Poređenje algoritama kombinatorne optimizacije

Aleksandar D. Kupusinac

***Sadržaj* — Rešavanje problema kombinatorne optimizacije ima veliku primenu u inženjerstvu. Zajedničko za ove probleme jeste nalaženje ekstremnih vrednosti funkcije definisane na konačnom skupu. U ovom radu se bavimo rešavanjem tzv. NP-teških problema kombinatorne optimizacije, konkretnije uradićemo postavku i analiziraćemo rešenja jednodimenzionalnog *bin-packing* problema. Rešavajući ovaj problem uporedićemo algoritme detaljne i stohastičke pretrage i permutacionog genetskog algoritma, što je i glavni cilj ovog rada.**

***Ključne reči* — jednodimenzionalni *bin-packing* problem, kombinatorna optimizacija, teorija algoritama.**

I. Uvod

KOMBINATORNA optimizacija se bavi nalaženjem ekstremnih vrednosti funkcije definisane na konačnom skupu [1], [2]. NP-teški problemi [3] koje razmatra kombinatorna optimizacija se mogu podeliti u tri grupe:

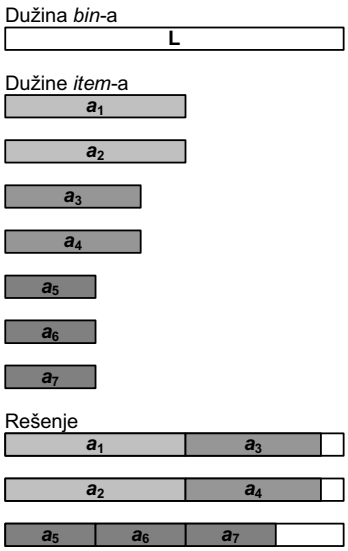
- Problemi trasiranja (*routing*), gde spadaju problem trgovačkog putnika (*travelling salesman problem*) [4], problem prevoza robe (*transportation problem*), planiranje putanje robota i sl.
- Problemi raspoređivanja (*scheduling*), koji se bave raspoređivanjem poslova (*job scheduling*), raspoređivanjem časova, raspoređivanjem poslova kod multiprocesiranja i sl.
- Problemi pakovanja (*packing*), gde spadaju jednodimenzionalni, dvodimenzionalni i trodimenzionalni *bin-packing* problemi [5].

U ovom radu fokusiraćemo se na rešavanje jednodimenzionalnog *bin-packing* problema. Razmotrićemo rešenje ovog problema pomoću algoritama detaljne i stohastičke pretrage i permutacionog genetskog algoritma [6]. Postupak rešavanja jednodimenzionalnog *bin-packing* problema se svodi na nalaženje optimalne permutacije. Detaljna pretraga ispituje redom sve moguće permutacije, stohastička pretraga na slučajan način izvlači permutacije, a permutacioni genetski algoritam je evolucijom vođen proces, pa se od njega očekuje da će brže konvergirati. Cilj ovoga rada jeste da prikaže sva tri algoritamska rešenja i napravi njihovo poređenje.

Aleksandar D. Kupusinac (autor za kontakte), Fakultet tehničkih nauka, Trg Dositeja Obradovića 6, 21000 Novi Sad, Srbija (telefon: 381-21-485-2441, e-mail: sasak@uns.ac.rs).

II. Jednodimenzionalni *bin-packing* problem

Jednodimenzionalni *bih-packing* problem ima veoma široku primenu u inženjerstvu (punjenje teretnih vagona, sečenje cevi, rezanje drvene građe, sečenje kablova, rezanje papira, raspoređivanje televizijskih reklama u pauzama između emisija, pravljenje finansijskog plana firme itd.). Ovaj problem spada u klasu poznatih NP-teških problema kombinatorne optimizacije. Na Sl. 1. je prikazan jednodimenzionalni *bin-packing* problem. Data je fiksna dužina *bin*-a L i dužine *item*-a $(a_1,a_2,...,a_n)$. Potrebno je rasporediti dužine *item*-a u sekvencu *bin*-ova $(B_1,B_2,...,B_m)$, tako da broj upotrebljenih *bin*-ova m bude minimalan.



Sl. 1. Jednodimenzionalni *bin-packing* problem.

