HEURISTIČKE METODE OPTIMIZACIJE

Školska 2020/21

Talbi, El-Ghazali. Metaheuristics: from design to implementation. Vol. 74. John Wiley & Sons, 2009.

Heuristike i metaheuristike

- Specijalizovane heuristike
- Metaheuristike
  - *Rešavanje velikih problema*
  - *Brže rešavanje probema*
  - *Robustni algoritmi*
- Razvijaju se u više oblasti
  - *Veštačka inteligencija*
  - *Soft computting*
  - *Matematičko programiranje*
  - *Operaciona istraživanja*

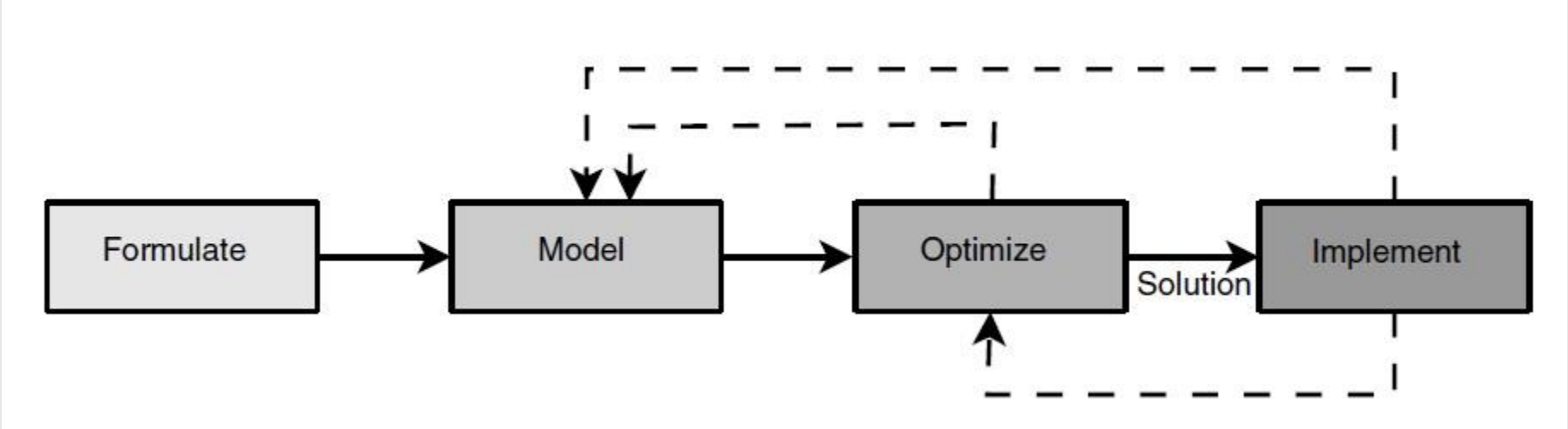
ZAJEDNIČKI POJMOVI ZA METAHEURISTIKE

heuriskein – umetnost otkrivanja nove strategije za rešavanje problema

meta – metodologija višeg nivoa

Optimizacioni modeli

- Pronalaženje „dobrih“ rešenja

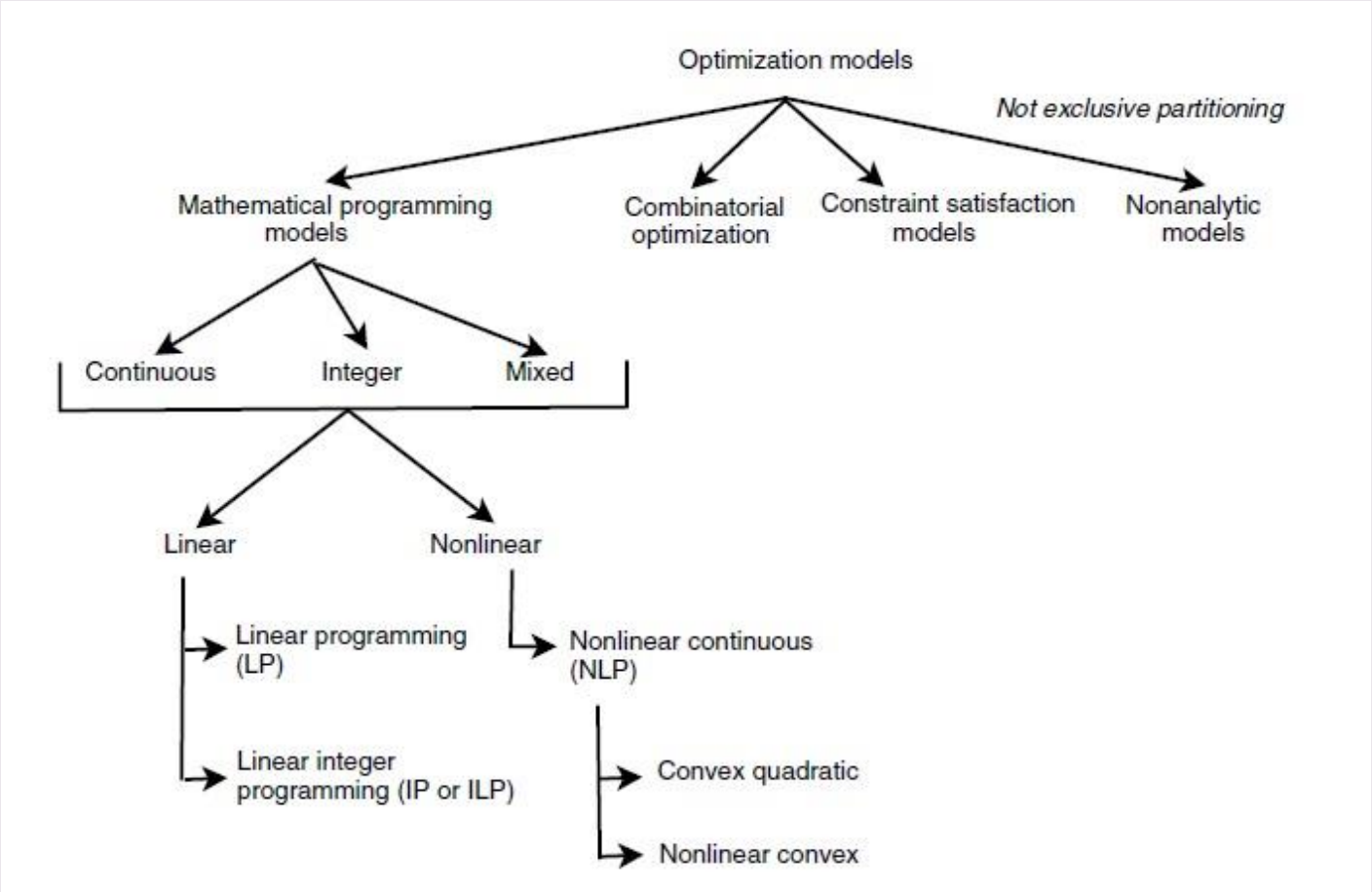


Klasični optimizacioni modeli

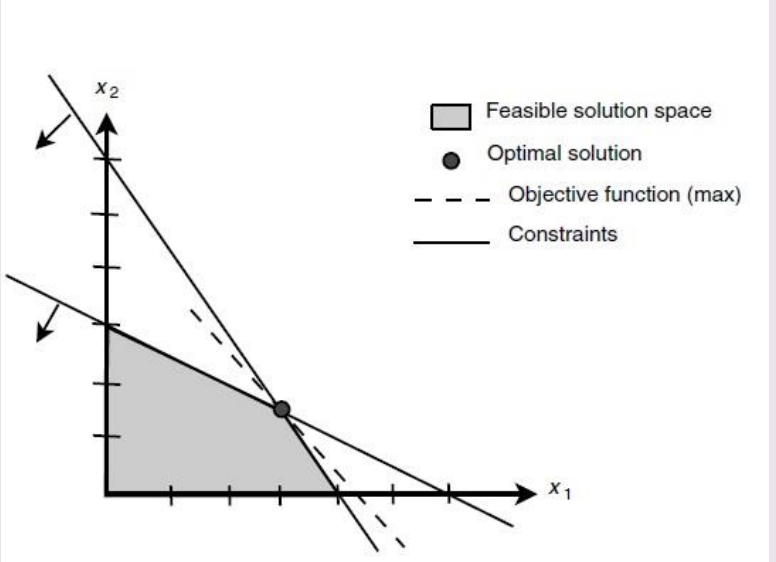
- Optimizacioni model je par (S, f), gde je S skup dopustivih rešenja i $f: S \to \mathbb{R}$ funkcija cilja
- Funkcija cilja omogućava da se definiše relacija uređenja za svaki par rešenja iz prostora pretrage

Definicija. Rešenje $s^* \in S$ je *globalni optimum* ako ima bolju vrednost funkcije cilja od svih ostalih rešenja iz prostora pretrage, tj. $\forall s \in S, f(s^*) \leq f(s)$

Klasični optimizacioni modeli



Linearno programiranje (LP)



$$\text{Min } c \cdot x$$
$$\text{pri ograničenjima}$$
$$A \cdot x \geq b$$
$$x \geq 0$$

gde je x vektor iz skupa dopustivih rešenja, c i b vektori i A matrica.

