Primerjava osnovnega algoritma po vzoru obnašanja netopirjev in njegove hibridne različice HBA

Žan Grajfoner
Fakulteta za elektrotehniko,
računalništvo in informatiko
Koroška cesta 46
2000 Maribor, Slovenija
zan.grajfoner@student.um.si

Lucija Brezočnik
Fakulteta za elektrotehniko,
računalništvo in informatiko
Koroška cesta 46
2000 Maribor, Slovenija
lucija.brezocnik@um.si

POVZETEK

V prispevku smo se osredotočili na algoritme inteligence roja, ki črpajo navdih iz obnašanja roja živali v naravi. Primerjali smo osnovni algoritem po vzoru obnašanja netopirjev in njegovo hibridno različico. Raziskali smo razlike med osnovnima arhitekturama obeh algoritmov, pripadajoče parametre, kot tudi nekaj drugih hibridnih različic algoritma po vzoru obnašanja netopirjev. Primerjavo smo izvedli na praktičnem primeru optimizacije desetih testnih funkcij na treh različnih dimenzijah problema (10, 20, 30). Prav tako smo raziskali vpliv različnih velikosti populacije (20, 30, 50) pri obeh algoritmih. Ugotovili smo, da so rezultati optimizacije hibridne različice algoritma boljši od osnovne različice algoritma.

Ključne besede

algoritem po vzoru obnašanja netopirjev, evolucijski algoritmi, hibridizacija, inteligenca roja, računska inteligenca

1. UVOD

Algoritmi po vzoru iz narave posnemajo delovanje različnih naravnih in bioloških sistemov [17, 3]. Večinoma se uporabljajo za reševanje kompleksnih optimizacijskih problemov [3], kot so na primer sestava urnikov, izbira najboljše lokacije za postavitev antene in iskanje najkrajše poti na grafu. Med tovrstne algoritme uvrščamo tudi algoritme inteligence rojev [9], ki so dandanes prisotni v številnih realnih aplikacijah [8].

Inteligenca roja (angl. Swarm Intelligence, krajše SI) [1, 2] predstavlja študije kolektivnega obnašanja množičnih sistemov, ki se razvijajo s časom. Takšne sisteme v naravi tvorijo razni insekti (npr. čebele in mravlje) in druge živali (npr. ptice in ribe). Predstavimo primer inteligence roja na čebelah [7]. Čebela sledi splošnim in enostavnim pravilom, vendar lahko v roju z drugimi čebelami rešuje kompleksnejše naloge, kot je na primer izbira prave lokacije za njihov novi dom. Več izvidniških čebel odleti na iskanje potencialne lokacije, vendar dokončno odločitev o najprimernejši lokaciji sprejmejo skupaj.

V prispevku, ki je povzetek diplomskega dela [6], se bomo osredotočili na algoritem po vzoru obnašanja netopirjev (angl. Bat Algorithm, krajše BA), ki ga uvrščamo v skupino SI. Obstoječe študije so pokazale, da lahko algoritem BA hitreje doseže boljše rezultate pri reševanju optimizacijskih problemov glede na njegove predhodnike, kot je na primer genetski algoritem [10]. Algoritem BA posnema naravni fenomen netopirjev – eholokacijo. Netopirji med letenjem spuščajo visoke zvoke, kakršnih človeško uho ne zaznava. Zvok potuje skozi zrak kakor val in se odbije od vsakega objekta. Netopirji poslušajo odbiti zvok in na podlagi tega določijo oddaljenost, velikost ter obliko predmeta.

Ker osnovni algoritem BA ni primeren za reševanje vseh družin problemov, so različni avtorji predlagali več hibridnih različic, ki razširjujejo osnovni algoritem BA z mehanizmi drugih algoritmov [15, 11, 13]. Prav tako nekateri avtorji navajajo različne nastavitve nadzornih parametrov, kot so na primer velikost populacije in minimalna/maksimalna frekvenca.

Glavni cilj prispevka je primerjava osnovnega algoritma BA, ki ga je predlagal Yang [16], in hibridnega algoritma po vzoru obnašanja netopirjev (angl. Hybrid Bat Algorithm, krajše HBA), ki ga so ga predlagali Fister in ostali [5]. Algoritma bomo uporabili pri optimizaciji več testnih funkcij in s pomočjo eksperimenta skušali odgovorili na dve raziskovalni vprašanji (RV).

RV1: Ali hibridni algoritem po vzoru obnašanja netopirjev dosega boljše rezultate kot osnovni algoritem po vzoru obnašanja netopirjev pri večji velikosti populacije?

RV2: Ali hibridni algoritem po vzoru obnašanja netopirjev dosega boljše rezultate kot osnovni algoritem po vzoru obnašanja netopirjev pri optimizaciji funkcij višjih dimenzij?

