Matematički fakultet  
Univerziteta u Beogradu

Seminarski rad u okviru kursa  
Naučno izračunavanje

Profesor: dr Mladen Nikolić  
Asistent: dr Stefan Mišković

**Tabu pretraga**

**za lokacijski problem ograničenih kapaciteta**

**sa jednostrukim alokacijama**

Marija Mijailović 1093/2017  
[mi14199@alas.matf.bg.ac.rs](mailto:mi14199@alas.matf.bg.ac.rs)

Miroslav Mišljenović 1110/2018  
[mr12260@alas.matf.bg.ac.rs](mailto:mr12260@alas.matf.bg.ac.rs)

Beograd, septembar 2019.

1. Prost lokacijski problem ograničenih kapaciteta

sa jednostrukim alokacijama

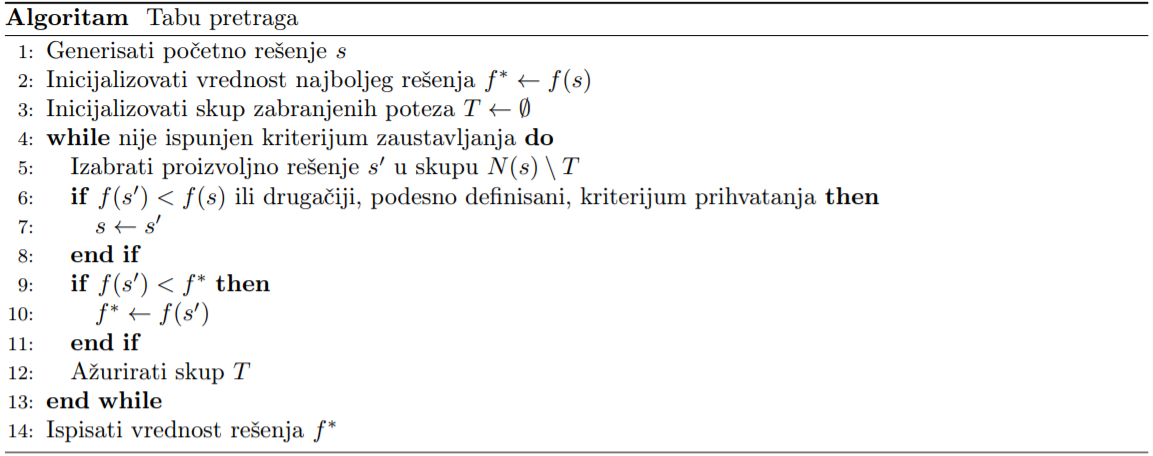
Prost lokacijski problem ograničenih kapaciteta sa jednostrukim alokacijama (eng. single-source capacitated facility location problem – SSCFLP) bavi se problematikom dodeljivanja korisnika određenim resursima. Svaki korisnik ima svoje zahteve (eng. demands), dok resurse karakteriše kapacitet, kao i fiksna cena korišćenja pojedinačnog resursa, odnosno cena uspostavljanja tog resursa. Poznata je i matrica cena, tj. matrica koja govori o tome koliko korisnika košta korišćenje nekog resursa. Svaki korisnik mora biti pridružen tačno jednom resursu, dok jedan resurs može opsluživati više korisnika. Zadatak je minimizovati ukupnu cenu korišćenja takvog sistema raspoređivanja korisnika i uspostavljanja resursa.

1. Tabu pretraga

Tabu pretraga je metaheuristika koja se zasniva na poboljšavanju vrednosti tekućeg rešenja. Za razliku od lokalne pretrage, ovde se uvodi skup zabranjenih poteza T, koji sužava izbor novog rešenja. U svakoj iteraciji se skup T može ažurirati na različite načine. Na početku algoritma se proizvoljno ili na neki drugi način generiše početno rešenje i izračunava vrednost njegove funkcije cilja. Vrednost najboljeg rešenja se najpre inicijalizuje na vrednost početnog. Zatim se algoritam ponavlja kroz nekoliko iteracija. U svakom koraku se razmatra rešenje u okolini trenutnog. Ukoliko je vrednost njegove funkcije cilja bolja od vrednosti funkcije cilja trenutnog rešenja, ažurira se trenutno rešenje.

Po potrebi se ažurira i vrednost najboljeg dostignutog rešenja. Algoritam se ponavlja dok nije ispunjen kriterijum zaustavljanja. Kriterijum zaustavljanja može biti, na primer, dostignut maksimalan broj iteracija, dostignut maksimalan broj ponavljanja najboljeg rešenja, ukupno vreme izvršavanja, itd.

Tabu pretraga se može prikazati sledećim pseudokodom:



Memorija se kod tabu pretrage vezuje za dužinu čuvanja rešenja u skupu T. U opštem slučaju, ona može biti kratkoročna, srednjoročna i dugoročna.

Kratkoročna memorija se implementira kao lista nedavno razmatranih rešenja. Kada se potencijalno rešenje pojavi u tabu listi, ne može se ponovo razmatrati u glavnom delu algoritma, sve dok se to rešenje ne izbaci iz tabu liste, po nekom unapred dogovorenom kriterijumu.

Srednjoročna memorija predstavlja pravila intenziviranja namenjena pristranosti prema obećavajućim područjima prostora pretrage.

Dugoročna memorija sadrži pravila diverzifikacije tj. pravila koja pretražuju nove regione (kada se pretraga zaglavi “na platou” ili u lokalnom optimumu, ova pravila određuju kako se tabu lista prazni ili osvežava).

