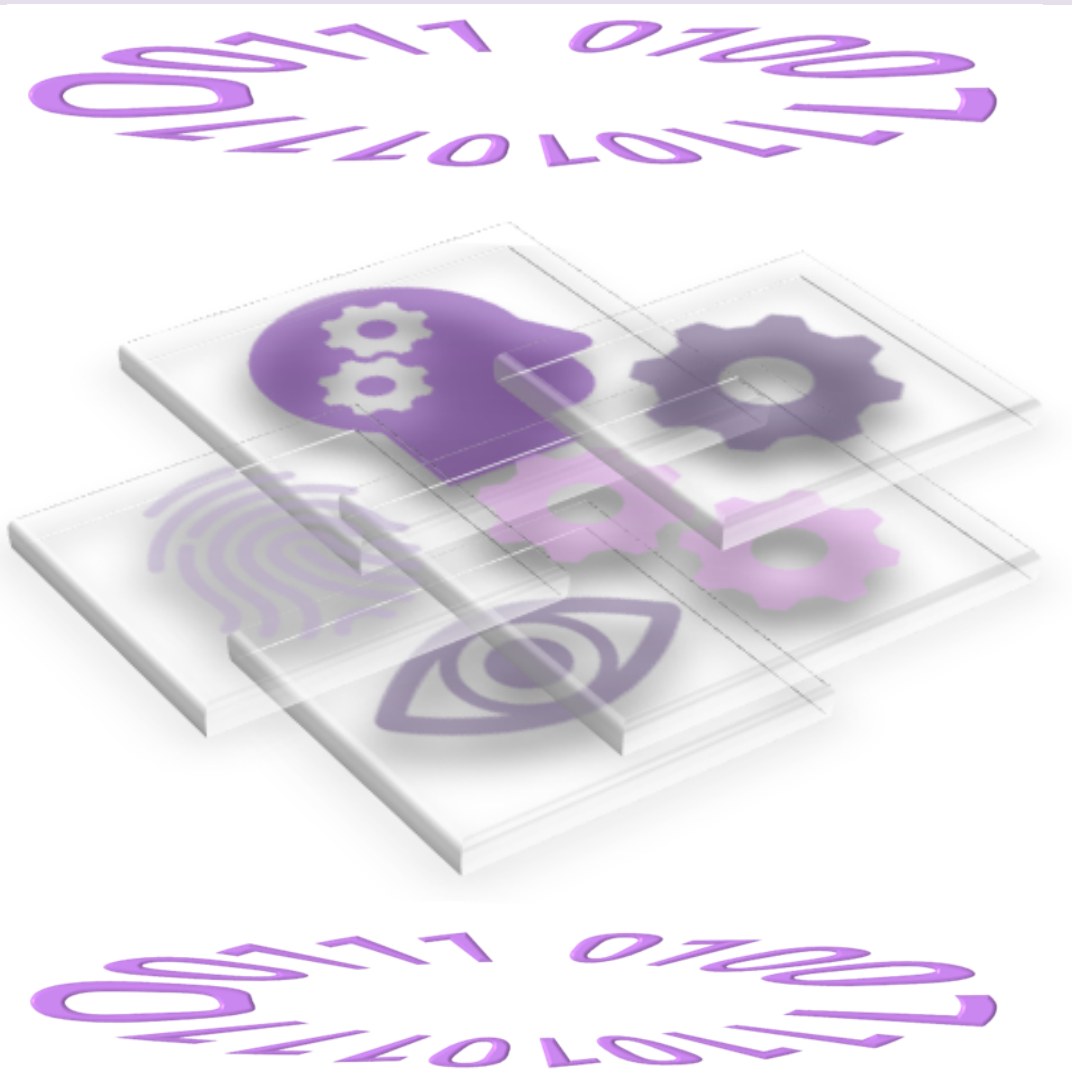
AGENȚIA NAȚIONALĂ PENTRU CERCETARE ȘI DEZVOLTARE
MINISTERUL EDUCAȚIEI ȘI CERCETĂRII