V drugi sekciji predstavimo algoritem BA in njegovo hibridno različico, tretja sekcija je namenjena orisu zasnove eksperimenta, analiza dobljenih rezultatov pa je zbrana v četrti sekciji. Prispevek strnemo v peti sekciji in podamo smernice za prihodnji razvoj.

DOI: https://doi.org/10.26493/978-961-7055-82-5.85-90

2. ALGORITMI PO VZORU OBNAŠANJA NETOPIRJEV

V sekciji 2.1 predstavimo osnovno različico algoritma BA, v sekciji 2.2 njegovo hibridno različico, sekcija 2.3 pa zajema predstavitev nekaterih drugih hibridnih različic BA.

2.1 Algoritem po vzoru obnašanja netopirjev

Algoritem po vzoru obnašanja netopirjev spada v družino algoritmov SI in ga je leta 2010 razvil Xin-She Yang na univerzi v Cambridgeu [16]. Njegov namen je bil razviti enostaven in učinkovit algoritem, ki bi bil primeren za uporabo v različnih aplikacijah za reševanje optimizacijskih problemov. V algoritmu je poskušal posnemati pojav eholokacije pri mikronetopirjih. Originalni algoritem BA obravnava netopirje kot roj, ki se pomika po prostoru in išče svoj plen. Obnašanje netopirjev je kompleksno in zahtevno, zaradi česar je neposredni prenos v t. i. računalniški sistem prezahteven. Zato je njihovo obnašanje opisal v algoritmu z naslednjimi poenostavljenimi pravili [4]:

1. Vsi netopirji uporabljajo eholokacijo za spremljanje razdalje do ciljnih objektov.
2. Netopirji pri iskanju hrane letijo naključno s hitrostjo v_i na pozicijo x_i s fiksno frekvenco f_{min}, variabilno valovno dolžino λ_i in glasnostjo A_i. Samodejno lahko prilagajajo valovno dolžino (ali frekvenco) pulzov, ki jih oddajajo, in prilagajajo raven oddajanja pulzov $r\_i \in [0, 1]$ glede na bližino cilja.
3. Glasnost variira od največje (pozitivne) A_0 do minimalne vrednosti A_{min}.

BA vzdržuje populacijo virtualnih netopirjev med delovanjem. Vsak položaj netopirja predstavlja rešitev problema in vsaka rešitev problema je predstavljena kot vektor realnih števil (enačba 1):

$$x^{(t)} = \{x_{i,1}^{(t)}, \ldots, x_{i,D}^{(t)}\}, \text{za } i = 1, \ldots, Np, \tag{1}$$

v kateri D označuje dimenzijo problema, t predstavlja trenutno generacijo, Np pa število virtualnih netopirjev v populaciji. Vsak vektor določa položaj virtualnega netopirja v iskalnem prostoru. Netopirji se gibajo v območju proti učinkovitejšim rešitvam in pri tem odkrivajo nova obetavna področja. Algoritem BA omogoča dve strategiji preiskovanja.

Prvo strategijo za gibanje netopirjev prikazujejo naslednje enačbe 2:

$$\begin{aligned} f_i^t &= f_{min} + (f_{max} - f_{min})\beta, \\ v_i^{t+1} &= v_i^t + (x_i^t - x_{best}^t) f_i^t, \\ x_i^{t+1} &= x_i^t + v_i^{(t+1)} \end{aligned} \tag{2}$$

V intervalu $f_i^t \in (f_{max}, f_{min})$ se spreminja frekvenca pulza. Iz uniformne porazdelitve $\beta \in [0, 1]$ se generira naključno število, ki definira velikost izhodnega pulza. Parameter x_{best}^t definira trenutno najboljšo globalno pozicijo oziroma rešitev.

Druga strategija izboljša trenutni položaj virtualnega netopirja po enačbi 3:

$$x^t = x_{best}^t + \epsilon A_i^t \varphi, \tag{3}$$

v kateri $\varphi \in [0, 1]$ definira naključno število iz Gaussove distribucije s srednjo vrednostjo 1 in standardnim odklonom 0, $\varphi > 0$ je skalirni faktor in A_i^t predstavlja glasnost.

Ta strategija je namenjena izkoriščanju najboljše rešitve, saj je usmerjena k raziskovanju območja najboljše rešitve in ponazarja vrsto naključnega sprehoda (angl. Random Walk Direct Exploitation, krajše RWDE) [4]. Algoritem BA nadzorujeta dva parametra in sicer stopnja pulza r_i in glasnost A_i. Matematično lahko to prikažemo z enačbo 4:

$$A_i^{(t+1)} = \alpha A_i^t, r_i^t = r_i^0[1 - exp(-\gamma\epsilon)], \tag{4}$$

v kateri α in γ predstavljata konstante. Obe vplivata na stopnjo konvergence algoritma [16].

