Boško Ristović

Generisanje ratnika za igricu Corewar genetskim algoritmom

Korišćenjem genetskog algoritma generisani su ratnici za igricu Corewar i praćena promena u performansama ratnika u odnosu na parametre algoritma. Corewar je računarska igrica u kojoj se dva ili više računarskih programa takmiče za kontrolu nad virtuelnim memorijskim prostorom. Merili smo uticaj dva parametra genetskog algoritma i kako se zavisno od njih menja učinak dobijenih ratnika. Prvi parametar je broj gena ili instrukcijâ koji čine jednog ratnika. Drugi parametar je funkcija za generisanje gena, pri čemu su korišćene dve funkcije, jedna ima uniformnu raspodelu, a druga generiše vrednosti koje imaju normalnu raspodelu. U radu je pokazano da su ratnici kojima je broj gena 20 i imaju implementiranu normalnu distribuciju u svojoj funkciji za generisanje gena, znatno bolji od preostale tri grupe koje su testirane u ovom radu. Zapaženo je da se ratnici koji pripadaju izgenerisanoj generaciji sa najboljim performansama ponašaju po pasivnoj strategiji u kojoj im je glavni cilj da prežive, a ne da unište protivnika.

Uvod

Corewar (ili Core War) je računarska igrica napravljena 1984. godine. Napravili su je D. G. Jones i A. K. Dewdney koji su u martu te godine objavili Core War Guidelines (Dewdney i Jonese 1984; Dewdney 1984) i time definisali pravila igrice. U ovoj igrici dva ili više računarskih programa, poznatiji kao ratnici, takmiče se za kontrolu nad virtuelnim računarom. Ovi ratnici se pišu u apstraktnom asemblerskom kodu koji se zove Redcode. Na početku svake igre, svaki ratnik se učitava u memorijski prostor virtuelnog računara na nasumično izabranoj lokaciji, posle toga on izvršava jednu instrukciju po potezu. Cilj igre je da se prekine izvršavanje procesa protivničkog ratnika, što se dešava kad taj proces pokuša da izvrši nedozvoljenu instrukciju. Jedna lokacija u memoriji označava tačno jednu Redcode instrukciju. Postoji nekoliko Redcode standarda, a za svrhe ovog istraživanja korišćen je '94 standard (ICWS 1994) bez P-prostora. P-prostor je najnoviji dodatak Redcode-u koji dozvoljava ratnicima da čuvaju podatke u posebnu memoriju.

Ideja ovog rada je pokušaj da se dođe do ratnika koji neće samo gubiti većinu svojih borbi, već će biti kompetitivan protiv nekih već dokazano dobrih ratnika, a to su oni „ručno napravljeni" ratnici koji ostvaruju dobre rezultate na turnirima. Cilj istraživanja je, da se korišćenjem genetskog algoritma ispitaju karakteristike nastalih ratnika, kao i razlike u rezultatima između ratnika koji su nastali korišćenjem različitih parametara u algoritmu. Parametri koji se u ovom radu razmatraju su broj gena jednog ratnika i funkcija za generisanje gena. Genetski algoritam je metod rešavanja optimizacionih problema koji je zasnovan na prirodnoj selekciji, procesu koji pokreće biološku evoluciju.

Opis Corewar-a

Set instrukcija za Redcode '94 standard je sledeći (Karonen 1997):

DAT – uklanja (ubija) trenutni izvršavajući proces iz procesnog reda
MOV – kopira podatak (instrukciju) sa jedne adrese na drugu
ADD – jednom broju dodaje drugi
SUB – od jednog broja oduzima drugi

Boško Ristović (2000), Čačak, učenik 4. razreda Tehničke škole u Čačku

MENTOR: Igor Šikuljak, student Fakulteta tehničkih nauka Univerziteta u Novom Sadu

