Tabu pretraživanje za rješavanje višedimenzionalnog 0-1 problema ranca

Muhamed Parić

Seminarski rad iz predmeta Optimizacija resursa

Predmetni profesor:
Prof. dr. Samim Konjicija

Predmetni asistent:
Jasmina Zubača, MoE

Sažetak

Problem ranca je tip optimizacijskog problema, pri čemu je cilj pronaći optimalan podskup predmeta iz nekog skupa koji ne narušavaju ograničenja. S obzirom na to da je problem klasifikovan kao NP-hard problem, u praksi se za rješavanje koriste heurističke metode koje su mnogo brže od optimalnih, a dobiveno rješenje je najčešće dovoljno dobro. Tabu pretraga je jedna od takvih heurističih metoda, koja je pogodna upravo za rješavanje problema ranca. U radu su opisani i konkretan problem i algoritam tabu pretrage, neki pristupi optimalnom rješavanju problema, kao i modifikacije na tabu pretragu koje su urađene s ciljem povećanja istraženog problemskog prostora. Algoritam je testiran na nasumično generisanim primjerima, te su rezultati tih eksperimenata prikazani u posljednjem poglavlju..

Abstract

The knapsack problem is an optimization problem, where the goal is to find an optimal subset of a set of items that does not exceed the given constraints. Considering that the problem is classified as an NP-hard problem, heuristic methods for solving it are used in practice, which are much faster than methods that give the optimal solution, while still giving a sufficiently good solution in most cases. Tabu search is one such heuristic method, which is especially well suited for solving the knapsack problem. The paper describes both the specific variant of the problem and the tabu search algorithm, approaches to solving the problem optimally, as well as modifications to tabu search that were done with the goal of expanding the problem space to be explored. The algorithm was tested on randomly generated test cases, the results of which were shown in the final chapter.

Sadržaj

1. Opis problema

Problem ranca (eng. *Knapsack problem*) je optimizacijski problem izbora određenog broja predmeta iz skupa čiji je zbir vrijednosti maksimalan, a zauzeće resursa ne veće od maksimalno dozvoljenog. Problem možemo vizualizirati kao problem ubacivanja predmeta u ranac, tako da zbir težina svih predmeta ne prelazi nosivost ranca, a zbir vrijednosti svih predmeta je maksimalan. Problem ranca ima više varijanti, koje, za naše potrebe, dijelimo po maksimalnom broju svakog predmeta i po broju ograničenja.

Višedimenzionalni 0-1 problem ranca ograničava broj svakog predmeta na 1 (drugim riječima, svaki predmet možemo ili staviti u ranac ili ga ostaviti), dok je broj različitih ograničenja (ili broj dimenzija) proizvoljan. Intuitivno možemo dva ograničenja zamisliti kao ograničenje na maksimalnu težinu i maksimalnu zapreminu svih predmeta. Broj ograničenja ne utiče značajno na pristup rješavanju, barem idejno, jer je svako rješenje (koje je zapravo neka kombinacija predmeta iz skupa) ili dozvoljeno ili nije. Rješenje koje prekorači bilo koje od ograničenja je nedozvoljeno.

Problem ranca je vrsta problema cjelobrojnog linearnog programiranja, dok je 0-1 problem ranca vrsta 0-1 linearnog programiranja, tako da ga možemo opisati kao problem linearnog programiranja na sljedeći način:

$$
\begin{aligned}
&\arg\max \mathrm{Z(x)} = \sum_{i=1}^{n} x_i v_i \\
&\text{p.o.} \\
&\sum_{i=1}^{n} W_{1i} x_i \leq b_1 \\
&\sum_{i=1}^{n} W_{2i} x_i \leq b_2 \\
&\vdots \\
&\sum_{i=1}^{n} W_{mi} x_i \leq b_m \\
&\mathrm{x}_i \in \{0,1\}\ za\ i = 1..n
\end{aligned}
$$

Pri tome je n broj predmeta, m broj ograničenja, x vektor koji predstavlja rješenje (pri čemu je x_i=1 ako je i-ti predmet dio rješenja, dok je u suprotnom x_i=0), v vektor vrijednosti svakog od predmeta, W je mxn matrica koja takva da W_{ij} predstavlja količinu i-tog resursa koja je potrebna za j-ti predmet, a b vektor ograničenja na maksimalnu količinu svakog resursa, dužine m. Za potrebe ovog rada, elementi matrice W, te vektora v i b su pozitivni realni brojevi.

