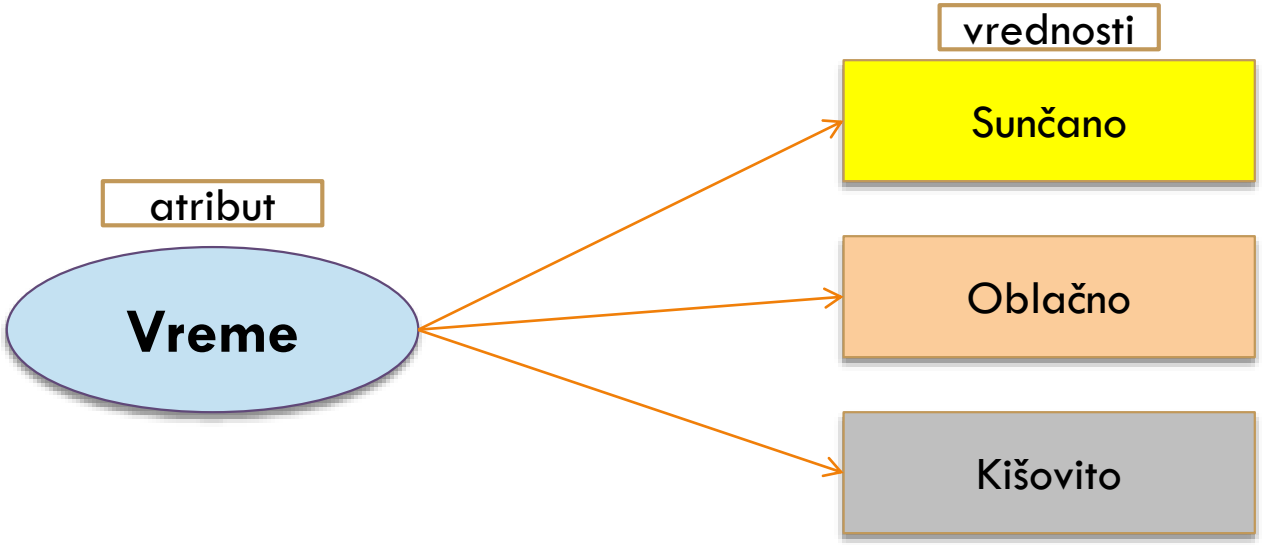
Kodiranje hromozoma

- Kodiranje hromozoma je prvo pitanje sa kojim se suočavamo pri pokušaju da problem rešavamo GA-om.
- Tehnike koje se koriste za rekombinovanje i mutaciju uglavnom zavise od načina kodiranja hromozoma.
- Postoje sledeći načini za kodiranje:
  - binarno kodiranje,
  - permutaciono kodiranje,
  - kodiranje vrednostima i
  - kodiranje stablom.

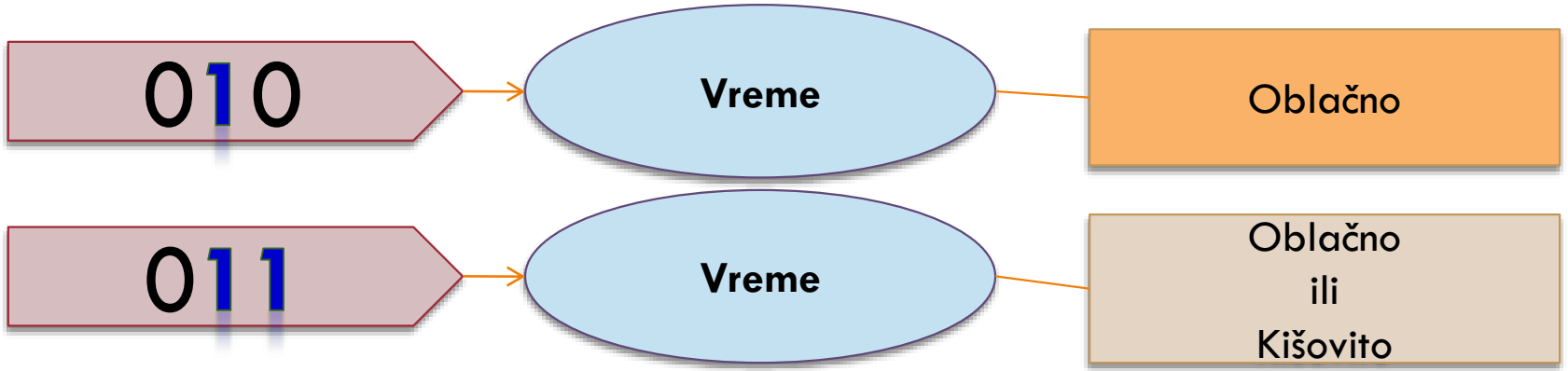
Binarno kodiranje

- Predstavljanje hromozoma nizom bitova (0 i 1)
  - Najčešće se koristi zbog jednostavnosti.
  - jednostavno mogu da se koriste u operatorima ukrštanja i mutacije.
- Primer:
  - Hromozom A: 01101011
  - Hromozom B: 11000010
- Problem kod binarnog kodiranja je taj što ono nije prirodno za određene primene.
  - Zbog toga su često neophodne korekcije nakon rekombinacije i/ili mutacije.