MUL – množi dva broja
DIV – deli jedan broj drugim
MOD – deli jedan broj drugim i vraća ostatak deljenja
JMP – proces „skače" i nastavlja izvršavanje na zadatoj adresi
JMZ – proverava da li je neki broj 0, a ako jeste „skače" na zadatu adresu
JMN – proverava da li je neki broj različit od 0, ako jeste „skače" na zadatu adresu
DJN – dekrementira broj za 1 i „skače" na zadatu adresu, osim ako je rezultat 0
SPL – stvara novi proces, koji počinje izvršavanje na zadatoj adresi
SEQ – ako su dve zadate instrukcije jednake, preskače izvršavanje sledeće po redu
SNE – ako dve zadate instrukcije nisu jednake, ne izvršava sledeću po redu
SLT – poredi dve vrednosti, ukoliko je prva manja od druge, ne izvršava sledeću instrukciju
NOP – nema operacije (ne radi ništa)

Broj mogućih ratnika je beskonačan, ali su poznate neke opšte strategije koje su konkurentne u borbama i turnirima: **Kamen** (Stone): ovi ratnici su mali po broju instrukcija i brzi u njihovom izvršavanju, obično razbacuju DAT bombe po memoriji u određenim prostornim intervalima u nadi da će tako pogoditi protivnika i naterati ga da izvrši tu nedozvoljenu instrukciju. Obično su dobri protiv *makaza* ali gube od *papira*. **Papir** (Paper) kopira svoj kod na više mesta u lokaciji i izvršava ga paralelno; ovi ratnici su obično veoma izdržljivi i dobri protiv *kamena* ali gube od *makaza*. Strategija poznata kao **makaze** (Scissors) prvo skenira memorijski prostor tražeći protivnika i kada pronađe njegovu lokaciju vrši jak i precizan napad, prvo „ošamuti" protivnika SPL instrukcijama, pa ga ubije DAT instrukcijama. Dobra je protiv *papira*, ali gubi od *kamena*. Takođe je moguće praviti ratnike koji su hibridi ovih strategija u cilju da se napravi strategija koja će funkcionisati dobro protiv većine, ali to uvek ima svoju cenu.

Metode

U ovom radu genetski algoritam je korišćen kao generator ratnika na kojima je vršeno testiranje njihovih performansi. Jedan ratnik označava jednu jedinku u populaciji koju ćemo posmatrati, a gen je jedna Redcode instrukcija unutar ratnika. Dakle, više gena čini jednog ratnika, a više ratnika čini jednu populaciju. Genetski algoritam kao ulaz dobija veličinu populacije, veličinu samog ratnika, odnosno broj instrukcija u jednom ratniku, funkciju kojom može da pravi nasumične gene, funkciju koja vraća stepen prilagodjenosti, koeficijent za elitizam, i šansu za mutaciju u procentima. Koeficijent elitizma određuje broj najboljih ratnika koji će nepromenjeni biti kopirani u narednu generaciju, a mutacijom se postojeći gen zamenjuje novim, nasumično odabranim genom.

Kroz sva testiranja korišćene su iste vrednosti za veličinu populacije, šansu za mutiranje i koeficijent elitizma. Veličina populacije je 50, šansa za mutaciju je 5%, a koeficijent elitizma 2.

Funkcija stepena prilagođenosti određuje način na koji objektivno određujemo koliko je neka jedinka u populaciji uspešna, i pomoću nje određujemo da li će ta jedinka imati šanse da prosledi svoje gene na narednu generaciju. Ako se pokaže dobro, te šanse će biti veće, a u suprotnom biće manje. Zbog toga što je ovo istraživanje vezano za pravljenje što boljih ratnika za igricu Corewar, ocenjivaćemo ih po tome koliko su dobri u samoj igrici. Iz tog razloga za funkciju određivanja stepena prilagođenosti ovog genetskog algoritma korišćena je upravo ova igrica. Igrica Corewar se pokreće u programu zvanom pMARS (Portable Memory Array Redcode Simulator). Ovde je korišćena Windows verzija ovog programa koji je napravio Joonas Pihlaja (Pihlaja 2003). Podrazumevana podešavanja za pMARS, koja su takođe korišćena u ovom radu, su:

- veličina memorije (prostor za borbu): 8000 instrukcija
- maksimalna veličina ratnika: 100 instrukcija
- maksimalan broj aktivnih procesa po jednom ratniku: 8000 procesa

Konačan broj poteza koji se mogu odigrati: 80 000 poteza (ako se posle ovog broja poteza ne dođe do pobednika ishod bitke je nerešen)

