

# O · P · T · I · M · I · Z · A · C · I · J · A

## PODJELA PROBLEMA

### 1. DOMENA

a) kontinuirani

bilo koja vrijednost iz  $\mathbb{R}$

b) diskretni (kombinatorni problem)

TSP problem

### 2. OGRANIČENJE

a) bez ograničenja

b) s ograničenjem

ograničenja mogu pojavljati se u ograničenjima domene ili eksplisitno zadanim ograda

prostor prihvatljivih rješenja

prostor neprihvatljivih rješenja

težda ograničenja (moraš se zadovoljiti)

meka ograničenja

najmanje ograničenja

najviše ograničenja

1. ugradimo u algoritam

2. ciljna funkcija + karba

mogu biti zahtijevni za izgradnju  
i neefikasni zbog podjele prostora  
pretrage u više nepovezanih dijelova

### 3. BROJ KITERIJA

a) jednokriterijska ocjena rješenja

b) višekriterijska ocjena rješenja

multi-objective (2,3)

many-objective (4...)

jednokriterijska vraćaju jedan broj,  
a višekriterijska vektor brojeva i onda je  
rješenja teško mjerljivo usporediti

## MIN-MAX $f(\vec{x})$

$$\text{UZ OGRANIČENJA} \quad g_j(\vec{x}) \geq 0, j=1,2,\dots,J \\ h_k(\vec{x}) = 0, k=1,2,\dots,K$$

IMPLICITNA OGRANIČENJA

$$x_i^L \leq x_i \leq x_i^U, i=1,2,\dots,n$$

EKSPLISITNO OGRANIČENJE

Np-teški problemi  $\rightarrow$  broj rješenja raste  
barem eksponencijalno pa ne postoji  
efikasni algoritam za pronađak  
optimalnog rješenja

## OPTIMIZACIJSKI POSTUPCI

1. egzaktni  $\rightarrow$  pronađavaju optimalno rješenje iskoristujući pretragu, dinamičko programiranje...

aproximativni  $\rightarrow$  daju garantiju o ogranicama u kojima se rješenje nalazi u odnosu na neoptimalne

2. aproksimativni  $\rightarrow$  heuristički  $\rightarrow$  ne daju garantiju, ne ispituju sva moguća rješenja

pronađavaju dovoljno dobro rješenje

## DOBIVENO RJEŠENJE

1. konstrukcijske heuristike  $\rightarrow$  započinju od pravnog rješenja i grade ga korak po koraku

2. unapredavačke heuristike  $\rightarrow$  imaju početno rješenje koje nastaje unaprijediti

$\hookrightarrow$  metaheuristike

## VRSTE HEURISTIKA

1. problemski specifične

čelično nešto optimizirati, neki specifični problem

jedno rješenje

genetski algoritam, algoritmi lokalne  
pretrage, tabu pretraga, simulirano klijanje...

2. metaheuristike  $\rightarrow$  rješenje

nema pojma koji je problem

populacijske kombinira više rješenja

3. hiperheuristike

pretražujemo u prostoru heuristika, a ne rješenja  $\rightarrow$  izgradnju heuristiku

## PROCES PRETRAŽIVANJA - faze

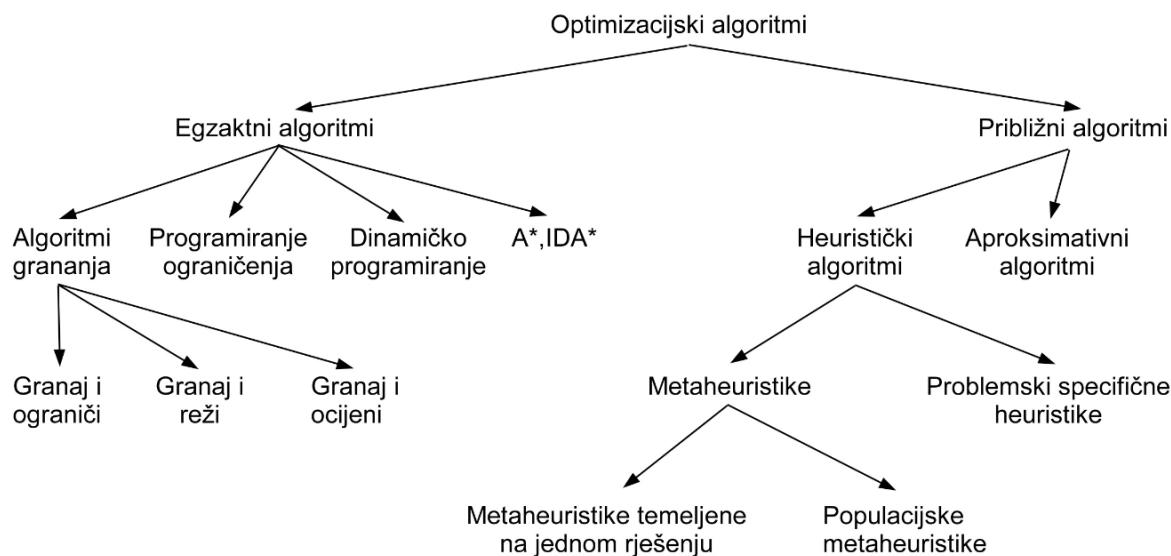
1. grubor pretraga → nasumična rješenja iz početnog čitavog prostora rješenja; locira obećavajući potprostor
2. fina pretraga → istraživanje okolice pronađenog podprostora