Liubomir Chiriac, Eugenia Chiriac,
Natalia Lupașco, Maria Pavel

Itinerar Elementar în Inteligența Artificială și Algoritmii Genetici



Chișinău 2023

CZU: 519(07)

Studiu realizat în cadrul proiectului de cercetări științifice „Metodologia implementării TIC în procesul de studiere a științelor reale în sistemul de educație din Republica Moldova din perspectiva inter/transdisciplinarității (concept STEAM)”, inclus în „Program de stat” (2020-2023), Prioritatea IV: Provocări societale, cifrul 20.80009.0807.20
Coordonator de proiect: Liubomir Chiriac, dr. habilitat, profesor universitar

Lucrarea a fost aprobată de către Senatul
Universității Pedagogice de Stat „Ion Creangă” din Chișinău

Recenzenți:
Inga Țițchiev, dr., conferențiar universitar, IMI
Teodora Vascan, dr., conferențiar universitar, UPSC

DESCRIEREA CIP A CAMEREI NAȚIONALE A CĂRȚII DIN REPUBLICA MOLDOVA

Itinerar elementar în inteligența artificială și algoritmii genetici / Liubomir Chiriac, Eugenia Chiriac, Natalia Lupașco, Maria Pavel ; coordonator de proiect: Liubomir Chiriac ; Agenția Națională pentru Cercetare şi Dezvoltare, Ministerul Educației şi Cercetării. – Chişinău : [S. n.], 2023 (CEP UPSC). – [109] p. : fig., fot., tab. color.

Bibliogr.: p. 108-109. – [100] ex.

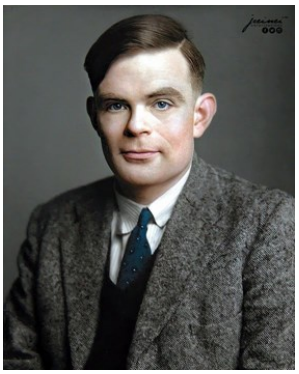
ISBN 978-9975-46-818-3.
004.89+519(075)
I-58

Tiparul: CEP al Universității Pedagogice de Stat „Ion Creangă”

CUPRINS

INTRODUCERE
FORMULA INTELIGENȚEI: PROVOCĂRI ȘI TENDINȚE

Omenirea întotdeauna a încercat să producă instrumente, dispozitive, tehnologii care, fiind puse în aplicație, imitau inteligența umană. Formula inteligenței umane a fost, este și va rămâne în continuare un mare mister, o mare provocare pentru acei care încearcă să o substituie prin intermediul inteligenței artificiale (IA). Chiar dacă încă nu există criterii clare care ar face distincție intre inteligența umană și cea artificială, omul face eforturi colosale pentru a dezvolta domeniul inteligenței artificiale și al pune în serviciu omenirii. Teama că Inteligența Artificială își va subordona Inteligența Umană este discutată tot mai des de cele mai ilustre minți ale omenirii. Chiar dacă Inteligența Artificială se dezvoltă cu o viteză uimitoare, în procesul de predare-învățare a informaticii se iscă o serie de întrebări și probleme privind dezvoltarea domeniului la care nu se poate da răspunsuri univoce.



Alan Turing (1912-1954)

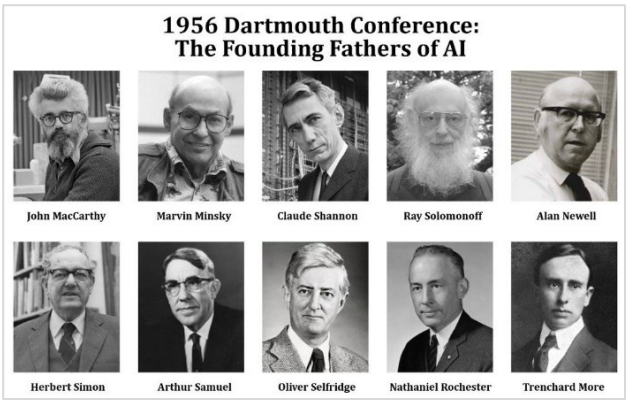
Ar fi oportun în acest context să menționăm de geniala convenție a lui Alan Turing, precum că:

„Nu ar trebui să decidem dacă o mașină poate „gândi"; trebuie doar să clarificăm dacă o mașină poate funcționa inteligent ca o ființă umană".