Psevdokoda algoritma BA je prikazana v algoritmu 1.

Algoritem 1 Algoritem BA

Vhod: populacija $\mathbf{x_i} = (x_{i,1}, x_{i,2}, \ldots, x_{i,D})$ for $i = 1 \ldots Np$, MAX_FE.
Izhod: najboljša rešitev x_{best} in njena vrednost f_{min}

```
inicializacija_populacije;
eval = oceni_novo_populacijo;
f_min = najdi_najboljšo_rešitev (x_best);
while ustavitveni_pogoj_ni_dosežen do
    for i = 1 to Np do
        generiraj_novo_rešitev (x_i);
        if rand(0,1) > r_i then
            y = izboljšaj_najboljšo_rešitev (x_best);
        end if
        f_new = oceni_novo_rešitev (y);
        if f_new ≤ f_i and rand(0,1) < A_i then
            x_i = y;
            f_i = f_new;
        end if
        f_min = poišči_najboljšo_rešitev (x_best);
    end for
end while
```

2.2 Hibridni algoritem po vzoru obnašanja netopirjev

Hibridni algoritem po vzoru obnašanja netopirjev je eden od prvih hibridnih različic BA. HBA je hibridiziran z mutacijskimi strategijami diferencialne evolucije [12]. Strategija lokalnega iskanja osnovnega algoritma BA je zamenjana s strategijo mutacije DE, ki je prikazana v algoritmu 2.

V algoritmu HBA se na začetku izberejo tri naključne rešitve v populaciji. S pomočjo diferencialne evolucije "DE/rand/1/

bin" generiramo novega potomca y. Izbran potomec je podvržen delovanju normalne selekcije v algoritmu BA [4].

Algoritem 2 Lokalno iskanje v algoritmu HBA

```
if rand(0,1) > r_i then
    r_{i=1...3} = rand(0,1) * Np + 1;
    n = rand(1, D);
    for i = 1 to D do
        if rand(0,1) < CR or n == D then
            y_n = x_{r1,n} + F * (x_{r2,n} + x_{r3,n});
            n = (n + 1)%(D + 1);
        end if
    end for
end if
```

2.3 Primeri ostalih različic hibridnega BA

Prednost BA je, da lahko zelo hitro doseže konvergenco v začetni fazi. Kadar je potrebna hitra rešitev, se algoritem BA izkaže kot zelo učinkovit. Če pa mu dovolimo prehiter preklop v fazo izkoriščanja, tako da se glasnost A in impulz r prehitro spremenita, lahko to povzroči stagnacijo v začetni fazi. Za izboljšanje učinkovitosti in adaptacije osnovnega algoritma so številni avtorji ustvarili nekaj različic. Kot primer lahko omenimo avtorje v [15], ki so hibridizirali osnovni algoritem BA z algoritmom iskanja harmonij (angl. Harmony Search), avtorji v [11] so hibridizirali osnovni algoritem BA z algoritmom ABC (angl. Artificial Bee Colony), avtorji v [13] pa so hibridizirali osnovno različico algoritma BA z algoritmom kresnic (angl. Firefly Algorithm).

3. OPIS EKSPERIMENTOV

V sekciji opišemo vzpostavitev razvojnega okolja in vključitev programske knjižnice NiaPy ter definiramo testne funkcije, ki smo jih uporabili v eksperimentih.

3.1 Vzpostavitev razvojnega okolja

Program je bil napisan v odprtokodnem programskem jeziku Python v integriranem razvojnem okolju PyCharm (IDE: Jetbrains). Pri razvoju smo uporabili odprtokodno knjižnico Python microframework for building nature-inspired algorithms (krajše NiaPy) [14], katere namen je enostavnejša in hitrejša uporaba algoritmov po vzorih iz narave.

3.2 Testne funkcije

Eksperimente smo izvedli na naslednjih desetih različnih testnih funkcijah [14]: Ackley, Griewank, Levy, Rastrigin, Ridge, Rosenbrock, Salomon, Schwefel, Sphere in Whitley.

4. REZULTATI

V nadaljevanju je prikazana statistika zagonov testnih funkcij. V stolpcu *Alg.* imamo kratice imen algoritmov, ki jih testiramo, in sicer algoritma BA in HBA. V stolpcu *Nast.* so prikazane nastavitve parametrov, ki smo jih spreminjali pri zagonih eksperimentov. Spreminjali smo dva parametra: dimenzijo testnih funkcij in velikost populacije. D predstavlja dimenzijo problema testnih funkcij. Testne funkcije smo zaganjali na treh dimenzijah (10, 20, 30). Višja kot je dimenzija, zahtevnejša je optimizacija. Velikost populacije predstavlja število kandidatnih rešitev, ki so uporabljene v iskalnem prostoru. Uporabili smo tri različne velikosti populacij (20, 30, 50). Preostale vrednosti nadzornih parametrov so: $Np = y$, $A = 0{,}5$, $r = 0{,}5$, $f_{min} = 0{,}0$, $f_{max} = 2{,}0$, $F = 0{,}5$ in $CR = 0{,}9$.

