Genetski algoritmi

Magušić-Dumančić, Iva

Sveučilište J. J. Strossmayera u Osijeku

Filozofski fakultet

Preddiplomski studij Informatologije

Iva Magušić-Dumančić

Genetski algoritmi

Završni rad

Mentor: izv. prof. dr. sc. Gordana Dukić
Sumentor: dr. sc. Anita Papić, viša asistentica

Osijek, 2014.

Sažetak:

Cilj rada je dati pregled algoritama te pobliže pojasniti genetske algoritme, njihov razvoj i strukturu te prikazati njihovu primjenu. Ukratko se govori o povijesnom razvoju algoritama, od Euklidovog algoritma do programskih jezika. Definira se i objašnjava sam pojam „algoritam“, izvode ključni pojmovi i neka njegova svojstva te navode primjeri istoga. Danas su algoritmi usko vezani uz računalnu inteligenciju, koja se smatra granom umjetne inteligencije. Znanstvenici su uvidjeli prednosti biološke i prirodne inteligencije te načine na koje se one mogu implementirati u računalni sustav. Jedno od područja kojim se bavi računalna inteligencija je evolucijsko računarstvo koje koristi računalne metode evolucijskih procesa. Ukratko se objašnjava i računalna inteligencija te se pobliže objašnjava evolucijsko računarstvo, s obzirom da je ono karika u lancu razvoja genetskih algoritama. Rad nadalje objašnjava genetski algoritam, njegove prednosti i nedostatke. Također, prikazana je i objašnjena njegova struktura, operatori (selekcija, križanje i mutacija) i parametri o kojima ovisi njegov učinak. S tim u svezi, navedene su najpoznatije postavke parametara takvih algoritama prema američkom znanstveniku De Jongu. U radu je prikazan primjer primjene genetskog algoritma. Zaključuje se da su genetski algoritmi u mogućnosti riješiti mnoge probleme, ali da mjesta za napredak i razvoj i dalje ima.

Ključne riječi: algoritam, genetski algoritam, računalna inteligencija, evolucijsko računarstvo

Sadržaj

1. Uvod

Jedan od temeljnih pojmova u računarstvu danas je pojam algoritma. Najjednostavnije rečeno algoritam je uputa kako obaviti neki zadatak. U današnje vrijeme, pojam algoritma usko se veže uz računarstvo, međutim, algoritam je postojao puno prije prvih računala.

Drugo poglavlje rada donosi uvid u povijesni razvoj algoritama. Ukratko se govori o prvom, Euklidovom algoritmu te podrijetlu riječi „algoritam“, a spominje se i prvi računalni program koji je kreirala Ada Byron, prva programerka. Treće poglavlje rada određuje pojam algoritma. Navodi se nekoliko definicija od kojih je najjednostavnija ona da je algoritam uputa kako riješiti neki zadatak. Također spominju se ključni pojmovi u definiciji kao i neka svojstva algoritma: konačnost, definitnost, ulaz, izlaz i efektivnost. U okviru ovog poglavlja prikazani su i jednostavni primjeri algoritma s ciljem boljeg razumijevanja definicije algoritma. Također, navedeni su neki od problema koje algoritmi rješavaju, poput: sortiranja, upravljanja informacijama na internetu, zaštite podataka itd. Budući da algoritmi mogu rješiti različite vrste problema postoji i mnogo vrsta algoritama, a navedeno i objašnjeno je nekoliko (pohlepni, algoritmi sortiranja, kriptografski, optimizacijski). Koliko su algoritmi napredovali dokazuje i činjenica kako su algoritmi, danas, usko vezani uz računalnu inteligenciju, kojoj je posvećeno četvrto poglavlje ovoga rada, a koja se smatra granom umjetne inteligencije, područja koje se proučava posljednjih pedesetak godina, međutim, do danas nije dalo veće rezultate. Mnogi znanstvenici uvidjeli su prednosti biološke i prirodne inteligencije te načine na koje se te inteligencije mogu implementirati u računalni sustav. Računalna inteligencija bavi se mnogim područjima, a jedno od njih je i evolucijsko računarstvo koje koristi računalne metode evolucijskih procesa, a opisano je u sklopu petog poglavlja ovog rada. Također, u okviru petog poglavlja objašnjavaju se evolucijski algoritmi kao podskup evolucijskog računarstva iz kojih su razvijena tri postupka: evolucijsko programiranje, evolucijske strategije te genetski algoritmi. Ukratko je opisan i povijesni razvoj algoritama s posebnim naglaskom na genetske algoritme.

Posljednje poglavlje rada posvećeno je baš genetskim algoritmima. Naime, genetski algoritmi razvijeni su tijekom šezdesetih i sedamdesetih godina, a radi se o heurističkoj metodi optimiranja koja imitira prirodni evolucijski proces. Nad genetskim algoritmima djeluju tri operatora: selekcija, križanje i mutacija, a za rad genetskog algoritma vrlo su važni i parametri. Genetski algoritmi pogodni su za rješavanje mnogih problema uključujući i problem trgovačkog putnika. Na kraju rada zaključuje se o najvažnijim idejama predstavljenima u radu.

2. Povijesni razvoj algoritama

Danas se pojam „algoritam“ uglavnom vezuje uz računarstvo. Međutim, algoritmi su se pojavili puno prije prvih računala. Prvi poznati algoritam, koji se još uvijek koristi, nastao je tri stoljeća prije nove ere, a poznat je kao Euklidov algoritam. Opisao ga je grčki matematičar Euklid u svojoj knjizi „Elementi“.[1]

Riječ algoritam dolazi od imena arapskog matematičara Abu Ja'far Mohammed ibn Musa al Khowarizmi (čiji je nadimak bio Alchwarizmi), iz 9. stoljeća te se on ujedno smatra i ocem algoritma. O njegovu životu zna se vrlo malo. Bio je učenik „Kuće mudrosti“ u Bagdadu te je ondje prevodio Grčke znanstvene tekstove i studirao i pisao o algebri, geometriji i astronomiji.[2] Vjerovao je da se svaki matematički problem može razbiti na manje cjeline i korake te da se na taj način može ubrzati i olakšati njegovo rješavanje.[3] Ovakva ideja pojednostavljenja zadataka, preteča je onoga što danas poznajemo kao algoritam.

Pojam algoritma prvo se koristio samo u matematici, no s pojavom računala, proširuje se i na računalstvo. Prvom osobom koja je algoritam prilagodila za računalo smatra se Ada Byron (kći poznatog pjesnika Lord Byrona). Ada je postala zainteresirana za Charles Babbageovu ideju o računalnom, tj. analitičkom stroju za koji je pretpostavljao kako bi mogao ne samo predviđati, već i reagirati na predviđaj. 1843. godine prevela je članak s francuskog koji je o Babbagovu stroju napisao talijanski matematičar Luigi Federico Manabrea, uz prijevod je napisala i svoje bilješke, koje su, na kraju, bile tri puta duže od samoga članka. U bilješkama su se mogli pronaći i naputci prema kojima bi analitički stroj mogao računati Bernoullijeve brojeve.[4] Ti naputci danas su poznati i kao prvi računalni program, a Ada Byron je danas poznata kao prvi programer. Ministarstvo obrane Sjedinjenih Američkih Država Adi Byron u čast, svoj je programski jezik, razvijen početkom osamdesetih godina prošlog stoljeća nazvalo „Ada“.[5]

Sredinom 20. stoljeća izrađena su prva elektronička računala, samim time rasla je potreba za programskim jezicima te razvijanjem istih (algoritam napisan umjetnim jezikom naziva se program, a umjetni jezik naziva se programskim jezikom). Prva računala programirala su se u