## PRIKAZ RJEŠENJA

1. izravni prikaz
2. neizravni prikaz (dekodiranje po originalu)

razlog:  
priredostavljanje algoritma



Slika 2.1: Podjela optimizacijskih algoritama

SELEKCIJSKI PRITISAK - parametar vezan uz brzinu fokusiranja populacije; veći pritisak → brže fokusiranje

OPERATOR MUTACIJE - diverzifikacija populacije

KUANT PRETRAGUE - minimalni korak kojim algoritam može promijeniti trenutnu vrijednost rješenja

$$\tilde{x} = \frac{1}{2^n - 1} (x_{\max} - x_{\min})$$

Problem s binarnim prikazom je što se dva susjedna broja mogu razlikovati u svim bitovima (0111 i 1000).

Zato je bolje koristiti Grayev kod, ali on stvara dodatni utrošak kod dekodiranja.

Bitovi → prirodni binarni kod

$$X = X_{\min} + \frac{k_x}{2^{n_x} - 1} (x_{\max} - x_{\min})$$

*k<sub>x</sub>* = binarni broj u dekadski

## VRSTE KRIŽANJA

1. s jednom točkom prekida

2. s k točaka prekida

3. uniformno križanje

$$D_1 = R_1 \cdot R_2 + R(R_1 \oplus R_2)$$

$$D_2 = R_1 \cdot R_2 + !R(R_1 \oplus R_2), \quad R \rightarrow \text{slučajno generirani niz od } k \text{ bitova}$$

## PRIKAZ POLJEM DECIMALNIH BROJEVA

1. diskretno križanje  $\rightarrow$  stvara v i-dijete, i-ta komponenta dijeteta će se izravno prenijeti od jednog roditelja

2. ravno križanje  $\rightarrow$  i-ta komponenta postavi se na vrijednost koja je izvučena iz  $U([\min(c_i^1, c_i^2), \max(c_i^1, c_i^2)])$

3. jednostavno križanje  $\rightarrow$  kao s jednom točkom prekida

$$h_i^1 = \lambda c_i^1 + (1-\lambda)c_i^2$$

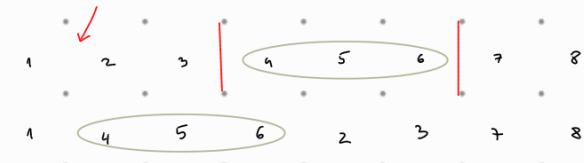
4. aritmetičko križanje  $\rightarrow$  linearna kombinacija komponenti roditelja

$$h_i^2 = \lambda c_i^1 + (1-\lambda)c_i^2$$

... Blx-d križanje, linearno križanje, prošireno unijuško križanje, prošireno komponentno križanje

## PRIKAZ PERMUTACIJAMA I MATRICAMA

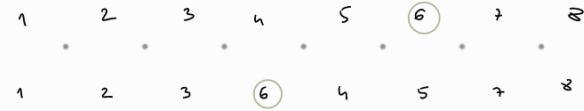
1. mutacija premještanjem



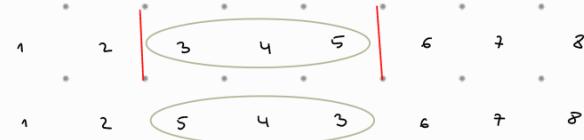
2. mutacija zamjenom



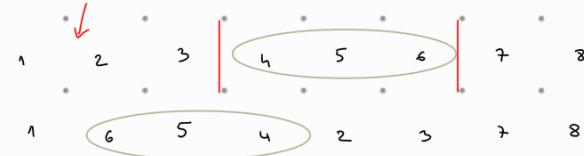
3. mutacija umetanjem



4. jednostavna mutacija inverzijom



5. mutacija inverzijom



6. mutacija miješanjem



### KOMBINIRANJE RJEŠENJA

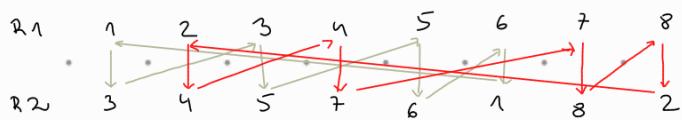
#### 1. djelomično preslikano knižanje

R1	1	2	3	4	5	6	7	8
R2	3	4	7	1	8	2	5	6

↑ Već imamo. Preslikavan se u 4.

