Veštački roj pčela

Nikola Milosević 197/2018

Mila Jakovljević 117/2018

Sadržaj

Predgovor

Algoritam veštačkog roja pčela je vrsta optimizacionog algoritma koja se zasniva na inteligentnom ponašanju grupa. Ovaj rad uvodi algoritam veštačkog roja pčela za rešavanje problema rutiranja vozila sa kapacitetom. Porede se originalna verzija i verzija poboljšana elitizmom. Takođe poredi se kvalitet rešenja u odnsu na izabrani operator koji trazi neko novo rešenje u okolini već postojećeg.

1. Uvod

Algoritmi optimizacije zasnovani na populaciji nalaze približno optimalna rešenja za teške optimizacione probleme motivisani prirodom. Zajednička karakteristika svih ovih algoritama je da se populacija, koja se sastoji od mogućih rešenja problema, modifikuje primenom određenih operatora nad rešenjima, zavisno od njihovog fitnesa. Na taj način, populacija se kreće ka boljim područjima rešenja u prostoru pretrage. Dve važne klase algoritama optimizacije zasnovanih na populaciji su evolutivni algoritmi i algoritmi zasnovani na inteligenciji roja.

Bonabeau je definisao inteligenciju roja kao "svaki pokušaj dizajniranja algoritama ili uređaja za rešavanje problema inspirisan ponašanjem kolektiva socijalnih insekata i drugih društava životinja". On i saradnici su svoj fokus stavili na socijalne insekte poput termita, pčela, osa i različitih vrsta mrava. Međutim, roj se može smatrati bilo kojom kolekcijom međusobno delujućih agenata ili jedinki. Kolonija mrava može se posmatrati kao roj čije su jedinke mravi; jato ptica je roj ptica. Imunološki sistem može se smatrati rojem ćelija i molekula, baš kao što je i gužva roj ljudi. Popularni algoritam zasnovan na inteligenciji roja je Algoritam Optimizacije Rojem Čestica (PSO) koji su 1995. godine predstavili Eberhart i Kennedy. PSO je takođe populaciona stohastička tehnika optimizacije i dobro se prilagođava optimizaciji nelinearnih funkcija u višedimenzionalnom prostoru. Modeluje socijalno ponašanje jata ptica ili jata riba. PSO je privukao značajnu pažnju istraživača iz različitih oblasti i uspešno je primenjen na nekoliko stvarnih problema.

Klasičan primer roja su pčele, ali se koncept roja može proširiti i na druge sisteme sa sličnom arhitekturom. Neki pristupi su predloženi za modeliranje specifičnih inteligentnih ponašanja rojeva pčela i primenjeni su za rešavanje kombinatornih problema. Tereshko je razmatrao koloniju pčela kao dinamički sistem koji prikuplja informacije iz okoline i prilagođava svoje ponašanje u skladu s tim. Tereshko i Loengarov su razvili minimalni model za izbor hrane koji vodi ka pojavi kolektivne inteligencije, a sastoji se od tri osnovna elementa: izvori hrane, zaposleni skupljači hrane i nezaposleni skupljači hrane, definišući dva vodeća oblika ponašanja: regrutacija ka izvoru nektara i napuštanje izvora.

Izvori hrane: Da bi izabrala izvor hrane, prikupljačka pčela procenjuje nekoliko svojstava vezanih za izvor hrane, kao što su blizina košnice, bogatstvo energije, ukus nektara i lakoća ili težina ekstrakcije ove energije. Radi jednostavnosti, kvalitet izvora hrane može se predstaviti samo jednom vrednošću, iako zavisi od različitih parametara.

Zaposleni skupljači: Zaposleni skupljač radi na određenom izvoru hrane koji trenutno eksploatiše. Ona nosi informacije o tom specifičnom izvoru i deli ih sa drugim pčelama koje čekaju u košnici. Informacije uključuju udaljenost, smer i profitabilnost izvora hrane.

Nezaposleni skupljači: Pčela koja traži izvor hrane za eksploatisanje naziva se nezaposlenom. Može biti izviđač koji traži okolinu nasumično ili posmatrač koji pokušava pronaći izvor hrane putem informacija koje je dobila od zaposlene pčele. Prosečan broj izviđača je oko 5-10%.