Binarno kodiranje – primer



Kodiranje: Treba da se koristi niz bitova dužine tri, gde svaki bit odgovara jednoj od tri moguće vrednosti



Binarno kodiranje – primer

- Predstavljanje konjunkcije više atributa: jednostavno dopisivanjem odgovarajućih nizova bitova.

- Primer: dodatni atribut *Vetar* koji može da ima vrednosti *Jak* ili *Slab*.

  $$(\textit{Vreme} = \textit{Oblačno} \vee \textit{Kišovito}) \wedge (\textit{Vetar} = \textit{Jak})$$

  predstavljamo nizom bitova:

Vreme	*Vetar*
011	10

Binarno kodiranje – primer

- Predstavljanje pravila

IF *Vetar* = *Jak* THEN *IgrajTenis* = *da*

nizom bitova:

Vreme	***Vetar***	***IgrajTenis***
111	**10**	**10**

Dužina reprezentacije je fiksirana, a podniz na određenoj lokaciji opisuje vrednosti odgovarajućeg atributa.

Binarno kodiranje (primer)

- Primer problema:
  - Problem popune ranca
- Opis problema:
  - Postoji lista stvari za koje se zna vrednost i veličina i ranac određenog kapaciteta.
  - Cilj je napuniti ranac što vrednijim stvarima.
- Kodiranje:
  - Svakoj stvari iz liste se dodeljuje po jedan bit.
  - Vrednost 1 znači da je u rancu, a vrednost 0 da nije.
  - Primer: 0110100110

Permutaciono kodiranje

- Koristi se kod problema uređivanja.
- Svaki hromozom predstavlja sekvencu elemenata čije optimalno uređenje se traži.
  - Hromozom A: 2 7 4 1 3 5 8 6
  - Hromozom B: 5 3 7 2 8 1 4 6
- Pri rekombinovanju i mutiranju hromozoma mora se voditi računa o očuvanju konzistencije hromozoma.

Permutaciono kodiranje (primena)

- Primer problema:
  - Problem trgovačkog putnika
- Opis problema:
  - Dat je skup gradova i njihova međusobna rastojanja koje trgovački putnik treba da obiđe.
  - Cilj je naći redosled gradova tako da se pređe najkraći put.
- Kodiranje:
  - Hromozom opisuje redosled obilaska gradova.
  - Primer: 8 5 6 7 2 3 1 4 9

Kodiranje vrednostima

- Direktno kodiranje vrednostima se koristi kod problema gde se koriste složenije vrednosti.
  - Korišćenje binarnog kodiranja u ovim slučajevima može biti nezgodno.
- Kod kodiranja vrednostima hromozom predstavlja sekvencu vrednosti određenog tipa:
  - Hromozom A: 1,25 -5,24 -0,23 0,01 -3,32 2,74
  - Hromozom B: AHGKATWEM
  - Hromozom C: (nazad), (nazad), (levo), (napred)
- Ovaj tip kodiranja zahteva da se razviju specijalizovane operacije rekombinacije i mutacije koje odgovaraju konkretnom problemu koji se rešava.

Kodiranje vrednostima (primena)

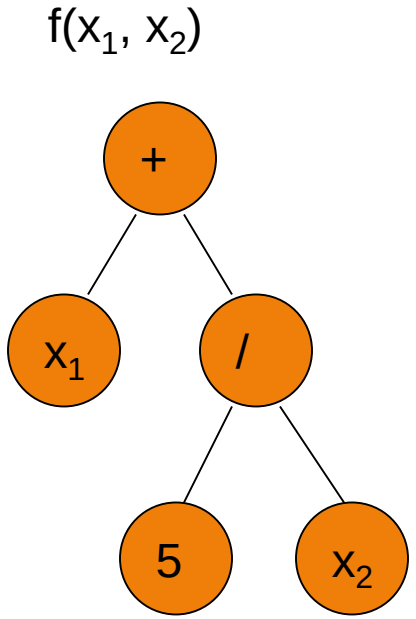
- Primer problema:
  - Nalaženje težina za neuronsku mrežu
- Opis problema:
  - Data je neuronska mreža sa predefinisanom arhitekturom.
  - Cilj je naći težine neurona tako da mreža daje zadovoljavajući izlaz.
- Kodiranje:
  - Hromozom se sastoji iz niza realnih vrednosti koje definišu vrednosti težina neurona u mreži.
  - Primer: 1,25 -5,24 -0,23 0,01 -3,32 2,74

Kodiranje stablom

- Kodiranje stablom se uglavnom koristi za razvoj programa ili izraza
  - **genetsko programiranje**
- Svaki hromozom predstavlja stablo koje u čvorovima sadrži određene objekte:
  - funkcije, operatore, konstante, promenljive, komande prog. jezika i dr.
- Genetsko programiranje je jednostavno u LISP-u jer se tamo program i podaci predstavljaju isto – pomoću stabla (ugnježdene liste).