[1] Usp. Divjak, Blaženka; Lovrenčić Alen. Diskretna matematika s teorijom grafova. Varaždin: 2005., str. 125.
[2] Abu Ja'far Muhammad ibn Musa al-Khwarizmi. URL: http://www-history.mcs.st-and.ac.uk/Biographies/Al-Khwarizmi.html (12-8-2014)
[3] Usp. Hoško, T.; Slamić, M. Strukture podataka i algoritmi: Priručnik. Zagreb : Algebra, 2009., str. 8.
[4] Ada Byron, Lady Lovelace. URL: http://www.cs.yale.edu/homes/tap/Files/ada-bio.html (12-8-2014)
[5] Computer Programming languange. // Encyclopaedia Britannica Online. URL: http://www.britannica.com/EBchecked/topic/130670/computer-programming-language/248115/SQL#toc248123 (12-8-2014)

strojnom jeziku. Takvi jezici nazivaju se još i asemblerskim jezicima, budući da su kao pomoć u programiranju razvijeni „asembleri“, odnosno programi koji prevode simboličke naredbe u strojne. Prvi viši programski jezik bio je FORTRAN razvijen 1945. godine. Nedugo nakon njega, u sedamdesetim godinama 20. stoljeća razvijeni su specijalizirani programski jezici. Jedan od njih je i C, jezik razvijen za sistemsko programiranje, koji se i danas koristi u razne svrhe.[6]

3. Definicija algoritama

Postoji mnogo definicija algoritama. Mnogi autori slažu se kako je svaka od tih definicija bitna za razumijevanje algoritama te da svaka definicija, na svoj način, definira algoritam. Razumijevanje definicije ključno je u razumijevanju samog algoritma. Jedna od poznatijih definicija algoritma je da je on „dobro definirana računalna procedura koja uzima neke vrijednosti ili skupove vrijednosti (apstraktne tipove podataka) kao ulaz i isto tako rezultira nekim vrijednostima ili skupovima vrijednosti kao izlazima“[7]. Međutim, najjednostavnije, bilo bi reći kako je algoritam „naputak kako riješiti neki zadatak ili obaviti neki posao“[8]. To jest, algoritme možemo definirati kao alate uz pomoću kojih se rješavaju određeni problemi. Oni su detaljni i precizni opisi potrebnih stvari i postupaka pri rješavanju problema, te cijeli zadatak svode na manje i jednostavnije zadatke.

Prema Tomislavu Hošku i Miroslavu Slamiću (2009.) „algoritam je konačan skup preciznih, razumljivih i jednoznačnih instrukcija koje djelotvorno i učinkovito dovode do rješenja u konačnom vremenu“[9]. Iz ove definicije moguće je istaknuti ključne dijelove koji pomažu pri razumijevanju algoritma. Instrukcije algoritma moraju biti precizne (dakle, instrukcije tipa „dodajte malo ulja“ nisu dobre, trebalo bi napisati točnu količinu te iako se npr. određeni korak u rješavanju nekog problema podrazumijeva, u zapisu algoritma ga je izuzetno bitno navesti). Instrukcije algoritma moraju biti i razumljive (svakome čovjeku, odnosno računalu) te jednoznačne (jedna instrukcija uvijek treba biti shvaćena na isti način). Instrukcije algoritma, odnosno njihov broj, mora biti konačan (one ukazuju na slijed operacija kojim se dolazi do cilja, ukoliko instrukcije nisu konačne, ne možemo doći do cilja, tj. algoritam ne

[6] Usp. Krčadinac, Vedran. Osnove algoritama: predavanja, 2013./2014. URL: http://web.math.pmf.unizg.hr/nastava/oa/oa-skripta.pdf (12-8-2014)
[7] Hoško, T.; Slamić, M. Nav.dj., str. 5.
[8] Grundler, Darko; Blagojević, Lidija. Informatika: udžbenik za 1. razred gimnazije. Zagreb : Školska knjiga, 2005., str. 144.
[9] Hoško, T.; Slamić, M. Nav.dj., str. 9.

rješava problem). Također, algoritam mora biti djelotvoran (mora dati rezultat, postići cilj) te učinkovit (u smislu memorije, vremena itd.). [10]

Prema američkom računalnom znanstveniku, matematičaru i profesoru Donaldu Knuthu, algoritam mora zadovoljiti pet svojstava: konačnost, koja je ranije opisana; definitnost (svaki algoritam mora sadržavati nedvosmislene, rigorozno definirane operacije) koja se može usporediti s prije navedenom preciznošću. Također, svaki algoritam mora imati ulaz (0 ili više ulaza); izlaz (1 ili više izlaza); te mora biti efektivan, odnosno takav da se može izvesti uz pomoć olovke i papira u konačnom vremenu.[11]

3.1. Jednostavni primjeri algoritma

Kako bi se algoritmi što bolje shvatili, najbolje ih je prikazati tj. objasniti primjerima. Jedan od najčešćih i najjednostavnijih primjera algoritma jest kuhanje čaja. Ako se nekome postavi zadatak da skuha čaj za četiri osobe, a ta osoba nikada nije kuhala čaj, može mu se pomoći algoritmom: „U posudu za kuhanje uliti približno litru vode. Zagrijavati vodu dok ne zavri. U vodu staviti četiri vrećice čaja. Prestati zagrijavati. Pričekati deset minuta. Izvaditi vrećice čaja. Uliti čaj u četiri šalice.“[12] Dakle, u ovome slučaju, uz pomoć algoritma, pomoglo se osobi da riješi problem - kuhanje čaja. Preciznim i detaljnim opisima, podijeljenima u više manjih, jednostavnih zadataka. Također, osobe koje su kuhale čaj, podrazumijevaju da se na primjer u vodu, nakon što je prokuhala, stave vrećice čaja, osoba koja nikada nije napravila čaj, to ne zna – dakle važno je da je algoritam precizan te da se ne smije izostaviti niti jedan korak. Iz ovog primjera vidljiva su također i svojstva konačnosti (čaj je skuhan, dakle problem je riješen) i efektivnosti. Ovaj primjer algoritma također ima i skup ulaznih podataka (upute kako napraviti čaj) te izlazni „podatak“, u ovome slučaju čaj. Ovim primjerom prikazano je točno ono što algoritam je: „točno opisana pravila za postizanje željenog rezultata“[13].

Prethodni primjer, primjer je jednostavnog algoritma iz svakodnevnog života. Međutim, algoritmi su se prvo počeli koristiti u matematici, stoga je potrebno prikazati i jednostavan algoritam iz tog područja. Na neke algoritme toliko smo se navikli, da ih koristimo bez da smo svjesni. Jedan primjer takvog algoritma je i zbrajanje višeznamenkastih cijelih brojeva. Zbrajanje

[10]Usp. Isto.
[11] Usp. Divjak, Blaženka; Lovrenčić Alen . Nav. dj., str. 126.
[12] Grundler, Darko; Blagojević, Lidija. Nav. dj., str. 144.
[13] Budin, Leo. Inteligentno izračunavanje. // Poslovno računarstvo / Vlatko Čerić… [et al.]. Zagreb : Znak, 1998., str. 459.

višeznamenkastih brojeva uči se još u osnovnoj školi, a budući da se radi o zbrajanju velikih brojeva, da bi se došlo do rješenja koristi se vrlo jednostavan algoritam. Taj algoritam glasi ovako: „a) ulaz algoritma su dva pribrojnika; b) potpisati brojeve jedan ispod drugog, tako da znamenke odgovarajućih težina budu jedna ispod druge (jedinice ispod jedinica, desetice ispod desetica ...); c) zamisliti broj koji ćemo zvati „pamtim“ i za početak ga postaviti na 0; d) za svako težinsko mjesto počevši od jedinica, napraviti sljedeće: 1) zbrojiti dvije znamenke i broj „pamtim“, a ako jedna od znamenki nedostaje tretirati je kao nula; 2) na odgovarajuće težinsko mjesto rezultata upisati znamenku jedinica tako dobivene sume; 3) ako je ta suma dvoznamenkasta, tada znamenki desetica dodijeliti broju „pamtim“; e) ako je na kraju pamtim veći od 0, onda ga upisujemo na početak rezultata kao znamenku najveće težine; f) izlaz algoritama je zbroj dvaju pribrojnika.“[14] Na konkretnom primjeru, algoritam zbrajanja višeznamenkastih brojeva izgledao bi ovako:

Zbrojimo 2576 i 578

t	s	d	j		
2	5	7	6		
+	5	7	8		pamtim=0
			4	6+8+0=14	pamtim=1
		5	4	7+7+7=15	pamtim=1
	1	5	4	5+5+1=11	pamtim=1
3	1	5	4	2+0+1=3	

Sličan, jednostavni matematički algoritam je onaj za oduzimanje višeznamenkastih cijelih brojeva, koji radi po istome principu.