1. Implementacija algoritma

Okosnica našeg seminarskog rada se sastoji, u osnovi, od koda sa vežbi iz oblasti lokalne pretrage, uz izmene funkcije “solutionValue” i dodatak funkcije “tabuSearch”.

U glavnom delu programa imamo spoljašnju petlju, gde se skup uspostavljenih resursa inicijalizuje na slučajan način.

“SolutionValue” je funkcija koja izračunava vrednost funkcije cilja za zadatu konfiguraciju, tj. uspostavljene resurse. Tu je implementirana unutrašnja petlja za naš program, tako što se skup korisnika permutuje na slučajan način, za koren broja korisnika. Ovo je urađeno u cilju postizanja što boljih rezultata. Ispostavilo se da su dobijena mnogo bolja rešenja, nego bez pomenute unutrašnje petlje i permutacije korisnika.

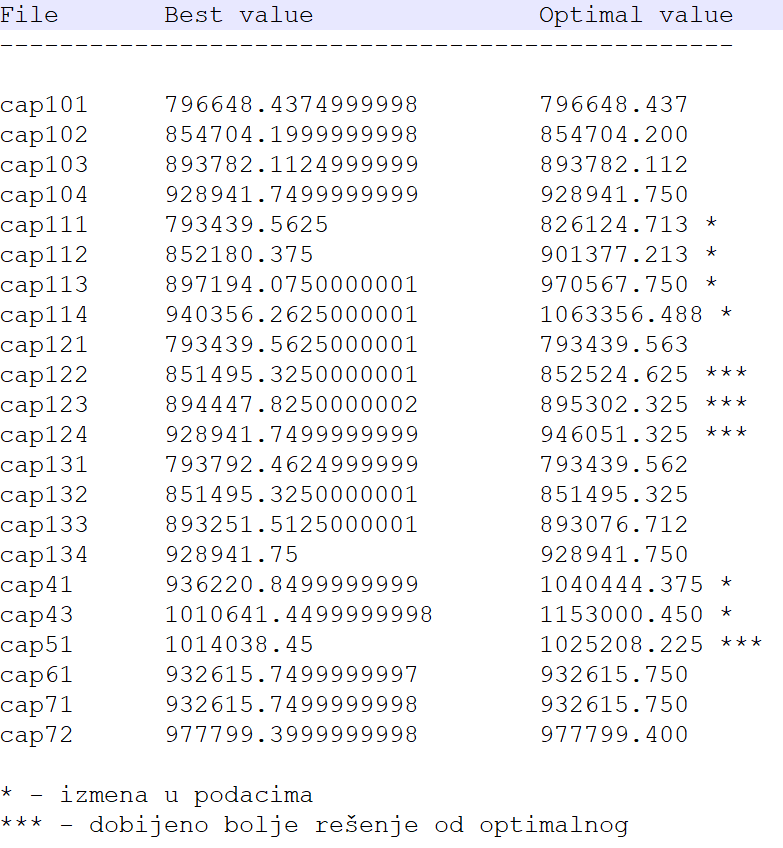
Funkcija “tabuSearch” sadrži implementaciju tabu liste bazirane na kratkoročnoj memoriji.

1. Rezultati

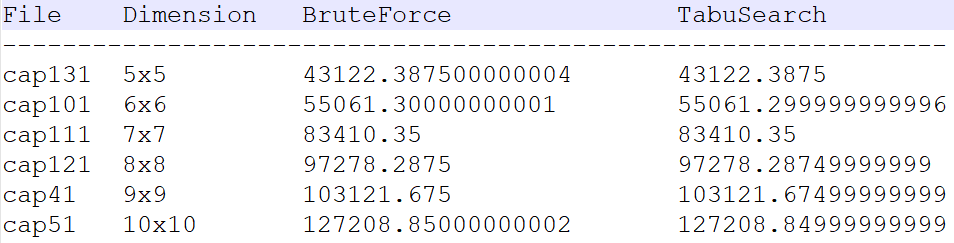
Algoritam smo testirali na 22 test primera, preuzetih sa sajta: <http://people.brunel.ac.uk/~mastjjb/jeb/orlib/capinfo.html>

Neki od test primera su sadržali greške, tako da nisu mogla da se dobiju zadovoljiva rešenja, jer su neki zahtevi korisnika bili veći nego kapaciteti resursa. Te podatke smo korigovali kako bi se dobilo zadovoljivo rešenje, stoga su dobijeni različiti rezultati od optimalnih. Optimalni rezultati dati su na linku:

<http://people.brunel.ac.uk/~mastjjb/jeb/orlib/files/capopt.txt>



Takođe, testirali smo algoritam na primerima od 5x5 do 10x10 redom. Optimalna rešenja za te test primere smo dobili pomoću algoritma grube sile, a našim algoritmom dobili smo poklapanja sa tim rešenjima.



1. Literatura

* Izvorni kod:

<https://github.com/miroslav-misljenovic/Tabu-search-heuristic-for-the-Single-Source-Facility-Location-Problem>

* Sadržaji kursa Naučnog izračunavanja:

<http://ni.matf.bg.ac.rs/>

* Tabu pretraga:

<https://en.wikipedia.org/wiki/Tabu_search>

* Test primeri:

<http://people.brunel.ac.uk/~mastjjb/jeb/orlib/files/>