Svaka jedinka u genetskom algoritmu je testirana na isti način, tako što će se boriti sa 12 ručno napravljenih ratnika, i sa svakim od njih odigrati 50 borbi. Izabran je skup od 12 ručno napravljenih ratnika koji su raznovrsni po strategijama

kojim igraju, tako da imamo po 4 ratnika iz svake od tri osnovne grupe strategija koje su prethodno navedene. Dakle, ne merimo koliko su naše jedinke dobre protiv jedne strategije, već gledamo performanse protiv svih strategija. Ovaj skup ručno napravljenih ratnika nije slučajno izabran, već je korišćen Wilkie's benchmark (Koth.org 2018), pošto već postoje naučni radovi koji su koristili isti skup za svoja istraživanja, kao na primer u eksperimetima koje je sproveo Andersen (2001). Sistem bodovanja je takav da se za pobedu u borbi dobijaju 3 poena, za nerešen ishod 1 poen, a za izgubljenu borbu 0 poena. Svi poeni neke jedinke se sabiraju i dele sa 12, i to daje njen konačni rezultat, odnosno broj koji će nam reći koliko je ta jedinka dobra. Minimalan mogući broj bodova je 0, ako jedinka izgubi svaku od 600 test borbi, a maksimalan 150, ako jedinka dobije svaku odigranu borbu.

U ovom radu smo pratili uticaj dve promenljive u genetskom algoritmu na uspeh jedinki u našoj populaciji. To su veličina ratnika, odnosno broj gena jednog ratnika, i funkcija za dobijanje nasumičnih gena. Za broj gena smo koristili vrednosti 5 i 20. Kod funkcije za dobijanje nasumičnih gena koristili smo dve različite metode. Prva metoda pravi instrukcije koje u sebi sadrže potpuno slučajne brojeve za adrese od –4000 do 4000, a druga metoda za svoje adrese slično koristi slučajne brojeve, ali ovoga puta tako da oni imaju normalnu (Gausovu) raspodelu sa sredinom u nuli i standardnom devijacijom od 1000.