D1	x	x	7	8	x	x	4	2	7	1	8	6	3	5	
D2	x	x	3	4	5	x	x	7	1	3	4	5	2	8	6

#### 2. knižanje ciklusa



D1	1	(4)	3	(7)	5	6	(8)	(2)
D2	3	(2)	5	(4)	6	1	(7)	(8)

#### 3. knižanje poretkom

R1	1	2	3	4	5	6	7	8
R2	3	4	7	1	8	2	5	6

D1	x	x	3	4	5	x	x	x	1	8	3	4	5	2	6	7
D2	x	x	7	1	8	x	x	x	4	5	7	1	8	6	2	3

#### 4. knižanje temeljeno na poretku

R1	1	2	3	4	5	6	7	8
R2	3	4	7	1	8	2	5	6

→ x na pozicije koje se nalaze na 2, 4, 6 poziciji u R2

D1	x	x	3	x	5	6	7	8	4	1	3	2	5	6	7	8
D2	3	x	7	1	8	x	5	x	3	2	7	1	8	4	5	6

### 5. Križanje temeljeno na poziciji

R1	1	2	3	4	5	6	7	8
R2	3	4	7	1	8	2	5	6

→ samo dobruto

→ svi iz R1 redom od početka

D1	x	4	x	1	x	2	x	⇒ 3	4	5	1	6	2	7	8
D2	x	2	x	4	x	6	x	3	2	7	4	1	6	8	5

### GENOTIPSKI I FENOTIPSKI PRIKAZ RJEŠENJA

- fenotipski → ako su 3 znacajke i imamo O1O pa znamo da je samo 2. prisutna

- genotipski → ako trebamo i dekodirati dobiveno rješenje

- LOKALNOST PRIKAZA RJEŠENJA - koliko dobro genotipsko susjedstvo odgovara fenotipskom susjedstvu

# NUMERIČKE METODE

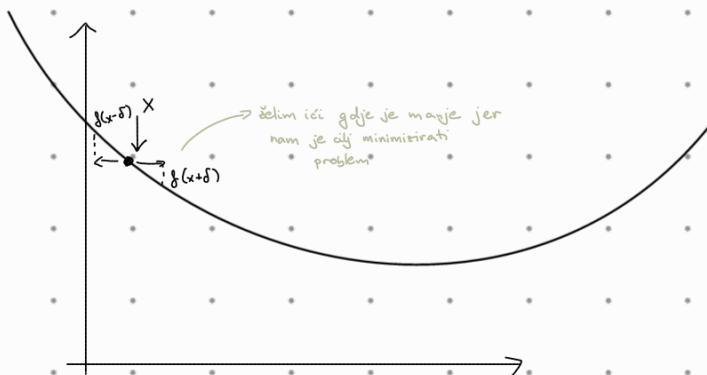
## OPTIMIZACIJE

- domena je  $\mathbb{R}^n$
- matematičke metode
- funkcija je konveksna ili unimodalna

 samo jedan minimum

### PODJELA

1. samo vrijednosti funkcije cilja  $x \rightarrow f(x)$
2. derivacije   
*ne radimo*
3. ograničenja



### PRIMJER FUNKCIJE JEDNE VARIJABLE

$$f(x) = 3x^2 - 6x - 45 \quad \text{traži minimum}$$

$$2 \cdot 3x - 6 = 0 \quad \text{derivacija}$$

$$x = 1$$

lokalni minimum / točka infleksije...

$$6x - 6 = 0$$

→ sve što mi je desno od minimuma imat će derivaciju  $> 0$ , a lijevo derivaciju  $< 0$

$$f'(x) < 0 \quad x \leftarrow x + \delta$$

$$f'(x) > 0 \quad x \leftarrow x - \delta$$

$$f(x) \quad \text{minimum}$$

Za traženje minimuma funkcije tada se treba kretati u smjeru negativne derivacije.

$$x \leftarrow x - \eta f'(x), \eta > 0 \quad \text{korak}$$

### PRIMJER FUNKCIJE VIŠE VARIJABLJI

$$\nabla f(\vec{x}) = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \frac{\partial f}{\partial x_3} \end{bmatrix}$$

$$\vec{x} \leftarrow \vec{x} - \eta \nabla f(\vec{x})$$

$$f(\vec{x}) = (x_1 - 4)^2 + (x_2 + 8)^2 - (x_3 + 5)^2$$

$$\nabla f(\vec{x}) = \begin{bmatrix} 2x_1 - 8 \\ 2x_2 + 16 \\ 2x_3 + 10 \end{bmatrix} \rightarrow \begin{array}{l} 2x_1 - 8 = 0 \quad x_1 = 4 \\ 2x_2 + 16 = 0 \quad x_2 = -8 \\ 2x_3 + 10 = 0 \quad x_3 = -5 \end{array} \quad \vec{x} = [4, -8, -5] \quad \text{minimum / sedlo!...}$$