3.2. Problemi koje algoritmi rješavaju

U prijašnjem poglavlju navedeni su jednostavni primjeri algoritama te jednostavni problemi koje algoritmi rješavaju. Međutim, algoritmi, kao alati uz pomoć kojih se rješavaju problemi, mogu se koristiti u mnoge svrhe, odnosno za rješavanje mnogih problema.

[14] Hoško, T.; Slamić, M. Nav.dj., str. 10.

Promatramo li algoritam kao alat za rješavanje precizno definiranog računalnog problema, jedan od najčešćih problema na koje se nailazi u računalnoj znanosti jesu problemi sortiranja tj. razvrstavanja. Zbog učestalosti problema razvijen je veliki broj sortirajućih algoritama koji su nam na raspolaganju.[15] Naravno, sortiranje nije jedini problem zbog kojeg su algoritmi razvijeni. Praktična primjena algoritama danas je sveprisutna, a uključuje mnogo primjera.

Projekt ljudskog genoma postigao je veliki napredak u identificiranju svih gena u ljudskom DNK, određivanjem nizova od 3 milijarde kemijskih baza koje čine ljudski DNK, pohranjujući te informacije u baze podataka, te razvijajući alate za analizu podataka. Za svaki od ovih koraka potreban je sofisticirani algoritam.[16] Svi ti algoritmi pomažu znanstvenicima u ostvarivanju njihovih zadataka učinkovito koristeći sredstva tj. resurse.

Današnji internet ljudima omogućuje da brzo pristupe i preuzmu velike količine informacija. Uz pomoć pametnih algoritama mrežne stranice su u mogućnosti upravljati i manipulirati, već spomenutom, velikom količinom informacija. Osim za upravljanje informacijama, algoritmi na internetu koriste se i za pronalaženje dobrih ruta na kojima će podaci putovati, a koriste se i u tražilicama kako bi se brzo pronašle mrežne stranice na kojima se nalazi tražena informacija.[17]

Elektroničke trgovine također koriste algoritme. Ovakva usluga ovisi o privatnosti osobnih podataka poput brojeva kreditnih kartica, lozinke, bankovnih izvoda itd. Temeljna tehnologija koja se koristi u elektroničkoj trgovini je „public-key“ kriptografija i digitalni potpisi, koji se temelje na brojevnim tj. numeričkim algoritmima i teoriji brojeva.[18]

3.3. Vrste algoritama

Ranije su navedeni samo neki od problema koje je moguće riješiti algoritmima. Budući da algoritmi pomažu u rješavanju različitih problema, postoje i različiti algoritmi, odnosno vrste algoritama. Postoji mnogo vrsta algoritama, a navest će se samo neke od njih.

Pohlepni algoritmi pripadaju algoritmima za traženje puta. Pohlepni algoritmi, poput ostalih algoritama su pravila, postupci ili proces koji nudi rješenje problema. Pohlepni algoritmi

[15]Usp. Cormen, T. H.; Leiserson, C. E.; Rivest, R. L. Introduction to algorithms. The MIT Press, 2001., str. 5.
[16]Usp. Isto, str. 6.
[17]Usp. Isto.
[18]Usp. Isto, str. 7.

su heuristički algoritmi, što znači da se kod rješavanja problema ne polazi od unaprijed zadanih pravila, već se do rješenja dolazi empirijskim znanjem. Jedan od heurističkih načina rješavanje problema je i metoda pokušaja i pogrešaka.[19] Pohlepni algoritmi uzimaju seriju neposrednih, lokalnih optimuma, u nadi da će, kada proces završi, također postići globalni optimum. Jednostavan primjer pohlepnog algoritma bio bi vraćanje ostatka novca (npr. u trgovini), budući da, obično, svi žele dobiti što manje kovanica. Ako netko treba vratiti 74 centa, a kovanice koje ima pri ruci su one od 25, 10 i 5 centi te 1 centa, osoba koja prima novac dobit će dvije kovanice od 25 centi, dvije kovanice od 10 centi te četiri kovanice od 1 cent. Bilo koje drugo rješenje, osobi koja prima novac, dalo bi veći broj kovanica.[20] Računalne mreže također mogu koristiti pohlepne algoritme kako bi obradile poruke i datoteke koje se šalju između čvorova na mreži. Svakome rubu, koji povezuje dva čvora može se dati važnost/težina koja predstavlja vrijeme potrebno da poruka stigne od jednog čvora do drugog. Spomenuta težina može uzeti u obzir udaljenost, kompoziciju materijala kroz koju poruka putuje, brzinu procesora itd. Različiti putovi koji spajaju dva čvora, mogu se tada usporediti kako bi se odredilo koji je put najkraći.[21]

Ranije su, u kontekstu problema koji algoritmi rješavaju, spomenuti algoritmi sortiranja. Sortiranje je proces uzimanja neuređenog niza zapisa te uređivanje tih zapisa prema nekom specifičnom ciljanom redu.[22] Glavno opravdanje za proces sortiranja jest to da kada se jednom učini, relativno je jednostavno pronaći zapise koristeći brze algoritme. Za algoritme sortiranja vrlo je bitno vrijeme rada koje se mjeri brojem računalnih koraka potrebnih algoritmu da završi sortiranje n-broja zapisa. Neki od najpoznatijih algoritama za sortiranje su „bubble sort“, „selection sort“ te „insertion sort“. Najjednostavniji od navedenih algoritama je „selection sort“. Ako se pretpostavi da je veličina zapisa bitan kriterij sortiranja, „selection sort“ algoritam skenira listu tražeći najmanji zapis te prebacuje poziciju zapisa, zajedno sa zapisom, na prvo mjesto. Nakon toga, skenira preostale n – 1 zapise, pronalazi najmanji te ga prebacuje na drugo mjesto. „Selection sort“ algoritam nastavlja prebacivanje zapisa sve dok svi zapisi nisu poredani rastućim redoslijedom.[23]

U današnje vrijeme internet, odnosno online trgovine postaju sve popularnije (npr. eBay, Amazon). Teško će se pronaći stvar koja se ne može kupiti/naručiti preko interneta – od hrane i

[19]Usp. Heuristic approach // Kiš, Miroslav. Englesko-hrvatski i hrvatsko-engleski informatički rječnik. Zagreb : Naklada Ljevak, 2000., str. 451
[20] Usp. Greedy algorithm. // World of computer science. Detroit: Gale Group, cop. 2002., str. 269.
[21] Usp. Isto.
[22] Usp. Sorting algorithms. // World of computer science. Detroit: Gale Group, cop. 2002., str. 520.
[23] Usp. Isto.

odjeće do računala i mobitela. Kod ovakve, online kupovine, ljudi često plaćaju karticama, stoga je vrlo važna zaštita podataka: broj kartice, pin, korisnička lozinka itd. Da bi se zaštitili privatni podaci korisnika, koriste se kriptografski algoritmi. Kriptografija je sustavno zamjenjivanje i razmještanje grafičkih znakova nekog teksta sa svrhom sakrivanja informacije[24], a znanost koja se bavi sakrivanjem i otkrivanjem informacija zove se kriptologija.[25] Kriptografija se koristi još od vremena prije nove ere, a koristila se prije svega kako bi se od neprijatelja sakrile zapovjedi i planovi. Moderna kriptografija koristi se u razne svrhe od osiguravanja satelitske komunikacije i mobilnih telefona pa sve do potvrde autentifikacije, tj. potvrde da je primljena poruka jednaka onoj poslanoj. Moderni kriptografski sustavi koriste algoritme koji obavljaju transformaciju prema skupu pravila modificiranih od strane ključa. Kriptografski algoritmi koriste se kako bi transformirali podatke (iz njihove normalne forme u neku drugu formu) pod kontrolom ključa[26] (ključ je u ovome slučaju skup simbola kojima se tekst šifrira ili dešifrira[27]).