Kodiranje stablom (primena)

- Primer problema:
  - Nalaženje funkcije koja će da aproksimira zadati skup uređenih parova (x_1, ..., x_n, $f(x_1, ..., x_n)$)
- Opis problema:
  - Dat je skup ulaznih i izlaznih vrednost.
  - Cilj je naći matematičku funkciju koja najbolje opisuje taj skup.
- Kodiranje:
  - Hromozomi opisuju funkcije koje se zadaju stablom.



(+ (x1 (/ 5 x2)))

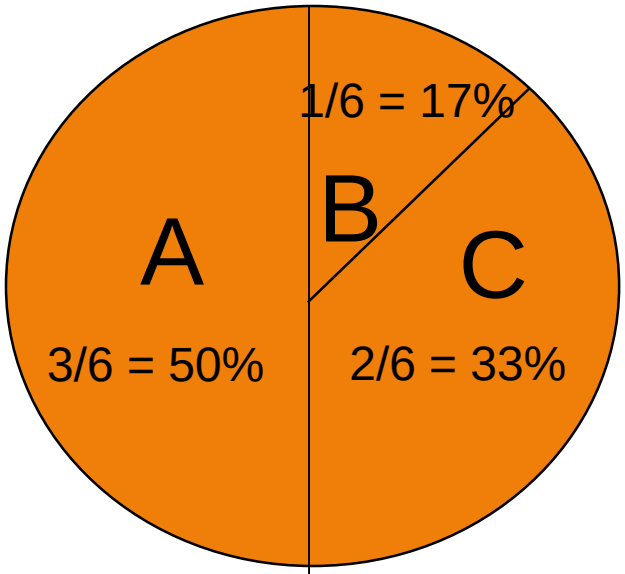
Izbor roditelja – Rulet selekcija

- **Rulet selekcija**
  - Roditelji se biraju na osnovu svoje dobrote.
  - Što je veća dobrota jedinke veća je šansa da bude izabrana za roditelja.

- **Rangiranje**
  - Rešava problem kada postoje velike razlike u dobrotama jedinki.
  - Jedinke se sortiraju po dobroti pa im se dodele rangovi od 1 do N (veličina populacije).
  - Pri izboru roditelja se ne gleda dobrota već rang.
  - Na ovaj način sve jedinke imaju šanse da budu izabrane (mana je sporija konvergencija).

Rulet selekcija – izbor roditelja

- Osnovna ideja: bolje individue imaju veću šansu
  - Šansa je proporcionalna fitness-u
  - Implementacija: rulet (roulette wheel technique)
    - Dodeli svakoj individui deo „točka„ (roulette wheel)
    - Zavrti točak n puta za selekciju n individua



fitness(A) = 3

fitness(B) = 1

fitness(C) = 2

Izbor roditelja – Metod stabilnog stanja

- **Metod stabilnog stanja** se zasniva na ideji da novu generaciju treba da u velikoj meri čine najbolje jedinke iz prethodne generacije.
- Metod funkcioniše na sledeći način:
  - Iz tekuće populacije izbacuje se određen broj najlošijih jedinki.
  - Bira se par najboljih jedinki koje daju onoliko novih potomaka koliko je izbačeno.
  - Ostatak populacije se prenosi u novu generaciju.

Izbor roditelja - Elitizam

- **Elitizam** je uveden da bi se sprečilo gubljenje najboljih rešenja pri kreiranju nove populacije.
- Kod elitizma prvo se kopira par najboljih rešenja u novu populaciju, dok se ostatak nove generacije dobija rekombinacijom i mutacijom.

Rekombinacija

Binarno kodiranje

- Operator *rekombinacije* formira potomke od dva roditeljska niza bitova, tako što kopira određene bitove iz jednog, odnosno drugog roditelja.

- Bit na poziciji *i* svakog potomka je kopiran sa bita sa pozicije *i* jednog od roditelja.

- Izbor od kog roditelja će bit na poziciji *i* biti kopiran određuje se na osnovu dodatnog niza bitova koji se naziva ***maska rekombinacije*** ili ***maska ukrštanja***.