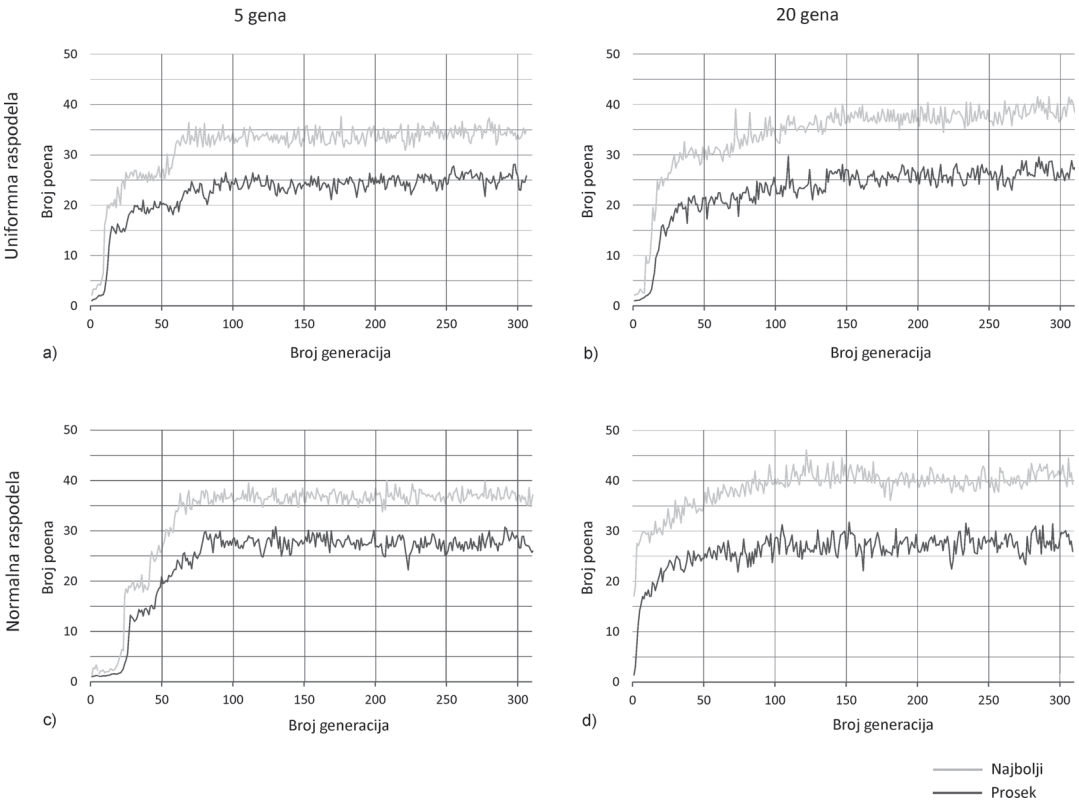
Rezultati i diskusija

Podaci koji su mereni u ovom istraživanju su stepen prilagođenosti, ili broj poena koje je ostvarila najbolja jedinka u nekoj generaciji, kao i prosek broja poena svih jedinki u jednoj generaciji. Takođe, bitno je napomenuti da, iako maksimalni broj poena koji se može ostvariti prethodno opisanom šemom testiranja iznosi 150 poena u praksi taj rezultat nije dostižan jer je nemoguće napraviti ratnika koji igra dobro protiv svake strategije sa kojima se suočava u skupu naših protivničkih test ratnika.

Najbolji ratnici na svetu, oni koji osvajaju organizovane turnire i uvek se nalaze na vrhu rang liste istih na korišćenoj šemi testiranja dobijaju rezultat od približno 90 poena. Da bi se neki ratnik smatrao dobrim, odnosno da bi imao šanse da uspešno učestvuje u nekim slabijim turnirima, trebao bi da ima minimalan rezultat od 50 poena. Kao što je navedeno u radu Davida Andersena (Andersen 2001), gde je korišćen isti skup test radnika, do sada niko nije uspeo da iz početne generacije potpuno nasumičnih ratnika razvije jednog ratnika koji je na ovom testu uspeo da ostvari rezultat od preko 50 poena.

Na slici 1a su prikazani podaci iz grupe, koja je napravljena sa veličinom ratnika od 5 instrukcija, a njihova funkcija za dobijanje nasumčnih gena je implementirala uniformnu raspodelu. Ako pogledamo rezultate do prvih 50 generacija, vidimo nagli napredak naše populacije ratnika. Na samom početku, rezultati su očekivano dosta slabi, pošto tek napravljeni ratnici imaju potpuno nasumične instrukcije u sebi. Velika većina njih ne radi apsolutno ništa, i jednostavno umru na početku borbe tako što njihovi procesi slepo zalutaju u slobodan memorijski prostor virtuelnog računara koji je ispunjen nedozvoljenim DAT instrukcijama, ili naivno sami imaju DAT instrukciju u sebi i izvrše je. Oni koji ne umru odmah sede i čekaju da ih protivnik ubije. Procesom selektivnog uparivanja loše osobine se gube, i već posle tridesete generacije ratnici nauče da je samoubistvo loša taktika za pobedu. Oko 70. generacije dolazimo do vrhunca performansi ove populacije, nakon čega sledi stagnacija u njihovom razvoju. Najbolji ratnik ove populacije imao je rezultat od 37.58 u 176. generaciji, što je u kontekstu ovog rada i ostalih grupa koje su ispitivane, najlošiji rezultat.

Na slici 1b su prikazani rezultati grupe u kojoj je za dobijanje gena takođe korišćena uniformna raspodela, ali je napravljena sa veličinom ratnika od 20 instrukcija. U ranim fazama ova grupa pokazuje slične rezultate kao prethodna, ali bitna razlika za ovu grupu je što ona ne stagnira sa razvojom posle 70. generacije. Ova populacija ima vidan rast u broju ostvarenih poena sve do neke 150. generacije, i tek posle toga stagnira. Moja pretpostavka je da je razlog za to taj što zbog povećanog broja instrukcija ovi ratnici imaju više više elemenata koji mogu da se unaprede, u odnosu na prvu grupu. Najbolji predstavnik ove grupe je ostvario rezultat od 43



Slika 1. Evolucija broja poena po generacijama. a) 5 gena, uniformna raspodela; b) 20 gena, uniformna raspodela; c) 5 gena, normalna raspodela; d) 20 gena, normalna raspodela.