Razmena informacija među pčelama najvažniji je događaj u formiranju kolektivnog znanja. Proučavajući celu košnicu, moguće je razlikovati neke delove koji zajednički postoje u svim košnicama. Komunikacija između pčela u vezi sa kvalitetom izvora hrane dešava se na plesnom podijumu. Pošto su informacije o svim aktuelnim bogatim izvorima dostupne posmatraču na plesnom podijumu, verovatno bi mogla da gleda brojne plesove i odluči da zaposli se na najprofitabilnijem izvoru. Veća je verovatnoća da će posmatrači birati profitabilnije izvore jer kruži više informacija o profitabilnijim izvorima. Zaposleni prikupljači dele svoje informacije sa verovatnoćom koja je proporcionalna profitabilnosti izvora hrane, a deljenje te informacije putem plesa traje duže. Stoga je regrutacija proporcionalna profitabilnosti izvora hrane.

2. Algoritam veštačkog roja pčela

Algoritam veštačkog roja pčela pripada klasi evolutivnih algoritama inspirisanih inteligentnim ponašanjem medonosnih pčela u potrazi za izvorima nektara oko svojih košnica. U svim varijantama pčelinjih algoritama se primenjuju neke zajedničke strategije pretrage; to jest, kompletna ili parcijalna rešenja se smatraju izvorima hrane, a grupe pčela pokušavaju iskoristiti izvore hrane u nadi da će

pronaći nektar visokog kvaliteta za košnicu. Osim toga, pčele komuniciraju međusobno o prostoru pretrage i izvorima hrane izvodeći ples. U ABC algoritmu, pčele se dele u tri vrste: zaposlene pčele, posmatrače i izviđače. Zaposlene pčele odgovorne su za iskorišćavanje dostupnih izvora hrane i prikupljanje potrebnih informacija. Takođe dele informacije s posmatračima, a posmatrači biraju postojeće izvore hrane koje treba dalje istražiti. Kada zaposlena pčela napusti izvor hrane, postaje izviđač i počinje tražiti novi izvor hrane u blizini košnice. Napuštanje se dešava kada kvalitet izvora hrane nije poboljšan nakon izvođenja maksimalnog dozvoljenog broja iteracija. Glavni koraci algoritma su sledeći:

1.Inicijalizacija populacije
2.Ponaljaj dok nisu ispunjeni uslovi zaustavljanja:
2.1. Rasporediti zaposlene pčele na njihove izvore hrane
2.2. Rasporediti pčele posmatrače na izvore hrane u zavisnosti od količine nektara
2.3. Poslati pčele izviđače u prostor pretrage za pronalaženje novih izvora hrane
2.4. Zapamtite najbolji izvor hrane do sada

U ABC algoritmu, pozicija izvora hrane predstavlja moguće rešenje optimizacionog problema, a količina nektara izvora hrane odgovara kvalitetu odnosno fitnesu odgovarajućeg rešenja. ABC algoritam je iterativni algoritam i počinje generisanjem nasumičnih rešenja kao izvora hrane i dodelom svake zaposlene pčele izvoru hrane. Zatim, tokom svake iteracije, svaka zaposlena pčela pronalazi novi izvor hrane u blizini svog prvobitno dodeljenog ili starog izvora hrane korišćenjem operatora susedstva. Zatim se procenjuje količina nektara novog izvora hrane. Ako novi izvor hrane ima više nektara od starog, onda se stari zamenjuje novim. Nakon što sve zaposlene pčele završe sa gore navedenim procesom eksploatacije, dele informacije o nektaru izvora hrane s posmatračima. Zatim, svaki posmatrač bira izvor hrane ruletskom selekcijom. Nakon toga, svaki posmatrač pronalazi izvor hrane u blizini svog izabranog izvora hrane i izračunava količinu nektara susednog izvora hrane. Verovatnoća izbora izvora hrane x_i se tada definiše kao $p(x_i)= \frac{f(xi)}{\sum_{j=0}^{\tau} f(xj)}$.

Zatim, za svaki stari izvor hrane, određuje se najbolji izvor hrane među svim izvorima hrane u blizini starog izvora hrane. Zaposlena pčela povezana sa starim izvorom hrane dodeljuje se najboljem izvoru hrane i napušta stari ako je najbolji izvor hrane bolji od starog izvora hrane. Izvor hrane takođe napušt zaposlena pčela ako se kvalitet izvora hrane nije poboljšao tokom određenog broja uzastopnih iteracija unapred određenog broja.