Posmatrajmo ponovo datu n-torku *item*-a $(a_1, a_2, \ldots, a_n)$. Neka je A skup svih permutacija date n-torke *item*-a. Sada možemo definisati funkciju λ koja za svaku permutaciju iz skupa A određuje broj potrebnih *bin*-a m, tj. $\lambda: A \rightarrow N$, gde je N skup prirodnih brojeva. Očigledno, rešenje jednodimenionalnog *bin-packing* problema je permutacija $p_{min} \in A$ za koju važi $\lambda(p_{min}) \leq \lambda(p_i)$, $p_i \in A \wedge p_i \neq p_{\min}$. Permutaciju p_{min} ćemo zvati *optimalna permutacija*. Primetimo da ona ne mora biti jedinstvena, tj. u opštem slučaju može postojati više optimalnih permutacija. Rešenje primera na Sl. 1 jeste permutacija $(a_1,a_3,a_2,a_4,a_5,a_6,a_7)$, ali isto tako rešenja jesu i permutacije $(a_3,a_1,a_2,a_4,a_5,a_6,a_7)$, $(a_3,a_1,a_4,a_2,a_5,a_6,a_7)$ itd.

III. ALGORITAM DETALJNE PRETRAGE

Detaljna pretraga podrazumeva ispitivanje ponaosob svake permutacije iz skupa A, izračunavanje funkcije λ i ispitivanje da li je data permutacija optimalna, tj. da li se za nju dobija minimalna vrednost funkcije λ. Detaljnom pretragom sigurno se dolazi do optimalne permutacije. Međutim, s obzirom da je broj različitih permutacija $n!$, a sa druge strane da su resursi računara ograničeni, detaljna pretraga gubi smisao za slučajeve $n>10$. Npr. za $n=20$ broj mogućih permutacija iznosi $20!\approx 2.4\cdot 10^{17}$, pa ako pretpostavimo da je za ispitivanje samo jedne permutacije računaru potrebna 1 milisekunda, tada će za celu detaljnu pretragu računaru biti potrebno oko 77 miliona godina.

IV. ALGORITAM STOHASTIČKE PRETRAGE

Stohastička pretraga podrazumeva slučajan izbor permutacije iz skupa A, izračunavanje funkcije λ i ispitivanje da li je data permutacija optimalna, tj. da li se za nju dobija minimalna vrednost funkcije λ. Kriterijum zaustavljanja stohastičke pretrage je broj pokušaja slučajnog izbora permutacija, kojeg zadaje korisnik. Dakle, stohastičkom pretragom se do optimalne permutacije dolazi sa određenom verovatnoćom.

Neka je n broj *item*-a, tada je $n!$ broj različitih permutacija. Neka je poznato da postoji samo jedna optimalna permutacija. Ukoliko bi se samo jednom birala permutacija, verovatnoća da će biti izvučena baš optimalna permutacija je $p=1/(n!)$. To je verovatnoća povoljnog događaja, dok verovatnoća nepovoljnog događaja, tj. izvlačenja permutacije koja nije optimalna iznosi $q=1-p=1-1/(n!)$. Ukoliko se izvlačenje ponavlja više puta, tj. u više iteracija, verovatnoća da je bar jednom izvučena optimalna permutacija računa se po sledećoj formuli:

$$P(N)=1-q^N=1-\left(1-\frac{1}{n!}\right)^N$$

gde je N broj izvlačenja. Tabela 1 prikazuje verovatnoće da je bar jednom izvučena optimalna permutacija $P(N)$ za različite vrednosti broja izvlačenja N, u slučaju kada je broj *item*-a $n=20$.

TABELA 1: VEROVATNOĆE DA JE BAR JEDNOM IZVUČENA OPTIMALNA PERMUTACIJA.

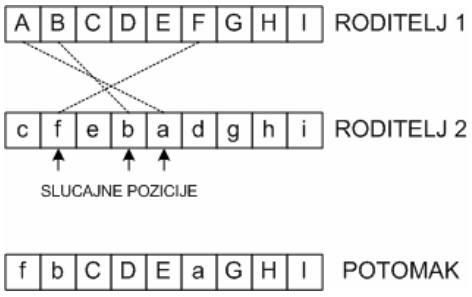
N	***P(N)***
1	$4.1103\cdot 10^{-19}$
10^6	$4.1103\cdot 10^{-13}$
10^{12}	$4.1103\cdot 10^{-7}$
10^{18}	0.3370
10^{19}	0.9836
10^{20}	0.9999

Optimistički deluje da korisnik može da izabere broj izvlačenja N i na taj način utiče na trajanje stohastičke pretrage. Za dovoljno velik broj izvlačenja N verovatnoća da će stohastička pretraga pronaći optimalnu permutaciju će biti blizu 1. Ipak iz Tabele 1 vidimo da ako želimo da verovatnoća $P(N)$ bude blizu 1 potrebno je 10^{20} izvlačenja i to već za slučaj kada je broj *item*-a samo $n=20$, čime stohastička pretraga gubi praktičan značaj.