Această abordare, asociată cu inteligența artificială, constituie baza testului Turing.

Din această perspectivă în cadrul Conferinței de la Dartmouth, din anul 1956, s-a înaintat următorul deziderat:

„Fiecare aspect al învățării sau orice altă caracteristică a inteligenței poate fi atât de exact descrisă încât poate fi construită o mașină pentru a-l simula".

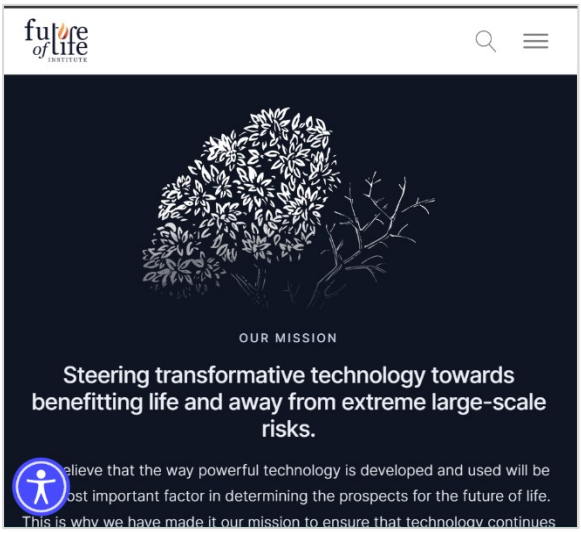


Chiar dacă în prezent sunt unii experți din domeniul IA care nu agreează abordarea respectivă, conceptul menționat reprezintă, la moment, poziția multor altor cercetători în domeniul IA.

Faptul că inteligența artificială generală (IAG – eng. „Artificial General Intelligence" - AGI), în conformitate cu opinia unor experți în domeniul IA, poate genera, în anumite circumstanțe, un risc existențial pentru umanitate, conducând la o catastrofă umanitară, rămâne a fi o ipoteză reală.

Se consideră că specia umană domină în prezent alte specii, deoarece creierul depășește prin puterea inteligenței umane, nivelul inteligenței ființelor respective. În contextul dat, dacă IA va depăși umanitatea în inteligență și va deveni „superinteligentă", atunci umanitatea nu ar fi în stare că controleze IAG (Inteligența Artificială Puternică). Din această perspectivă soarta umanității ar putea depinde de IAG, exact la fel cum în prezent, de exemplu, soarta gorilei de munte (*Gorilla beringei beringei*) depinde de acțiunile și inteligența speciei umane, altfel spus de bunăvoința umană.

Anticipând derularea acestor scenarii, în SUA a fost constituit Institutul Viitorului Vieții (*The Future of Life Institute*, https://futureoflife.org/) ca să efectueze cercetări privind înțelegerea procesului decizional în domeniul IA.



Scopul institutului este ***„să crească competențele și înțelepciunea cu care gestionăm"*** puterea tot mai mare a tehnologiilor moderne.

Dezvoltarea Inteligenței Artificiale se bazează pe rezultatele și conceptele unor ramuri ale științelor, precum: informatică, matematică, inginerie informațională, psihologie, lingvistică, filozofie, robotică, algoritmi genetici etc.

În această lucrare, sunt examinate și unele aspecte care țin de calculul evolutiv, în mod special ***algoritmii genetici***.

Abordarea privind aplicarea principiilor evolutive (calculul evolutiv) în soluționarea automată a problemelor (*Problem Solving*) datează cu mult timp mai înainte comparativ cu apariția și dezvoltarea calculatoarelor moderne.