Zaposlena pčela tada postaje izviđač i traži novi izvor hrane nasumično. Kada izviđač pronađe novi izvor hrane, postaje ponovo zaposlena pčela. Nakon što je svaka zaposlena pčela dodeljena izvoru hrane, počinje nova iteracija ABC algoritma. Ceo proces se ponavlja dok se ne ispuni uslov zaustavljanja.

Koraci ABC algoritma mogu se sažeti na sledećih 5:

1. *Nasumično generiši skup rešenja kao početne izvore hrane x_i, i = 1,...,s. Dodeli svakoj zaposlenoj pčeli jedan izvor hrane.*
2. *Oceni fitnes f(xi) svakog od izvora hrane x_i i = 1,...,s.*
3. *Postavi v = 0 i $l_1 = l_2 = ... = l_s = 0$*
4. *Ponavljaj*
   - *4.1. Za svaki izvor hrane x_i*
     - *4.1.1. Primjeni operator susedstva na $x_i \rightarrow \tilde{x}$.*
     - *4.1.2. Ako f(x) > f(xi), zameni x_i sa $\tilde{x}$ i postavi $l_i = 0$, inače $l_i = l_i + 1$.*
   - *4.2. Postavi $G_i = \emptyset$, i = 1,...,s, gde je G_i je skup susednih rešenja izvora hrane i.*
   - *4.3. Za svakog posmatrača*
     - *4.3.1. Izaberi izvor hrane x_i koristeći metod izbora na osnovu fitnesa pomoću ruleta.*
     - *4.3.2. Primjeni operator susedstva na $x_i \rightarrow \tilde{x}$.*
     - *4.3.3. $G_i = G_i \cup \{\tilde{x}\}$.*
   - *4.4. Za svaki izvor hrane x_i i $G_i \neq \emptyset$*
     - *4.4.1. Postavi $\hat{x} \in arg\ max_{\sigma \in Gi}\ f(\sigma)$*
     - *4.4.2. Ako $f(x_i) < f(\hat{x})$, zameni x_i sa $\hat{x}$ i postavi $l_i = 0$, inače $l_i = l_i + 1$*
   - *4.5. Za svaki izvor hrane x_i*
     - *4.5.1. Ako je l_i= limit, zameni x_i nasumično generisanim rešenjem.*
   - *4.6. v = v + 1.*
5. *Dokle god (v = MaxIteracija).*

3. Problem rutiranja vozila sa kapacitetom

Teodorović je predložio korišćenje inteligencije roja pčela u razvoju veštačkih sistema usmerenih na rešavanje složenih problema u saobraćaju i transportu.

Problem rutiranja vozila sa kapacitetom (CVRP) definisan je na potpuno neusmerenom grafu G = (V, E), gde je V = {0,1,...,n} skup čvorova, a E = {(i,j): i,j $\in$ V, i < j} skup grana. Čvorovi 1,...,n predstavljaju klijente; svaki klijent i je povezan sa nenegativnom potrebom d_i i nenegativnim vremenom usluge s_i. Čvor 0 predstavlja skladište u kojem se nalazi m homogenih vozila kapaciteta Q. Svaka grana (i,j) je povezana sa nenegativnim troškom putovanja ili vremenom putovanja c_{ij}. CVRP ima za cilj određivanje m ruta vozila tako da:

(a) svaka ruta počinje i završava se u skladištu,

(b) svaki klijent se posećuje tačno jednom,

(c) ukupna potreba bilo koje rute vozila ne premašuje Q, i

(d) ukupni trošak svih ruta vozila je minimizovan.

U nekim slučajevima, CVRP takođe postavlja ograničenja trajanja, gde trajanje bilo koje rute vozila ne sme preći zadatu granicu L. S obzirom na to da je CVRP NP-težak problem, samo instance malih veličina mogu se rešiti optimalno korišćenjem tačnih metoda rešenja, i to možda čak nije moguće ako je potrebno koristiti ograničenu količinu vremena za računanje. Kao rezultat toga, heurističke metode se koriste kako bi se pronašla dobra, ali ne nužno optimalna rešenja uz razumnu količinu vremena računanja.

Kako bi se očuvala jednostavnost ABC algoritma, usvojena je prilično jednostavna šema prikaza rešenja. Pretpostavimo da n klijenata posećuje m ruta vozila. Prikaz ima oblik vektora dužine (n + m). U vektoru postoje n celih brojeva između 1 i n, koji redom predstavljaju identitete klijenata. Takođe postoje i m nula u vektoru koji predstavljaju početak svake rute vozila iz skladišta. Sekvenca između dve nule je sekvenca kupaca koje treba posetiti vozilom.

