Metaheurističke metode 4.nedelja

November 3, 2024

(Meta)heuristike

- Na osnovu zdravog razuma, intuicije
- Ponekad se zasnivaju na fizičkim, biološkim pojavama (npr. simulirano kaljenje, genetski algoritami, mravlje kolonije,...)
- Obično veoma vremenski efikasane
- Nema rigorozne matematičke analize
- Ne garantujtu optimalno rešenje
- Neretko dostižu prilično dobra rešenja, čak i optimalna
- Zasnovane na skupu uopštenih pravila na visokom apstraktnom nivou
- Mogu se primeniti na veliki skup problema optimizacije

Metaheuristike: Podela

- S-metaheuristike (engl. Single-solution based) - zasnovane na iterativnom poboljšanju jednog rešenja
- P-metaheuristike (engl. Population based) - istovremeno rade nad skupom rešenja koji se naziva populacija

Lokalna pretraga (Local Search)

- Najstarija i najjednostavnija heuristika
- Pogodna i za diskretnu i za kontinualnu optimizaciju
- Koraci:
  - Odabrati reprezentaciju rešenja, kao i prostor rešenja X

    (Na primer: rešenje je vektor sa celobrojnim vrednostima koje predstavljaju lokacije na kojima se uspostavljaju centri. Prostor X je prostor svih celobrojnih vektora sa različitim elementima odgovarajuće dužine.)
  - Odabrati početno dopustivo rešenje x

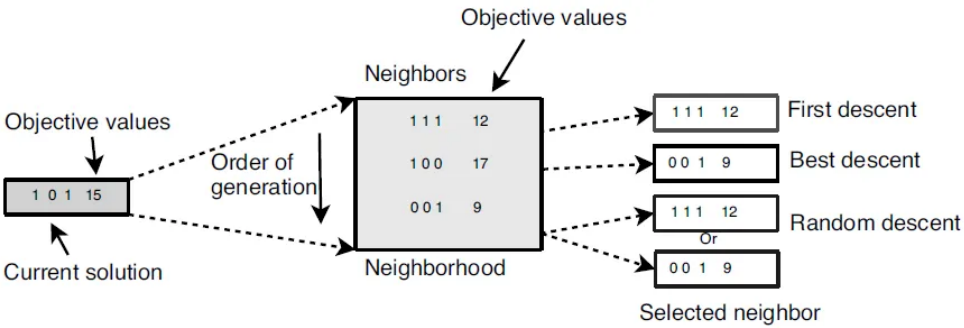
    (Generisati nasumično ili nekom heuristikom.)
  - Definisati okolinu reše nja koja će se pretraživati. U zavisnosti od topologije prostora, bira se metrika (euklidska, Hamingova, Menhetn...) pomoću koje se za proizvoljno dopustivo rešenje x definiše podskup $N(x) \subseteq X$ koji predstavlja okolinu rešenja x, a njeni članovi su susedi od x.

Metoda počinje pretraživanjem okoline početnog rešenja $N(x)$. Ako se naidje na bolje rešenje, ono postaje tekuće i nastavlja se sa pretraživanjem njegove okoline. Ukoliko takvo rešenje ne postoji (tekuće rešenje je lokalni optimum), LS se zaustavlja i poslednje tekuće rešenje se uzima za aproksimaciju optimalnog rešenja.

Strategije za izbor novog suseda

- **Best improvement**: Za sledećeg suseda bira se najbolji sused iz okoline $N(x)$, tj. onaj sa najboljom vrednošću funkcije cija. Potrebno je za svakog suseda izračunati vrednost funckije cilja, što može dosta povećati vreme izvršavanja algoritma u celini, posebno ukoliko se pretražuje veća okolina.
- **First improvement**: Bira se prvi sused koji je bolji od tekućeg rešenja. Na ovaj način ne ocenjuju se sva rešenja koja se nalaze u okolini tekućeg, već samo odredjeni broj njih dok se ne pronadje rešenje sa boljom vrednošću funkcije cilja. Prateći odredjeni redosled generisanja suseda, računaju se njihove vrednosti funkcije cilja i čim se naidje na poboljšanje, vrši se zamena tekućeg rešenja susedom na kom se to poboljšanje postiže. U najgorem slučaju (ako je tekuće rešenje lokalni optimum) evaluiraju se sva rešenja.
- **Random selection**: Nasumično se bira sused koji je bolji od tekućeg rešenja i on postaje novo tekuće rešenje.