Vsako konfiguracijo smo zagnali 25-krat, ustavitveni pogoj pa je bil 1000 * D ovrednotenj funkcije uspešnosti.

Stolpec *Min* predstavlja najboljšo dobljeno rešitev, stolpec *Max* predstavlja najslabšo dobljeno rešitev, stolpec *Mean* pa predstavlja povprečje dobljenih rešitev. Stolpec *Median* predstavlja središče med zgornjo in spodnjo mejo dobljenih rešitev, stolpec *Std* pa predstavlja izračun standardnega odklona iz njih.

Tabela 1: Rezultati Ackley

Alg.	Nast.	Min	Max	Mean	Median	Std
BA	D=10,	14,304	18,630	17,003	17,116	1,187
HBA	Np=20	2E-06	2,013	0,782	1,155	0,809
BA	D=10,	13,772	18,273	16,701	17,002	1,189
HBA	Np=30	2E-06	3,404	0,908	1,155	1,035
BA	D=10,	10,837	18,760	16,925	17,192	1,607
HBA	Np=50	2E-06	2,580	0,999	1,155	0,987
BA	D=20,	15,997	18,960	17,970	18,034	0,690
HBA	Np=20	2,171	5,473	3,494	3,490	0,935
BA	D=20,	16,959	19,360	18,032	18,154	0,679
HBA	Np=30	1,155	5,240	3,240	3,222	1,020
BA	D=20,	16,553	19,120	17,861	17,948	0,710
HBA	Np=50	4E-07	5,650	3,070	3,223	1,203
BA	D=30,	15,879	18,940	17,950	18,062	0,650
HBA	Np=20	3,222	8,236	5,300	4,919	1,504
BA	D=30,	17,211	19,360	18,202	18,162	0,529
HBA	Np=30	2,887	10,020	6,021	6,182	1,831
BA	D=30,	17,255	19,440	18,135	18,003	0,510
HBA	Np=50	2,580	9,475	5,857	5,503	1,913

Tabela 2: Rezultati Griewank

Alg.	Nast.	Min	Max	Mean	Median	Std
BA	D=10,	1,856	4,835	3,244	3,260	0,757
HBA	Np=20	0,027	0,577	0,151	0,140	0,115
BA	D=10,	1,575	4,887	3,122	3,064	0,788
HBA	Np=30	1E-11	0,344	0,122	0,116	0,069
BA	D=10,	1,736	4,698	3,083	3,217	0,876
HBA	Np=50	0,027	0,310	0,124	0,106	0,074
BA	D=20,	3,159	10,540	6,852	6,640	1,678
HBA	Np=20	5E-14	0,101	0,033	0,025	0,035
BA	D=20,	4,725	9,525	6,718	6,560	1,242
HBA	Np=30	2E-14	0,096	0,027	0,020	0,025
BA	D=20,	3,953	11,700	7,220	7,249	1,741
HBA	Np=50	1E-13	0,118	0,026	0,020	0,028
BA	D=30,	5,226	18,170	10,970	10,501	3,019
HBA	Np=20	1E-12	0,127	0,031	0,025	0,035
BA	D=30,	6,596	16,460	10,993	10,982	2,738
HBA	Np=30	2E-12	0,107	0,025	0,015	0,027
BA	D=30,	7,005	15,310	10,785	10,685	2,348
HBA	Np=50	1E-12	0,069	0,020	0,012	0,021

Iz vseh zagonov optimizacij testnih funkcij (tabele 1-10) je razvidno, da je hibridna različica algoritma po vzoru obnašanja netopirjev učinkovitejša od osnovne različice algoritma. Naša raziskava je tako potrdila že obstoječe raziskave [5].

Dodatno smo preizkusili delovanje obeh algoritmov z raz-

Tabela 3: Rezultati Levy

Alg.	Nast.	Min	Max	Mean	Median	Std
BA	D=10,	8E-07	1,002	0,190	0,032	0,320
HBA	Np=20	3E-15	0,182	0,022	4E-14	0,047
BA	D=10,	5E-07	1,793	0,249	0,005	0,475
HBA	Np=30	4E-16	0,851	0,041	4E-14	0,170
BA	D=10,	9E-07	1,215	0,124	0,003	0,307
HBA	Np=50	8E-16	0,942	0,049	1E-14	0,188
BA	D=20,	4E-06	4,734	0,797	0,0003	1,253
HBA	Np=20	2E-16	2,065	0,286	0,091	0,514
BA	D=20,	4E-06	3,430	0,683	1E-05	1,024
HBA	Np=30	4E-16	1,701	0,278	0,182	0,388
BA	D=20,	3E-06	4,842	0,776	2E-05	1,157
HBA	Np=50	2E-16	1,123	0,272	0,091	0,382
BA	D=30,	1E-05	6,243	1,706	1,715	1,446
HBA	Np=20	2E-14	2,246	0,637	0,272	0,676
BA	D=30,	8E-06	4,614	1,179	0,907	1,369
HBA	Np=30	1E-14	2,065	0,466	0,272	0,525
BA	D=30,	1E-05	5,005	1,094	0,851	1,473
HBA	Np=50	6E-15	2,156	0,496	0,272	0,586