Inicijalno rešenje se konstruiše dodelom jednog kupca jednom od m ruta vozila. Odabir kupca se vrši nasumično. Zatim se kupcu dodeljuje lokacija koja minimizuje

trošak dodele tog kupca trenutnom skupu ruta vozila. Postupak se ponavlja sve dok se svi kupci rutiraju. Ukupno se generiše s inicijalnih rešenja ovim postupkom.

3.1. Prostor pretrage i funkcije troškova

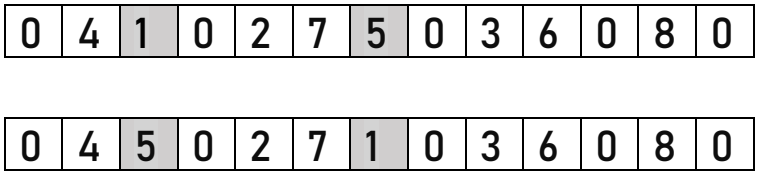
Pretraga može postati restriktivna ako se istražuje samo dopustivi deo prostora rešenja. Zbog toga takođe dopuštamo da se pretraga sprovodi u nedopustivom delu prostora rešenja, kako bi pretraga mogla oscilovati između dopustivih i nedopustivih delova prostora rešenja X. Svako rešenje x u X sastoji se od m ruta vozila, pri čemu svaki od njih počinje i završava se u skladištu, a svaki klijent se posećuje tačno jednom. Stoga, x može biti nedopustivo u vezi s kapacitetom i/ili vremenskim ograničenjima. Za rešenje x, neka c(x) označava trošak putovanja, a q(x) i t(x) označavaju ukupno kršenje kapaciteta i vremenskih ograničenja, redom. Trošak rute vozila k odgovara zbiru troškova c_{ij} povezanih s ivicama (i,j) koje prolazi to vozilo. Ukupno kršenje kapaciteta i vremenskih ograničenja računa se na osnovu ruta u odnosu na Q i L. Svako rešenje x pronađeno tokom pretrage ocenjuje se pomoću funkcije troška $z(x) = c(x) + \alpha q(x) + \beta t(x)$. Koeficijenti α i β su prilagodljivi pozitivni parametri koji se menjaju pri svakoj iteraciji. Parametar α prilagođava se na sledeći način: Ako je broj rešenja bez kršenja kapaciteta veći od τ/2, vrednost a se deli sa 1 + δ, inače se množi sa 1 + δ. Ista pravila važe i za β u vezi s vremenskim ograničenjima.

4. Operateri susedstva

Operater susedstva se koristi kako bi se dobilo novo rešenje $\tilde{x}$ iz trenutnog rešenja x u koraku 4 ABC heuristike. Unapred se biraju nekoliko operatera susedstva. Bilo koja kombinacija navedenih operatera je moguća, čak i jedan. Zatim, svaki put kada je potrebno novo rešenje $\tilde{x}$, operater susedstva se nasumično bira iz skupa prethodno odabranih operatera susedstva i primenjuje se jednom (tj. nema lokalne pretrage) na rešenje x. Mogući operatori uključuju:

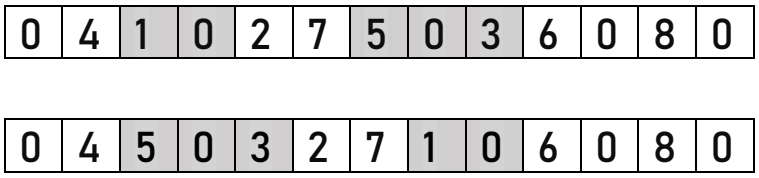
(1) Nasumične zamene

Ovaj operator nasumično bira pozicije (u vektoru rešenja) *i* i *j* sa *i* – *j* i zamenjuje kupce smeštene na pozicijama *i* i *j.*



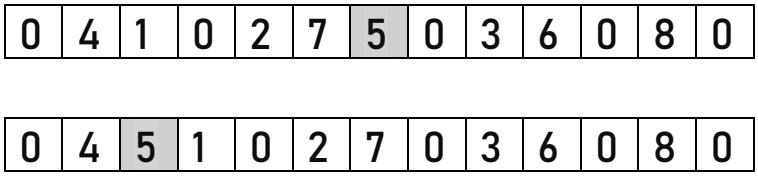
(2) Nasumične zamene podsekvenci

Ovaj operator je proširenje prethodnog, gde se biraju dve podsekvence kupaca i skladišta nasumičnih dužina i zamene ih.



