Univerzitet u Banjoj Luci

Prirodno-matematički fakultet

Informatika

Uvod u vještačku inteligenciju

Iterativno pohlepni algoritam u rješavanju problema višedimenzionalnog ruksaka

Profesor

Dr Marko Đukanović

Student

Teodora Milanović 22/19

Sadržaj

1. Uvod 1
2. Opis algoritma 2
   2.1 Inicijalizacija 2
   2.2 Faza destrukcije i konstrukcije 3
   2.3 Lokalna pretraga 3
   2.4 Kriterijum za prihvatanje i kriterijum za prekid 4
3. Rezultati 6
4. Zaključak 8
5. Literatura 10

1. Uvod

Problem višedimenzionalnog ruksaka (eng. Multidimensional Knapsack Problem - MKP) je dobro proučen, NP-težak, optimizacioni problem koji ima različite primjene u logistici, finansijama, telekomunikaciji i drugim poljima. Kao primjeri se mogu navesti utovar tereta, budžet kapitala, alokacija procesora u ogromnim distribuiranim računarskim sistemima, dostava namirnica u vozilima sa više odjeljaka itd.

Problem višedimenzionalnog ruksaka može se predstaviti sljedećim modelom ILP-a:

$$z = \sum_{j=1}^{n} p_j x_j \rightarrow \max \qquad (1)$$

$$s.t. \sum_{j=1}^{n} w_{ij} x_j \leq c_i \quad i \in \{1, \dots, m\} \qquad (2)$$

$$x_j \in \{0,1\} \quad j \in \{1, \dots, n\} \qquad (3)$$

gdje n predstavlja broj predmeta, m broj dimenzija, p_j profit j-tog predmeta, w_{ij} težinu j-tog predmeta u i-toj dimenziji, c_i kapacitet i-te dimenzije i x_j logičku promjenljivu koja predstavlja prisustvo j-tog predmeta u ruksaku.

Slijedi da je dat skup koji sadrži n predmeta sa profitima $p_j > 0$ i m dimenzija sa kapacitetima $c_i > 0$. Svakom predmetu p_j je pridružena težina w_{ij} u i-toj dimenziji. Vrijednost promjenljive x_j je 1 ukoliko je j-ti predmet izabran, u suprotnom je 0 (3). Funkcija cilja z predstavlja ukupan profit izabranih predmeta. Cilj problema jeste maksimizacija funkcije cilja. Drugim riječima, potrebno je izabrati podskup skupa predmeta sa maksimalnim ukupnim profitom (1) tako da sume težina izabranih predmeta zadovoljavaju kapacitete c_i (2).

S obzirom da je problem višedimenzionalnog ruksaka NP-težak bilo je brojnih doprinosa tokom proteklih godina u razvoju različitih metoda rješavanja. Kao primjer se mogu navesti:

(1) Egzaktne metode rješavanja
- Plateau i Elkihel (1985) su predstavili algoritam dinamičkog programiranja kada je broj dimenzija jedan;
- Pisinger (1995) su predstavili algoritam koji se temelji na branch-and-bound tehnici;
- Martello i Toth (1997) su predstavili algoritam koji se temelji na branch-and-cut pristupu;
- Wilbaut i Hanafi (2009) su predstavili algoritam koji se temelji na relaksaciji linearnog programiranja;

(2) Heurističke i metaheurističke metode rješavanja

- Hanafi i Fréville (1998) su predstavili genetski algoritam;
- Chu i Beasley (1998) su predstavili tabu search algoritam;
- Wilbaut i Hanafi (2009) i Gao, Li, i Chen (2014) su predstavili hibridni heuristički algoritam koji kombinuje genetski algoritam i postupak lokalne pretrage;
- Li, Xu, and Wang (2017) su predstavili iterirani algoritam lokalne pretrage;

2. Opis algoritma

Iterativno pohlepni algoritam je metaheuristički algoritam koji se koristi za rješavanje kombinatornih optimizacionih problema. Sastoji se iz dvije glavne faze:

1. Faza djelimičnog uništenja rješenja u kojoj se uklanjaju određene komponente rješenja,
2. Faza rekonstrukcije rješenja u kojoj se primjenjuju pohlepne heuristike kako bi se rekonstruisalo potpuno rješenje.