- Funkcija cilja i ograničenja su linearni
- Prostor rešenja – kontinualan, diskretan (IP), mešovit (MIP)
- LP modeli su razvijeni tokom II svetskog rata za rešavanje logističkih problema
- Zbog velike efikasnost algoritama Simplex tipa nema razloga za korišćenjem metaheuristika za rešavanje kontinualnih LP problema

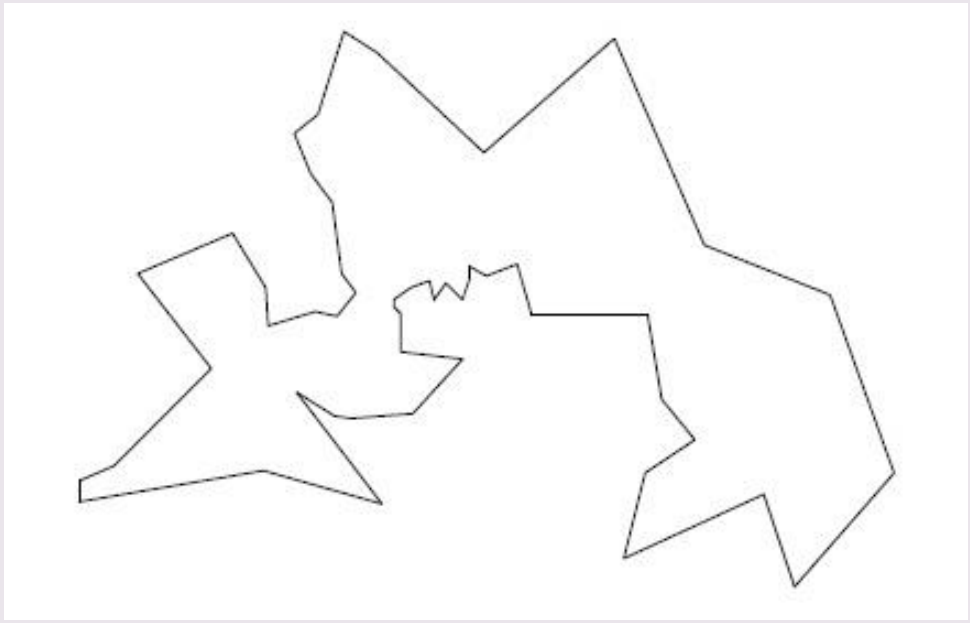
Nelinearno programiranje (NLP)

- Funkcija cilja i/ili ograničenja nisu linearna
- Metode linearizovanja problema u nekim slučajevima
- Nelder-Mead algoritam – Simplex-inspirisana heuristička metoda
- Za male kvadratne i konveksne kontinualne probleme postoje egzaktne metode rešavanja
- Metaheuristike su dobar kandidat za rešavanje ove klase problema

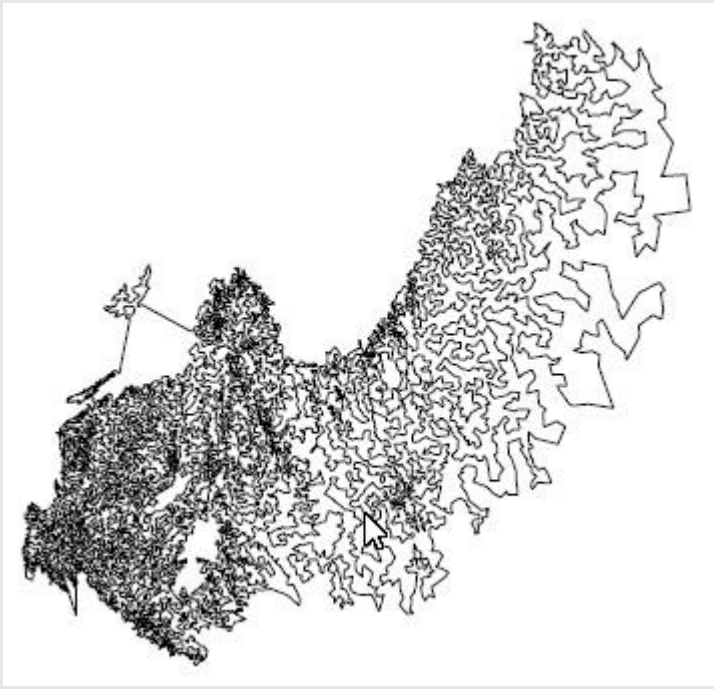
Kombinatorni optimizacioni problemi

- Opštija klasa IP problema
  - *Diskretan i konačan prostor pretrage*
  - *Funkcija cilja i ograničenja mogu biti u bilo kom obliku*
- Problem trgovačkog putnika (*Traveling salesman problem* – TSP)
  - *Dato je n gradova i matrica rastojanja $d_{n,n}$, gde svaki element $d_{i,j}$ predstavlja rastojanje između gradova i i j. Pronaći putanju sa minimalnim rastojanjem, takvu da se kroz svaki grad prolazi tačno jednom (Hamiltonov put)*
  - *Veličina prostora pretrage je $n!$*

TSP



Primer za 52 grada



Primer za 24 978 gradova

Number of Cities n	Size of the Search Space
5	120
10	3, 628, 800
75	2.5×10^{109}

Programiranje sa zadovoljenjem ograničenja (*Constraint programming* – CP)

- Model se sastoji od skupa promenljivih, svaka promenljiva ima konačan domen vrednosti
- Problem raspoređivanja
  - *Cilj je rasporediti n objekata $\{o_1, o_2, \dots, o_n\}$ na n lokacija $\{l_1, l_2, \dots, l_n\}$, gde se svaki objekat smešta na različitu lokaciju*
  - *Matematički rebusi, raspored, 4 osobe u 4 kuće tako da…*

Kompleksnost algoritama

- **Definicija. *O*-notacija.** Algoritam je kompleksnosti $f(n) = O\big(g(n)\big)$ ako postoje pozitivni brojevi n_0 i c takvi da $\forall n > n_0, f(n) \leq c \cdot g(n)$.
  - *Funkcija $f(n)$ je sa gornje strane ograničena funkcijom $g(n)$*
- **Definicija. Polinomni algoritmi.** Algoritam je polinomne kompleksnosti ako je njegova kompleksnost $O\big(p(n)\big)$, gde je $p(n)$ polinom po n.
- **Definicija. Eksponencijalni algoritmi.** Algoritam je eksponencijalne kompleksnosti ako je njegova kompleksnost $O(c^n)$, gde je c realna konstanta strogo veća od 1.