Tabela 4: Rezultati Rastrigin

Alg.	Nast.	Min	Max	Mean	Median	Std
BA	D=10,	15,920	77,610	45,052	44,773	15,620
HBA	Np=20	5,970	31,840	17,073	16,914	7,416
BA	D=10,	18,904	80,590	48,236	47,758	19,660
HBA	Np=30	5,970	38,800	16,312	14,742	6,937
BA	D=10,	18,904	79,600	41,828	38,804	16,340
HBA	Np=50	5,970	28,850	16,079	15,919	4,880
BA	D=20,	44,775	153,200	87,000	81,588	26,970
HBA	Np=20	17,909	62,680	42,107	45,768	13,240
BA	D=20,	36,815	161,200	88,712	87,558	31,680
HBA	Np=30	21,889	80,590	49,827	47,758	14,730
BA	D=20,	24,876	170,100	75,339	70,643	32,670
HBA	Np=50	22,884	91,540	48,992	47,758	14,730
BA	D=30,	67,661	242,800	143,120	148,250	46,420
HBA	Np=20	38,803	137,400	78,647	76,612	28,770
BA	D=30,	82,584	241,800	136,910	120,390	47,600
HBA	Np=30	42,783	130,300	73,348	74,622	21,150
BA	D=30,	59,700	222,900	126,680	123,380	37,490
HBA	Np=50	38,803	134,400	77,450	77,607	20,570

Tabela 5: Rezultati Ridge

Alg.	Nast.	Min	Max	Mean	Median	Std
BA	D=10,	7,5E2	6,5E6	4,1E6	4,3E6	1,5E6
HBA	Np=20	3E-07	0,002	0,0002	4E-05	4E-04
BA	D=10,	1E6	7,5E6	4,1E6	4E6	1,6E6
HBA	Np=30	2E-06	0,004	0,0003	3E-05	9E-04
BA	D=10,	1,2E6	8.9E6	4,1E6	3,5E6	2E6
HBA	Np=50	3E-06	2E-04	4E-05	2E-05	5E-05
BA	D=20,	1,9E6	2,5E7	1,4E7	1,4E7	6,6E6
HBA	Np=20	0,0085	4,181	0,397	0,105	0,862
BA	D=20,	5,2E6	2,9E7	1,7E7	1,7E7	6E6
HBA	Np=30	0,0082	2,566	0,302	0,071	0,549
BA	D=20,	5,8E6	2,4E7	1,5E7	1,5E7	4,7E6
HBA	Np=50	0,009	1,668	0,283	0,129	0,423
BA	D=30,	1,1E7	7,9E7	3,4E7	3,2E7	1,6E7
HBA	Np=20	0,464	2,1E6	90,873	5,618	409,032
BA	D=30,	9,5E6	6,6E7	3E7	2,8E7	1,3E7
HBA	Np=30	0,253	67,270	10,771	8,240	12,940
BA	D=30,	8,9E6	4,9E7	2,8E7	2,6E7	1,1E7
HBA	Np=50	0,414	31,860	9,629	4,938	8,573

Tabela 6: Rezultati Rosenbrock

Alg.	Nast.	Min	Max	Mean	Median	Std
BA	D=10,	1,9E8	4E7	1E7	7E6	1E7
HBA	Np=20	6E-05	100,789	7,077	2,633	19,636
BA	D=10,	5E8	2E7	1E7	9E6	6E6
HBA	Np=30	0,0065	7,903	3,641	3,430	2,340
BA	D=10,	2E6	3E7	1E7	8E6	7E6
HBA	Np=50	0,0133	65,412	5,365	2,524	12,642
BA	D=20,	2E7	1E8	5E7	5E7	2E7
HBA	Np=20	1,248	121,418	21,271	12,315	27,305
BA	D=20,	6E6	6E7	3E7	3E7	2E7
HBA	Np=30	0,495	78,608	12,600	10,839	14,593
BA	D=20,	3E6	7E7	4E7	4E7	2E7
HBA	Np=50	1,874	69,171	13,313	11,927	12,300
BA	D=30,	1E7	2E8	6E7	6E7	4E7
HBA	Np=20	4,326	567,123	61,398	22,023	109,953
BA	D=30,	2E7	2E8	7E7	6E7	4E7
HBA	Np=30	0,033	106,160	32,547	21,400	30,564
BA	D=30,	2E7	2E8	8E7	7E7	5E7
HBA	Np=50	9,143	74,178	29,963	24,145	20,055

ličnimi nastavitvami populacije. Eksperimenti so pokazali, da:

- velikost populacije pri obeh uporabljenih algoritmih nima enakega vpliva na rezultate optimizacije;
- ne moremo nedvoumno potrditi, da z večanjem populacije dosegamo boljše rezultate;
- je za vsako testno funkcijo potrebno eksperimentalno določiti optimalno nastavitev populacije, saj glede na rezultate ne moremo potrditi, da obstaja optimalna nastavitev za vse funkcije;
- je algoritem BA pri optimizaciji funkcij višjih dimenzij dosegel zelo slabe rezultate, kar so potrdile tudi že druge študije [5];
- se je algoritem HBA izkazal kot zelo učinkovit pri vseh nastavitvah parametrov. Najboljše rezultate je dosegel pri dimenzijah 10 in 20, pri nekaterih funkcijah pa tudi pri dimenziji 30;
- so bile pri funkcijah Rosenbrock, Sphere in Whitley razlike v rezultatih optimizacije največje, pri funkcijah Ackley (slika 1) in Rastrigin pa najmanjše;

Na podlagi dobljenih rezultatov in njihove analize lahko odgovorimo na zastavljeni raziskovalni vprašanji. Z večanjem populacije pri nekaterih testnih funkcijah dosežemo boljše rezultate, vendar pa to ne velja za vse primere. S tem smo odgovorili na RV1.

HBA dosega boljše rezultate na testnih funkcijah višjih dimenzij kakor njegov predhodnik BA. Razlog za to leži v hibridizaciji s strategijami DE, saj se osnovna različica algo-

Tabela 7: Rezultati Salomon

Alg.	Nast.	Min	Max	Mean	Median	Std
BA	D=10,	507E3	1054E3	828E3	800E3	158E3
HBA	Np=20	0,099	3,582	0,891	0,895	0,759
BA	D=10,	258E3	1603E3	954E3	948E3	353E3
HBA	Np=30	0,099	1,592	0,621	0,398	0,344
BA	D=10,	474E3	1390E3	870E3	871E3	247E3
HBA	Np=50	0,099	2,487	0,605	0,398	0,481
BA	D=20,	1207E3	3342E3	2153E3	2009E3	566E3
HBA	Np=20	0,895	14,326	4,740	3,582	3,006
BA	D=20,	901E3	3124E3	2260E3	2312E3	631E3
HBA	Np=30	0,895	12,038	4,000	2,487	3,171
BA	D=20,	1370E3	3732E3	2225E3	2137E3	641E3
HBA	Np=50	0,895	9,949	4,115	3,582	2,906
BA	D=30,	1950E3	5542E3	3,4E6	336E4	1E6
HBA	Np=20	4,875	39,790	10,912	8,059	7,293
BA	D=30,	265E4	666E4	396E4	386E4	970E3
HBA	Np=30	2,487	39,791	13,733	9,949	8,835
BA	D=30,	2,2E6	6E6	3,9E6	3,8E6	1E6
HBA	Np=50	3,582	35,911	12,070	9,949	8,059

Tabela 8: Rezultati Schwefel

Alg.	Nast.	Min	Max	Mean	Median	Std
BA	D=10,	1559E3	3213E3	2644E3	2707E3	354E3
HBA	Np=20	385E3	2024E3	1246E3	1308E3	481E3
BA	D=10,	2041E3	3101E3	2723E3	2757E3	249E3
HBA	Np=30	355E3	2026E3	1139E3	1108E3	408E3
BA	D=10,	2232E3	3223E3	2685E3	2697E3	270E3
HBA	Np=50	355E3	1906E3	1087E3	952E3	450E3
BA	D=20,	5414E3	6850E3	6164E3	6265E3	415E3
HBA	Np=20	2195E3	3852E3	2752E3	2696E3	424E3
BA	D=20,	5379E3	6607E3	6195E3	6233E3	298E3
HBA	Np=30	1782E3	3990E3	2893E3	2926E3	502E3
BA	D=20,	4477E3	6860E3	6088E3	6130E3	490E3
HBA	Np=50	1682E3	4051E3	2654E3	2645E3	541E3
BA	D=30,	8983E3	1059E4	9778E3	9805E3	350E3
HBA	Np=20	2338E3	6235E3	4483E3	4448E3	839E3
BA	D=30,	9025E3	1084E4	9997E3	10003E3	365E3
HBA	Np=30	3759E3	6319E3	4655E3	4602E3	732E3
BA	D=30,	8424E3	1051E4	9790E3	9780E3	528E3
HBA	Np=50	2694E3	5577E3	4455E3	4379E3	718E3

ritma hitro ujame v lokalni optimum. S tem smo odgovorili na RV2.