### ODGOVARAJUĆI POMAK

$$f(x) = x^2 \quad x_0 = 2$$

$$f'(x) = 2x$$

$$\theta(\eta) = f(\vec{x}^{(0)} + \eta \vec{d}^{(0)})$$

*konstante*

$\eta = 1$

$$x = 2 - 1 \cdot 2 \cdot 2 = -2$$

$$x = -2 - 1 \cdot 2 \cdot (-2) = 2$$

$$\begin{aligned} \eta = 2 & \quad x = 2 - 2 \cdot 2 \cdot 2 = -6 \\ & \quad x = -6 - 2 \cdot 2 \cdot (-6) = 18 \end{aligned} \quad \text{divergencija}$$

$\eta$  prevelika

$$\begin{aligned} \eta = 0.1 & \quad x = 2 - 0.1 \cdot 2 \cdot 2 = 1.6 \\ & \quad x = 1.6 - 0.1 \cdot 2 \cdot 1.6 = 1.28 \end{aligned} \quad \text{prespon pomak}$$

$\eta$  premala

$\Rightarrow \lambda = 0.5$  idealno (optimalno) jer bi došli odmah do 0

## ISPITNI ZADATAK

1. analitički

$$f(\vec{x}) = (x_1 - 3)^2 + (x_2 + 5)^2 \quad x_0 = (0, 0) \quad d = \begin{bmatrix} 6 \\ -10 \end{bmatrix}$$

2. linijskim pretraživanjem

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} + \lambda \begin{bmatrix} 6 \\ -10 \end{bmatrix}$$

$$g(x) = (0 + 6\lambda - 3)^2 + (0 - 10\lambda + 5)^2 / = 0$$

$$g'(\lambda) = 2(6\lambda - 3) \cdot 6 + 2(-10\lambda + 5)(-10) = 0$$

$$\lambda = \frac{1}{2}$$

smjer pretrage

moglo je i  $\begin{bmatrix} 3 \\ -5 \end{bmatrix}$ , negdje je

$$f(x) = (x_1 - 4)^2 + (x_2 + 8)^2 + (x_3 + 5)^2$$

$$x_0 = (6, -10, -9)$$

$\lambda$	$\vec{x}' = \vec{x} + \lambda d$	$g(x')$
0	6, -10, -9	24
1	5, -8, -8	10
2	4, -6, -7	8
3	3, -4, -6	18

$$d = \begin{bmatrix} -1 \\ 2 \\ 1 \end{bmatrix}$$

tv negdje je minimum

$$(6-4)^2 + (-10+8)^2 + (-9+5)^2 = 24$$

↪ traženje unimodalnog intervala

$$x \leftarrow x + \lambda d$$

$$\lambda_{upper} = 1 \quad \lambda_{lower} = 2$$

$$f'(x + \lambda d) > 0$$

minimum između  $\lambda_L$  i  $\lambda_U$

→ uzmememo točku na pola  $\lambda_M$

$f' = 0 \rightarrow$  nagni su minimum  
 $f' \neq 0 \rightarrow$  ili  $\lambda_L$  ili  $\lambda_U$  zamjenjujemo  
 $\lambda_M$

$$f(x) = (x - 5)^2 \quad x_0 = 0 \quad d = 1 \quad \text{ISPITNI ZADATAK}$$

$\lambda$	$f'(\lambda)$
0	-10
1	-8
2	-6
4	-2
8	6

$$f'(x + \lambda d) = 2(x - 5)$$

svakako uzmite 0 za  $\lambda_L$   
 negdje je

$\lambda_L$	$\lambda_M$	$\lambda_U$	$f'(\lambda_M)$
0	4	8	-2
4	6	8	2
4	5	6	0

## PRETRAŽIVANJE U ZADANOM SMJERU

dok neki kriterij zaustavljanja nije zadovoljen

odredi  $d$

pronadi  $\lambda$

$$\vec{x} \leftarrow \vec{x} + \lambda d$$

/

primjer  $f(x) = 2(x_1 - 3)^2 + 3(x_2 - 2)^2$

$$x_0 = (0, 0)$$

## ISPITNI ZADATAK

1 iteracija postupka gradijentnog spusta

gradijentni spust

$$d = -\nabla f(x)$$

$$\nabla f(\vec{x}) = \begin{bmatrix} 4(x_1 - 3) \\ 6(x_2 - 2) \end{bmatrix}$$

$$\nabla f(x_0) = \begin{bmatrix} -12 \\ -12 \end{bmatrix}$$

$$x \leftarrow x_0 - \lambda \nabla f(x_0) = \begin{bmatrix} 0 \\ 0 \end{bmatrix} + \lambda \begin{bmatrix} 12 \\ 12 \end{bmatrix}$$

$$f(\lambda) = 2(12\lambda - 3)^2 + 3(12\lambda - 2)^2$$

$$f'(\lambda) = 4(12\lambda - 3) \cdot 12 + 6(12\lambda - 2) \cdot 12$$

$$0 = (18\lambda - 12)12 + (12\lambda - 12)12$$

$$\lambda = 0.2$$

$$\vec{x} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} + 0.2 \begin{bmatrix} 12 \\ 12 \end{bmatrix} = \begin{bmatrix} 2.4 \\ 2.4 \end{bmatrix}$$