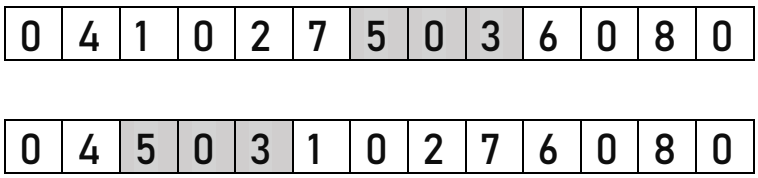
(3) Nasumična umetanja

Ovaj operator sastoji se od nasumičnog biranja pozicija *i* i *j* i premeštanja kupca sa pozicije *i* na poziciju *j*.



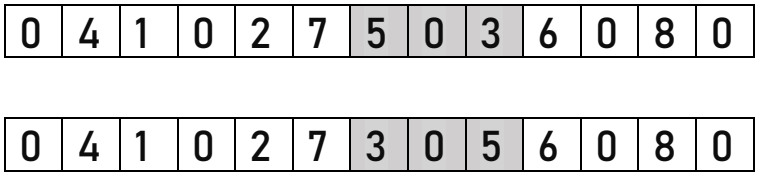
(4) Nasumična umetanja podsekvenci

Ovaj operator je proširenje operatora nasumičnih umetanja gde se podsekvenca kupaca I skladišta nasumične dužine, počevši od pozicije *I*, premešta na poziciju *j*. Pozicije *I* I *j* se biraju nasumično.



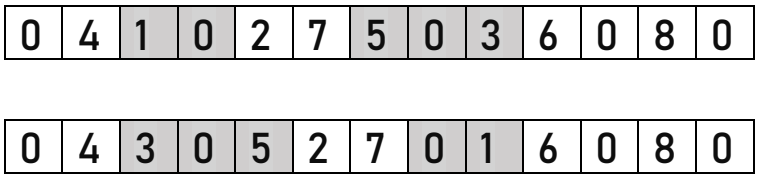
(5) Okretanje podsekvence

Bira se podsekvenca uzastopnih kupaca I skladišta nasumične dužine, a zatim se redosled odgovarajućih kupaca I skladišta obrće.



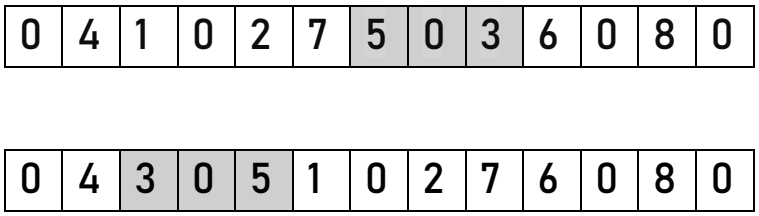
(6) Nasumične zamene obrnutih podsekvenci

Ovaj operator je kombinacija dva prethodno pomenuta operatora. Biraju se dve podsekvence kupaca i skladišta nasumičnih dužina i zamene. Zatim, svaka od zamijenjenih podsekvenci može biti obrnuta sa verovatnoćom od 50%.



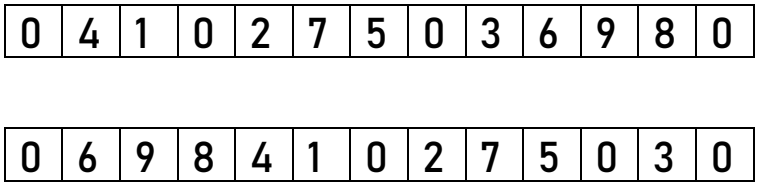
(7) Nasumična umetanja obrnutih podsekvenci

Podsekvenca kupaca i skladišta nasumične dužine počevši od pozicije *i* premešta se na poziciju *j*. Zatim, premeštena podsekvencija ima 50% šanse da bude obrnuta.



(8) Ciklično pomeranje elemenata

Kupci i skladišta se ciklično pomeraju za k pozicija, gde je k random broj. Ako je na primer k=4, niz ce posle primecnjivanja ovog operatora izgledati ovako:



(9) Promena mesta susednim elementima

Kupci na poziciji *i* i *i+1* zamenjuju mesta.