Figure 1. Evolution of the score by generations: a) 5 genes and uniform distribution; b) 20 genes and uniform distribution; c) 5 genes and normal distribution; d) 20 genes and normal distribution. The darker line shows the average, and the lighter line the best scores.

poena u generaciji 317, što je znatno bolje od prethodne grupe.

Na slici 1c su prikazani podaci grupe manjih ratnika, dakle onih koji sadrže 5 instrukcija u sebi, ali ovoga puta u njihovoj funkciji za dobijanje nasumičnih gena je implementirana normalna distribucija, sa standardnom devijacijom od 1000. To za ratnike iz ove populacije znači da će najveći broj njihovih instrukcija pokazivati na memorijske lokacije koje su bliže njima, a samo mali broj njih pokazivaće na daleke lokacije. Kao u prvoj grupi primećuje se stagnacija u razvoju posle sedamdesete generacije, ali kod ove grupe imamo primetno poboljšanje u odnosu na prvu grupu. To poboljšanje je u proseku rezultata svih ratnika, kao i u rezultatima najboljih ratnika u generaciji. Ovom metodom je uspešno napravljen ratnik koji je dostigao čak 40 poena u generaciji 107, što nije bilo moguće postići metodom sa uniformnom raspodelom, koja je korišćena kod prve grupe ratnika.

Iz prethodno navedenih rezultata mogli smo da vidimo da sa većim brojem instrukcija i implementacijom normalne distribucije dobijamo bolje rezultate. U četvrtoj grupi ratnika smo iskoristili obe metode koje su davale poboljšanje u rezultatima, sa pretpostavkom da će to doneti još bolje rezultate. Na slici 1d vidimo da se pretpostavka pokazala tačnom. Ovde vidimo poboljšanje rezultata sve do vrhunca u 121. generaciji, kada je jedan ratnik ostvario rezultat od 46.08, čak 10 poena više nego najbolji ratnik u prvoj

grupi. Posle toga, kao i u ostalim grupama, vidimo stagnaciju u razvoju. Ako pogledamo strategiju koju je ovaj ratnik razvio, vidimo nešto interesantno. On u vrlo malom broju bitaka ostvaruje pobedu, njegov glavni cilj jeste da borbu završi nerešenim rezultatom, u čemu je veoma dobar. Konkretno, njegova strategija je da na početku napravi više svojih procesa i održava ih u životu tako što ih vrti u zatvorenoj petlji, to mu daje otpornost na tuđe napade, jer ako mu jedan proces umre ostali će da nastave da se izvršavaju. Dok radi na ojačavanju sebe, u isto vreme pokušava da „ošamuti" protivnika, dakle ne i da ga ubije. To radi tako što dekrementira pokazivače na adrese u instrukcijama ispred sebe kroz celu memoriju, što može da onesposobi protivničke procese tako što poremeti instrukcije koje oni izvršavaju. Ako uspe da onesposobi protivnika, dva ratnika će samo sedeti ne radeći ništa korisno, dok ne istekne maksimalno dozvoljenih 80000 procesa i borba se završi nerešenim rezultatom. Strategija koja se razvila u ovoj grupi ratnika je vrlo interesantna. Sve što ovi ratnici žele je da ostvare najveći broj poena moguć, i izgleda da su pronašli najlakši put da to ostvare. Razvili su takvu strategiju da je njihov cilj u borbi da prežive do kraja, ostvarujući po jedan poen za nerešenu bitku, a ne da nastoje da pobede protivnika i time dobiju tri poena. Ovo se dešava verovatno zato što je stepen kompleksnosti kod agresivnih strategija dosta veći, nego kod pasivnih strategija koje smo dobili. Sve što je potrebno za uspešnu pasivnu strategiju je nekoliko SPL instrukcija na početku ratnika i DJN instrukcija koja će da napravi beskonačnu petlju, tako što će vraćati procese unazad kroz memoriju gde se nalaze te SPL instrukcije, i praviti još više procesa. Velika prednost u korišćenju DJN instrukcije, u odnosu na ostale instrukcije skoka koje postoje u Redcode-u, je ta što DJN, pre nego što izvrši skok, dekrementira neko polje u memoriji, što može da poremeti protivnika.