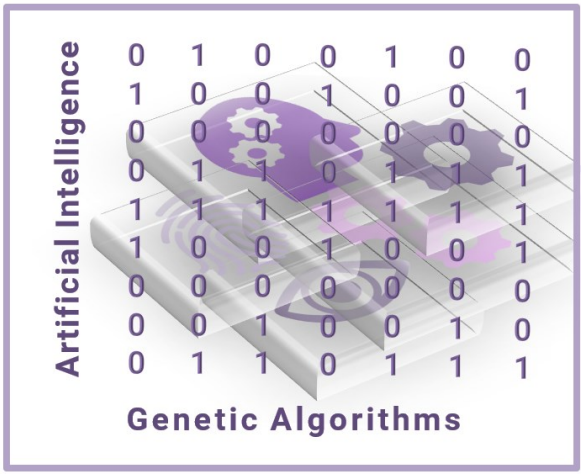
Alan Turing, încă în anul 1948, lansează o abordarea nouă, aplicată la soluționarea unor tipuri de probleme numită abordare evolutivă ori genetică. Așa

dar, calculul evolutiv este un domeniu al informaticii moderne, cu accent puternic în matematică, inspirat din procesul evoluției naturale, iar conceptul de bază care ține de calculul evolutiv este interconexiunea evoluția naturală – tehnica de rezolvare a problemelor de tip experiment-eroare.

În acest sens, în prezent, un domeniu important de cercetare al informaticii îl reprezintă calculul evolutiv. După cum se știe, acest domeniu își „trage seva", adică își are rădăcinile în procesul evoluției naturale. Algoritmii care apar și se dezvoltă în acest domeniu se numesc *algoritmi evolutivi* și ei includ subdomenii importante de mare perspectivă precum: *Programarea evolutivă; Strategii evolutive; Programare genetică; Algoritmi genetici*.

În această lucrare se vor examina, în mod special, algoritmii genetici.

Algoritmul genetic este o metodă care se referă la rezolvarea problemelor de optimizare și care se bazează pe selecția naturală, încrucișări și mutații, proces ce repetă evoluția biologică.



Algoritmul genetic modifică în mod repetat o populație de soluții individuale. Referitor la algoritmul genetic, pe parcursul acestei lucrări, vom examina în profunzime mai multe aspecte care țin de dezvoltarea și aplicarea algoritmului.

În genere, ținem să menționăm că un algoritm evolutiv este considerat o componentă a calculului evolutiv în inteligența artificială. Un algoritm evolutiv funcționează prin intermediul procesului de selecție în care sunt excluși membrii cei mai puțin adaptați (cei mai puțin în formă) din grupul de populație examinat, în timp ce membrii apți supraviețuiesc și sunt luați în considerație până când sunt determinate soluții mai bune. Cu alte cuvinte, algoritmii evolutivi sunt aplicații computerizate care imită procesele biologice pentru a rezolva probleme complexe. În timp, membrii de succes evoluează pentru a prezenta soluția optimizată a problemei. Algoritmii evolutivi utilizează concepte în biologie, cum ar fi selecția, reproducerea și mutația.

Un algoritm evolutiv are o populație de soluții candidate, spre deosebire de metodele clasice, care încearcă să mențină o singură soluție cât mai bună. Există numeroase beneficii asociate cu algoritmi evolutivi. Unul dintre cele mai mari avantaje vine în câștigurile de flexibilitate, deoarece majoritatea conceptelor de

algoritm evolutiv sunt adaptabile chiar și la probleme complexe. Există câteva dezavantaje asociate algoritmilor evolutivi. Un dezavantaj în acest sens, ține de faptul că soluția oferită de un algoritm evolutiv este mai bună doar în comparație cu alte soluții cunoscute. Ca atare, algoritmul nu poate dovedi că nici o soluție este total optimă, doar că este optimă în comparație cu celelalte rezultate.

În contextul celor punctate mai sus, autorii dezvoltă următoarele 6 capitole:

Capitolul 1. *Paradigme moderne în dezvoltarea și învățarea IA*

Capitolul 2. *Abordări istorico-didactice în studierea IA*

Capitolul 3. *Paradigme actuale în procesul de dezvoltare a roboților de chat*

Capitolul 4. *Dezvoltarea Algoritmilor Genetici*

Capitolul 5. *Implementarea Algoritmilor Genetici*

Capitolul 6. *Aplicații practice ale Algoritmului Genetic*

În cadrul fiecărui capitol se face o sinteză analitică, tratând cele mai semnificative teorii, concepte și idei care au contribuit la dezvoltarea unor ramuri care țin de Inteligența Artificială și Algoritmii genetici și, totodată, se scot în evidență conexiunile interdisciplinare ce au influențat pozitiv ascensiunea Inteligenței Artificiale și Algoritmilor Genetici, domenii de mare perspectivă ce captivează și atrage în prezent tot mai mulți tineri talentați.