V. PERMUTACIONI GENETSKI ALGORITAM

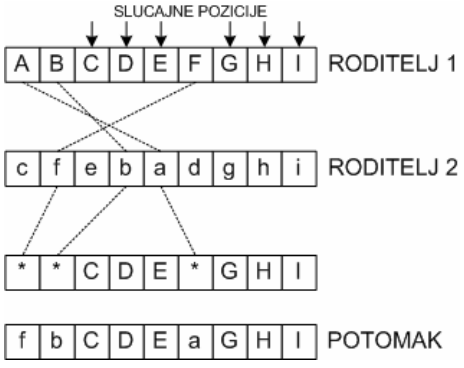
Genetski algoritam [7], [8], [9] koji koristi reprezentaciju problema preko permutacije naziva se permutacioni genetski algoritam (kraće PGA). PGA poseduje karakteristične genetske operatore i njegov krajnji cilj jeste nalaženje optimalne permutacije za dati kombinatorni problem. Očigledan problem kod ukrštanja dve permutacije je da klasično ukrštanje u opštem slučaju ne može da generiše validnu permutaciju koja bi predstavljala potomka, tj. jedinku nove generacije. Zbog toga je potrebno definisati nove operatore ukrštanja za PGA, a to su: uređeno i pozicione ukrštanje.

Kod uređenog ukrštanja na slučajan način se izabere K pozicija u drugom roditelju i uoči se redosled izabranih elemenata u permutaciji. Zatim, odgovarajući elementi koji se nalaze u prvom roditelju se poređaju redosledom po kojem stoje u drugom roditelju i na taj način se dobija potomak. Na Sl. 2 prikazano je uređeno ukrštanje. U drugom roditelju na slučajan način odabrane su tri pozicije i uočeno da se elementi na datim pozicijama pojavljuju redom F, B, A, pa će odgovarajući elementi u prvom roditelju biti poređani u datom redosledu i tako će nastati potomak.



Sl. 2. Uređeno ukrštanje.

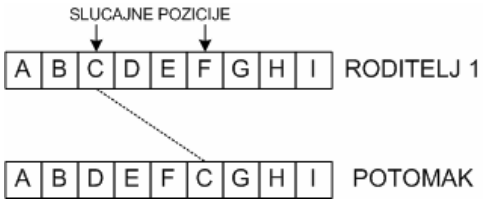
Kod pozicionog ukrštanja na slučajan način se izabere K pozicija u prvom roditelju. Elementi na da tim pozicijama se prepišu na odgovarajuće pozicije u potomku. Zatim se posmatra drugi roditelj s leva u desno i svaki element, koji se ne pojavljuje u potomku, ubacuje se po redosledu kakav je u drugom roditelju. Na Sl. 3 prikazano je pozicione ukrštanje.



Sl. 3. Poziciono ukrštanje.

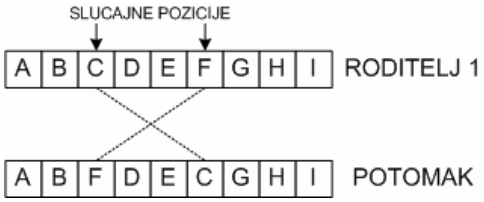
Takođe, klasičan operator mutacije se ne može primeniti na permutaciju, već je potrebno definisati posebne opera-

tore mutacije za PGA, a to su: ubacivanje i razmena. Kod ubacivanja na slučajan način se biraju dve pozicije, a zatim se element sa prve pozicije ubacuje iza elementa na drugoj poziciji i na taj način se dobija potomak. Na Sl. 4. je prikazano ubacivanje.



Sl. 4. Ubacivanje.

Kod razmene na slučajan način se biraju dve pozicije, a zatim se razmene elementi koji se nalaze na datim pozicijama i tako se dobija potomak. Na slici 5. je prikazana razmena.



Sl. 5. Razmena.