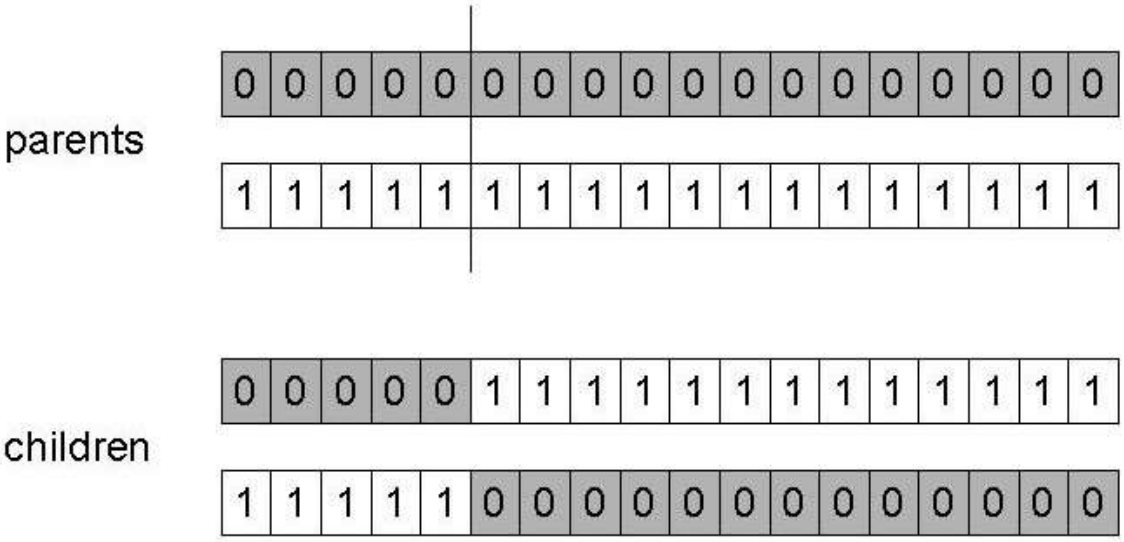
Jednostruko ukrštanje
– sa jednom tačkom prelaska

- Maska ukrštanja se formira tako da počinje nizom od ***n*** **uzastopnih jedinica** i potrebnim brojem nula da bi se kompletirao niz.
- Primenom ove maske dobija se potomak kod kojeg je prvih *n* bitova iz prvog roditelja, a preostali bitovi iz drugog roditelja.
- Svaki put kada se primenjuje jednostruko ukrštanje broj *n* se bira na slučajan način.

Jednostruko ukrštanje

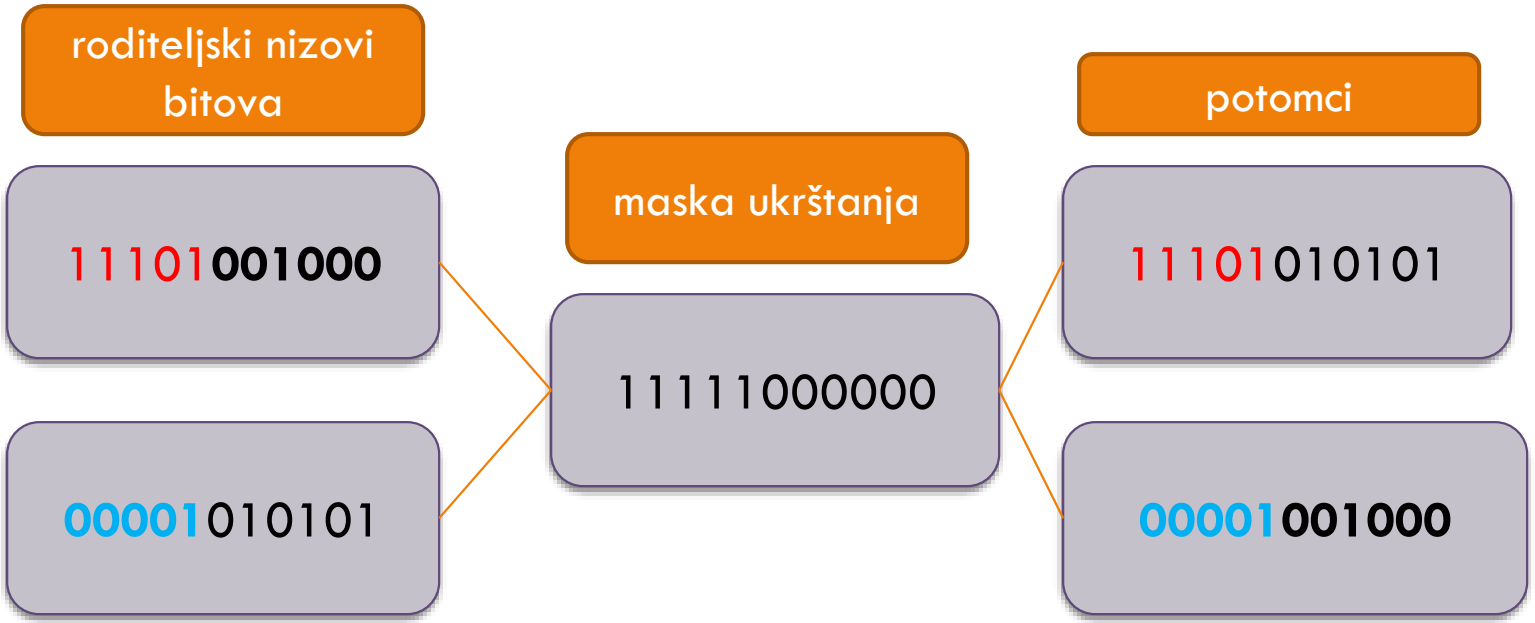
– sa jednom tačkom prelaska (SGA)