Uz navedene pohlepne algoritme, algoritme sortiranja te kriptografske algoritme bitno je spomenuti i optimizacijske algoritme. Optimizacija je postupak pronalaženja najpovoljnijeg rješenja, tj. rješenja koje najbolje ispunjava određene uvjete, odnosno postupak pronalaženja rješenja problema u nekom intervalu, pri čemu je željeno rješenje maksimum ili minimum funkcije.[28] Teorija optimizacije označava skup postupaka za pronalaženje rješenja (maksimum ili minimum) u definiranom skupu mogućih rješenja, a uključuje linearno, nelinearno, dinamičko i stohastičko programiranje te neke dijelove kombinatorike.[29] Optimizacijskim algoritmima pripadaju, između ostalih, i evolucijski algoritmi te genetski algoritmi.

4. Računalna inteligencija

Većina znanstvenih disciplina poput biologije, fizike, matematike, kemije dobro su definirane znanstvene discipline. Međutim, to nije slučaj s umjetnom inteligencijom, koja je do današnjeg dana ostala zagonetnom. Iako se ova znanstvena disciplina proučava već 50 godina, još uvijek ne postoji opće prihvaćena definicija umjetne inteligencije.[30] Iz navedenog razloga, ne postojanja definicije umjetne inteligencije, ona je dan danas relativno nejasna disciplina i može

[24] Usp. Cryptography. // Kiš, Miroslav. Nav. dj., str. 253.
[25] Usp. Cryptology. // Kiš, Miroslav. Nav. dj., str. 253.
[26] Usp. Cryptography. // World of computer science. Detroit: Gale Group, cop. 2002., str. 174.
[27] Usp. Key. // Kiš, Miroslav. Nav. dj., str. 538
[28] Usp. Optimisation. // Kiš, Miroslav. Nav. dj., str. 685
[29] Usp. Optimisation theory. // Kiš, Miroslav. Nav. dj., str. 685
[30] Usp. Fogel, B. David. Evolutionary Computation: Toward a New Philosophy of Machine Intelligence. Wiley-IEEE Press, 1995., str. 1.

se reći kako se javlja alternativa ovome izrazu, odnosno znanstvenoj disciplini – računalna inteligencija.

Jedna od definicija računalne inteligencije bila bi da se radi o proučavanju prilagodljivih mehanizama kako bi se omogućilo ili olakšalo inteligentno ponašanje u složenim i promjenjivim okolinama.[31] Veliki udar na razvoj algoritama je dizajniranje algoritamskih modela koji rješavaju rastuće kompleksne probleme. Veliki uspjeh postignut je modeliranjem biološke i prirodne inteligencije, što je rezultiralo tzv. „inteligentnim sustavima“[32], tj. inteligentnim algoritmima, odnosno računalnoj inteligenciji.

Računalna inteligencija uključuje istraživanja o neuronskim mrežama, „fuzzy“ (nejasnim) sustavima te evolucijskom računarstvu, a smatra se granom umjetne inteligencije. Osim navedenih istraživanja u području umjetnih neuronskih mreža, „fuzzy“ sustavima te evolucijskom računarstvu, računalna inteligencija uključuje još i „swarm“ (roj) inteligenciju te umjetni imunološki sustav. Sva ova područja podrijetlom su iz bioloških sustava. Neuronske mreže oponašaju živčani sustav, evolucijsko računarstvo temelji se na evoluciji, roj inteligencija temelji se na društvenom ponašanju organizama koji žive u roju ili kolonijama, umjetni imunološki sustav temelji se na ljudskom imunološkom sustavu, a „fuzzy“ inteligencija vuče korijene iz studija o odnosu organizama i njihova okruženja.[33] Neke od glavnih značajki računalne inteligencije su: prilagodljivost okolišu, robustnost, numerička ili znakovna predodžba znanja, oponašanje postupaka iz prirode, mogućnost dobivanja dobrih približnih rješenja uz prihvatljiv utrošak vremena i sklopovlja, mogućnost rješavanja zadataka teško rješivih deterministčkim postupcima te široko područje primjene.[34]

5. Evolucijsko računarstvo i evolucijski algoritmi

Evoluciju možemo definirati kao optimizacijski proces čiji je cilj poboljšati sposobnost organizma (ili sustava) da preživi u dinamički promjenjivom i konkurentnom okolišu.[35] Evolucijsko računarstvo odnosi se na računalno bazirane sustave rješavanja problema koji

[31] Usp. Engelbrecht, P. Andries. Computational intelligence: An introduction. Second Edition. Chichester: John Wiley & Sons Ltd, cop. 2007., str. 4.
[32] Usp. Isto, str. 3
[33] Usp. Isto, str. 5
[34] Grundler, Darko. Evolucijski algoritmi (I): pobude i načela. // Automatika 42, 1/2 (2001.), str. 13-22., str. 16
[35] Usp. Engelbrecht, P. Andries. Nav. dj., str. 127.

koriste računalne modele evolucijskih procesa kao temeljne dijelove. Radi se o evolucijskim procesima poput prirodne selekcije, opstanka najjačih te reprodukcije.[36]

Evolucijski algoritmi su podskup evolucijskog računarstva. To su računalni programi koji pokušavaju riješiti kompleksne probleme oponašajući proces evolucije.[37] Evolucijski algoritmi nastali su prvenstveno iz dva razloga: želja za boljim razumijevanjem prirodne evolucije te pokušaja primjene načela prirodne evolucije pri rješavanju različitih zadaća.[38] Princip rada evolucijskog algoritma najjednostavnije se može objasniti ovako: u evolucijskom algoritmu broj umjetnih stvorenja (jedinki) pretražuju problemski prostor, neprestano se međusobno natječu kako bi otkrili optimalna područja prostora koji pretražuju, u nadi da će se tijekom vremena najuspješnije od stvorenja razviti i otkriti optimalno rješenje.[39]

Evolucijski algoritmi spadaju u šire područje znanosti o spoznaji te u uže područje inteligentnih algoritama, odnosno računalne inteligencije. Iz evolucijskih algoritama razvijena su tri postupka zasnovana na načelima biološke evolucije: evolucijsko programiranje, evolucijske strategije te genetski algoritmi.[40]

5.1. Povijesni razvoj evolucijskih algoritama

Proučavanje evolucijskih sustava započelo je u pedesetim i šezdesetim godinama prošlog stoljeća. Nekoliko računalnih znanstvenika, zasebno su počeli proučavati evolucijske sustave s idejom da bi se evolucija mogla koristiti kao optimizirajući alat za tehničke, inženjerske probleme. Ideja u svim sustavima je bila razviti populaciju mogućih rješenja određenog problema, koristeći operatore inspirirane prirodnim genetskim varijacijama i prirodnom selekcijom.[41]

U šezdesetim godinama Rechtenberg je predstavio evolucijske strategije, metodu koju je koristio za optimizaciju stvarnih vrijednosti parametara za uređaje kao što su aeroprofilna krila. Ovu ideju kasnije je dalje razvijao Schwefel.[42] Evolucijske strategije imaju mnogo zajedničkih

[36] Usp. Isto, str. 128.
[37] Jones, Gareth. Genetic and Evolutionary algorithms. URL: http://www.wiley.com/legacy/wileychi/ecc/samples/sample10.pdf (21-8-2014)
[38] Usp. Grundler, Darko. Nav. dj. str. 13.
[39] Jones, Gareth. Nav. dj. URL: http://www.wiley.com/legacy/wileychi/ecc/samples/sample10.pdf (21-8-2014)
[40] Usp. Grundler, Darko. Nav. dj., str. 14.
[41] Usp. Mitchell, Melanie. An introduction to Genetic Algorithms. Cambridge, Massachusetts: The MIT Press, cop. 1998., str. 2.
[42] Usp. Isto.