Zatim se na osnovu kriterijuma za prihvatanje utvrđuje da li dobijeno rješenje treba da bude prihvaćeno ili odbačeno. Ovaj algoritam može da uključuje i fazu lokalne pretrage čija iterativna primjena omogućava poboljšavanje kvaliteta rješenja. Prethodni koraci se ponavljaju sve dok se ne ispuni kriterijum za prekid.

2.1 Inicijalizacija

Prvi korak u rješavanju ovog problema jeste inicijalizacija iliti generisanje početnog rješenja. Početno rješenje se generiše na pohlepan način tako što se za svaki predmet računa fitnes vrijednost kao odnos između profita predmeta i suma težina predmeta u svim dimenzijama. Slijedi da je

$$f_j = \frac{p_j}{\sum_{i=1}^{m} w_{ij}} \quad j \in \{1, \dots, n\}$$

Potom se vrši sortiranje predmeta u opadajućem poretku na osnovu fitnes vrijednosti f. Dalje se bira predmet sa najvećom vrijednošću f_j i dodaje u ruksak ukoliko njegovim dodavanjem ne dolazi do prekoračenja kapaciteta c_i, $i \in \{1, \dots, m\}$. U suprotnom se odbacuje. Prethodni postupak se ponavlja za sve predmete. Dobijeno rješenje se potom poboljšava lokalnom pretragom.

2.2 Faze destrukcije i konstrukcije

Destrukcija i konstrukcija su dvije glavne procedure u iterativno pohlepnom algoritmu. Tokom faze destrukcije se vrši nasumično izbacivanje d predmeta iz ruksaka. Faktor destrukcije d je jedan od ulaznih parametara.

Faza konstrukcije uključuje računanje fitnes vrijednosti predmeta koji nisu ubačeni u ruksak na isti način kao u fazi inicijalizacije. Dakle

$$f_j = \frac{p_j}{\sum_{i=1}^{m} w_{ij}} \quad j \in \{1, \dots, n\}$$

Zatim se na osnovu fitnes vrijednosti f vrši sortiranje predmeta u opadajućem poretku. Dalje se bira predmet sa najvećom vrijednošću f_j i dodaje u ruksak ukoliko njegovim dodavanjem ne dolazi do prekoračenja kapaciteta c_i, $i \in \{1, \dots, m\}$. U suprotnom se odbacuje. Prethodni postupak se ponavlja za sve predmete.

Dobijeno rješenje se potom poboljšava lokalnom pretragom.

2.3 Lokalna pretraga

Lokalna pretraga se koristi za poboljšavanje rješenja, te funkcioniše na sljedeći način:

(1) Nasumično odaberemo jedan ubačeni predmet, te ga izbacimo iz ruksaka.

(2) Biramo predmet koji nije ubačen i dodajemo ga u ruksak ako se njegovim dodavanjem ne narušavaju ograničenja. Dalje, računamo ukupan profit koji se dobije ubacivanjem ovog predmeta umjesto prethodno izbačenog. Slijedi da je

$$ukupan\ profit = \sum_{j=1}^{n} p_j x_j$$

(3) Ovaj postupak se ponavlja za sve predmete koji nisu ubačeni.

(4) Konačno, biramo predmet čijim ubacivanjem se maksimizuje ukupan profit.

(5) U slučaju da je dobijeni profit veći od ukupnog profita prije izbacivanja predmeta u koraku (1), dato rješenje se postavlja kao novo najbolje.

Do prekida lokalne pretrage će doći u slučaju da dođe do zaglavljivanja u lokalnom optimumu.

2.4 Kriterijum za prihvatanje i kriterijum za prekid

Nakon faza destrukcije, konstrukcije i lokalne pretrage, potrebno je provjeriti da li je ukupan profit dobijenog rješenja veći od najvećeg ukupnog profita koji je do tada pronađen. U slučaju da jeste, dolazi do ažuriranja najboljeg rješenja.
Do prekida izvršavanja algoritma dolazi kada se dostigne unaprijed određen broj iteracija koji je ujedno i ulazni parametar.

```
procedure iterativnoPohlepniAlgoritam(faktorDestrukcije, brojIteracija)
        rjesenje := inicijalizacija();
        rjesenje := lokalnaPretraga(rjesenje);
        for i := 1 to brojIteracija do
                rjesenje' := rjesenje;
                rjesenje' := destrukcija(rjesenje');
                rjesenje' := konstrukcija(rjesenje');
                rjesenje' := lokalnaPretraga(rjesenje');
                if ukupanProfit(rjesenje') > ukupanProfit(rjesenje) then
                        rjesenje := rjesenje';
                endif
        endfor
        return rjesenje
end
```