- Slučajan izbor tačke za dva roditelja
- Podela roditelja po izabranoj tački prelaska
- Kreiranje dece zamenom ostataka (repova)
- Verovatnoća P_c obično u opsegu (0.6, 0.9)



Jednostruko ukrštanje - Primer

n=5



Dvostruko ukrštanje
– sa dve tačke prelaska

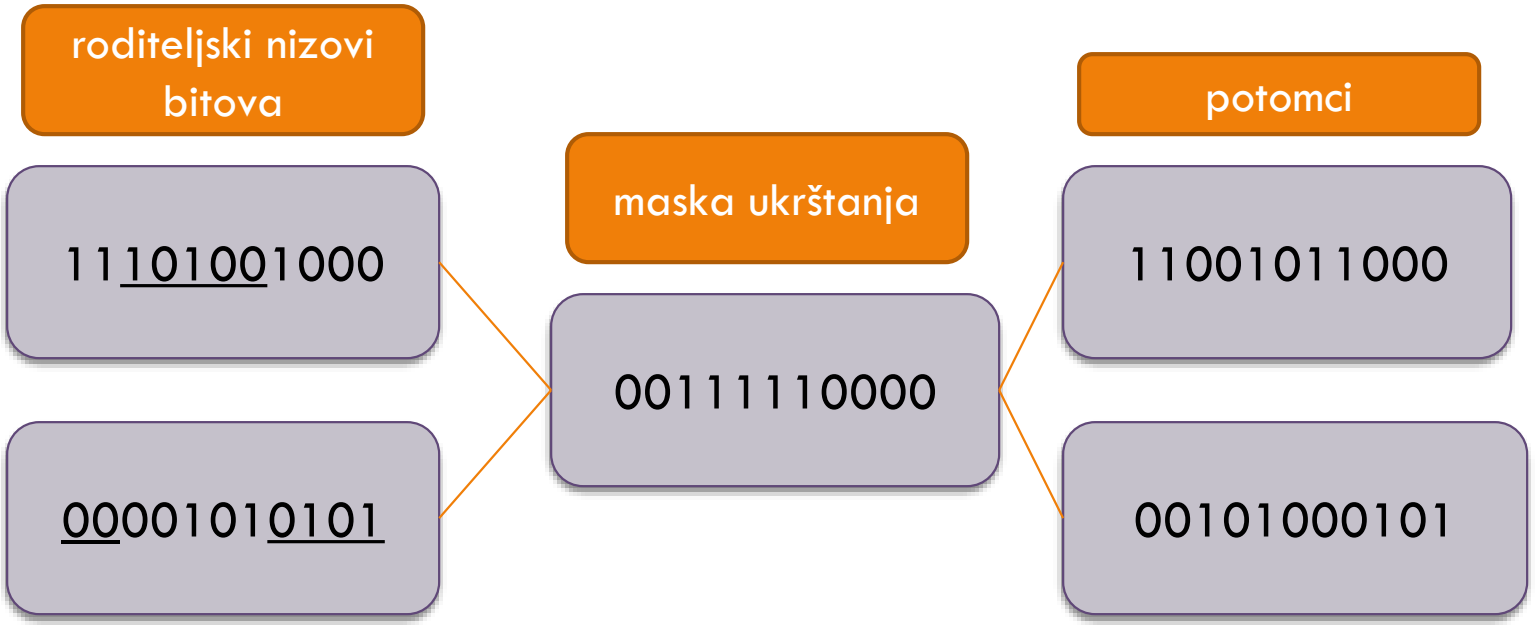
- Maska ukrštanja je niz bitova koji počinje sa n_0 nula, za kojim sledi niz od n_1 jedinica i na kraju je neophodan broj nula da se kompletira niz.

- Primenom ove maske dobija se potomak kod kojeg je središnji deo jednog roditelja upisan na odgovarajuću poziciju u sredinu drugog roditelja.

- Svaki put kada se maska primenjuje, vrednosti n_0 i n_1 se biraju na slučajan način.