osobina s genetskim algoritmima te im se često, teško određuju granice. Obje metode odražavaju populaciju rješenja nad kojima se provode operacije koje se periodički ponavljaju.[43] Pa, iako su ove metode izuzetno slične, donedavno, su se razvijale potpuno neovisno jedna o drugoj.[44]

U ranim šezdesetim godinama, L. J. Fogel je predstavio alternativni pristup umjetnoj inteligenciji – evolucijsko programiranje. Umjetna inteligencija u šezdesetim godinama bila je koncentrirana na modelima ljudske inteligencije. Fogel je inteligenciju promatrao kao svojstvo koje omogućuje sustavu da prilagodi svoje ponašanje kako bi zadovoljio željene ciljeve u različitim okruženjima. Stoga je evolucijsko programiranje razvijeno kao model koji imitira bihevioralne osobine.[45] Imitiranje bihevioralnih osobina je ujedno i glavna razlika između evolucijskog programiranja i evolucijskih strategija te genetskih algoritama, budući da evolucijske strategije i genetski algoritmi stavljaju naglasak na genetske modele, dok evolucijsko programiranje naglašava modele ponašanja (iako sve tri metode za glavni cilj imaju imitiranje prirodnih evolucijskih procesa).

Genetske algoritme osmislio je John Holland, također u šezdesetim godinama, a razvio ih je zajedno sa svojim studentima na Sveučilištu Michigan u šezdesetim i sedamdesetim godinama prošlog stoljeća. Hollandov izvorni cilj nije bio dizajnirati algoritme za rješavanje specifičnih problema, već proučiti fenomen prilagodbe kao što se događa u prirodi i razviti načine na koji bi mehanizmi prirodne prilagodbe mogli biti implementirani u računalne sustave.[46] Hollandova knjiga „Adaptation in Natural and Artificial Systems“ („Adaptacija u prirodnim i umjetnim sustavima“) predstavila je genetski algoritam kao apstrakciju prirodne biološke evolucije te je dala teoretski okvir za adaptaciju pod genetskim algoritmom. Hollandov genetski algoritam je metoda pomicanja od jedne populacije kromosoma (nizova jedinica i nula) do nove populacije koristeći prirodnu selekciju zajedno s genetski inspiriranim operatorima – križanja, mutacije i inverzije.[47] Svaki kromosom sastoji se od gena (npr. bitova), svaki gen je instanca određene alele (npr. 0 ili 1). Selekcijski operator odabire one kromosome u populaciji kojima će kasnije biti dopušteno razmnožavanje, u prosjeku sposobniji kromosomi daju više potomaka. Križanje razmjenjuje poddijelove dva kromosoma, grubo oponašajući biološku rekombinaciju (kod genetskih algoritama križanje i rekombinacija su obično jedno te isto) kod haploidnih

[43] Usp. Golub, Marin. Genetski algoritmi: prvi dio. 2010. URL: http://www.zemris.fer.hr/~golub/ga/ga_skripta1.pdf (23-8-2014)
[44] Mitchell, Melanie. Nav. dj., str. 2.
[45] Usp. Engelbrecht, P. Andries. Nav. dj. str. 187.
[46] Usp. Mitchell, Melanie. Nav. dj., str. 3.
[47] Usp. Isto, str. 3.

organizama; mutacija nasumično mijenja vrijednosti alela na nekim mjestima u kromosomu, dok inverzija obrće redoslijed graničnog dijela kromosoma, čime se mijenja redoslijed kojim su raspoređeni geni.[48]

U posljednjih petnaestak godina zabilježen je značajan razvoj genetskih algoritama. Genetski algoritam se primjenjuje i daje dobre rezultate u području učenja kod neuronskih mreža, pri traženju najkraćeg puta, problemu trgovačkog putnika, strategiji igara, problemima sličnim transportnom problemu itd.[49] Pojam genetskog algoritma danas, koristi se kako bi se opisalo nešto, što je vrlo daleko od Hollandovog originalnog koncepta.

6. Genetski algoritmi

Genetski algoritam je heuristička metoda optimiranja koja imitira prirodni evolucijski proces.[50] Kao što je ranije navedeno, ideja za genetske algoritme dolazi od procesa evolucije. Evolucija kao prirodni proces traženja najbolje i najprilagodljivije jedinke na okolinu i uvjete u prirodi je zapravo metoda optimiranja.[51] U prirodi, najčešće, preživljavaju one jedinke koje se bolje prilagode snalaženju u okolini s oskudnim životnim sredstvima. One jedinke koje se pokažu najsposobnije u teškim životnim uvjetima dobivaju priliku za dominacijom nad manje sposobnim jedinkama, a također se i reproduciraju. Ako sposobnost pojedinca da preživi poistovjetimo s nasljednim materijalom koji nosi u sebi (genima), tada se može reći kako geni sposobnijih pojedinaca opstaju, dok geni slabijih izumiru, jer nemaju potomstva. Kod svake reprodukcije dolazi i do rekombinacije gena zbog koje dolazi do različitosti među pojedincima iste vrste, ali i sličnosti s roditeljima jedinke. Promatraju li se geni, može se reći kako se u svakoj novoj generaciji dobiva novi skup gena. Naravno, neki pojedinci mogu biti lošiji od prethodne generacije, a neki bolji. Izmjena gena koja nastaje reprodukcijom se u genetskim algoritmima naziva križanjem. Osim križanja u evoluciji postoji još jedna pojava koja se naziva mutacija, koja nastaje mijenjanjem genetskog materijala djelovanjem vanjskih uzroka. Navedeni pojmovi, križanje i mutacija, se u genetskim algoritmima nazivaju i genetskim operatorima, dok se izdvajanje najsposobnijih jedinki unutar generacije naziva odabirom, to jest, selekcijom.[52] Genetski algoritam oponaša evolucijski proces, za koji se može ustanoviti sljedeće: „ a) postoji populacija jedinki; b) neke jedinke su bolje; c) bolje jedinke imaju veću vjerojatnost

[48] Usp, Isto, str. 3.
[49] Golub, Marin. Nav. dj. URL: http://www.zemris.fer.hr/~golub/ga/ga_skripta1.pdf (23-8-2014) str. 4
[50] Isto, str. 8.
[51] Isto.
[52] Usp. Isto.

preživljavanja i reprodukcije; d) svojstva jedinki zapisana su u kromosomima s pomoću genetskog koda; e) djeca nasljeđuju svojstva roditelja; f) nad jedinkom može djelovati mutacija.“[53] Kod genetskog algoritma jedinke predstavljaju moguća rješenja (npr. plan rada tvornice, najkraći put...) te se jedinke nazivaju kromosomima. Okolina predstavlja funkciju cilja (funkcija koja određuje mjeru kvalitete, dobrote). Želi li se genetski algoritam implementirati kao metoda optimiranja, potrebno je definirati proces dekodiranja i preslikavanja podataka zapisanih u kromosomu te se mora odrediti funkcija dobrote.[54]

6. 1. Prednosti i nedostatci genetskog algoritma

Genetski algoritmi su tijekom posljednjih nekoliko godina postali izuzetno popularni, a tome je više razloga. Osim što su primjenjivi na velik broj problema, velika prednost genetskog algoritma je to što su oni intrizično paralelni. Većina drugih algoritama je serijska te mogu istražiti problemski prostor samo u jednome smjeru te ako je pronađeno rješenje ispod optimalnog, napuštaju sav prethodno obavljeni posao te kreću ispočetka. Genetski algoritmi, međutim, imaju više potomaka koji pretražuju problemski prostor u više smjerova odjednom. Ako se jedan put pokaže kao slijepa ulica, „potomci“ taj put eliminiraju te nastavljaju s radom, a eliminacija im daje veću vjerojatnost da će sljedećim pretraživanjem pronaći optimalno rješenje.[55] Također, u slučaju da genetski algoritam ne nađe globalni optimum i dalje daje dobro rješenje koje može zadovoljiti. Osim toga, genetski algoritam kao rezultat ne daje jedno rješenje, već skup rješenja.[56] Još jedna od prednosti genetskih algoritama je i to što mogu podnijeti problemska ograničenja, tako što ih jednostavno ugrade u kod kromosoma te su genetski algoritmi vrlo jednostavna tehnika s vrlo malo matematike.[57]