În secolul nostru conceptele și tehnicile de IA au înregistrat un progres semnificativ în urma creșterii puterii de calcul a calculatoarelor moderne și posibilității prelucrării unor acumulări de cantități mari de date. Astfel, tehnicile IA au devenit o parte esențială a industriei tehnologice, despre care se discută în lucrare, contribuind astfel la rezolvarea multor probleme dificile din domeniul informaticii, matematicii, roboticii și ingineriei software.

Autorii speră că această lucrare va fi de bun augur pentru elevi, studenți, profesori și toți cei interesați de dezvoltarea Inteligenței Artificiale.

Liubomir Chiriac,
Doctor habilitat, profesor universitar

CAPITOLUL 1.
PARADIGME MODERNE ÎN DEZVOLTAREA ȘI ÎNVĂȚAREA INTELIGENȚEI ARTIFICIALE

În acest capitol sunt examinate diverse aspecte care țin de conceptele de bază ale Inteligenței Artificiale (IA), se studiază unele aspecte privind realizarea cercetărilor în domeniu IA din perspectiva Testului Turing, cât și metodele de învățare ale IA. Sunt punctate avantajele și dezavantajele IA comparativ cu Inteligența Umană.

1.1. Conceptul de Inteligență Artificială (IA). De ce este importantă IA?

Elemente care se referă la tehnologiile de IA există în lume mai bine 50 de ani. Însă odată cu creșterea puterii de calcul a calculatoarelor, posibilitatea de a prelucra cantități impresionante de date și elaborarea unor algoritmi noi de ultima generație au condus la progrese importante în domeniul IA. În prezent IA este considerată un domeniu prioritar al transformării digitale a societății și este o prioritate pentru lumea modernă [1-8].

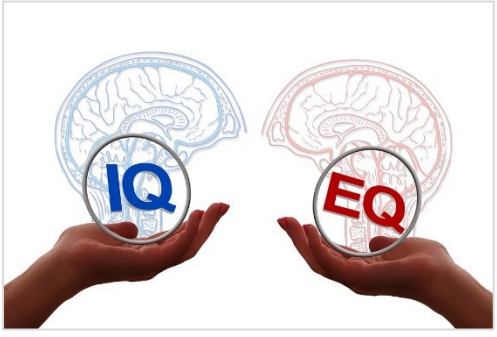
În acest sens, au fost multe încercări de a răspunde la întrebarea:

Ce este Inteligența Artificială (IA)? Dificultatea cea mai mare vine de la faptul, credem noi, că nu există o definiție foarte exactă referitor la Inteligența naturală care, ulterior, prin analogie ar putea fi definită și IA. Așa dar, să examinăm mai detaliat aspectele menționate.

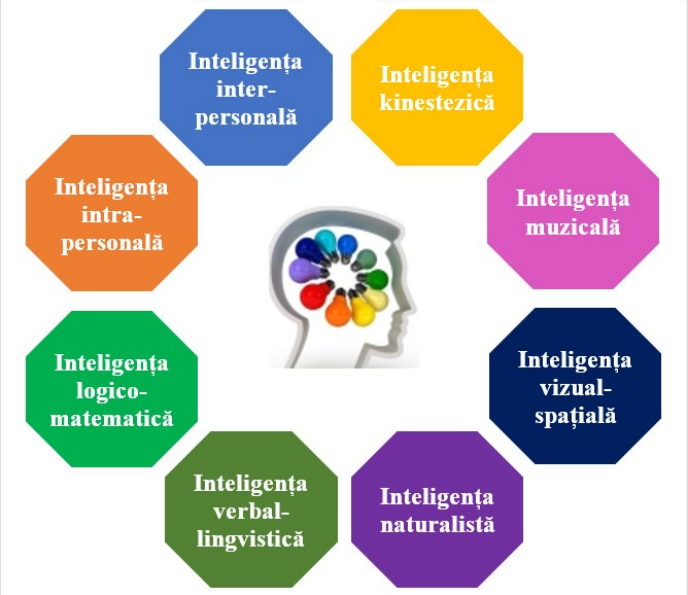
Conform dicționarului explicativ, inteligența este

„capacitatea de a înțelege ușor și bine, de a sesiza ceea ce este esențial, de a rezolva situații sau probleme noi pe baza experienței acumulate anterior”

(https://dexonline.ro/definitie/inteligenta).