Dvostruko ukrštanje – Primer

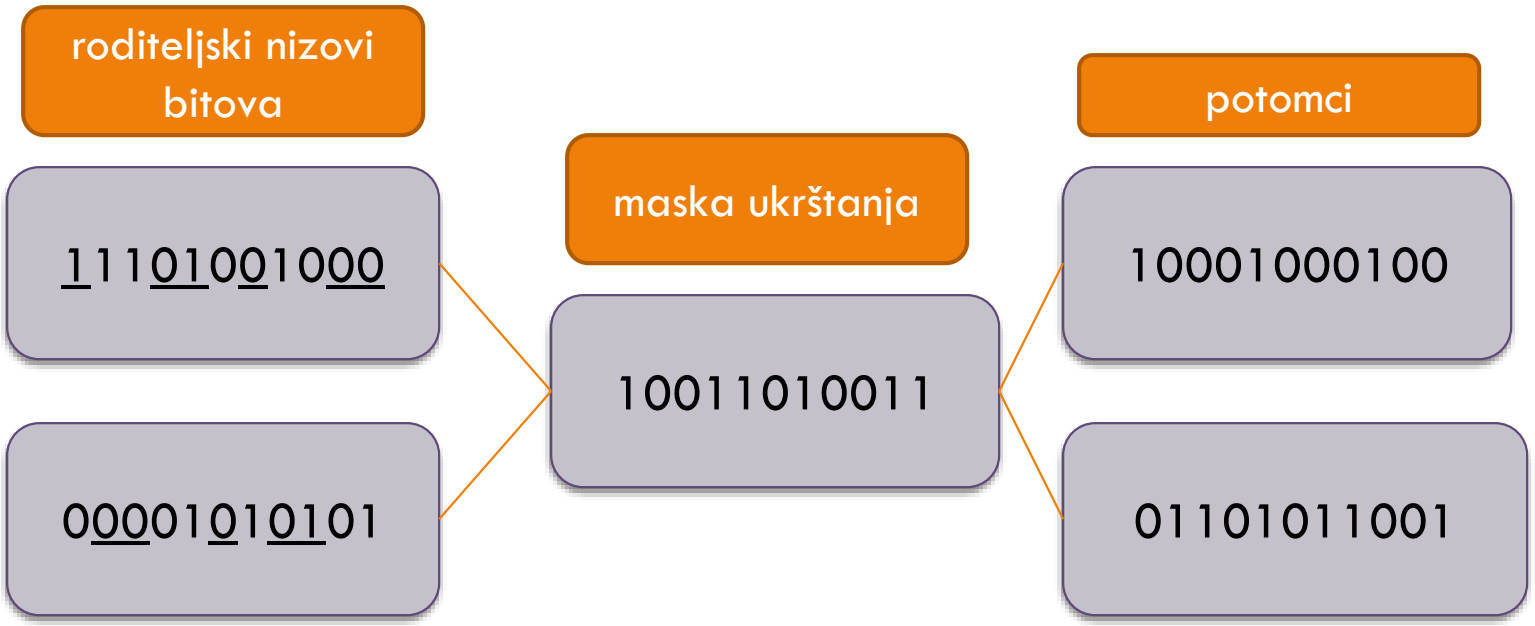
$n_0 = 2$ i $n_1 = 5$



Uniformno ukrštanje

- Maska se formira kao slučajan niz nula i jedinica.
- Potomci se formiraju tako što se bitovi uniformno kopiraju od oba roditelja.

Uniformno ukrštanje – Primer



Rekombinacija, Binarno kodiranje - Pregled

- Sa jednom tačkom prelaska:
  - R1: 0100100110|11011
  - R2: 1110010011|10101
  - P1: 0100100110|10101
  - P2: 1110010011|11011
- Sa dve tačke prelaska:
  - R1: 0100|10011|011011
  - R2: 1110|01001|110101
  - P1: 1110|10011|010101
  - P2: ??
- Uniformno – bitovi potomka se nasumično biraju iz jednog od roditelja
  - R1: 0100100110111011
  - R2: 1110010011010101
  - P1: 0110000010010111
  - P2: ??

Rekombinacija

Permutaciono kodiranje

- Sa jednom tačkom prelaska
  - Bira se tačka u hromozomu.
  - Kopira se sekvenca iz prvog roditelja do tačke prelaska.
  - Nakon toga se dodaje one stavke iz drugog roditelja koje se već ne nalaze u potomku.
- Primer:
  - R1: 1 6 8 2 3 7 9 5 4
  - R2: 2 5 8 4 1 3 7 9 6
  - P : 1 6 8 2 3 5 4 7 9