Iako genetski algoritmi imaju mnogo prednosti, postoji mnogo ograničenja, odnosno nedostataka genetskih algoritama. Prva i najvažnija stvar koja se mora uzeti u obzir pri kreiranju genetskog algoritma je definiranje prikaza rješenja. Jezik koji se koristi kako bi se specificiralo rješenje mora biti snažan, u suprotnome dolazi do nezadovoljavajućih i nedosljednih rezultata. Također, problem pisanja funkcije dobrote mora se pažljivo razmotriti tako da je veća funkcija

[53] Isto, str.9
[54] Usp. Isto.
[55] Genetic Algorithms and Evolutionary Computation. URL: http://www.talkorigins.org/faqs/genalg/genalg.html#history (23-8-2014)
[56] Usp.Golub, Marin. Nav. Dj. URL: http://www.zemris.fer.hr/~golub/ga/ga_skripta1.pdf (23-8-2014) str. 32.
[57] Usp. Man, K.F.; Tang, K.S.; Kwong, S. Genetic algorithms: Concepts and Application. // IEEE Transactions on Industrial Electronics 43, 5 (1996), str. 519-534., URL: ftp://vm1-dca.fee.unicamp.br/pub/docs/vonzuben/ia707_1s06/textos/00538609.pdf (23-8-2014)

dobrote dostižna te da se ravna prema boljem rješenju za određeni problem. U suprotnome, genetski algoritam može biti u nemogućnosti pronaći rješenje ili može riješiti pogrešan problem.[58] Jedan od problema s kojima se genetski algoritmi teško nose je i problem „obmane“, odnosno varljivih funkcija dobrote. Varljive funkcije dobrote bile bi lokacije poboljšanih točaka koje daju pogrešne informacije o tome gdje bi se mogao pronaći lokalni optimum.[59] Međutim, većina problema s kojima se susreće u radu s genetskih algoritmima su rješiva.

6. 2. Struktura genetskog algoritma

Slika 1. prikazuje strukturu genetskog algoritma, tj. pseudokod genetskog algoritma.

```
Genetski_algoritam
{
   t = 0
   generiraj početnu populaciju potencijalnih rješenja P(0);
   sve dok nije zadovoljen uvjet završetka evolucijskog procesa
   {
        t = t + 1;
        selektiraj P'(t) iz P(t-1);
        križaj jedinke iz P'(t) i djecu spremi u P(t);
        mutiraj jedinke iz P(t);
   }
   ispiši rješenje;
}
```

Slika 1. Struktura genetskog algoritma[60]

Prilikom uvođenja generira se početna populacija jedinki. Obično se početna populacija jedinki generira slučajnim odabirom rješenja. Međutim, moguće je početnu populaciju generirati i uniformno (ravnomjerno), međutim, u početku evolucijskog procesa sve jedinke su iste, stoga genetski algoritam nije učinkovit te se taj postupak ne preporuča. U početnu populaciju se također može usaditi početno rješenje dobiveno nekom drugom optimizacijskom metodom. Nakon generiranja početne populacije slijedi proces koji se ponavlja sve do isteka vremena ili dok se ne zadovolji određeni uvjet. Proces se sastoji od djelovanja genetskih operatora (selekcija, križanje i mutacija) nad populacijom jedinki. Tijekom selekcije loše jedinke odumiru, a bolje opstaju. Nakon selekcije slijedi križanje u kojemu se bolje jedinke

[58] Genetic Algorithms and Evolutionary Computation. URL: http://www.talkorigins.org/faqs/genalg/genalg.html#history (23-8-2014)
[59] Isto.
[60] Usp.Golub, Marin. Nav. Dj. URL: http://www.zemris.fer.hr/~golub/ga/ga_skripta1.pdf (23-8-2014) str. 9.

razmnožavaju te se križanjem svojstva prenose s roditelja na djecu. Nakon križanja, slijedi mutacija kojom se mijenjaju svojstva jedinke slučajnom promjenom gena. Takvim postupkom se postiže iz generacije u generaciju sve veća i veća prosječna dobrota populacije.[61]

6. 3. Genetski operatori

Selekcija je prvi genetski operator koji djeluje pri evolucijskom procesu. Uloga selekcije je prenošenje dobrih svojstava na sljedeću generaciju jedinki. Selekcijom se odabiru dobre jedinke koje sudjeluju pri sljedećem koraku. Selekcijom se dobri geni prenose na sljedeću generaciju, dok loši geni odumiru. Genetski algoritmi, s obzirom na vrstu selekcije mogu se podijeliti na generacijske i eliminacijske, a karakteristične vrste selekcije koje koristi generacijski genetski algoritam su jednostavna i turnirska selekcija. Jednostavna selekcija generira novu populaciju P'(t) i P(t-1) koja ima jednak broj jedinki kao i populacija P(t-1). Odnosno, VEL_POP(P'(t))=VEL_POP(P(t-1)), VEL_POP označava veličinu populacije, dok t označava redni broj generacije. Cilj jednostavne selekcije je odabir roditelja čija je vjerojatnost selekcije proporcionalna njihovoj dobroti.[62] Turnirska selekcija u svakom koraku generira novu populaciju iz stare populacije, na način da VEL_POP puta odabire s jednakom vjerojatnošću k jedinki iz stare populacije, uspoređuje ih i najbolju jedinku kopira u bazen za reprodukciju na kojem će u sljedećem koraku djelovati ostali genetski operatori.[63] Generacijski genetski algoritmi nepotrebno udvostručavaju populaciju, jer u jednom koraku raspolažu s dvije populacije jedinki. Jedna populacija jedinki je populacija iz prijašnjeg koraka, dok se druga generira selekcijom. Populacija iz prijašnjeg koraka briše se tek nakon što selekcija generira novu populaciju. Nepotrebno udvostručavanje populacije može se izbjeći eliminacijskom selekcijom, tj. reprodukcijom. Eliminacijska selekcija, za razliku od jednostavne, ne bira dobre kromosome, već loše, koje treba eliminirati te reprodukcijom zamijeniti novima. Dakle, loši kromosomi umiru, a njih nadomještaju djeca nastala reprodukcijom dobrih kromosoma. Algoritam eliminacijske selekcije prikazan na Slici 2. glasio bi: a) selektiraj m loših kromosoma; b) izbriši selektirane kromosome; c) genetskim operatorima generiraj nove kromosome; d) dodaj nove kromosome.[64] Eliminacijski algoritam zapravo je vrlo sličan jednostavnoj selekciji, međutim postoje dvije bitne razlike. Vjerojatnost selekcije nije proporcionalna funkciji dobrote, već funkciji kazne (što manja dobrota). Na taj način riješen je problem očuvanja kromosoma s

[61] Usp. Golub, Marin. Nav. dj. URL: http://www.zemris.fer.hr/~golub/ga/ga_skripta1.pdf (23-8-2014) str. 9.
[62] Usp. Isto, str. 12.
[63] Usp. Isto. str. 13.
[64] Usp. Isto, str. 14.

najvećom dobrotom, budući da je njihova kazna jednaka nuli, pa je vjerojatnost eliminiranja najbolje jedinke, nula. Također, jednom odabrani kromosom za eliminaciju, ne može se više odabrati. [65]

```
Eliminacijski genetski algoritam{
   generiraj početnu populaciju;
   sve dok nije zadovoljen uvjet završetka evolucijskog procesa{
      izračunaj vjerojatnost eliminacije za svaku jedinku;
      M puta jednostavnom selekcijom odaberi i izbriši jedinku;
      parenjem preživjelih jedinki nadopuni populaciju;
   }
}
```

Slika 2. Struktura genetskog algoritma s eliminacijskom reprodukcijom[66]