VI. EKSPERIMENT I REZULTATI

U programskom paketu MATLAB 7.0 za potrebe izvođenja eksperimenata realizovana su rešenja pomoću kojih se mogu posmatrati uporedne karakteristike detaljne pretrage, stohastičke pretrage i PGA prilikom rešavanja jednodimenzionalnog *bin-packing* problema. Prvo se unosi fiksna dužina *bin*-ova, a zatim se unose količine i dužine *item*-a. Kada su unešeni svi neophodni podaci formira se polazna permutacija čiji su elementi dužine *item*-a. U programskom paketu MATLAB 7.0 napisana je funkcija koja izračunava vrednost funkcije λ za datu permutaciju. Svaka dužina *item*-a pojavljuje se onoliko puta u permutaciji koliko je korisnik uneo vrednost za količinu. Kada je polazna permutacija formirana prelazi se na detaljnu pretragu, stohastičku pretragu i PGA sa ciljem da se odredi optimalna permutacija.

Detaljna pretraga podrazumeva ispitivanje svake moguće permutacije ponaosob i određivanje optimalne permutacije za koju se dobija minimalna vrednost funkcije λ. Može se zaključiti da se detaljnom pretragom sigurno dolazi do tražene optimalne permutacije, međutim, s obzirom da broj različitih permutacija iznosi $n!$, za veće vrednosti broja n (npr. $n>10$) detaljna pretraga gubi smisao.

Stohastička pretraga podrazumeva zadavanje permutacija na slučajan način i određivanje najbolje ponuđene permutacije (što u opštem slučaju ne mora biti optimalna permutacija) za koju se dobija minimalna vrednost funkcije λ. U svakoj iteraciji na slučajan način zadaje se jedna permutacija, a zatim se vrši njena analiza. Kriterijum zaustavljanja stohastičke pretrage jeste broj iteracija koje korisnik zadaje. Algoritam pamti uvek najbolju permutaciju koju je pronašao, dok ostale ignoriše. Stohastičkom pretragom se do tražene permutacije dolazi sa određenom verovatnoćom. Za dovoljno veliki broj iteracija verovatnoća da će stohastička pretraga pronaći optimalnu permutaciju je blizu 1.

Permutacioni genetski algoritam je, za razliku od stohastičke pretrage, evolucijom vođen proces, pa se zbog toga od njega očekuje da će brže konvergirati ka traženom rešenju. Broj jedinki u generaciji, kao i broj generacija određuje korisnik. Početna generacija se postavlja na slučajan način. U svakoj generaciji jedinke (tj. permutacije) se sortiraju po količini otpadnog materijala i biraju se one najbolje, koje prelaze u narednu generaciju. Nove jedinke se dobijaju primenom genetskih operatora na postojeće jedinke, a takođe u svaku generaciju ubacuju se i nove jedinke koje se dobijaju na slučajan način. Na primerima se mogu analizirati uporedne karakteristike opisanih algoritama.

A. Primer

Neka je dužina *bin*-ova 6000 i neka je data sekvenca *item*-a (4200, 4200, 3500, 3500, 1800, 1800, 2500, 2500), tada se dobija rešenje koje je dato na Sl. 6.

```
*** DETALJNA PRETRAGA ***
*********************************************
Resenje problema:
Resenje =
        3500        2500           0
        4200        1800           0
        3500        2500           0
        4200        1800           0
*********************************************
Kolicina otpada:
Otpad = 0

*** STOHASTICKA PRETRAGA ***
Broj iteracija slucajne pretrage: 1000
*********************************************
Resenje problema:
Resenje =
        4200        1800           0
        3500        2500           0
        3500        2500           0
        4200        1800           0
*********************************************
Kolicina otpada:
Otpad = 0

*** PERMUTACIONI GENETSKI ALGORITAM ***
Broj jedinki u populaciji [>=20]: 25
Broj generacija: 40
*********************************************
Resenje problema:
Resenje =
        3500        2500           0
        4200        1800           0
        4200        1800           0
        3500        2500           0
*********************************************
Kolicina otpada:
Otpad = 0
```

Sl. 6. Rešenje.

Stohastička pretraga i permutacioni genetski algoritam su takođe pronašli optimalnu permutaciju, pa je u sva tri slučaja otpad materijala jednak 0. Ovakvo rešenje se moglo očekivati, s obzirom da je mali broj *item*-a, tj. $n=8$.