Rekombinacija

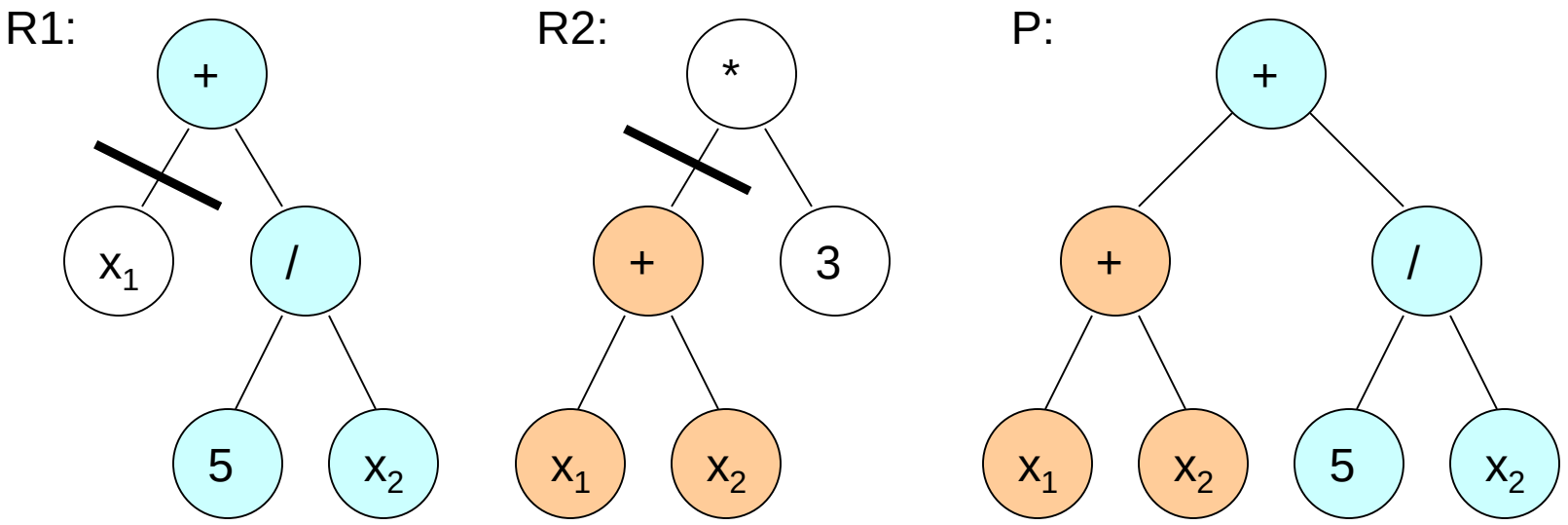
Kodiranje vrednostima

- Mogu se iskoristiti sve varijante rekombinacije iz binarnog kodiranja.
- Primer (sa jednom tačkom prelaska):
  - R1: 1,2 -0,2 | 9,8 5,2 -2,5
  - R2: -3,6 -1,5 | -0,8 7,2 -4,9
  - P : -3,6 -1,5 | 9,8 5,2 -2,5

Rekombinacija

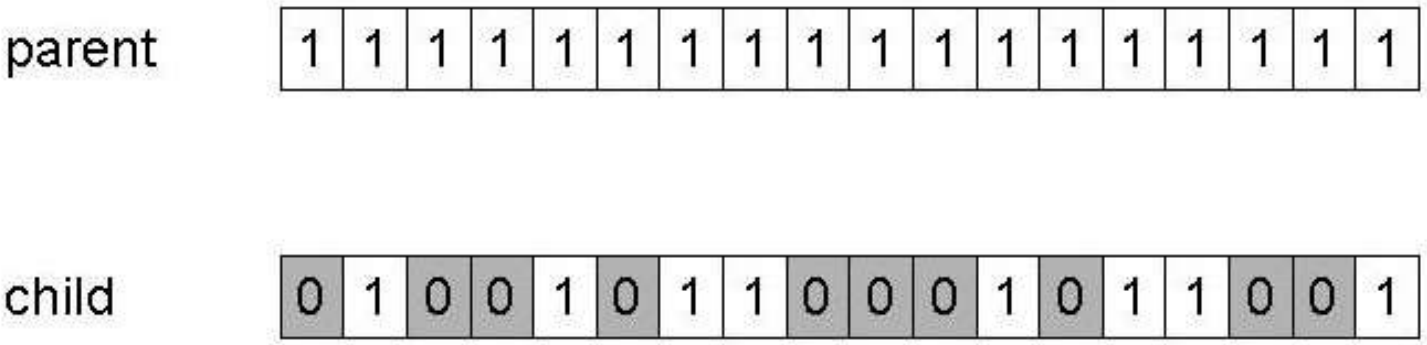
Kodiranje stablom

- Bira se po jedna tačka prelaska u stablima roditelja.
- Stablo potomka se dobija spajanjem "gornjeg" dela stabla jednog i "donjeg" dela stabla drugog roditelja.



Mutacija

- Operator mutacije formira potomka samo od jednog roditelja (menja jedinku mutacijom).
- Proizvodi male nasumične promene.
- Promena gena sa verovatnoćom p_m
- p_m se zove *mutation rate*
  - Vrednost obično između 1/pop_size i 1/ chromosome_length