## NEWTONOVA METODA

osim smjera, utima u obliku zaobljejanja

TAYLOR  $f(x)$

Hesgeova matrica

$$f(\vec{x} + \vec{\delta}) = f(\vec{x}) + \nabla f(\vec{x}) \vec{\delta} + \frac{1}{2} \vec{\delta}^T H \vec{\delta} + \dots$$

$$\vec{\delta} = -H^{-1} \nabla f(\vec{x})$$

primjer  $f(\vec{x}) = 2(x_1 - 3)^2 + 3(x_2 - 2)^2$

$$\nabla = \begin{bmatrix} 4(x_1 - 3) \\ 6(x_2 - 2) \end{bmatrix} = \begin{bmatrix} -12 \\ -12 \end{bmatrix}$$

## ISPITNI ZADATAK

$$x_0 = (0, 0)$$

$$H = \begin{bmatrix} 4 & 0 \\ 0 & 6 \end{bmatrix}$$

$$\vec{\delta} = -H^{-1} \nabla f(\vec{x})$$

$$H \vec{\delta} = -\nabla f$$

$$4 \delta_1 = 12 \quad \delta_1 = 3$$

$$6 \delta_2 = 12 \quad \delta_2 = 2$$

$$\vec{x} = \vec{x} + \lambda \vec{\delta} = \begin{bmatrix} ? \\ ? \end{bmatrix}$$

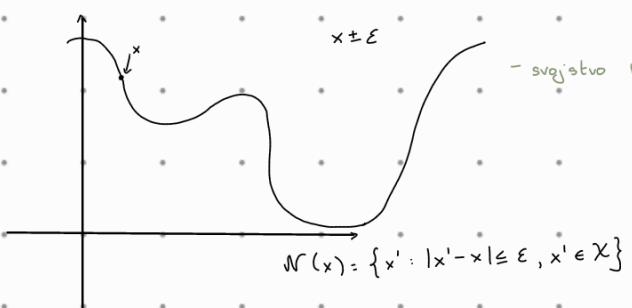
$$H = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$$

# A L G O R I T M I I L O K A L N E P R E T R A G E

- jedno rješenje (najbolje pronađeno rješenje)
- operatori lokalne pretrage
- jednostavni algoritmi
- lokalni optimumi

SUSJEDSTVO       $S \rightarrow (S')$        $N(x)$        $N: X \rightarrow 2^X$

## - KONTINUIRANI PROBLEM



- svojstvo lokalnosti - male promjene u prikazu rješenja mogu dovesti do malih promjena u samom rješenju
- ako nije zadovljeno, to je neusmjerena pretraga

## BINARNA DOMENA

	1	0	1	1	0	1	1	0	$d=1$
0 1 2 3 , 2 1 0 3	1	1	1	1	0	1	1	0	

zamjene elemenata →  $n(n-1)/2$   
ubacivanje elemenata →  $n^2$       veličina susjedstva

PERMUTACIJE	5	3	1	4	2
0 1 2 3 4	5	4	1	3	2
0 1 4 2 3					

LOKALNI OPTIMUM - rješenje  $x^*$  za koje vrijedi da su sva rješenja iz njegovog susjedstva lošija od njega

- diskretni prostor -  $d(S', S) \leq \varepsilon$

## LOKALNA PRETRAGA

- početno rješenje  $x$
- u svakoj iteraciji napravi susjedstvo od  $x$
- odaberi najboljeg susjeda
- ako je  $x'$  bolji od  $x$   $x=x'$ , inače stani

3 varijante algoritma za povećanje robustnosti s ogromom na lokalne optimume

1. iterativno pokretanje s različitim početnim rješenjima
2. prihvatanje i rješenja koja nisu bolja od trenutnog
3. promjena susjedstva

## STRATEGIJE ODABIRA SUSJEDA

1. odaberi najboljeg → počepni algoritam uspona na vrh

2. odaberi prvoj bolje

- + jednostavan, gotovo bez parametara
- potlepan, lako zapne u lokalnom optimumu

3. odaberi nasumično, ali bolje

## LOKALNA PRETRAGA S VIŠESTRUKIM POKRETANJEM

→ pokreni lokalnu pretragu N puta iz različitih početnih rješenja, na kraju uzmi najbolje rješenje

+ jednostavan, paralelizacija

- ne izbjegava lokalne optimume, pokretanja ne dijele informacije

## ITERATIVNA LOKALNA PRETRAGA

1. PERTURBACIJA → promjeni rješenje

2. LOKALNA PRETRAGA X

3. PRIHVACANJE

} u svakoj iteraciji

- kvaliteta lokalnog optimuma ovisi o inicijalnom rješenju

+ jednostavna, koristi informacije iz prethodnih pretraživanja  
- potrebno je dobro odrediti omjer perturbiranja i lokalne pretrage