În conformitate cu clasificările actuale, care vin din psihologia modernă, există următoarele tipuri de inteligență:



Cele 8 inteligențe ale lui Gardner

1. Inteligența vizual-spațială

Caracteristici: Pasiune pentru scris si citit; Înclinație către rezolvarea puzzle-urilor; Recunoașterea anumitor tipare; Talent la desen sau arte vizuale.

2. Inteligența lingvistică

Caracteristici: Memorare si reproducere a informației orale sau scrise; Pasiune pentru citit si scris; Talent la dezbateri sau discursuri persuasive; Utilizare a umorului in istorisire.

3. Inteligența logic-matematică

Caracteristici: Capacitate sporită de rezolvare a problemelor; Inclinație către gândirea abstractă; Plăcere pentru experimente științifice; Talent la rezolvarea problemelor complexe.

4. Inteligența corporal-chinestezică

Caracteristici: Talent la dans și sport; Pasiune pentru confecționarea obiectelor cu propriile mâini; Coordonare fizică sporită; Capacitate de a memora lucruri prin acțiune.

5. Inteligența muzicală

Caracteristici: Pasiune pentru cântat si instrumente muzicale; Recunoaștere facilă a tiparelor muzicale și a tonurilor; Memorare și reproducere a melodiilor; Înțelegere sporită a structurii muzicale, a ritmului și a notelor.

6. Inteligența interpersonală

Caracteristici: Capacitate superioară de comunicare verbală; Talent la comunicarea non-verbală; Viziune din multiple perspective a situațiilor; Crearea unor relații sociale pozitive; Abilități la evaluarea emoțiilor, a motivațiilor, dorințelor și intențiilor celor din jurul lor; Talent la armonizarea conflictului in diverse grupuri.

7. Inteligența intrapersonală

Caracteristici: Talent la analizarea propriilor slăbiciuni și a punctelor tari; Pasiune pentru analizarea teoriilor și ideilor; Conștientizarea propriilor sentimente și gânduri; Înțelegerea clara a propriilor motivații; Înclinație către auto-reflecție și analiză.

8. Inteligența naturalistă

Caracteristici: *Inclinație către domenii precum botanica, zoologie, biologie; Talent pentru observarea detaliilor; Abilități pentru clasificarea și catalogarea informației; Pasiune pentru grădinărit, camping si explorarea mediului.*

În contextul celor punctate mai sus, cercetătorii în domeniu au evidențiat următoarele caracteristici importante pentru orice tip de inteligență:

- inteligența este măsurabilă;
- inteligența se manifestă ca rapiditate în învățare;
- inteligența se leagă de randament.

Aceste caracteristici pot fi atribuite, din anumite perspective, și IA.

Poate oare inteligența artificială să posede la un nivel foarte avansat una din cele 8 tipuri de inteligențe umane punctate? Să ne imaginăm un robot care posedă Inteligență logic-matematică și este un superb maestru la jocul de șah. Pe bune, un astfel de robot a fost construit în premieră în anul 1996 de către Compania IBM și care purta numele de Deep Blue.

Tot în anul 1996 campionul mondial de șah Garry Kasparov a jucat la Philadelphia primul meci împotriva supercomputer-ului ***IBM Deep Blue***, pe care la câștigat cu scorul 4-2. Meciul revanșă a fost jucat în New York City în 1997 și a fost câștigat de Deep Blue cu scorul 3½–2½.

Aceasta a fost prima înfrângere a unui campion mondial de șah de către un computer, demonstrând superioritatea IA.