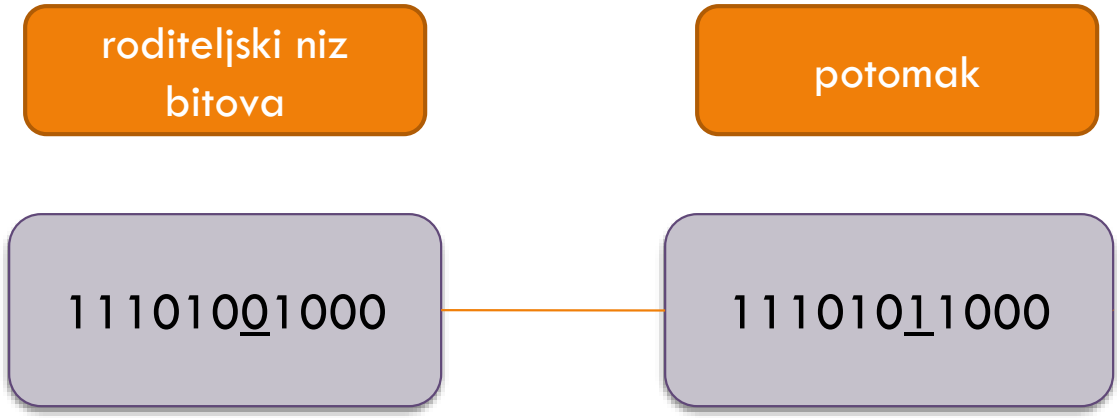
Mutacija

Binarno kodiranje

- Na slučajan način izabere jedan bit niza i promeni njegovu vrednost.
- Invertuju se pojedini slučajno izabrani bitovi u hromozomu:
  - `P  : 0100100110110101`
  - `Pm: 0101100000110100`
- Definiše se verovatnoća sa kojom se bit u hromozomu invertuje
  - **Verovatnoća mutacije**

Operator mutacije – Primer

Binarno kodiranje



Mutacija

Permutaciono kodiranje

- Vrši se zamena (permutacija) određenih slučajno izabranih stavki iz hromozoma:
  - `P  : 1 6 8 2 3 5 4 7 9`
  - `Pm: 1 6 8 7 3 5 4 2 9`
- Verovatnoća mutacije definiše koliko zamena će biti izvršeno.

Mutacija

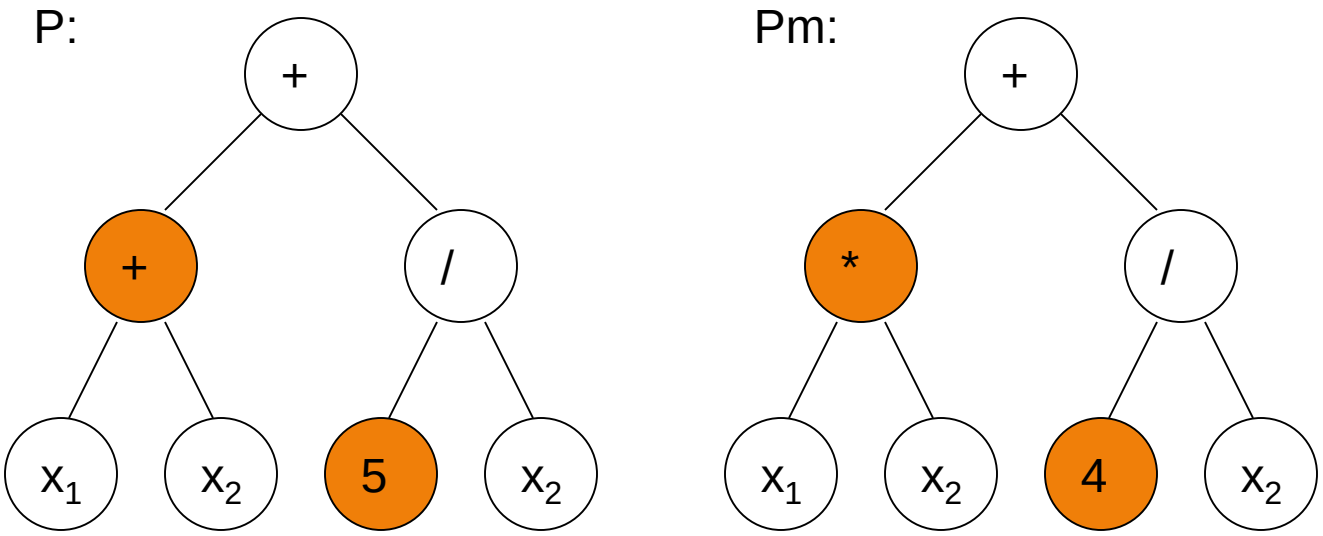
Kodiranje vrednostima

- Mutacija se vrši dodavanjem proizvoljne male vrednosti pojedinim vrednostima koje čine hromozom:
  - P : -3,6 -1,5 9,8 5,2 -2,5
  - +0,6 -0,2
  - Pm: -3,6 -0,9 9,8 5,2 -2,7
- Verovatnoća mutacije definiše koliko vrednosti iz hromozoma će biti modifikovano i kolika će veličina modifikacije biti.

Mutacija

Kodiranje stablom

- Mutacija se vrši izmenama u proizvoljim čvorovima stabla.
- Mogu se menjati operatori, funkcije, konstante, promenljive i dr.



PITANJA?

Dileme?

Komentari?