## PERTURBACIJA

↳ statički 0.1

↳ fiksiran pretraga

↳ dinamički 0.5 → 0.05 → 0.5

↳ nasumično

↳ adaptivno

## VARIJABILNO PRETRAŽIVANJE SUSJEDSTVA

### SPUST VARIJABILNOG SUSJEDSTVA (VND)

→ Prelazi se u novo susjedstvo do pronađenja bolje rješenja, onda se prelazi na prvo susjedstvo

→ Efikasno ako su susjedstva komplementarni

### ALGORITAM VARIJABILNE PRETRAGE SUSJEDSTVA (VNS)

↳ 3 koraka iteracije: shake (1 susjed), lokalna pretraga, pomak (ako je bolje)

↳ ako novodobiveno rješenje nije bolje, mijenja se susjedstvo

↳ koriste se ugnjeđena susjedstva

+ efikasan, koristi podatke iz prethodnih pretraživanja

- potrebno dobro definirati susjedstva i hiperstrukturnu između njih

## GRASP

### POHLEPNA RANDOMIZIRANA ADAPTIVNA PROCEDURA PRETRAGE

- pohlepna izgradnja rješenja → element po element koristeći neku pohlepnu proceduru

- lokalna pretraga

- RESTRICTED CANDIDATE LIST → generira se u svakom koraku i goriši koje kandidati sad imamo na raspolaganju

↳ na temelju kardinalnosti ili na temelju vrijednosti

} svaka iteracija rezervisana

## GUIDED LOCAL SEARCH

## NAVODENA LOKALNA PRETRAGA

$f'(x) = f(x) + \lambda \sum_i^m p_i l_i(x)$  → algoritam dinamički mijenja djel cijela na temelju dosadašnjih lok. optimuma  
 dobrota rješenja kazna  
 težina kazne je li značajka u rješenju

→ ideja je kazniti značajke koje su prigutne u lokalnom optimumu

## LARGE NEIGHBOURHOOD SEARCH

## LNS

→ Operator uništavanja → nasumično, težinski

→ Operator po pravljancima → 1. ISCRPNA PRETRAGA

2. GREEDY HEURISTIKA

→ IDEJA: ovako se može pretražiti šire susjedstvo

## ALNS

## ADAPTIVE LARGE NEIGHBOURHOOD SEARCH

iskoristi više operatora uništavanja i popravljanja, svaki ima vjerojatnost primjene koja se prilagođava dinamički (proporcionalna selekcija)

težine se dinamički prilagođavaju

# M E T A H E U R I S T I K E

## S J E D N I M R J E Š E N J E M

→ radimo algoritme koji prihvataju i lošija rješenja

### SIMULIRANO KALJENJE

→ lošija rješenja češće se prihvataju u početku → diverzifikacija (istražuje se prostor rješenja)

$$P = e^{-\frac{\Delta E}{T}}$$

$T \rightarrow \infty$   $P \rightarrow 1$  (prihvati bilo što, random search)

$T \rightarrow 0$   $P \rightarrow 0$  (prihvati samo bolja rješenja)

⇒ algoritam će smanjivati  $T$  tijekom vremena

$\Delta E \rightarrow$  što je manja udaljenost između dobrote susjeda, to je vjerojatnost veća

prebroj hlađenje: zapinjanje u lok. optimumu  
presporo hlađenje: šteti konvergenciji

### PLAN HLAĐENJA

1) linearan plan hlađenja:  $T$  se u svakom koraku smanjuje za fiksni iznos

$$\begin{aligned} T &= T - \beta \\ T_i &= T - i\beta \end{aligned}$$

2) geometrijski plan hlađenja:  $\alpha \in [0, 1]$   $T = \alpha T$

↳ obično  $[0.5, 0.95]$

koristan samo za dokazivanje globalne konvergencije algoritma

3) logaritamski plan hlađenja: ne koristi se u praksi jer je prespor

$$T_k = \frac{T_0}{\log k}$$

4) vrlo sporo smanjenje

$$T = \frac{T_i}{1 + \beta T_i}$$

samo 1 iteracija unutarne petlje po temperaturi

→ ostali planovi: nemonoton, adaptivno

### STANJE EKVILIBRIJA

KRITERIJ ZAUSTAVLJANJA → "šta god vam padne na pamet"

→ metode slične sim. kaljenju: threshold acceptance, record to record travel, algoritam demona

→ nitko normalan ne koristi, neće biti na ispitu

→ svi skupa izumrli

• samo rješenja koja su lošija od trenutnog za određenu razinu

• isto, ali naspram najboljeg dosad

## TABU PRETRAGA

TABU LISTA → proti sva dosadašnja rješenja i/ili poteze

→ rješenje prihvacamo ako se ne nalazi u tabu listi ni u blizini nekog rješenja iz tabu liste

→ problemi: memorija, pretraživanje liste

→ praćenje poteza?  $123456 \rightarrow 143256$  → zašto ne bi god zamijenili 2. i 4.?