Figura 2.1. Pseudokod procedure za iterativno pohlepni algoritam

```
procedure konstrukcija(rjesenje)
        fitnesi := lista dobijena generisanjem fitnes vrijednosti predmeta koji nisu ubačeni u ruksak;
        sortirati vrijednosti liste u opadajućem poretku
        veličina := broj elemenata liste;
        for i := 1 to  veličina do
                rjesenje := rješenje dobijeno ubacivanjem predmeta sa fitnesom i u slučaju da se
                njegovim dodavanjem ne prekoračuju kapaciteti;
        endfor
        return rjesenje
end
```

Figura 2.2. Pseudokod procedure za konstrukciju

```
procedure destrukcija(rjesenje)
        for i := 1 to faktorDestrukcije do
                rjesenje := rješenje dobijeno nasumičnim izbacivanjem jednog predmeta iz ruksaka;
        endfor
        return rjesenje
end
```

Figura 2.3. Pseudokod procedure za destrukciju

```
procedure lokalnaPretraga(rjesenje)
        improve := true;
        while (improve := true) do
                improve := false;
                for i := 1 to n do
                        rjesenje' := rješenje dobijeno nasumičnim izbacivanjem jednog predmeta iz
                        ruksaka;
                        rjesenje' := najbolje rješenje dobijeno ubacivanjem jednog predmeta koji se ne
                        nalazi u ruksaku uz poštovanje ograničenja kapaciteta;
                        if ukupanProfit(rjesenje') > ukupanProfit(rjesenje) then
                                rjesenje := rjesenje';
                                improve := true;
                        endif
                endfor
        endwhile
        return rjesenje
end
```

Figura 2.4. Pseudokod procedure za lokalnu pretragu

3. Rezultati

Svi prikazani rezultati su dobijeni korišćenjem računara sa Intel Core i3 procesorom, 8GB RAM-a i Windows 10 operativnim sistemom. Algoritmi su pisani u programskom jeziku Java i kompajlirani u Eclipse razvojnom okruženju.

Algoritam je testiran na 15 instanci koje se mogu pronaći na sljedećem linku: http://people.brunel.ac.uk/~mastjjb/jeb/orlib/files/. Prvih pet instanci sadrži 100 predmeta i 5 dimenzija, narednih pet instanci sadrži 100 predmeta i 10 dimenzija i posljednjih pet instanci sadrži 500 predmeta i 5 dimenzija.

Rezultati iterativno pohlepnog algoritma su upoređeni sa nekim od najboljih rezultata iz literature. Prvih deset instanci je upoređeno sa rezultatima algoritma koji se temelji na algoritmu optimizacije kolonijom mrava – DMMAS, Liangjun Ke, Zuren Feng, Zhigang Ren i Xiaoliang Wei (2008). Posljednjih pet instanci je upoređeno sa rezultatima hibridnog DMMAS algoritma označenog sa DMMAS+ls koji uključuje fazu lokalne pretrage - Liangjun Ke, Zuren Feng, Zhigang Ren i Xiaoliang Wei (2008), te sa rezultatima KNN quantum cuckoo search algoritma - José García, Carlos Maureira (2021). U svrhe poređenja su takođe prikazani optimalni rezultati. Za pronalaženje optimalnog rješenja je korišten optimizacioni rješavač CPLEX. Implementacija modela u CPLEX-u uključuje:

a) Definisanje varijabli odluke (treća stavka modela ILP-a),
b) Definisanje ograničenja (druga stavka modela ILP-a),
c) Definisanje i maksimizacija funkcije cilja (prva stavka modela ILP-a).