Complexity	Size = 10	Size = 20	Size = 30	Size = 40	Size = 50
$O(x)$	0.00001 s	0.00002 s	0.00003 s	0.00004 s	0.00005 s
$O(x^2)$	0.0001 s	0.0004 s	0.0009 s	0.0016 s	0.0025 s
$O(x^5)$	0.1 s	0.32 s	24.3 s	1.7 mn	5.2 mn
$O(2^x)$	0.001 s	1.0 s	17.9 mn	12.7 days	35.7 years
$O(3^x)$	0.059 s	58.0 mn	6.5 years	3855 centuries	2×10^8 centuries

Kompleksnost problema

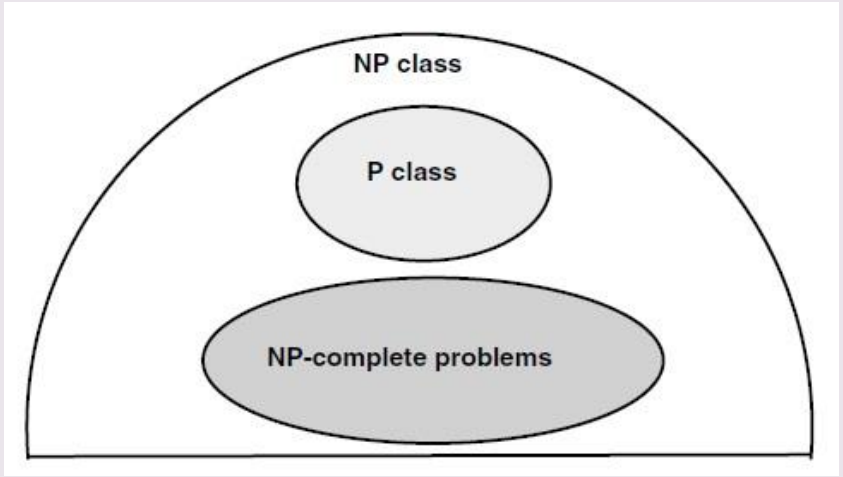
- Kompleksnost problema je ekvivalentna kompleksnosti najboljeg algoritma koji rešava taj problem
- Problem je traktabilan (efikasno rešiv) ako postoji polinomni algoritam koji ga rešava
- Problemi odlučivanja čiji je odgovor uvek DA ili NE
  - *Problem prostog broja. Da li je dati broj Q prost?*
  - *TSP. Da li za dati broj D postoji Hamiltonov put čija je dužina manja ili jednaka D?*

Kategorizacija problema. P i NP

- Klasa P – Skup svih problema odlučivanja koji se mogu rešiti determinističkom Tjuringovom mašinom u polinomnom vremenu
  - *Minimalno razapinjuće stablo, Problem najkraćeg puta, Problem maksimalnog toka, Kontinualni problem lineranog programiranja…*
- Klasa NP – Skup svih problema odlučivanja koji se mogu rešiti nedeteminističkom Tjuringovom mašinom u polinomnom vremenu
  - *0-1 problem ranca. Dato je N objekata. Svaki objekat O ima određenu težinu i vrednost. Za dati kapacitet ranca (maksimalna ukupna težina) da li postoji izbor objekata čija je ukupna vrednost bar Q?*
- Pitanje za 1 000 000 $: Da li je P = NP ?

Kategorizacija problema. P i NP

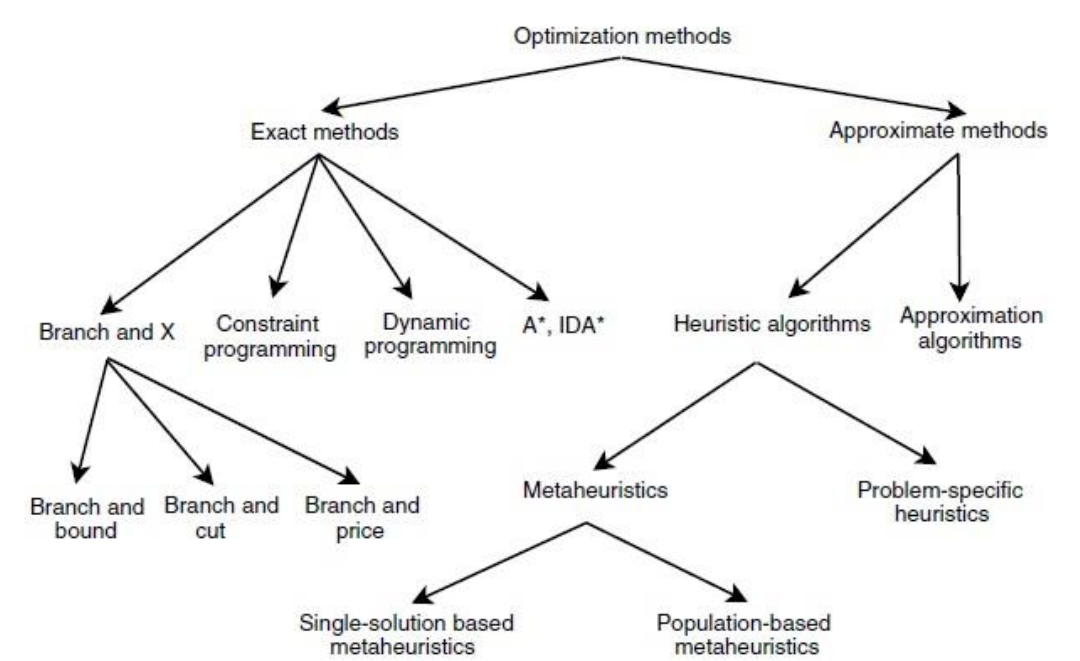
- Klasa NP-kompletni. Svi problemi na koje je moguće svesti bilo koji problem iz klase NP u polinomnom vremenu.
- NP-teški problemi su oni optimizacioni problemi čiji su odgovarajući problemi odlučivanja NP-kompletni
  - *SAT je prvi problem za koji je pokazano da je NP-kompletan*
  - *Problem ranca, trgovačkog putnika, sume podskupa, Hamiltonov ciklus, bojenje grafa,*
  - *Izomorfizam grafa – otvoreno pitanje da je P ili NP?*



Ostali optimizacioni modeli

- Optimizacija sa nepozdanostima
  - *Prisustvo verovatnoće u funkciji cilja*
- Dinamička optimizacija
  - *Ulazni parametri sistema se menjaju tokom vremena*
- Robustna optimizacija
  - *Kreiranje prihvatljivog rešenja uz prisustvo malih promena sistema*

Optimizacione metode



- Kompletnim algoritmima je dobijaju optimalna rešenja i garantuje se njihova optimalnost
- Heurističkim metodama se dobija visoko kvalitetno rešenje za razumno vreme za praktičnu upotrebu, ali ne postoji garancija da je pronađeno rešenje optimalno

Optimizacione metode

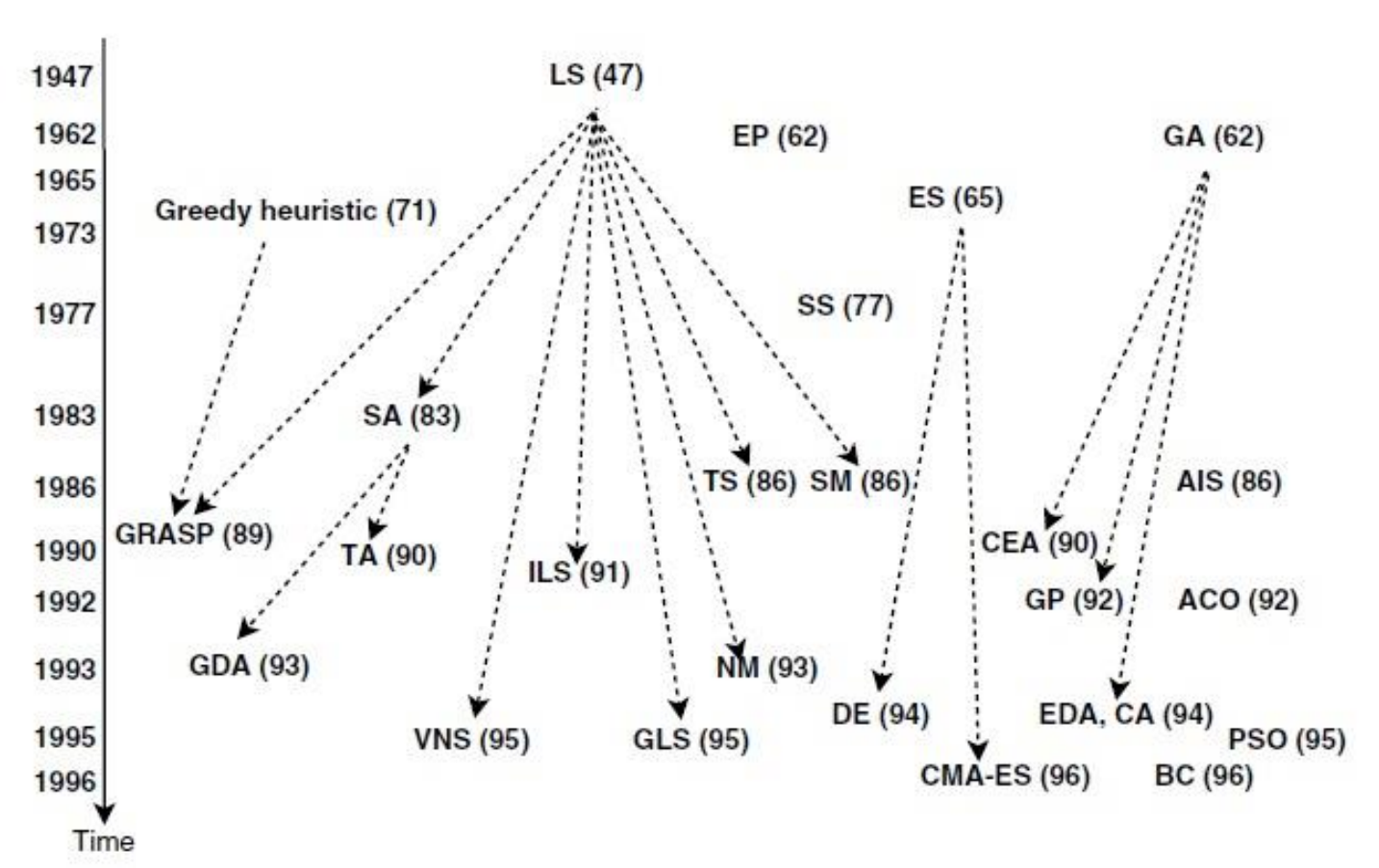
- Veličina problema za koje su pronađena optimalna rešenja kompletnim metodama
  - *Implentacije na grid platformama sa 2000 procesora i 2 meseca izračunavanja*