Strategije za izbor novog suseda

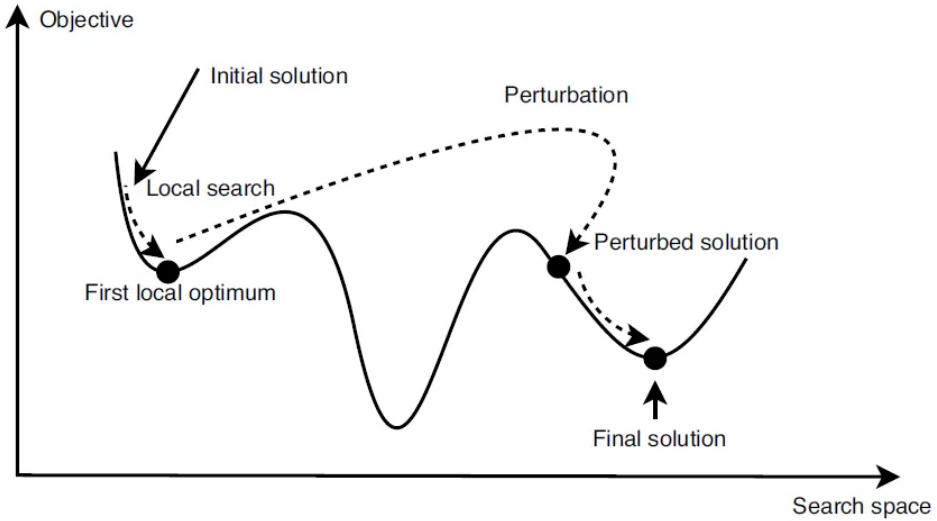


Modifikacije

Mana:Pretraga često može da zaglavi u lokalnom optimumu koji nije globalni.
Metoda osetljiva na izbor početnog rešenja.
Poboljšanja:

- **Multistartni pristup** (Multistart Local Search) Budući da kvalitet rešenja dobijenog LS zavisi i od početnog rešenja, ideja na kojoj se zasniva MLS je generisati više početnih rešenja (na slučajan način) i nad svakim primeniti LS. Sva izvršavanja su medjusobno nezavisna i za aproksimaciju optimalnog rešenja bira se najbolji lokalni optimum.
- **Iterirano lokalno pretraživanje** (Iterated Local Search) Dodatno poboljšanje možemo dobiti povezivanjem lokalnih optimuma dobijenih MLS metodom (npr. rezultat jedne MLS iteracije prosledimo kao početno rešenje sledeće MLS iteracije). Vrši se perturbacija rešenja dobijenog u jednoj MLS iteraciji i rezultat proslediti kao početno rešenje sledeće MLS iteracije.

ILS


Objective
Initial solution
Perturbation
Local search
Perturbed solution
First local optimum
Final solution
Search space

ILS

Algorithm 1 Iterated Local Search

s = Generisanje_inicijalnog_resenja;
s^* = Lokalna_pretraga(s);
while nije ispunjen kriterujm zaustavljanja **do**
    new_s =Perturbacija(s^*, istorija_pretrage);
    new_{s^*} = Lokalna_pretraga(new_s);
    s^* = Kriterijum_prihvatanja_resenja(s^*, new_{s^*},istorija_pretrage);
end while
return s^*

ILS

- LS: bilo koja S-metaheuristika (deterministička ili stohastička) može se koristiti za pretraživanje prostora umesto klasične lokalne pretrage.
- Perturbacija: Treba uneti dobar balans izmedju diverzifikacije i intenzifikacije u proces pretrage. Dužina perturbacije (broj komponenti rešenja koje menjamo) jedan je od parametara koji treba prilagoditi problemu koji se rešava. Male perturbacije tekućeg rešenja mogu vratiti proces pretrage u već posećeni lokalni optimum, dok suviše velike perturbacije bi izbrisale svu prethodnu istoriju pretrage menjajući dobre komponente tekućeg rešenja. Perturbacije mogu biti fiksnih i promenljivih dužina, nasumične i poludeterminističke perturbacije.
- Kriterijum prihvatanja rešenja: Najčešće se koristi stroga selekcija, tj. prihvatanje samo onih rešenja koja poboljšavaju vrednost funkcije cilja, i to sa ciljem pojačavanja intenzifikacije. Suprotan princip kojim se pojačava diverzifikacija je slaba selekcija, tj. prihvatanje svih rešenja, bez obzira na njihovu vrednost funkcije cilja. U upotrebi su i kriterijumi koji prave balans izmedju ova dva principa, prihvatanje rešenja sa odredjenom verovatnoćom koja zavisi od kvaliteta rešenja, deterministički kriterijum kojim se na osnovu unapred definisanog praga odlučuje da li se dobijeno rešenje prihvata ili ne.