Înseamnă oare că Inteligența Artificială depășește deja Inteligența Umană? Se schimbă interpretările despre conceptul IA? Să încercăm să înțelegem cum evaluează conceptul de IA.

Inteligența Artificială este un concept mai mult decât complex aflat într-o evoluție continuă. IA nu înseamnă nici pe de parte replicarea și transformarea minții umane într-un chip. IA, în opinia noastră, ar însemna mai degrabă înțelegerea și replicarea experienței umane vis-a-vis de responsabilitatea interacțiunilor dintre om-om și om-natură. Inteligența artificială este studiul procesului prin care sistemele tehnice, inclusiv calculatoarele, pot fi „învățate” să facă lucruri care,

pentru moment, sunt realizate mai bine de oameni. Bine, dar cum totuși poate fi definită IA?

Marvin Lee Minsky (1927-2016)

Celebrul informatician american Marvin Lee Minsky (9 august 1927 – 24 ianuarie 2016) unul din cercetătorii pionieri în domeniul inteligenței artificiale, co-fondator al laboratorului de Inteligență Artificială al Institutului de Tehnologie din Massachusetts, întrebat ce este IA, a răspuns:

„Există întotdeauna persoane care au nevoie să definească totul pentru a realiza ceva. De ce?"

Mai jos vom încerca totuși să identificăm o definiție relevantă pentru IA.

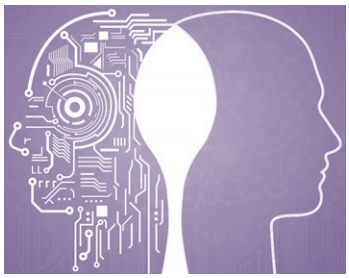
1.2. Ce este Inteligența Artificială?

Se consideră că Inteligența Artificială este un domeniu al informaticii care dezvoltă sisteme tehnice capabile să rezolve probleme dificile legate de inteligența umană.

În termeni tehnici, ***inteligenta artificială este o ramură a tehnologiei*** care permite programarea și proiectarea atât a sistemelor hardware cât și a celor care permit echiparea unor tehnologii cu anumite ***caracteristici care sunt considerate tipic umane***.

Oricum, în prezent, prin

Inteligență Artificială se înțelege capacitatea unei mașini de a imita funcții umane, cum ar fi raționamentul, învățarea, planificarea și creativitatea [7].



Sistemele tehnice dotate cu IA interacționează prin intermediul propriilor senzori cu mediul înconjurător și colectează informația primită, o prelucrează și, analizând datele recepționate, reacționează adaptându-și compartimentul și acționând autonom.

Ca orice știință, Inteligența Artificială studiază o serie de probleme cu caracteristici generale comune și dezvoltă tehnici specifice de rezolvare a acestor probleme.

Inteligența artificială versus inteligența naturală

Valoarea potențiala a IA poate fi înțeleasa mai bine dacă punem în contrast inteligența artificială și inteligența naturală, umană.

Tabelul 1. Avantajele și dezavantajele Inteligenței Artificiale comparativ cu Inteligența Umană Naturală

Avantajele Inteligenței Artificiale comparativ cu Inteligența Umană Naturală	
Inteligența artificială	***Inteligența naturală***
Este permanentă (informația procesată nu poate fi pierdută).	Inteligența umană naturală este perisabilă (unii angajați pot uita informația).
Poate fi repartizată, promovată și difuzată prin intermediul rețelelor ori calculatoarelor.	Transferul cunoștințelor de la o persoană fizică la altele este un proces al cunoașterii și necesită timp.
Poate presta o serie de servicii mai ieftine ca preț comparativ cu inteligența naturală. De exemplu, vinderea biletelor on line de către robot este un serviciu mai ieftin comparativ cu munca umană.	În schimb traducerea unui text în engleză prin intermediul serviciilor unui translator poate fi mai scump decât cu ajutorul unui program specializat.
IA, fiind o tehnologie computerizată, este consecventă și organizată.	Inteligența naturală este dez-organizată și nu întotdeauna consecventă.
IA poate soluționa mai multe sarcini și mai repede comparativ cu omul.	Inteligența umană este mai lentă în îndeplinirea anumitor sarcini, de exemplu calculul numeric ori calculul modular al numerelor mari.
Dezavantajele IA comparativ cu Inteligența Umană Naturală	
IA este, în general, este mai puțin inspirată și cu mare dificultate, ori chiar deloc, nu poate găsi soluții neordinare la probleme dificile.	Inteligența naturală este creativă și poate identifica soluții neordinare la probleme non standard.
IA nu este autodidactă. Cunoașterea pentru IA este construită cu mare atenție de experții în domeniu Machine Learning.	Omul este autodidact. Abilitatea omului de a căpăta cunoștințe este înnăscută pentru ființele umane.
IA este în afara oricăror emoții cu excepția cazurilor când emoțiile sunt exprimate prin algoritmii elaborați.	Inteligența umană nu poate să se dezvolte în afara spațiului emoțional și al culturii general-umane.