Tabu pretraga je pristup pronalaženju optimuma funkcije. U najjednostavnijoj varijanti, tabu pretraga je modifikacija lokalnog pretraživanja koja se razlikuje na dva načina: 1. Ako prelazak u bolje rješenje nije moguć, prelazi se u najbolje lošije rješenje i 2. Zadnjih N posjećenih rješenja se drži u tabu listi (pri čemu je vrijednost N određena empirijski), i prelazak u bilo koje rješenje koje se nalazi u tabu listi nije dozvoljen. Kombinacijom ove dvije modifikacije dobivamo pristup koji može "izaći" iz lokalnih optimuma, što omogućava pretragu znatno većeg područja pretrage.

Postoji više varijanti tabu pretrage, koje uključuju dodatne strategije. U ovom radu su opisane dvije strategije (diverzifikacija i strateško osciliranje), koje se koriste (uz prelazak u lošije rješenje) u situacijama kada trenutno rješenje predstavlja lokalni optimum.

2. Optimalno rješenje

Knapsack problem spada u klasu NP-hard problema, što znači, između ostalog, da u općem slučaju nije moguće pronaći rješenje u polinomijalnom vremenu. Problemi ovog tipa upravo i jesu pogodni za rješavanje heurističkim metodama jer bi klasičnim metodama trebala ogromna količina vremena i/ili radne memorije za njihovo rješavanje.

Najjednostavniji pristup rješavanju problema jeste tzv. *brute-force* pristup, koji uključuje isprobavanje svih mogućih rješenja i, od svih dozvoljenih, izbor onog koje daje najveću vrijednost funkcije cilja. Broj kombinacija skupa od n članova je 2^n, tako je to broj rješenja koje moramo testirati. Da bismo provjerili da li je neko rješenje dozvoljeno ili ne, moramo izvršiti množenje matrice i vektora, te ako jeste rješenje dozvoljeno, vršimo množenje dva vektora da bismo odredili vrijednost funkcije cilja za to rješenje. Zbog ogromne količine zahtijevanog računanja, ovaj pristup nije moguće koristiti za slučajeve sa više od veoma malog broja predmeta.

Drugi, mnogo brži pristup, se zasniva na tehnici dinamičkog programiranja. Vrijeme izvršavanja ovog algoritma je proporcionalno broju predmeta i proizvodu svih članova vektora ograničenja. Često je veći problem činjenica da je utrošena količina memorije također proporcionalna istim faktorima. Osim toga, za efikasno korištenje ovog pristupa, svi članovi matrice W moraju biti cijeli brojevi (pri čemu je u nekim slučajevima moguće problem preformulisati u oblik koji samo koristi cijele brojeve).

3. Tabu pretraga

3.1. Model problema

Za korištenje tabu pretrage, moramo odrediti kako ćemo predstaviti rješenje, te definisati susjedstvo nekog rješenja. Najjednostavniji način predstavljanja rješenja je korištenje vektora dužine n, kako je to već opisano. Susjedi rješenja x su sva rješenja koja se razlikuju od x u tačno jednom elementu. Odavde vidimo da je broj susjeda nekog rješenja n.

3.2. Diverzifikacija

Kod tabu pretraživanja, diverzifikacija uključuje korištenje tzv. dugoročne memorije za prelazak u rješenja u dijelu problemskog prostora koja do sada nisu bila posjećena. U implementaciji algoritma, dugoročna memorija je predstavljena preko dvije varijable: diverzifikacijskog brojača koji drži broj iteracija algoritma od posljednjeg poboljšanja najboljeg pronađenog rješenja, te vektora koji drži broj iteracija algoritma od posljednjeg korištenja predmeta u trenutnom rješenju za svakog od predmeta. Ako trenutno rješenje predstavlja lokalni maksimum (ili ako se sva bolja rješenja nalaze u tabu listi), prva korištena strategija je diverzifikacija. Ako je diverzifikacijski brojač dostigao određenu vrijednost, vršimo diverzifikaciju tako što određeni broj predmeta koji se najduže nisu nalazili u trenutnom rješenju dodamo u rješenje, te brojač postavimo na 0.

3.3. Strateško osciliranje

Da bismo vršili diverzifikaciju, opisani brojač mora dostići određenu vrijednost. U suprotnom vršimo strateško osciliranje, koje nam omogućava prelazak u nedozvoljeni dio problemskog prostora. Naime, ako u nekoj iteraciji ne možemo preći u bolje dozvoljeno rješenje bilo zbog nepostojanja takvog rješenja ili ograničenja tabu liste, i ako brojač diverzifikacije nije dostigao potrebnu vrijednost, prelazimo u najbolje od svih susjednih rješenja koja nisu u tabu listi. Ovim postupkom će trenutno rješenje postati nedozvoljeno.