Optimization Problems	Quadratic Assignment	Flow-Shop Scheduling (FSP)	Graph Coloring	Capacitated Vehicle Routing
Size of the instances	30 objects	100 jobs 20 machines	100 nodes	60 clients

Metaheuristike

- Primena u
  - *Inženjerski dizajn, optimizacija topologije i strukturna optimizacija u elektronici i VLSI, aerodinamika, dinamika fluida, telekomunikacije, automobilska industrija, robotika*
  - *Mašinsko učenje i data mining iz bioinformatike i računske biologije, finansije.*
  - *Modeliranje sistema, simulacija i identifikacija u hemiji, fizici i biologiji; kontrola, obrada signala i slike.*
  - *Planiranje problema sa rutiranjem, planiranje robota, zakazivanje i problemi u proizvodnji, logistiku i transport, upravljanje lancima snabdevanja, životnu sredinu.*

Metaheuristike



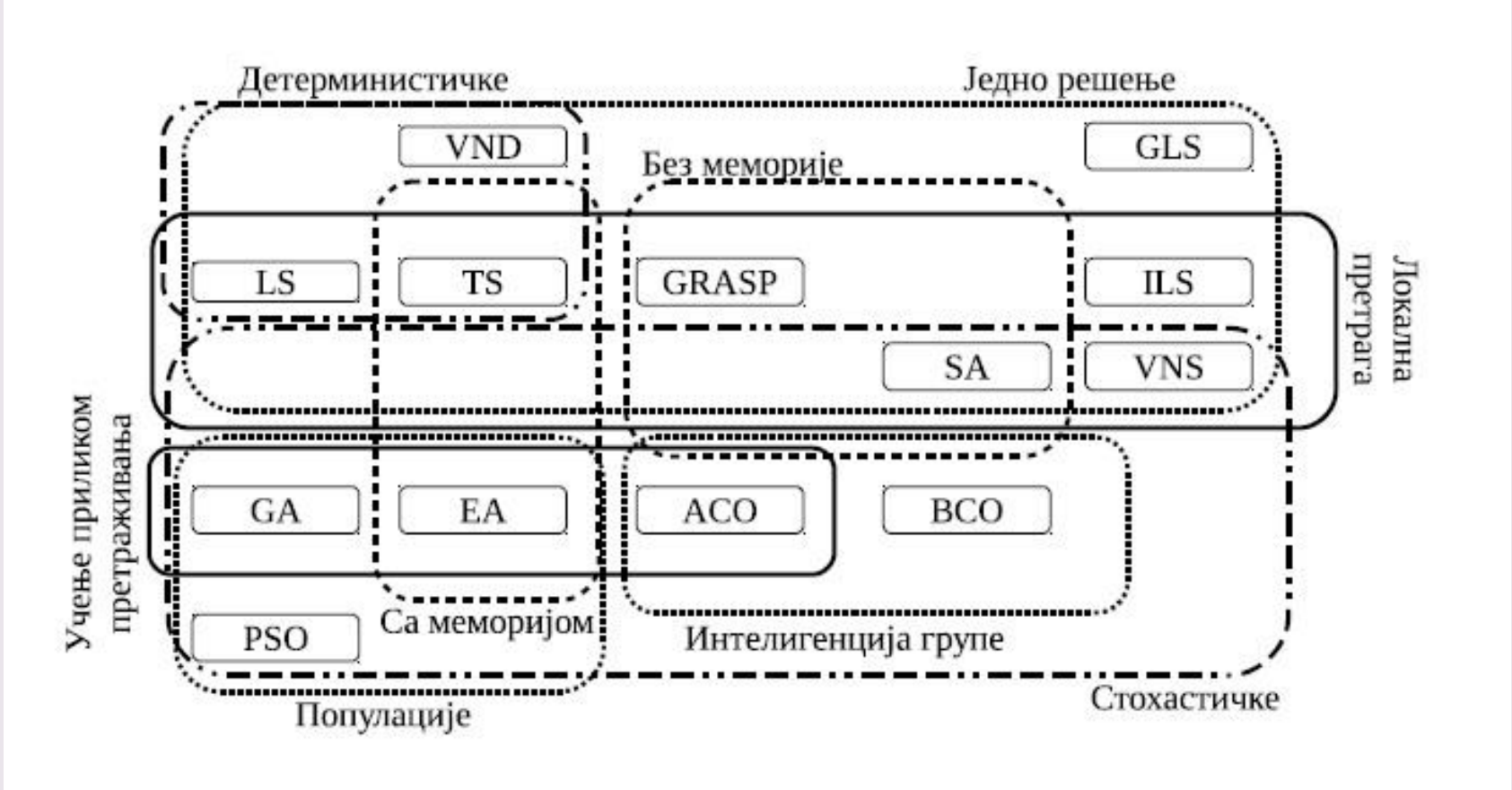
Klasifikacije metaheuristika

- Inspirisani prirodom vs. inspirisani „neprirodom“
  - *Evolutivni algoritmi, veštački imuni sistemi*
  - *Mravi, pčele, vukovi, veverice, rojevi čestica*
  - *Simulirano kaljenje*
- Sa memorijom vs. bez memorije
  - *Tabu pretraga, evolutivni algoritmi*
  - *Lokalna pretraga, GRASP, simulirano kaljenje*
- Determinističke vs. stohastičke
  - *Lokalna pretraga, tabu pretraga,*
  - *Simulirano kaljenje, evolutivni algoritmi, VNS, BCO, GA*

Klasifikacije metaheuristika

- Bazirane na populaciji vs. jedno rešenje
  - *GA, PSO, BCO*
  - *Lokalna pretraga, VNS, GRASP*
- Iterativne vs. Gramzive
  - *Počinju sa kompletnim rešenjem i popravljaju ga*
  - *Počinju sa prazni rešenjem i „grade“ ga*
- Hibridne metode
- Paralelizacija