Kao što je spomenuto, ovaj algoritam ima dva ulazna parametra – broj iteracija i faktor destrukcije. Faktor destrukcije je postavljen na 4 za sve instance, te je broj iteracija algoritma postavljen na 10000. Svaka instanca se izvršava 10 puta i računa se aritmetička sredina svih dobijenih rezultata.

		Iterativno pohlepni algoritam		DMMAS	
Instanca	CPLEX	Najbolji rezultat	Srednji Rezultat	Najbolji rezultat	Srednji Rezultat
5.100.00	24381	**24381**	24333	**24381**	24362
5.100.01	24274	**24274**	24229	**24274**	24273
5.100.02	23551	**23551**	23530	**23551**	23540
5.100.03	23534	**23534**	23488	**23534**	23482
5.100.04	23991	**23991**	23951	**23991**	23954
10.100.00	23064	23055	23051	**23064**	23045
10.100.01	22801	22684	22592	**22801**	22742
10.100.02	22131	21993	21931	**22131**	22091
10.100.03	22772	22693	22648	**22772**	22710
10.100.04	22751	22627	22620	**22751**	22617

Tabela 3.1. Rezultati dobijeni izvršavanjem instanci sa 100 predmeta, te 5 i 10 dimenzija

		Iterativno pohlepni algoritam		DMMAS+ls		KQCSA	
Instanca	CPLEX	Najbolji rezultat	Srednji Rezultat	Najbolji rezultat	Srednji Rezultat	Najbolji rezultat	Srednji Rezultat
5.500.00	120148	120013	119945	**120148**	120111	**120148**	120131.1
5.500.01	117879	117670	117483	**117879**	117841	117864	117853.8
5.500.02	121131	120954	120811	**121131**	121097	**121131**	121126.9
5.500.03	120804	120587	120424	**120804**	120776	**120804**	120794.4
5.500.04	122319	122089	121957	**122319**	122303	**122319**	122317.1

Tabela 3.2. Rezultati dobijeni izvršavanjem instanci sa 500 predmeta i 5 dimenzija

Primjećujemo da na malim instancama iterativno pohlepni algoritam pronalazi optimalna rješenja u 100% instanci. Na srednjim i velikim instancama ovaj algoritam daje dobre rezultate uzimajući u obzir brzinu izvršavanja i jednostavnost. Takođe, nije došlo do velikog pada performansi na velikim instancama.

4. Zaključak

Prethodno je predstavljen iterativno pohlepni algoritam sa fazom lokalne pretrage za rješavanje problema višedimenzionalnog ruksaka. Ovaj algoritam se sastoji od faze destrukcije u kojoj se nasumično izbacuje određeni broj predmeta iz ruksaka, te faze konstrukcije u kojoj se predmeti ubacuju u ruksak s ciljem maksimizacije ukupnog profita. Nakon uspoređivanja iterativno pohlepnog algoritma sa nekim od najboljih rezultata iz literature i optimizacionim rješavačem primijetili smo da je ovaj algoritam efikasan u rješavanju problema višedimenzionalnog ruksaka, pogotovo uzimajući u obzir njegovu jednostavnost. Ovaj algoritam pronalazi optimalna rješenja na malim instancama, te dobre rezultate za veće instance u kratkom vremenskom periodu pa se može koristiti kao prihvatljiva aproksimacija u situacijama gdje je brzina izvršavanja bitna.

5. Literatura

[1] Jakob Puchinger. The Multidimensional Knapsack Problem Structure and Algorithms.

[2] Said Hanafi, Jasmina Lazić, Nenad Mladenović, Christophe Wilbaut. Variable Neighbourhood Decomposition Search with Bounding for Multidimensional Knapsack Problem.

[3] Talbi (2009). Metaheuristics: From Design to Implementation.

[4] Ruben Ruiz, Thomas Stutzle. A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem.

[5] José García, Carlos Maureira (2021). A KNN quantum cuckoo search algorithm applied to the multidimensional knapsack problem.

[6] Liangjun Ke, Zuren Feng, Zhigang Ren i Xiaoliang Wei (2008). An ant colony optimization approach for the multidimensional knapsack problem.

[7] Freville A. i Hanafi S. (2005). The multidimensional 0–1 knapsack problem: An overview.

[8] Farhad Djannaty, Saber Doostdar (2008). A Hybrid Genetic Algorithm for the Multidimensional Knapsack Problem.