→ uzeti neku maksimalnu veličinu liste ✓

→ npr. ako je potez iz tabu liste, ali nije bolje od dosada najboljeg, onda ga slobodno uzmi

KRITERIJ ASPIRACIJE → dozvoljava da se pod određenim uvjetima neki pomici koji se nalaze u tabu listi ipak prihvate

### MÉMORIJA

1. kratkoročna → sprječava cikluse

tabu tenure → broj iteracija koliko će potez ostati u tabu listi

zao premašiti vrijednost češći su ciklusi,  
a zao preveliki je algoritam spor

2. srednjeročna → eksploracija (bolja rješenja)

3. dugoročna → eksploracija (širi prostor) : diverzifikacija ponovnim pokretanjem, kontinuirana diverzifikacija

uvodimo kaznu u funkciju cilja  
koja kaže na rješenja koja se  
često posjećuju

# GENETSKI ALGORITAM

KORACI → stvori početna rješenja JEDINKA → POPULACIJA

→ SELEKCIJA i KRIŽANJE daje JEDINKU (kromosoma)

→ MUTACIJA

→ evaluirati: FUNKCIJA DOBROTE → numerička vrijednost koja predstavlja kvalitetu rješenja

→ ubaciti dijete u populaciju

→ ponavljati

ježima  
PARALELNI

## VRSTE

- { 1. generacijski → stvara se nova populacija jedinki koja potpuno zamjenjuje staru  
2. eliminacijski → stvara se samo 1 nova jedinka, možemo definirati postotak eliminacije koji određuje koliko čemo jedinki eliminirati iz populacije

ELITIZAM - svojstvo da najbolja jedinka nikad ne bude eliminirana iz populacije

- u generacijski se treba ugraditi, u eliminacijskom je to automatski

- ako nema elitizma, dobrota oscilira

## STVARANJE POPULACIJE

1. nasumično → jednostavno, sporija konvergencija

2. heuristički → heuristike za stvaranje rješenja, problem lokalnog optimuma i osiguravanja raznolikosti

## PRIKAZI RJEŠENJA

1. binarni → problem deceptije (rješenje Greyer kod) problem diskretizacije prostora

$$n \geq \frac{\log(\frac{x_{\max} - x_{\min}}{\tilde{x}} + 1)}{\log 2}$$

realni → binarni

$$b = \frac{x - x_{\min}}{x_{\max} - x_{\min}} (2^n - 1)$$

binarni → realni

$$x = x_{\max} + \frac{b}{2^n} (x_{\max} - x_{\min})$$

2. realni

optimum možda ni ne možemo predstaviti u zadatom prikazu

3. permutacijski

4. cijelobrojni

5. miješani

1. FENOTIP - rješenje nekog problema

2. GENOTIP - kako je rješenje prikazano u jedinci

SELEKCIJA → odabir boljih jedinki za reprodukciju i lošijih za eliminaciju

↳ SELEKCIJSKI PRITISAK → omjer vjerojatnosti prečuvavanju boljih i lošijih jedinki; previelik → prebra konvergencija, zapinjanje u lok. opt. premalih → spora konvergencija, nasumična pretraga

$$\text{PROPORTIONALNA SELEKCIJA} \quad p_i = \frac{F_i}{\sum_j F_j} \quad \text{Relativna} \quad p_i = \frac{F_i - F_{\min}}{\sum_j (F_j - F_{\min})} \quad + \text{selekcijski pritisak} \quad F_i = 1 + (SP-1) \cdot f_i$$

+ vjerojatnost odabira ovisi o dobroti jedinke

- problem skale, prevladavanje najbolje jedinke ...

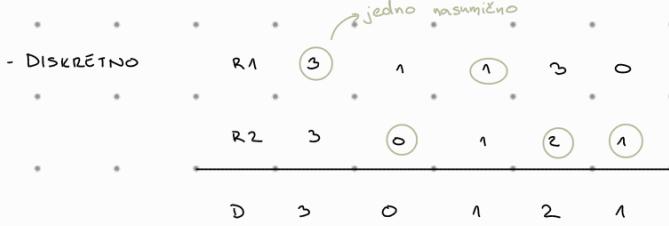
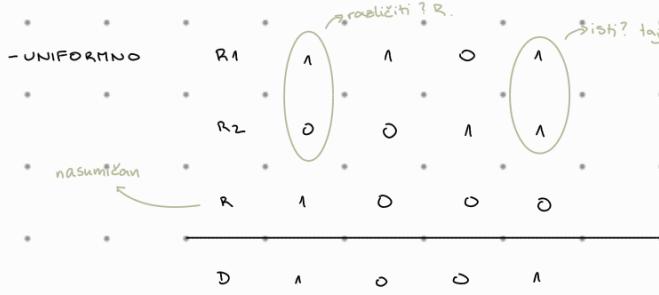
SELEKCIJA LINEARNIM RANGIRANJEM → sortiramo jedinke po njihovoj dobroti, rang 1 najbolje, rang n najbolje rješenje

$$p_i = \frac{2-SP+2(SP-1) \frac{i-1}{n-1}}{n}, SP \in [1,2]$$

TURNIRSKA SELEKCIJA → nasumično odaberemo k jedinki, od njih odaberemo najbolju; mali k → odabir loših jedinki