Križanje je sljedeći genetski operator koji sudjeluje u procesu. U procesu križanja sudjeluju jedinke, koje se još nazivaju i roditeljima, a križanjem nastaju jedinke koje se nazivaju djeca. Najvažnija karakteristika križanja je da djeca nasljeđuju svojstva roditelja, dakle, ako je roditelj prošao selekciju dijete će biti dobro, a moguće i bolje od roditelja.[67] Postoje tri glavne klase križanja, ovisno o broju roditelja. Aseksualno (jedan roditelj, jedno dijete), seksualno (dva roditelja, jedno ili dva djeteta) te multi-rekombinacija (više od dva roditelja, te jedno ili više djece).[68] Ako postoje dva roditelja te jedno ili dvoje djece tada se većinom govori o binarnom operatoru križanja. Križanje može biti definirano s proizvoljnim brojem prekidnih točaka, stoga postoji križanje s jednom točkom, križanje s dvije točke te uniformno križanje. Križanje s jednom točkom razvijeno je da nasumično odabere točku križanja te se nizovi bitova iza te točke zamjenjuju između dva roditelja. Kod križanja s dvije točke, nasumično se biraju dvije pozicije, a nizovi bitova zamjenjuju se između te dvije točke, odnosno pozicije.[69] Uniformno križanje je križanje s b-1 prekidnih točaka (b predstavlja bitove). Također, kod križanja se vrlo često mora kreirati, tj. zadati maska. Maska se zadaje ako je vjerojatnost nasljeđivanja različita za pojedine gene, maska definira vjerojatnost nasljeđivanja za svaki gen posebno. Ono što je bitno za operator križanja je to da se pretpostavlja kako je upravo taj operator ono što razlikuje genetske algoritme od ostalih metoda optimiranja.[70]

[65] Isto
[66] Isto
[67] Usp. Isto, str. 15.
[68] Usp. Engelbrecht, P. Andries. Nav. dj. str. 144.
[69] Usp. Isto, str. 145.
[70] Usp. Golub, Marin. Nav. Dj. URL: http://www.zemris.fer.hr/~golub/ga/ga_skripta1.pdf (23-8-2014) str. 15

Sljedeći i posljednji operator genetskog algoritma je mutacija. Cilj mutacije je uvesti novi genetski materijal u postojeću jedinku, odnosno dodati različitost u genetske karakteristike populacije.[71] Mutacija je unarni operator budući da djeluje nad samo jednom jedinkom, a rezultat mutacije je izmijenjena jedinka. Postoji nekoliko vrsta mutacije: jednostavna mutacija, miješajuća mutacija, potpuna miješajuća mutacija te invertirajuća miješana mutacija. Jednostavna mutacija svaki bit kromosoma mijenja s jednakom vjerojatnošću, dok miješajuća mutacija slučajnim postupkom odabire kromosom za mutaciju, prvu i drugu granicu i tada ili izmiješa gene ili ih slučajno generira (potpuna miješajuća mutacija), ili ih invertira (invertirajuća miješana mutacija).[72] Mutacija se može prikazati formulom: $P_M = 1-(1-p_m)^n$, p_m je u ovome slučaju vjerojatnost mutacije jednog gena, bita, a P_M je vjerojatnost mutacije kromosoma, dok je n broj bitova koji čine kromosom. Uzme li se za primjer da je vrijednost $p_m=0.01$, to znači da će jedan od sto bitova biti promijenjen, ako je kromosom predstavljen nizom od n=32 bita, tada je $P_M=0.275$, što znači da će 27 kromosoma biti promijenjeno. Mutacijom se pretražuje prostor rješenja i upravo je mutacija mehanizam uz pomoć kojeg se izbjegava lokalni minimum, jer ako cijela populacija završi u lokalnom minimumu, jedino se slučajnim pretraživanjem prostora može doći do boljeg rješenja.[73]

6. 4. Parametri genetskog algoritma

Učinak algoritma uvelike ovisi i o parametrima. Bez obzira o kakvom se genetskom algoritmu radi on ima sljedeće parametre: veličina populacije, broj generacija odnosno broj iteracija (ponavljanja) i vjerojatnost mutacije. Za generacijski genetski algoritam potrebna je još i vjerojatnost križanja, dok se kod eliminacijskog genetskog algoritma zadaje broj jedinki za eliminaciju. Za različite vrijednosti parametara algoritam daje različite rezultate, tj. može brže ili sporije doći do boljeg ili lošijeg rješenja.[74] Postoji velika rasprava o postavkama i pristupima parametrima, koja do sada nije donijela najbolji rezultat, stoga većina ljudi koristi parametre koji su bili dobri u nekim prijašnjim slučajevima.[75] Jedna od najpoznatijih postavki parametara je ona De Jonga. De Jongovi eksperimenti pokazali su kako je najbolja veličina populacije ona od 50-100 jedinki, najbolja vrijednost za križanje s jednom točkom 0.6 po paru roditelja, a najbolja

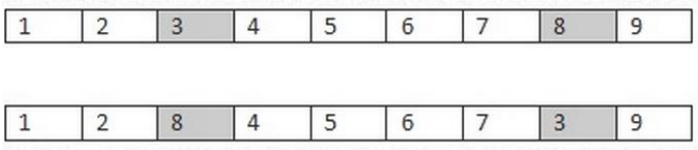
[71] Usp. Engelbrecht, P. Andries. Nav. dj. str. 153
[72] Usp. Golub, Marin. Nav. Dj. URL: http://www.zemris.fer.hr/~golub/ga/ga_skripta1.pdf (23-8-2014) str. 16
[73] Usp Isto.
[74] Usp. Isto.
[75] Mitchell, Melanie. Nav. dj., Str. 131.

vrijednost mutacije 0.001 po bitu. Ovi parametri jedni su od najraširenijih kod korištenja genetskog algoritma.[76]

6. 5. Genetski algoritam i problem trgovačkog putnika

Problem trgovačkog putnika jedan je od najsloženijih problema kombinatorne optimizacije. Problem trgovačkog putnika je da ima zadan broj gradova, od kojih svaki mora posjetiti točno jednom te se vratiti u početni grad najkraćim mogućim putem. Problem trgovačkog putnika pripada NP-teškom problemu, to su problemi za čije rješavanje ne postoji „pametni“ algoritam.[77]

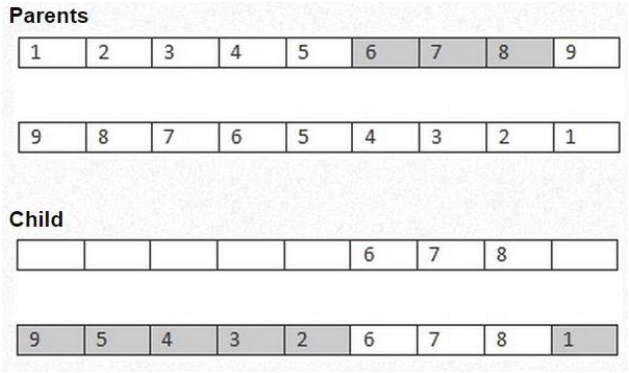
Kod problema trgovačkog putnika genetski algoritam je vrlo dobra metoda koja može pomoći pronaći rješenje. Genetski algoritam obično daje optimalna rješenja, a ako i ne daje, rješenja su i dalje dovoljno dobra, odnosno s malim odstupanjima. Pronalaženje rješenja za problem trgovačkog putnika zahtjeva postavljanje genetskog algoritma na specijalizirani način. Na primjer, valjano rješenje za ovaj problem trebalo bi predstaviti rutu u kojoj je jedan grad uključen barem jednom i samo jednom. To znači da operator mutacije treba prilagoditi na način da ne dodaje nasumični grad u rutu i moguće uzrokuje duplikacije. Mutacija koja se može koristiti kod ovog problema je miješajuća mutacija, na način da se nasumično odaberu dva objekta i mutacijom ih se samo zamijeni (Slika 3.).[78]



Slika 3. Mutacija korištena za problem trgovačkog putnika[79]

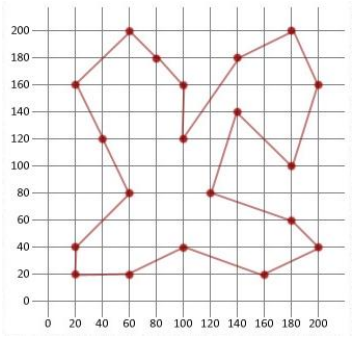
Operator križanja se također treba izmijeniti. Križanje koje se može koristiti je redno križanje u kojemu se odabire podskup prvog roditelja i dodaje djetetu. Nakon toga djetetu se dodaju objekti drugog roditelja u točnom redu (Slika 4.). [80]

[76] Usp. Mitchell, Melanie. Nav. dj., str. 131.
[77] Usp. Hoško, T.; Slamić, M. Nav.dj., str. 12
[78] The Project spot: Applying a genetic algorithm to the traveling salesman problem. URL: http://www.theprojectspot.com/tutorial-post/applying-a-genetic-algorithm-to-the-travelling-salesman-problem/5 (31-8-2014)
[79] Isto.
[80] Isto.