Zaključak

U ovom radu je praćen razvoj populacije ratnika u igrici Corewar zavisno od početnih parametara zadatih genetskom algoritmu, kao i karakteristikâ nastalih ratnika, odnosno njihove strategije. Pokazano je da veličina ratnika i tip funkcije za dobijanje nasumičnih gena imaju bitnu ulogu u eventualnom rezultatu jedne populacije ratnika. Konkretno, pokazano je da su ratnici koji imaju 20 instrukcija bolji od onih koji imaju 5, kao i da su oni koji implementiraju normalnu distribuciju u svojoj funkciji za dobijanje nasumičnih gena bolji od onih koji implementiraju ravnomernu raspodelu.

Takođe je otkriven tip ratnikâ koji se dobija ovom metodom, odnosno njhova strategija. To je jedna pasivna strategija, u kojoj cilj nije da se osvoje tri boda i pobedi protivnik, već održavanje u životu, i dobijanje jednog boda za nerešen rezultat. Ako jedan ratnik odigra nerešeno svaku borbu, to mu daje maksimalni mogući rezultat od 50 bodova, a ovi ratnici upravo to i pokušavaju. Ne pokušavaju da ostvare pobedu, i tako gube priliku da ostvare bolji rezultat. Naša pretpostavka je da baš zbog ove strategije u svakoj od grupa vidimo stagnaciju rezultata najboljeg predstavnika iz generacije pre nego što taj rezultat dostigne crtu od 50 bodova. Ovo bi se moglo izbeći drugačijim sistemom bodovanja, gde bi jedini način za ostvarivanje poena bila pobeda u borbi.

Dalja istraživanja se mogu fokusirati na pokušaj da se ovakvom metodom evolucije razvije agresivna strategija borbe. Agresivna strategija je ona strategija koja, za razliku od pasivne strategije koju smo dobili u ovom radu, aktivno pokušava da uništi svog protivnika. Jedan od načina na koji bi se to moglo izvesti je upravo izmena postojećeg sistema bodovanja u neki drugi, koji dodeljuje poene samo za pobede u borbi, a ne i za izjednačen rezultat.

Literatura

Andersen D. G. 2001. The Garden: Evolving Warriors in Core Wars. https://www.angio.net/res/garden.pdf

Dewdney A. K., Jonese D. G. 1984. *Core War Guidelines*. London (Canada): Department of computer science the University of Western Ontario

Dewdney A. K. 1984. Computer recreations: In the game called Core War hostile programs engage in a battle of bits. *Scientific American*, **250** (5): 14.

ICWS (International Core War Society) 1994. ICWS'94 standard. https://corewar.co.uk/standards/icws94.htm

Karonen I. 1997. The beginners' guide to Redcode. http://vyznev.net/corewar/guide.html

Koth.org. 2018. JKW's Beginner's Benchmark. http://www.koth.org/wilkies/

Pihlaja J. 2003. CoreWin (SDL pMARS). https://corewar.co.uk/pihlaja/pmars-sdl/index.htm

---

Boško Ristović

Generating Warriors for the Computer Game Corewar with a Genetic Algorithm

In this research paper warriors for the game Corewar were generated using a genetic algorithm and the change in their performance was measured based on the starting parameters of the genetic algorithm. Corewar is a computer game in which two or more computer programs compete for control over a virtual memory space. We measured the effect of two parameters and how the performance of the warriors change in connection to them. The first parameter is the number of genes or the number of Redcode instructions in a single warrior (5 or 20). The second parameter is a function for generating mutated genes, where one function has a uniform distribution of the address values in an instruction, and the second one implements normal distribution with a standard deviation of 1000. Each generated warrior in a population is tested as follows: it plays 50 battles against each of the 12 warriors in Wilkies benchmark (Koth.org 2018), a win in a battle is worth 3 points, a draw 1 and defeat 0 points, the score is then summed up and divided by 12. Research has shown us that warriors with 20 instruction and which implement normal distribution in their function for generating mutated genes have significantly better performance than the other 3 groups which were tested. It was identified that warriors which belong to the generation with the best performance have developed a passive strategy in which their main goal is to survive, and not to destroy the opponent. The theoretical maximum in the performance for this strategy is 50 points. That is precisely the reason for which we see stagnation in the development of warriors before they ever reach a fitness score of 50 points. The best warrior which was generated in this research had a score of 46 points.