JEDNOSTAVNA 3-TURNIRSKA SELEKCIJA → odaberemo 3, lošim eliminiramo, one 2 krišamo

### KRIŽANJE



- ARITMETIČKO

$$h_i^1 = \pi c_i^1 + (1-\pi) c_i^2$$

$$h_i^2 = \pi c_i^2 + (1-\pi) c_i^1$$

### MUTACIJA

- UNIFORMNA

- zamjena  $x=r$ ,  $r \in [x_{\min}, x_{\max}]$
- dodavanje uniformne vrijednosti  $x=x(b-a)+a$

- GAUSSOLIKA  $x=x+r$ ,  $r \in N(0,s)$

MEMETIČKI ALGORITAM - ugraditi operatore lokalne pretrage u GA

## GENETSKO PROGRAMIRANJE

↳ koristi se stablo, čvorovi mogu biti aritmetičke operacije, ugrađene funkcije, konstante ili varijable

↳ ideja je za ulazne podatke stvoriti netu funkciju koja ih dobro opisuje

# GENETSKO PROGRAMIRANJE

SIMBOLIČKA REGRESIJA → ne naučimo samo koeficijente, nego i oblik cijele funkcije

- genetski algoritam s drugačijim prikazom rješenja
- jedinke predstavljaju matematičke izraze ili programe (najčešći oblik stablo)

1. FUNKCIJSKI ČVOROVI → aritmetičke operacije, ugradene funkcije

2. TERMINALNI ČVOROVI → konstante, varijable

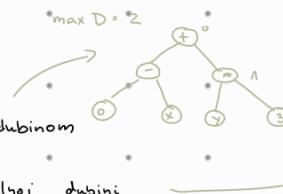
Potrebna svojstva skupa primitiva

- potpunost → skupom moguće rješiti problem
- zatvorenost PROTECTED OPERATORI → definirane sve kombinacije operacija-operand

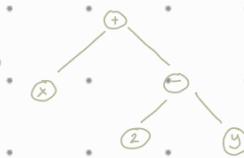
INICIJALIZACIJA POČETNE POPULACIJE → zada se maksimalna početna dubina

- metode izgradnje jedinki:

1. FULL - sva stabla potpuna s maksimalnom dubinom



2. GROW - ne moraju svi terminali biti na maksimalnoj dubini



3. RAMPED HALF-AND-HALF - 50% full, 50% grow ut rastuće maksimalne dubine

KRIŽANJE

1. one-point, uniform, context-preserving, size-fair

BLOAT → povećanje stabala tijekom evolucije

→ metode sprječavanja/popravljanja: parsimony pressure, mnogokriterijska optimizacija, editing

uklanjanje nebitnih dijelova

TIPOVI U GP

1. STGP (strongly typed) → dozvoli samo sintatsko ispravne programe

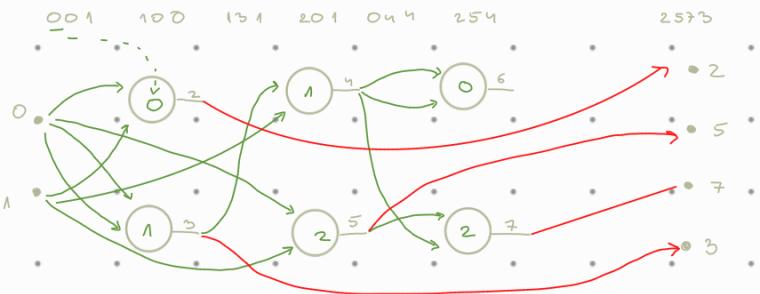
2. DAGP (dimensionally aware) → semantička ispravnost

## DRUGI OBLCI PRIKAZA RJEŠENJA

### 1. CGP

↳ Čvorovi raspoređeni u rešetku dimenzija  $n \times m$ , rješenje napisano kao niz cijelih brojeva

genotip:



Fenotip:

$0 \rightarrow +$   
 $1 \rightarrow -$   
 $2 \rightarrow *$

### 2. GEP

↳ Čvorovi zapisani u obliku niza, svaki niz sastavljen od gena, iz niza se gradi stablo

↳ bilo koji čvorovi

↳ gen = glava + rep

↳ samo terminalni čvorovi

$Q^* Q + bbaaa$

↳ glava      rep

↳ osigurano da se ne može izgraditi neispravni izraz