Klasifikacije metaheuristika



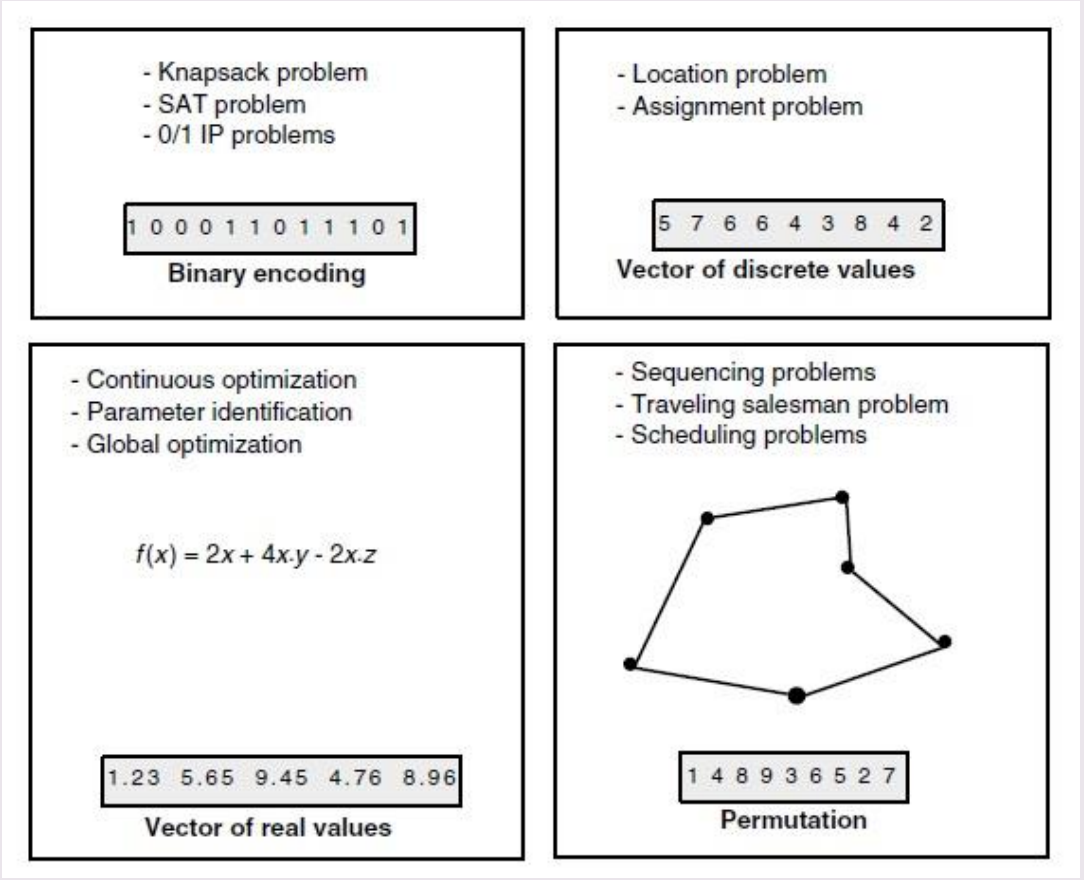
Osnovni pojmovi

- Reprezentacija rešenja
- Funkcija cilja

Reprezentacija rešenja

- Igra veliku ulogu u efikasnosti metaheuristike i predstavlja glavni korak prilikom implementacije neke heurističke metode
- Mora da odgovara prirodi optimizacionog problema
- Direktno je povezana sa operatorima pretrage, rekombinacije i sl.
- Više različitih reprezentacija može postojati za isti problem ali svaka mora biti
  - *Kompletna – svako rešenje može da se predstavi*
  - *Povezana – između svaka dva rešenja mora postojati put u prostoru pretrage*
  - *Efikasna – laka za primenu operatora*

Reprezentacija rešenja



Linearna reprezentacija

- Reč sačinjena od simbola datog alfabeta
- Binarno enkodiranje je pogodno za većinu problema *yes/no* odlučivanja
  - *SAT problem – raspodela istinitosnih vrednosti iskaznim slovima (interpretacija) se prestavlja se binarnim vektorom*
  - *Problem ranca – vektor s binarnih promenljivih dužine n*

$$\forall i, s_i = \begin{cases} 1 & \text{objekat } i \text{ je u rancu} \\ 0 & \text{inače} \end{cases}$$

- Binarno enkodiranje koristi binarni alfabet i može se proširiti na bilo koje diskretne vrednosti koje koriste n-arni alfabet

Linearna reprezentacija

- Diskretno ekodiranje predstavlja proširenje binarnog enkodiranja – binarni alfabet se može proširiti na bilo koje diskretne vrednosti koje koriste n-arni alfabet
- **Problem raspodele.** Potrebno je skup od k zadataka podeliti na m agenata u maksimizovanje ukupnog profita. Zadatak može biti dodeljen na kom agentu. Enkodiranje ove klase problem bazira se na diskretnom vektoru s dužine k gde je

$$s[i] = j \text{ ako je agentu } j \text{ dodeljen zadatak } i$$

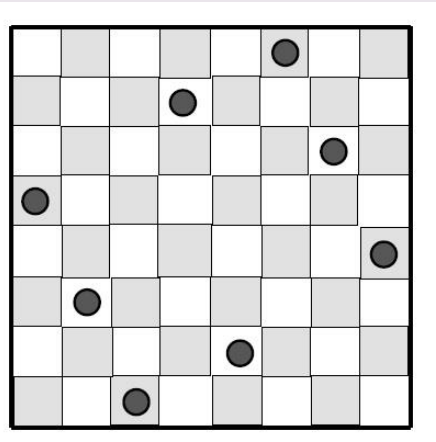
Linearna reprezentacija

- Enkodiranje permutacijama se koristi za permutacione probleme – problemi sekvenciranja, planiranja, usmeravanja.
- Rešenja su predstavljena permutacijom $\pi = (\pi_1, \pi_2, \dots, \pi_3)$ i svaki element problema mora da se pojavi jednom u permutaciji
  - *Problem trgovačkog putnika sa n gradova – rešenje predstavljeno permutacijom dužine n. Svaka permutacija predstavlja jedno rešenje. Prostor rešenja je reprezentovan skupom svih permutacija. Ako je prvi grad u ruti fiksira,. Tada je*

$$|S| = (n-1)!$$

  -

Linearna reprezentacija



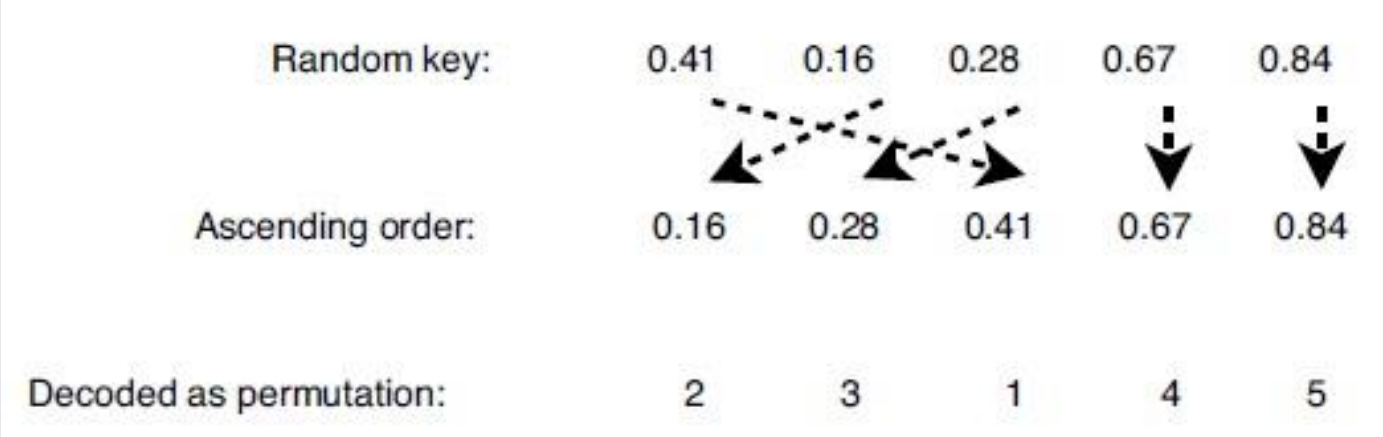
- Redukovanje prostora reprezentacije – izbor odgovarajuće reprezentacije pojednostavljuje problem
  - ***Problem N dama.*** *Rasporediti N dama na šahovsku tablu $N \times N$ tako da se međusobno ne napadaju.*
  - *Problem 8 dama je definisao šahista Max Bezzel 1848 godine*
  - *Ako se svako rešenje predstavi vektorom Dekartovih koordinata $x = (p_1, p_2, \dots, p_8)$ gde je $p_i = (x_i, y_i)$. Prostor rešenja je 64^8 čime se dobija preko 280 milijardi potencijalnih rešenja*
  - *Ako se u startu definiše da svaka dama mora bitu u drugoj vrsti, tada će imati 8^8 što je nešto preko 16 miliona potencijalnih rešenja*
  - *Ako se ne dozvoli da dve dame budu u istoj vrsti i koloni, tada se problem redukuje na problem permutacije n dama, tj. $8! = 40\,320$ mogućih rešenja*

Linearna reprezentacija

- Enkodiranje realnim vrednostima – prirodan način za predstavljanje kontinualnih problema
- Mešano enkodiranje parametaraske optimizacije –koristi se kod problema optimizacije parametara gde neki parametri imaju diskretne vrednosti, dok drugi imaju realne vrednosti.