B. Primer

Neka je dužina *bin*-ova 6000 i neka je data sekvenca *item*-a (4200, 4200, 4200, 4200, 3500, 3500, 3500, 3500, 1800, 1800, 1800, 1800, 2500, 2500, 2500, 2500), tada se dobija rešenje koje je dato na Sl. 7.

```
*** DETALJNA PRETRAGA ***
Broj item-a je suvise velik za detaljnu pretragu!

*** STOHASTICKA PRETRAGA ***
Broj iteracija slucajne pretrage: 1000
********************************************
Resenje problema:
Resenje =
      4200        1800           0
      3500        1800           0
      2500        1800           0
      4200           0           0
      4200           0           0
      3500        2500           0
      4200        1800           0
      3500        2500           0
      3500        2500           0
********************************************
Kolicina otpada:
Otpad = 6000

*** PERMUTACIONI GENETSKI ALGORITAM ***
Broj jedinki u populaciji [>=20]: 25
Broj generacija: 40
********************************************
Resenje problema:
Resenje =
        3500        2500           0
        4200        1800           0
        3500        2500           0
        4200        1800           0
        3500        2500           0
        4200        1800           0
        4200        1800           0
        3500        2500           0
********************************************
Kolicina otpada:
Otpad = 0
```

Sl. 7. Rešenje.

U ovom primeru broj *item*-a je veći od prethodnog, tj. n=16, pa zbog toga detaljna pretraga nema smisla, već ostaju samo stohastička pretraga i permutacioni genetski algoritam. Permutacioni genetski algoritam je pronašao optimalnu permutaciju za koju vrednost funkcije λ iznosi 0, dok stohastička pretraga je pronašao optimalnu permutaciju za koju vrednost funkcije λ iznosi 6000, što je posledica činjenice da su genetski algoritmi evolucijom vođeni procesi, pa brže konvergiraju ka optimalnom rešenju.

VII. ZAKLJUČAK

Kroz rešavanje jednodimenzionalnog *bin-packing* problema razmatrali smo i uporedili karakteristike algoritama detaljne i stohastičke pretrage i permutacionog genetskog algoritma. Permutacioni genetski algoritam daje najbolje rezultate, što se i moglo očekivati s obzirom da se radi o evolucijom vođenom procesu. Dalja istraživanja će se kretati u pravcu kombinovanja raznih algoritama u cilju što efikasnijeg rešavanja problema kombinatorne optimizacije.

LITERATURA

[1] D. Cvetković i S. Simić, *Diskretna matematika, Matematika za kompjuterske nauke*. Naučna knjiga, Beograd, 1990.

[2] D. Cvetković, M. Čangalović, Đ. Dugošija, V. Kovačević-Vučić, S. Simić, J. Vuleta, red. D. Cvetković i V. Kovačević-Vujčić, *Kombinatorna optimizacija, Matematička teorija i algoritmi*. Društvo operacionih istraživača Jugoslavije, Beograd, 1996.

[3] D. Hochbaum, *Approximation Algorithms for NP-hard Problems*, PWS Publishing, Boston, 1996.

[4] D. Cvetković, V. Dimitrijević and M. Milosavljević, *Variations on the travelling salesman theme*. Libra produkt, Beograd, 1996.

[5] A. Kupusinac, *Rešenje jednodimenzionalnog bin-packing problema primenom genetskog algoritma*, diplomski rad, Fakultet tehničkih nauka, Novi Sad, 2005.

[6] A. Kupusinac, "Rešavanje jednodimenzionalnog *bin-packing* problema primenom permutacionog genetskog algoritma", *7. DOGS, Digitalna obrada govora i slike*, Kelebija, 2-3 Oktobar, 2008, str. 225-228, ISBN 978-86-7892-136-0.

[7] T. Bäck, D. B. Fogel and Z. Michalewich, *Evolutionary Computation 1, Basic Algorithms and Operators*. Institute of Physics Publishing, Bristol and Philadelphia, 2000.

[8] T. Bäck, D. B. Fogel and Z. Michalewich, *Evolutionary Computation 2, Advanced Algorithms and Operators*, Institute of Physics Publishing, Bristol and Philadelphia, 2000.

[9] D. Whitley, *A Genetic Algorithm Tutorial*, Computer Science Department, Colorado State University, 1993.

ABSTRACT

Combinatorial optimization problems and their solutions are very important for engineering. Essence of these solutions is determination of extreme values of function over a finite set. In this paper we consider NP-hard problems of combinatorial optimization, actualy, we will define and consider one-dimensional bin-packing problem. We will describe solutions of this problem by brute force and stochastic searches and by permutation genetic algorithm. The goal of this research is to compare these algorithms.

A COMPARISON OF COMBINATORIAL OPTIMIZATION ALGORITHMS

Aleksandar D. Kupusinac