5. ZAKLJUČEK

V prispevku smo predstavili algoritem BA. Natančneje smo preučili eno izmed njegovih prvih različic, poimenovano HBA. V sklopu eksperimenta smo oba algoritma uporabili za optimizacijo desetih različnih testnih funkcij. Eksperimente smo zaganjali na različnih konfiguracijah, pri čemer smo spreminjali dimenzije problemov in velikost populacije. Ugotovili smo, da se je hibridna različica BA izkazala za uspešnejšo na vseh konfiguracijah nad vsemi testnimi funkcijami.

Za prihodnje delo bomo uporabili hibridno različico algoritma po vzoru obnašanja netopirjev v okviru realnih aplikacij, kot je na primer sestavljanje urnikov. Zanimiva bi bila tudi primerjava dobljenih rezultatov z rezultati drugih algoritmov, ki jih podpira knjižnica NiaPy.

Tabela 9: Rezultati Sphere

Alg.	Nast.	Min	Max	Mean	Median	Std
BA	D=10,	1,729	23,752	9,256	8,718	5,678
HBA	Np=20	8E-15	6E-12	6E-13	1E-13	1E-12
BA	D=10,	2,417	18,624	9,533	10,048	4,455
HBA	Np=30	1E-14	6E-12	8E-13	2E-13	1E-12
BA	D=10,	1,120	23,656	9,089	8,915	5,111
HBA	Np=50	9E-15	4E-12	7E-13	2E-13	1E-12
BA	D=20,	6,318	34,822	21,238	19,588	7,618
HBA	Np=20	2E-16	9E-14	2E-14	6E-15	3E-14
BA	D=20,	5,700	46,084	18,393	17,703	9,185
HBA	Np=30	5E-17	2E-13	5E-14	2E-14	6E-14
BA	D=20,	8,663	42,936	23,238	20,392	10,005
HBA	Np=50	1E-16	1E-12	1E-13	2E-14	3E-13
BA	D=30,	10,235	51,285	23,936	22,397	10,715
HBA	Np=20	2E-14	7E-12	1E-12	5E-13	2E-12
BA	D=30,	5,173	52,861	22,680	23,952	10,070
HBA	Np=30	1E-14	2E-11	2E-12	2E-13	4E-12
BA	D=30,	3,689	52,914	22,079	21,339	9,386
HBA	Np=50	3E-15	2E-11	2E-12	2E-13	5E-12

Tabela 10: Rezultati Whitley

Alg.	Nast.	Min	Max	Mean	Median	Std
BA	D=10,	2,8E8	1E8	1E7	2E6	2E7
HBA	Np=20	11E3	73E3	44E3	43E3	15E3
BA	D=10,	6,4E7	6E7	1E7	7E6	2E7
HBA	Np=30	19E3	62E3	44E3	46E3	11E3
BA	D=10,	1,3E7	4E7	7E6	4E6	1E7
HBA	Np=50	9E3	59E3	41E3	44E3	12E3
BA	D=20,	3,8E8	1E9	1E8	5E7	2E8
HBA	Np=20	71E3	328E3	249E3	250E3	51E3
BA	D=20,	3,9E8	5E8	9E7	6E7	1E8
HBA	Np=30	150E3	327E3	248E3	258E3	43E3
BA	D=20,	4,8E8	4E8	8E7	5E7	9E7
HBA	Np=50	124E3	343E3	240E3	248E3	52E3
BA	D=30,	3E6	2E9	3E8	2E8	3E8
HBA	Np=20	451E3	784E3	656E3	667E3	78E3
BA	D=30,	9E6	9E8	3E8	3E8	2E8
HBA	Np=30	408E3	855E3	663E3	663E3	98E3
BA	D=30,	8E6	6E8	2E8	2E8	1E8
HBA	Np=50	334E3	851E3	659E3	639E3	117E3

ZAHVALA

Raziskovalni program št. P2-0057 je sofinancirala Javna agencija za raziskovalno dejavnost Republike Slovenije iz državnega proračuna.