Problem trgovačkog putnika
Travelling salesman problem

- Dat je V skup čvorova koji predstavljaju gradove, $|V| = n$, i E skup grana, tj. skup uredjenih parova $(i, j), i, j \in V$.
- Svakoj grani (i, j) dodeljena je težina C_{ij} koja predstavlja trošak putovanja od grada i do grada j.
- Cilj je naći najkraći mogući put (turu sa minimalnim troškovima) tako da se svaki grad poseti tačno jednom i da se vrati u početni grad.
- Početni čvor (polazni grad) može, ali ne mora biti fiksiran.

Uvodimo skup promenljivih:

$$x_{ij} = \begin{cases} 1, \text{grana } (i,j) \in E \text{ je deo rešenja} \\ 0, \text{inače} \end{cases}$$

TSP: Model

$$\min \sum_{(i,j)\in E} c_{ij}x_{ij},$$

$$\sum_{i\in V} x_{ij} = 1,\ \forall j \in V,$$ (Iz svakog grada se odlazi u tačno jedan grad.)

$$\sum_{j\in V} x_{ij} = 1,\ \forall i \in V,$$ (U svaki grad se dolazi iz tačno jednog grada.)

$$\sum_{i,j\in S} x_{ij} \leq |S| - 1,\ \forall S \subset V, 2 \leq |S| \leq \frac{n}{2}.$$ (Podture se zabranjuju.)

$$x_{ij} \in \{0, 1\}$$

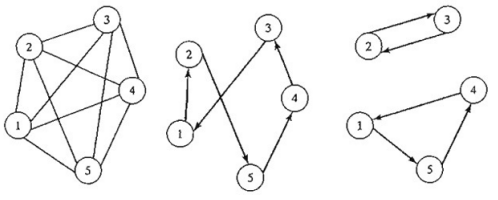


FIGURE 9.11

Primena LS, MLS, ILS i GRASP metode na rešavanje TSP problema

- Reprezentacija rešenja: rešenje je predstavljeno vektorom dužine n (broj gradova) čiji su elementi celi brojevi iz intervala $[0, n-1]$.
- Npr. ako je $n = 5$, rešenjem $x = [1\ 2\ 0\ 3\ 4]$ odredjena je tura koja počinje u gradu 1, redom posećuje gradove $2, 0, 3$ i 4 i završava u gradu 1.
- Okoline: Susedi rešenja x dobijaju se primenom jednog od operatora umetanja, inverzije i zamene.

Operatori

Figure: Operator umetanja

Figure: Operator zamene

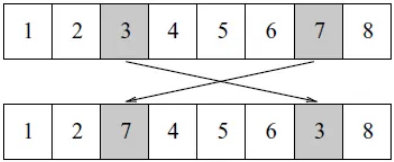
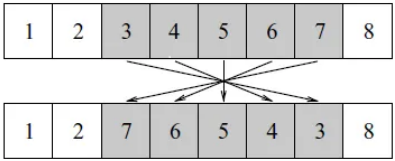
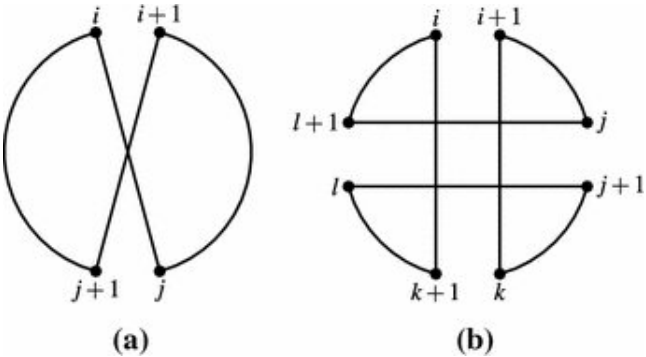


Figure: Operator inverzije



ILS: Perturbacija

Figure: 2-opt (a) vs double bridge (b)



2-opt: tura $[i, i+1, d, j, j+1, l]$ menja se turom $[i, j, d, i+1, j+1, l]$.

Questions?