0	4	1	0	2	7	5	0	3	6	9	8	0

0	1	4	2	0	5	7	3	0	9	6	8	0

5. Algoritam veštačkog roja pčela uz korišćenje elitizma

ABC algoritam smo pokušali da poboljšamo korišćenjem elitizma. Korak 4.5. osnovnog ABC algoritma kaže da ako je prekoračen limit l_i onda se x_i zamenjuje nasumično generisanim rešenjem. Ovaj korak menjamo tako što sortiramo sva rešenja u odnosu na fitnes i za prvih *e* elemenata koji predstavljaju elitni deo populacije ne proveravamo da li su prekoračili limit.

6.Računarski eksperimenti

Testirali smo algoritam grube sile nad problemom rutiranja vozila sa kapacitetom. Rezultati testiranja se nalaze u tabeli. Sem ova 4 skupa pokušali smo da testiramo i nad problemom gde je n=8 i k=3, ali isti nije uspeo da se izvraši u realnom vremenu. Primećujemo da se vreme izvršavanja eksponencijalno povećava sa povećanjem broja gradova i kamiona.

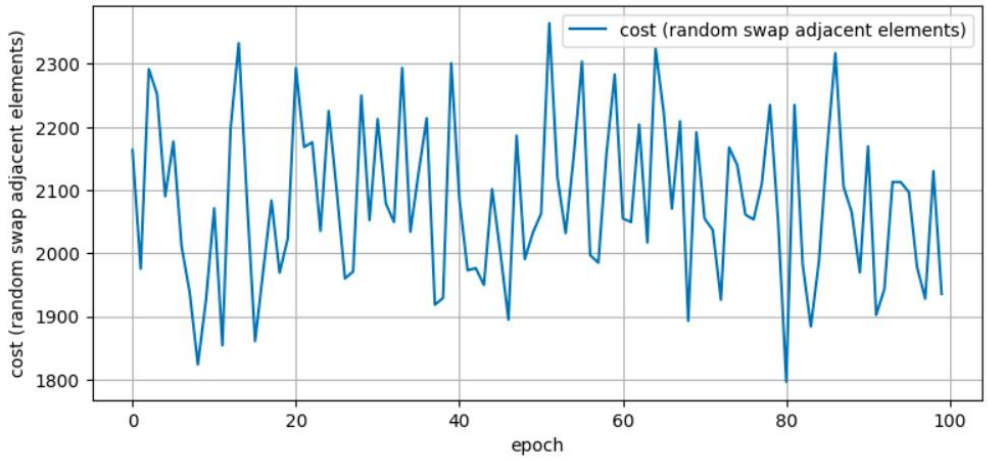
Takođe smo testirali oba algoritma, i originalni ABC i ABC sa elitizmom nad instatncama standardnog benčmark skupa. Broj zaposlenih pčela i pčela posmartača je jednak broju izvora hrane i jednak je 25. Parametar α i β smo stavili da budu jednaki, a parameter δ=0.001.

	n	k	vreme	najbolji fitnes
cvrp.vrp	5	2	0.0736	478.9970
cvrp2.vrp	8	2	19.9784	651.2520
cvrp3.vrp	7	2	2.3927	610.3251
cvrp4.vrp	7	3	22.0676	693.7150