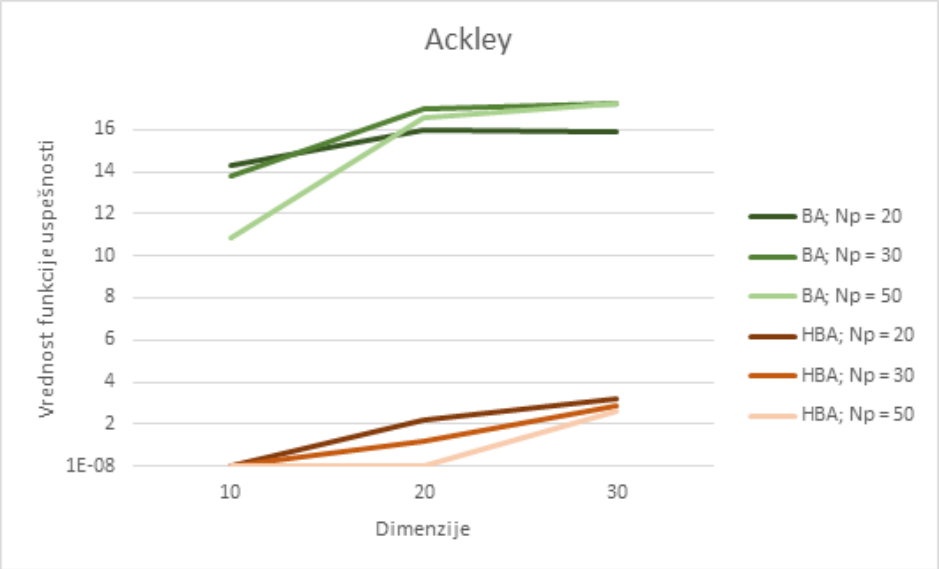
LITERATURA

[1] E. Bonabeau, D. d. R. D. F. Marco, M. Dorigo, G. Theraulaz, et al. *Swarm intelligence: from natural to artificial systems*. Number 1. Oxford university press, 1999.

[2] L. Brezočnik. Optimizacija z rojem delcev za izbiro atributov pri klasifikaciji. Master's thesis, University of Maribor, Slovenia, 2016.

[3] L. Brezočnik, I. Fister, and V. Podgorelec. Swarm intelligence algorithms for feature selection: a review. *Applied Sciences*, 8(9):1521, 2018.

[4] I. Fister Jr. *Algoritmi računske inteligence za razvoj*



Slika 1: Primerjava optimizacije testne funkcije Ackley glede na D in Np.

umetnega športnega trenerja. PhD thesis, University of Maribor, Slovenia, 2017.

[5] I. Fister Jr., D. Fister, and X.-S. Yang. A hybrid bat algorithm. *Elektrotehniški vestnik*, 80(1-2):1–7, 2013.

[6] Ž. Grajfoner. Primerjava različnih algoritmov po vzoru obnašanja netopirjev. University of Maribor, Slovenia, 2019.

[7] D. Karaboga. An idea based on honey bee swarm for numerical optimization. Technical report, TR06, Department of Computer Engineering, Engineering Faculty, Erciyes University, 2005.

[8] K. Ljubič and I. Fister Jr. Narava kot navdih računalništvu. *Življ. teh.*, 65(15):14–17, 2014.

[9] K. Ljubič and I. Fister Jr. Algoritem na osnovi obnašanja netopirjev. *Presek*, 42(3):26–28, 2015.

[10] S. Mirjalili, S. M. Mirjalili, and X.-S. Yang. Binary bat algorithm. *Neural Computing and Applications*, 25(3-4):663–681, 2014.

[11] J.-S. Pan, T.-K. Dao, M.-Y. Kuo, M.-F. Horng, et al. Hybrid bat algorithm with artificial bee colony. In *Intelligent Data analysis and its Applications, Volume II*, pages 45–55. Springer, 2014.

[12] A. K. Qin, V. L. Huang, and P. N. Suganthan. Differential evolution algorithm with strategy adaptation for global numerical optimization. *IEEE transactions on Evolutionary Computation*, 13(2):398–417, 2008.

[13] T. Sureshkumar, M. Lingaraj, B. Anand, and T. Premkumar. Hybrid firefly bat algorithm (hfba)–based network security policy enforcement for psa. *International Journal of Communication Systems*, 31(14):e3740, 2018.

[14] G. Vrbančič, L. Brezočnik, U. Mlakar, D. Fister, and I. Fister Jr. Niapy: Python microframework for building nature-inspired algorithms. *J. Open Source Softw.*, 3:613, 2018.

[15] G. Wang and L. Guo. A novel hybrid bat algorithm with harmony search for global numerical optimization. *Journal of Applied Mathematics*, 2013, 2013.

[16] X.-S. Yang. A new metaheuristic bat-inspired algorithm. In *Nature inspired cooperative strategies for optimization (NICSO 2010)*, pages 65–74. Springer, 2010.

[17] X.-S. Yang and M. Karamanoglu. Swarm intelligence and bio-inspired computation: An overview. In X.-S. Yang, Z. Cui, R. Xiao, A. H. Gandomi, and M. Karamanoglu, editors, *Swarm Intelligence and Bio-Inspired Computation*, pages 3 – 23. Elsevier, Oxford, 2013.