Netradicionalna linearna reprezentacija

- Definisane za potrebe evolucionih algoritama
- Random-key enkodiranje –koristi realne vrednosti za predstavljanje permutacija
  - *Svakom objektu se dodeljuje slučajna realna vrednost uniformo iz [0,1]. Rešenje se dobija dekodiranjem u rastućem redosledu*



Netradicionalna linearna reprezentacija

- „*Messy*“ reprezentacija – vrednost dodeljena promenljivoj zavisi od njene pozicije. Svaki element reprezentacije je par (*varijabla*, *vrednost*). Enkodiranje može biti promenljive dužine
- „Nekodirani“ regioni (intron) – inspirisano biologijom. Nekodirani regioni ne utiču na ocenu kvaliteta rešenja. Predstavlja se u obliku

$$x_1|\text{intron}|x_2|\ldots|\text{intron}|x_n$$

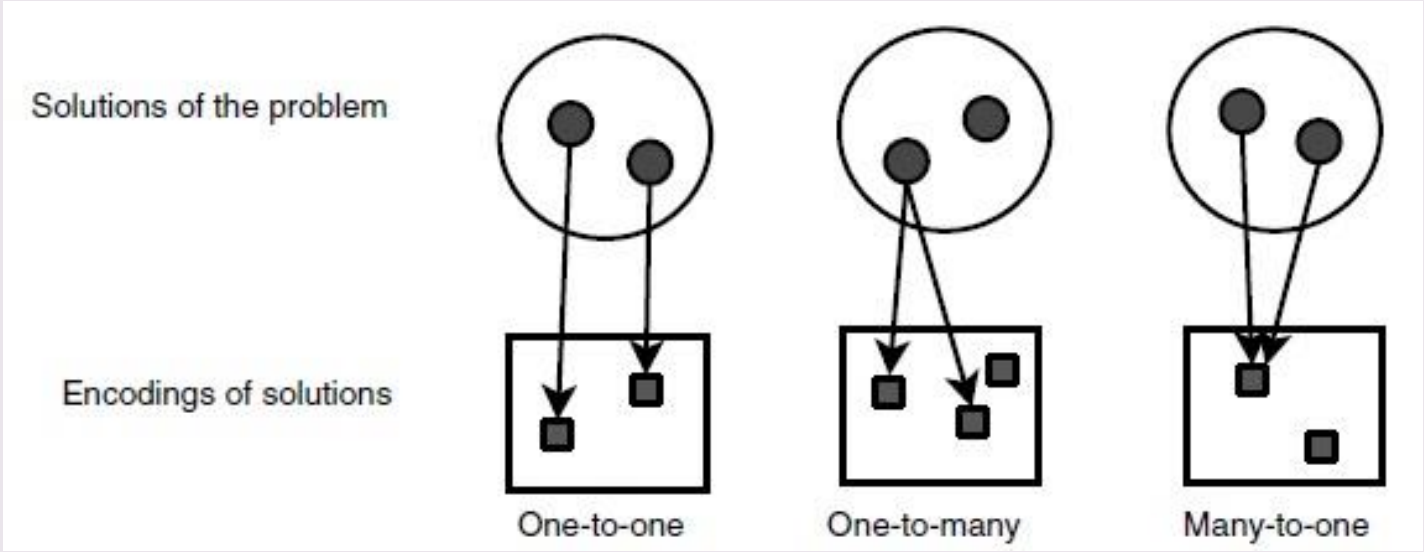
- „Dipliodna“ reprezentacija – uključuje više vrednosti za svaku poziciju u enkodiranju. Procedurom dekodiranja se određuje koja vrednost će biti dodeljena poziciji
- Kvantna reprezentacija – inspirisano kvantnim računarima. *Qubit* predstavlja super poziciju dve vrednosti u isto vreme. Pomoću n *qubit*-a se može predstaviti 2^n stanja u istom trenutku

Nelinearna reprezentacija

- Često bazirana na grafovima, posebno stablima
- Stabla su zgodna za predstavljanje hijerarhijskih struktura struktura (aritmetički izrazi, formule prvog reda, programi, ...)
  - ***Regresioni problem**. Za dati ulaz i izlaz pronaći funkciju koja za sve ulaze daje najbliže izlaze. Rešenje može biti predstavljeno u obliku stabla funkcija i terminala, pri čemu funkcije i terminali mogu biti definisani na različite načine*
- Mašina konačnih stanja, grafovi

Mapiranje *Reprezentacija→Rešenje*

- Ova funkcija treba da transformiše enkodiranje (*genotip*) u rešenje problema (*fenotip*)



Mapiranje *Reprezentacija→Rešenje*

- One-to-one – Tradicionalna klasa reperezentacije
  - *Svakoj reprezentaciji odgovara jedno rešenje i obrnuto*
  - *Ne postoji redudansa ni redukcija prostora pretrage*
  - *Za pojedine probleme zadovoljenja je teško definisati ovakvo preslikavanje*
- One-to-many
- Many-to-one

Mapiranje *Reprezentacija→Rešenje*

- One-to-one
- One-to-many
  - *Jednom rešenju odgovara više reprezentacija*
  - *Redudansa povećava prostor pretrage i utiče na efikasnost metaheuristike*
  - *Npr. simetrija kod problema vezanih za grafove*
- Many-to-one

Mapiranje *Reprezentacija→Rešenje*

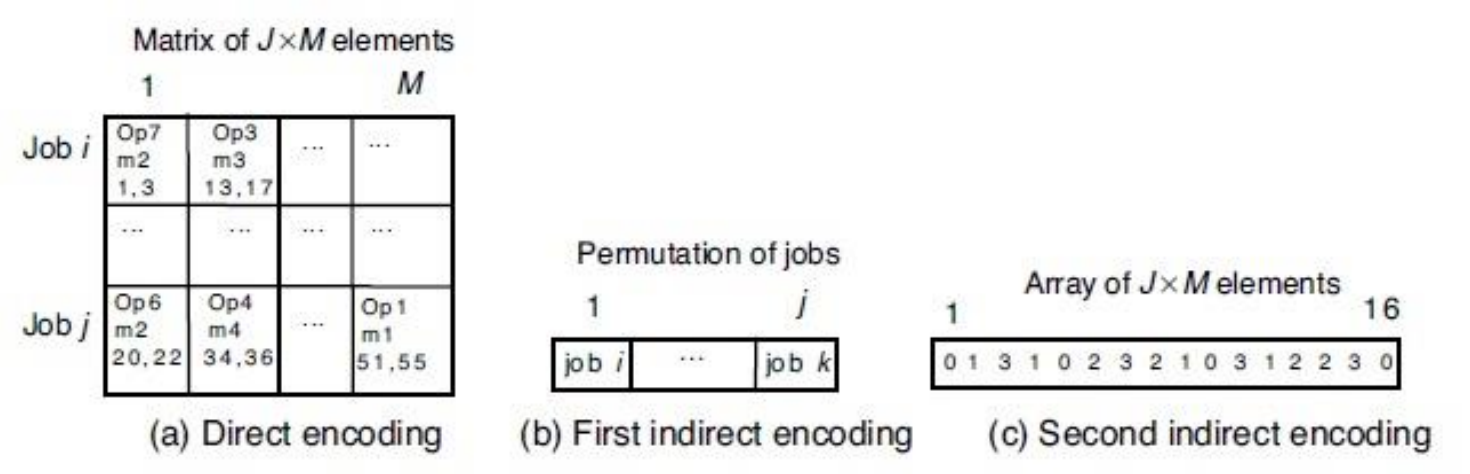
- One-to-one
- One-to-many
- Many-to-one
  - *Više rešenja reprezentovano na isti način*
  - *Nedostatak detalja u reprezentaciji rešenja – neke informacije nisu eksplicitno reprezentovanje*
  - *Smanjuje prostor pretrage, pa potencijalno povećava efikasnost metaheuristike*
  - *Indirektno enkodiranje*

Direktno vs. Indirektno enkodiranje

- Indirektrnim enkodiranjem dobijena reprezentacija nije kompletna
  - *Neophodan je dekoder da bi se dobilo rešenje od dobijene reprezentacije*
  - *Dekoder može biti nedeterministički*
  - *Često se koristi za probleme sa velikim brojem ograničenja – rasporedi,...*

Direktno vs. Indirektno enkodiranje

- **Job-shop problem raspoređivanja (JSP)** – Dato je j poslova. Svaki posao se sastoji od M operacija koje se realizuju na M mašina. Svaka operacija se realizuje na jednoj mašini. Svaki posao ima operaciju koja se realizuje na svakoj mašini. Funkcija cilja treba da minimizuje ukupno vreme procesa. Svaka mašina može da obavlja jednu operaciju u jednom trenutku. Operacije moraju da se obavljaju određenim redom. Rešenje treba da predstavlja raspored zauzeća na mašini.