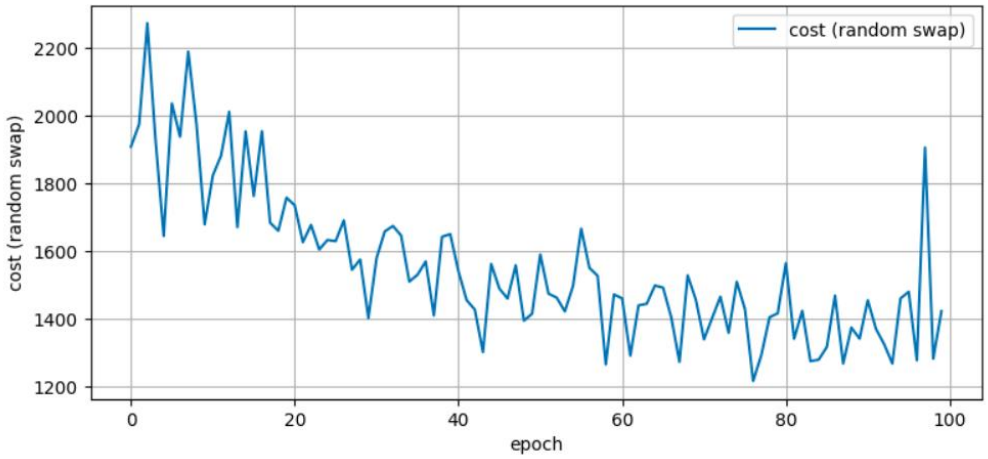
6.1. Upoređivanje operatora susedstva

Testirali smo običan ABC algoritam i ABC sa elitizmom koristeći različite operatore susedstva. Rezultati se nalaze u tabeli. Svaki od različitih operatora susedstva je pokrenut za epohe od 1 do 100. Može se videte da se sa povećanjem broja epoha smanjuje se cena obilaska grafa. Iako povećanje broja epoha dobro utiče na krajnji rezultat zbog početnog odabira nasumičnih rešenja, dešava se da rezultat mnogo odstupa od proseka. Kod upotrebe operatora 8 i 9 broj epoha ne igra veliku ulogu, a to je i logično jer su promene jako male.

	prosek	najbolji	prosek (elitizam)	najbolji (elitizam)
nasumične zamene	1552.6306	1216.3837	1452.8661	1107.0544
nasumične zamene podsekvenci	1471.3704	1093.0605	1509.3929	1037.8767
nasumična umetanja	1552.1820	1033.2257	1532.4009	1047.3443
nasumična umetanja podsekvenci	1653.5457	1291.0433	1669.8108	1169.4999
okretanje podsekvence	1474.6070	1070.9185	1470.1169	1077.0998
nasumične zamene obrnutih podsekvenci	1492.3979	1090.2017	1498.8250	1052.5073
nasumična umetanja obrnutih podsekvenci	1474.2046	1017.2936	1460.2207	1023.3294
ciklično pomeranje elemenata	2077.8762	1787.9357	1939.3525	1691.6722
promena mesta susednim elementima	2080.4202	1796.3327	1831.4976	1622.3843
nasumične zamene	1535.4543	1089.2516	1518.3917	1093.0294



Slika 1



Slika 2

6.2. Upoređivanje običnog ABC algoritma i ABC sa elitizmom

Naša istraživanja su pokazala da dodavanje elitizma u ABC aloritam nije donelo značajno poboljšanje u rezultatima. Iako se elitizam često smatra korisnim u evolutivnim algoritmima za očuvanje najboljih rešenja, naši rezultati ukazuju na to da njegova primena u ABC algoritmu možda nije od velike koristi u datom kontekstu.

Najveće poboljšanje prosečnog cost-a, uz korišćenje elitizma, je 1.29% u skupu A-n37-k6.vrp. Dok je najveće poboljšanje najboljeg cost-a 3.1% u skupu A-n34-k5.vrp.

	prosek	najbolji	prosek (elitizam)	najbolji (elitizam)
A-n32-k5.vrp	790.4518	687.5552	792.6044	690.3089
A-n33-k5.vrp	671.5033	600.3338	665.1414	583.2916
A-n33-k6.vrp	733.6733	648.3221	740.8181	659.3401
A-n34-k5.vrp	761.8601	674.2060	759.4225	653.3134
A-n36-k5.vrp	790.4572	712.9610	787.853	707.6672
A-n37-k5.vrp	762.2164	699.5413	762.1141	695.4556
A-n37-k6.vrp	866.2355	762.4032	855.0603	752.9449
A-n38-k5.vrp	777.5606	695.6125	775.6240	695.5882

7.Zaključak

U ovom radu predstavili smo heuristiku veštačkog roja pčela (ABC) za problem rutiranja vozila sa kapacitetom (CVRP). Prvo smo problem rešili algoritmom grube sile, koji daje uvek daje najbolje rešenje, ali za velike test primere ne može da se izvrši u realnom vremenu. Problem smo rešavali i uz pomoć ABC algortima i njegovom unapređenom verzijom koristeći elitizam. ABC algoritam može da da rešenje i za veće test primere, koje nije nužno najbolje, ali je dovoljno dobro. Kada

ABC algoritmu dodamo elitizam skoro uvek dobijemo malo bolje rezultate u odnosu na originalan algoritam.

8. Literatura

[1] [W.Y. Szeto, An artificial bee colony algorithm for the capacitated vehicle routing problem, 2011]

[2] [Karaboga, A comparative study of Artificial Bee Colony algorithm, 2009]