Ponašanje algoritma u nedozvoljenom prostoru je drugačije nego u dozvoljenom. Naime, algoritam više ne prelazi u bolje rješenje na osnovu rezultata funkcije cilja, nego na osnovu tzv. *mjere nedozvoljenosti*. Mjera nedozvoljenosti kvantificira udaljenost trenutnog rješenja od dozvoljenog problemskog prostora. U našem slučaju, mjera nedozvoljenosti je zbir svih prekoračenja ograničenja, pri čemu je prekoračenje normalizovano dijeljenjem sa ograničenjem. Odnosno:

$$\mathrm{M}=\sum_{j=1}^{m} max\{\sum_{i=1}^{n}(W_{ji}x_i - b_j)/b_j, 0\}$$

Dok se nalazi u nedozvoljenom prostoru, algoritam će uvijek preći u rješenje koje najviše smanjuje mjeru nedozvoljenosti, osim ako se sva takva rješenja nalaze u tabu listi, u kojem slučaju će preći u rješenje koje najmanje povećava mjeru nedozvoljenosti.

U procesu diverzifikacije, moguće je da će novo rješenje biti nedozvoljeno, nakon čega algoritam vrši isti postupak pokušaja izlaska iz nedozvoljenog prostora.

Jedini slučaj koji nam je preostao jeste situacija u kojoj sve sva susjedna rješenja (i dozvoljena i nedozvoljena) nalaze u tabu listi, a diverzifikacijski brojač nije dostigao potrebnu vrijednost za vršenje diverzifikacije. U tom slučaju, algoritam će jednostavno izbrisati sve elemente iz tabu liste.

4. Testiranje

U prvom testu, težina svih 100 predmeta je 1, te je sam problem jednodimenzionalan. 50 nasumično izabranih predmeta ima vrijednost od 2, dok ostalih 50 ima vrijednost od 1. Maksimalna težina je 30, što znači da optimalno rješenje ima vrijednost od 60. Testirane su vrijednosti dužine tabu liste od 10 do 370 (pri čemu je svaki put dužina povećana za 40), dok je broj iteracija algoritma 15 do 150, sa povećanjem od 15 u svakoj iteraciji. Svaki od 100 slučajeva je isproban 5 puta i uzeta je prosječna vrijednost rezultata, pri čemu je vektor vrijednosti nasumično izmiješan svaki put.

Dobivena matrica vrijednosti pronadjenih rješenja je:

```
[23 45 47 50 51 54 57 60 60 60]
[22 45 47 49 52 54 57 60 60 60]
[22 45 47 50 53 55 57 60 60 60]
[23 44 48 50 53 54 56 60 60 60]
[21 42 46 50 51 54 58 60 60 60]
[21 44 48 49 54 54 57 60 60 60]
[22 44 46 50 52 53 57 60 60 60]
[22 45 48 50 53 55 57 60 60 60]
[21 46 49 51 52 55 57 60 60 60]
[22 44 47 50 51 56 57 60 60 60]
```

Vidimo da dužina tabu liste (krećući se kroz kolone matrice) nema uticaja krajnji rezultat u ovom slučaju, dok se kvalitet rješenja povećava sa povećanjem broja iteracije, što je za očekivati.

U drugom testu se matrica težina sastoji od nasumično izabranih brojeva u intervalu [1, 100], vektor vrijednosti se sastoji od nasumičnih brojeva iz istog intervala, ukupan broj predmeta je 50, a vektor kapaciteta jeste jednak `[1000 1000]`, što odgovara težinama 20 prosječnih predmeta. Test je urađen sa 50, 100, 200, 500 i 1000 iteracija, sa dužinom tabu liste od 200.

Rezultati su:

```
[1745 1851 1858 1858 1858]
```

Vidimo da rast vrijednosti funkcije cilja ubrzo stagnira.

5. Zaključak

Tabu pretraga je efikasan heuristički pristup rješavanju problema ranca, na koji je tabu pretragu jednostavno primijeniti. Korištenjem strategije diverzifikacije dodatno ublažavamo problem lokalnog optimuma za algoritme lokalne pretrage, na kojima je tabu pretraga zasnovana. Uz diverzifikaciju, koristimo i strateško osciliranje, koje nam također pomaže u slučajevima kada diverzifikacija proizvede novo rješenje koje se nalazi van dozvoljenog problemskog prostora. U poglavlju 4 smo prikazali rezultate 2 testa nad nasumično konstruisanim primjerima.

Izvori

https://www.uv.es/rmarti/paper/docs/ts2.pdf

Glover, F. and Hao, J. (2009). The case for strategic oscillation. *Annals of Operations Research*, 183(1), pp.163-173.

Korišteni softver:

Python 3.6.3
Anaconda
NumPy
Matplotlib