Funkcija cilja

- Definiše cilj koji treba postići
- Povezuje svako rešenje iz prostora pretrage sa realnim brojem koji predstavlja kvalitet rešenja

$$f: S \to \mathbb{R}$$

- Omogućava uređenje svih rešenja iz prostora pretrage
- Rukovodi pretragom ka „dobrom“ rešenju
- Loše definisana funkcija cilja može dovesti do neprihvatljivih rešenja bez obzira na to koja je metaheuristika korišćena

„Lako definisane“ funkcija cilja

- Kod pojedinih optimizacionih problema funkcije cilja se „prirodno“ nameću
  - *TSP – Ukupan pređeni put*

$$f(s) = \sum_{i=1}^{n-1} d_{\pi(i),\pi(i+1)} + d_{\pi(n),\pi(1)}$$

  gde π reprezentaciju permutacijom putanje i n broj gradova
- Često je funkcija cilja sadržana u formulaciji problema

Vođene funkcije cilja

- Kod pojedinih problema definisanje funkcija cilja je bitno i teško pitanje
- Nekada je potrebno transformisati je u cilju bolje konvergencije metaheuristike
  - *k-SAT. Data je iskazna formula F u konjuktivnoj normalnoj formi, tj. sastoji se od m klauza (disjunkcija) C_i sa po k literala i ukupno n iskaznih slova. Naći interpretaciju iskaznih slova tako da formula F bude zadovoljena.*
    - Rešenje problema može da se predstavi kao binarni vektor dužine n
    - Prirodno nametnuta funkcija cilja bila bi

$$f = \begin{cases} 0 & F \text{ nije zadovoljeno} \\ 1 & \text{inače} \end{cases}$$

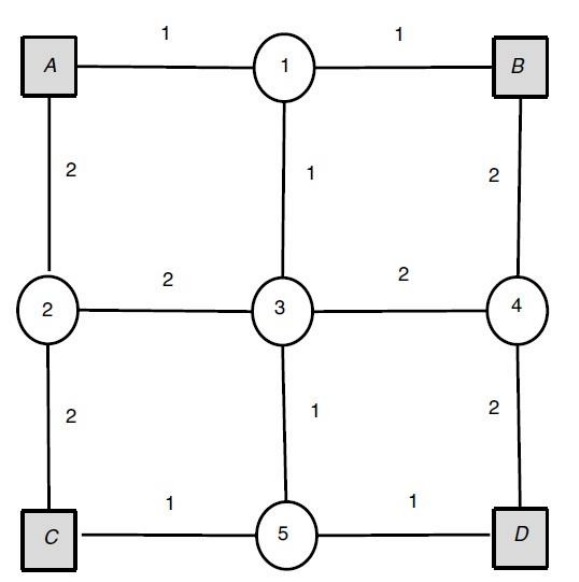
      - *Dva vesktora sa istom vrednošću funkcije $f = 0$*
    - Bolje bi bilo definisati funkciju koja za vrednost ima broj zadovoljenih klauza. Ovakva funkcija bi vodila MAX-SAT modelu

Dekodiranje reprezentacije

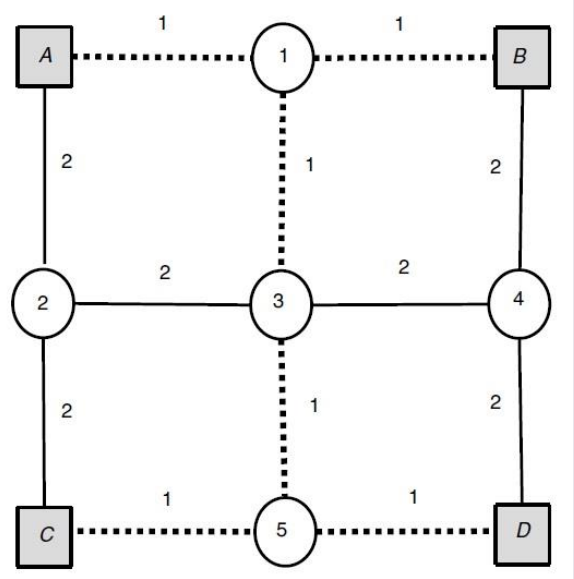
- Fukciju za dekodiranje potrebno je definisati kada mapiranje između repreznetacije i funkcije cilja nije očigledno, „pravolinijsko“
  - *Štajnerovo stablo (Steiner tree problem). Problem kombinatorne optimizacije sa velikom primenom u telekomunikacijama. Dat je neorijentisani težinski grah $G = (V, E)$ gde je V skup čvorova grafa i E skup ivica grafa. Težine dodeljene ivicama su sve pozitivne. Neka je $T \subseteq V$ skup terminala. Potrebno je naći minimalni težinski povezani graf koji uključuje sve terminale.*
  - *Za razliku od minimalnog razapinjućeg stabla (MST) koje je polinomne težine, Štajnerovo stablo je NP-težak problem*

Dekodiranje reprezentacije – Štajnerovo stablo

- Neka je $X \subseteq V$ neterminalnih čvorova.Rešenje se može predstaviti vektorom neterminala koji ne pripadaju rešenju.



Skup terminala dat sa $T = \{A, B, C, D\}$



Optimalno rešenje dato sa $s = \{1,5\}$

Interaktivna optimizacija

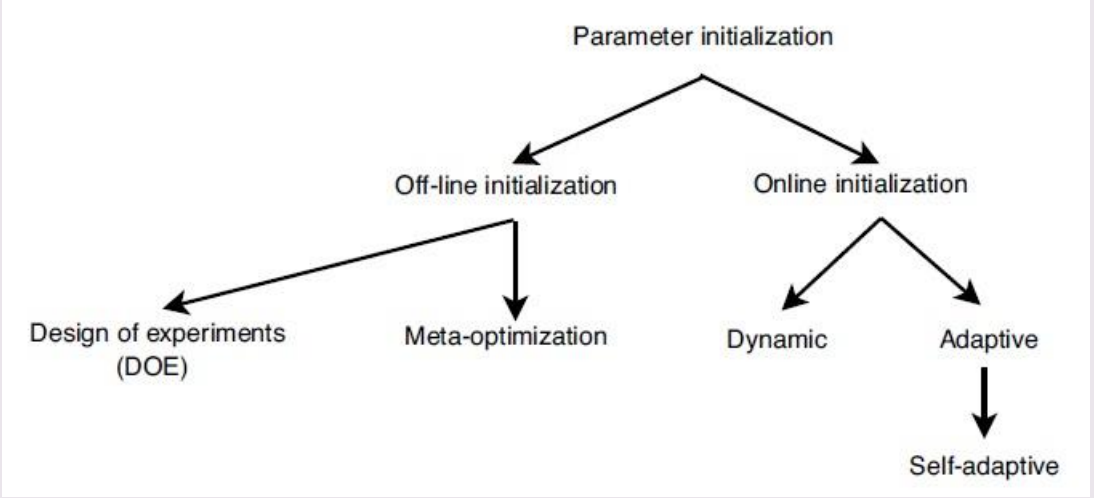
- Korisnik učestvuje „online“
- Dve glavne motivacije
  - *Intervencija korisnika u cilju rukovođena pretraživanjem – Cilj je postići bržu konvergenciju ka regionima koji obećavaju. Koristi se često pri višekriterijumskoj optimizaciji. Korisnik može da*
    - sugeriše na rešenja koja obećavaju na osnovu grafičke reprezentacije rešenja
    - sugeriše na promenu parametara heuristike
  - *Intervencija korisnika u evaluaciji rešenja – Kod nekih problema potrebna je subjektivna funkcija cilja (npr. ukus kafe) ili se funkcija ne može definisati analitički (npr. vizuelna atraktivnost)*
- Evaluacija od strane korisnika je spora i skupa, pa se mogu se koristiti kod rešenja koja se dobijaju u ograničenom broju iteracija ili ograničenoj populaciji

Meta-modeliranje

- U slučajevima da je izračunavanje funkcije cilja vremenski jako zahtevno, može se zameniti približnom, ali manje kompleksnom funkcijom cilja
- Jako zahtevne funkcija cilja se javljaju npr. pri trodimenzionom aerodinamičkom dizajniranju, proceni struktura zasnovanih na dinamici fluida i slično
- Postoje različite metodologije pri kreiranju aproksimacionih modela (Neuronske mreže, kvadratni polinomi, Gausov proces...)