Slika 4. Redno križanje[81]

Kod kreiranja koda (Java) prvi korak je kreirati klasu koja može kodirati ture gradova, nakon toga kreira se klasa koja sadrži sve destinacije gradova u turi. Sljedeći korak je klasa koja kodira rute. Budući da se radi o genetskom algoritmima potrebno je kreirati i klasu koja sadrži populaciju mogućih tura (Vidi Prilog 1, Slika 6.). Sljedeća klasa treba razviti populaciju rješenja, a nakon toga može se razviti glavna metoda, dodati gradovi i razviti tura. [82] Rješenje je prikazano na Slici 5.



Slika 5. Problem trgovačkog putnika – rezultat[83]

[81] Isto
[82] Isto.
[83] Isto.

7. Zaključak

Genetski algoritmi moćan su alat za rješavanje problema iz različitih područja te su obećavajuća metoda za rješavanje teških tehnoloških problema. Genetski algoritmi prikladni su za rješavanje najrazličitijih problema upravo zato što je ideja za genetski algoritam došla od prirodne evolucije. Živa bića se tijekom evolucije prilagođavaju prirodi, a genetski algoritam možemo prilagoditi problemu ili se problem može prilagoditi genetskom algoritmu. Po pitanju rješenja koje genetski algoritmi pružaju, oni možda neće uvijek pronaći najbolje rješenje, ali će pronaći rješenje dovoljno blizu dobrome, s vrlo malim odstupanjima. Evolucijsko računarstvo, samim time i genetski algoritmi, još uvijek je daleko od toga da bude afirmirana znanost, međutim, treba uzeti u obzir kako se ne radi o području koje se proučava već stoljećima i ima iza sebe mnoštvo utvrđenih teorija i saznanja. Evolucijsko računarstvo proučava se tek četrdesetak godina, a već sada su vidljivi obećavajući rezultati. Prirodna evolucija može biti neiscrpan izvor ideja, pogotovo uzme li se u obzir da npr. još uvijek nisu kartirani svi geni. Dakle, evolucijsko računarstvo i genetski algoritmi obećavajuće su metode, međutim, mjesta za napredak i dalje ima.

Prilog 1

```java
/*
* Population.java
* Manages a population of candidate tours
*/

package tsp;

public class Population {

    // Holds population of tours
    Tour[] tours;

    // Construct a population
    public Population(int populationSize, boolean initialise) {
        tours = new Tour[populationSize];
        // If we need to initialise a population of tours do so
        if (initialise) {
            // Loop and create individuals
            for (int i = 0; i < populationSize(); i++) {
                Tour newTour = new Tour();
                newTour.generateIndividual();
                saveTour(i, newTour);
            }
        }
    }

    // Saves a tour
    public void saveTour(int index, Tour tour) {
        tours[index] = tour;
    }

    // Gets a tour from population
    public Tour getTour(int index) {
        return tours[index];
    }

    // Gets the best tour in the population
    public Tour getFittest() {
        Tour fittest = tours[0];
        // Loop through individuals to find fittest
        for (int i = 1; i < populationSize(); i++) {
            if (fittest.getFitness() <= getTour(i).getFitness()) {
                fittest = getTour(i);
            }
        }
        return fittest;
    }

    // Gets population size
    public int populationSize() {
        return tours.length;
    }
}
```

Slika 6. Kreiranje klase koja sadrži populaciju mogućih tura (kod u programskom jeziku Java)[84]

[84] The Project spot: Applying a genetic algorithm to the traveling salesman problem. URL: http://www.theprojectspot.com/tutorial-post/applying-a-genetic-algorithm-to-the-travelling-salesman-problem/5 (31-8-2014)

8. Literatura

1. Abu Ja'far Muhammad ibn Musa al-Khwarizmi. URL: http://www-history.mcs.st-and.ac.uk/Biographies/Al-Khwarizmi.html (12-8-2014)

2. Ada Byron, Lady Lovelace. URL: http://www.cs.yale.edu/homes/tap/Files/ada-bio.html (12-8-2014)

3. Budin, Leo. Inteligentno izračunavanje. // Poslovno računarstvo / Vlatko Čerić… [et al.]. Zagreb : Znak, 1998.

4. Computer Programming languange. // Encyclopaedia Britannica Online. URL: http://www.britannica.com/EBchecked/topic/130670/computer-programming-language/248115/SQL#toc248123 (12-8-2014)

5. Cormen, T. H.; Leiserson, C. E.; Rivest, R. L. Introduction to algorithms. The MIT Press, 2001.

6. Divjak, Blaženka; Lovrenčić Alen. Diskretna matematika s teorijom grafova. Varaždin: 2005.

7. Engelbrecht, P. Andries. Computational intelligence: An introduction. Second Edition. Chichester: John Wiley & Sons Ltd, cop. 2007.

8. Fogel, B. David. Evolutionary Computation: Toward a New Philosophy of Machine Intelligence. Wiley-IEEE Press, 1995.

9. Golub, Marin. Genetski algoritmi: prvi dio. 2010. URL: http://www.zemris.fer.hr/~golub/ga/ga_skripta1.pdf (23-8-2014)

10. Grundler, Darko; Blagojević, Lidija. Informatika: udžbenik za 1. razred gimnazije. Zagreb : Školska knjiga, 2005.

11. Genetic Algorithms and Evolutionary Computation. URL: http://www.talkorigins.org/faqs/genalg/genalg.html#history (23-8-2014)

12. Grundler, Darko. Evolucijski algoritmi (I): pobude i načela. // Automatika 42, 1/2 (2001.), str. 13-22.

13. Hoško, T.; Slamić, M. Strukture podataka i algoritmi: Priručnik. Zagreb : Algebra, 2009.

14. Jones, Gareth. Genetic and Evolutionary algorithms. URL: http://www.wiley.com/legacy/wileychi/ecc/samples/sample10.pdf (21-8-2014)

15. Kiš, Miroslav. Englesko-hrvatski i hrvatsko-engleski informatički rječnik. Zagreb : Naklada Ljevak, 2000.

16. Krčadinac, Vedran. Osnove algoritama: predavanja, 2013./2014. URL: http://web.math.pmf.unizg.hr/nastava/oa/oa-skripta.pdf (12-8-2014)

17. Man, K.F.; Tang, K.S.; Kwong, S. Genetic algorithms: Concepts and Application. // IEEE Transactions on Industrial Electronics 43, 5 (1996), str. 519-534. URL: ftp://vm1-dca.fee.unicamp.br/pub/docs/vonzuben/ia707_1s06/textos/00538609.pdf (23-8-2014)

18. Mitchell, Melanie. An introduction to Genetic Algorithms. Cambridge, Massachusetts: The MIT Press, cop. 1998.

19. The Project spot: Applying a genetic algorithm to the traveling salesman problem. URL: http://www.theprojectspot.com/tutorial-post/applying-a-genetic-algorithm-to-the-travelling-salesman-problem/5 (31-8-2014)

20. World of computer science. Detroit: Gale Group, cop. 2002.