Podešavanje parametara

- Vrednost parametara metaheuristike u velikoj meri utiče na njenu efikasnost
- Optimalne vrednosti parametara zavise od
  - *Problema koji se rešava*
  - *Instanci problema*
  - *I vremenu pretrage koje korisnik želi da potroši u rešavanju problema*
- Univerzalne optimalne vrednosti parametara ne postoje



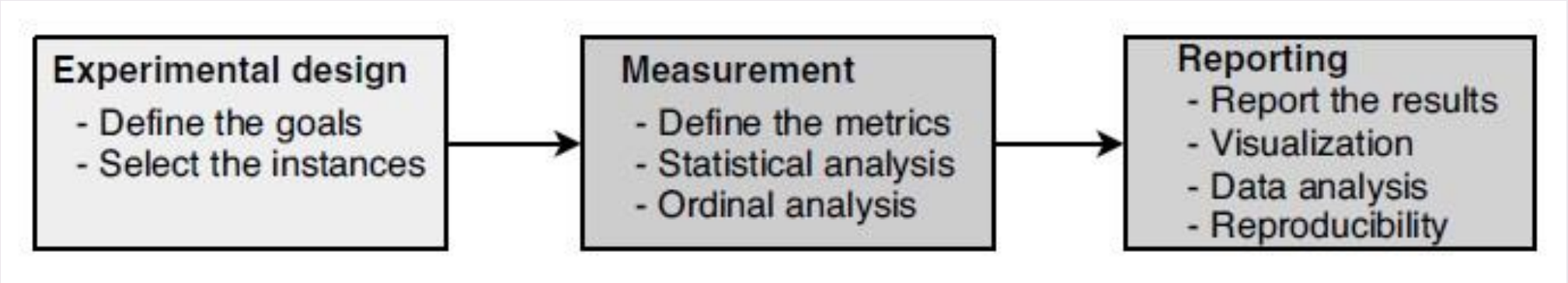
Off-line inicijalizacija parametara

- Meta-optimizacija – podešavanje jednog po jednog parametra, pri čemu se njihova vrednost određuje empirijski
- Ne proučava se interakcija među parametrima i ne garantuje se optimalna vrednost parametara
- Prevazilaženje ovih problema – dizajniranje eksperimenata
  - *Ako postoji n faktora/parametar i svaki ima k nivoa – potrebno je n^k eksperimenata*
  - *Vremenski vrlo zahtevno sa velikim brojem parametara i velikim domenom*
  - *Mali broj eksperimenata može postići korišćenjem **Latin hypercub** dizajna, sekvencijalnog dizajna, frakcijskog dizajna, **racing** algoritma...*
- Može se posmatrati kao optimizacioni problem
- Može se koristiti paralelno startovanje za različite vrednosti parametra

Online inicijalizacija parametara

- Dinamička promena
  - *Promena parametara se vrši ne uzimajući u obzir napredak pretrage*
  - *Promena može biti slučajna ili dinamička*
- Prilagodljiva promena
  - *Vrednosti parametara se menjaju u zavisnosti od napredka pretrage*
  - *Vrši se korišćenjem memorije pretrage*
  - *„samoprilagodljiva“ – procena parametrara tokom pretrage*

Analiza performansi



Dizajniranje eksperimenata

- Precizno definisanje ciljeva
- Eksperimenti, mera uspešnosti i statistička analiza zavise od namene heuristike
- Doprinos se može posmatrati kroz različite kriterijume
  - *Vreme pretrage, kvalitet rešenja, robusnost instanci, rešavanje širokog opsega problema, skalabilnost paralelizacije (broj procesora), jednostavnost implementacije, jednostavnost kombinovanja sa drugim algoritmimainovacije, automatsko podešavanje parametara…*
- Izbor vrednosti parametara

Dizajniranje eksperimenata

Function	Formulation
Sphere	$f(x) = \sum_{i=1}^{D} x_i^2$
Griewank	$f(x) = \frac{1}{4000} \sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$
Schaffer's f6	$f(x) = 0.5 - \frac{\left(\sin\sqrt{\left(x_1^2+x_2^2\right)}\right)^2 - 0.5}{\left(1+0.001\left(x_1^2+x_2^2\right)\right)^2}$
Rastrigin	$f(x) = \sum_{i=1}^{D} (x_i^2 - 10\cos(2\pi x_i) + 10)$
Rosenbrock	$f(x) = \sum_{i=1}^{D-1} (100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2)$

D represents the number of dimensions associated with the problem.

- Izbor skupa instanci za testiranje
  - *Instance iz realnog života vs. konstruisane instance*
  - *Moraju biti različite po veličini, težini rešavanja i strukturi*
  - *Neophodno ih je podeliti na dve podgupe*
    - Za podešavanje parametara
    - Za ocenu performansi pretrazivanja
- Prilikom izbora vrednosti parametara isti skup vrednosti se koristi za sve instance
- „Fino“ podešavanje za svaku instancu ponaosob može proizvesti *overfitting*
- Robusnost metaheuristike utiče na njeno rešavanje nepoznatih instanci
- Različite vrednosti parametara se mogu prihvatiti za različite strukture i veličine instanci

Ocena efikasnosti

- Kod kompletnih optimizacionih metoda glavni indikator efikasnosti vreme pretrage, obzirom da algoritmi garantuju optimalnost
- Kod heurističkog pristupa, u cilju kompletnog sagledavanja efikasnosti, neophodno je razmatrati
  - *Kvalitet rešenja – rastojanja među rešenjima*
  - *Vreme izračunavanja – empirijska analiza, vreme izvršavanja, broj izračunavanja funkcije cilja*
  - *Robusnost – neosetljivost na male promene primeraka ulaza ili parametara metaheuristike*

Statistička analiza

- Na dobijene eksperimentalne rezultate za različite indikatore neophodne je primeniti neku metodu statističke analize
  - *T-test, Eilcoxon, ANOVA, Kolmogorov, Levene, Mann-Whitney*
- Za dobijanje merodavni rezultat potrebno je istu metodu primeniti pod istim uslovima veći broj puta (bar 10, više od 100 ako je moguće)
- $\text{succes rate} = \frac{\text{number of successful runs}}{\text{total number of runs}}$
- $\text{performance rate} = \frac{\text{number of successful runs}}{\text{numbre of function evaluation} \times \text{total number of runs}}$

Izveštaji

- Nije dovoljno predstaviti tabelama velike količine podataka
- Alat za vuzuelizaciju podataka je poželjan
  - *Ineraction plots - Grafici koji prikazuju interakciju različitih faktora*
  - *Box plots – prokazuju najmanju vrednost, medijanu, najveću vrednost*
  - *Scatter plots – prikazivanje kompromisa izmedju različitih indikatora performansi*
- Za potrebe reprodukovanja mora biti dobro dokumentovana

Gotova okruženja

- Metaheuristike
  - *Evolutivni algoritmi – Galib*
  - *Lokalna pretraga – EasyLocal, Localizer*
- Optimizacioni problemi
  - *Nelinearna kontinualna optimizacija – GenocopIII*
  - *Kombinatorna optimizacija – iOpt*
  - *Jednokriterijumska optimizacija – BEAGLE*
  - *Višekriterijumska optimizacija – PISA*
- Paralelna i hibridne
  - *DREAN, ECJ, JDEAL…*
- …

Gotova okruženja

Framework or Library	Metaheuristic	Optimization Problems	Parallel Models	Communication Systems
EasyLocal++	S-meta	Mono	-	-
Localizer++	S-meta	Mono	-	-
PISA	EA	Multi	-	-
MAFRA	LS, EA	Mono	-	-
iOpt	S-meta, GA, CP	Mono, COP	-	-
OptQuest	SS	Mono	-	-
GAlib	GA	Mono	Algo-level Ite-level	PVM
GenocopIII	EA	Mono, Cont	-	-
DREAM	EA	Mono	Algo-level	Peer-to-peer sockets
MALLBA	LS EA	Mono	Algo-level Ite-level	MPI Netstream
Hotframe	S-meta, EA	Mono	-	-
TEMPLAR	LS, SA, GA	Mono, COP	Algo-level	MPI, threads
JDEAL	GA, ES	Mono	Ite-level	Sockets
ECJ	EA	Mono	Algo-level	Threads, sockets
Dist. BEAGLE	EA	Mono	Algo-level Ite-level	Sockets
ParadisEO	S-meta P-meta	Mono, Multi COP, Cont	Algo-level Ite-level Sol-level	MPI, threads Condor Globus