Așa dar, cum măsurăm, identificăm inteligența unui program ori a unei mașini?

1.3. Testul Turing

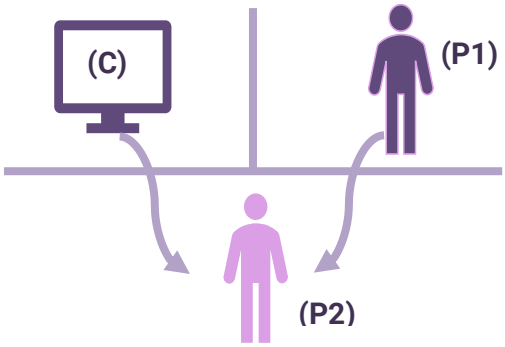
Unul dintre cele mai cunoscute teste privind determinarea gradului de inteligență a unui program este *testul Turing*. În anul 1950, matematicianul englez Alan Turing (1912-1954), a propus, în premieră, un test pentru a determina dacă o mașină (calculator) poate avea sau nu un comportament inteligent.

Turing a pornit de la o idee firească - dacă nu știm să definim în termeni preciși inteligența, însă putem zice despre om că este inteligent, atunci am putea să spunem și despre o altă „creatură" același lucru, în cazul în care s-ar comporta la fel ca o ființă umană. Să clarificăm care din aspectele comportamentului omenesc sunt într-adevăr relevante pentru inteligență.

În celebrul său articol *„Computing Machinery and Intelligence"*, publicat în anul 1950, Turing a lansat în premieră problema posibilității simulării ființei umane cu ajutorul calculatorului. În același articol, Turing descrie și testul care se reduce la determinarea de către Om a unui comportament inteligent și unul neinteligent în raport cu calculatorul dotat cu IA.

Testul Turing este un „joc al imitației" lansat de Turing, care presupune implicarea a 3 actori: o mașină (robot ori calculator) dotat cu IA (C) și două persoane (P1) și (P2). Este important de menționat că pentru acești actori se îndeplinesc următoarele condiții:

- (C), (P1) și (P2) se află în oficii diferite.
- (C), (P1) și (P2) comunică doar în scris prin intermediul terminalelor (calculatoarelor) folosind limbajul natural.
- (P2) nu știe care dintre ceilalți doi actori este calculator și care este om.
- (P2) nu știe și nu poate să îi vadă sau să le vorbească direct.



Conform testului Turing, obiectivul persoanei (P2) este să deosebească calculatorul (C) de omul (P1), pe baza răspunsurilor la orice fel de întrebări recepționate la terminal. Dacă (P2) nu reușește să deosebească calculatorul de Om, după răspunsurile recepționate atunci comportamentul calculatorului (C) poate fi considerată inteligent. În caz contrar comportamentul nu este inteligent.

Problema cea mai semnificativă în procesul de elaborare a unui astfel de program se referă la faptul că nu se cunoaște cantitatea de informație necesară care trebuie „încorporată” în program ca ulterior, calculatorului (C) să fie capabil să treacă testul Turing.

Programul Superinteligent de pe calculator trebuie să imite comportamentul uman, luând în considerare spectrul enorm de larg al întrebărilor posibile de adresat, cât și faptul că acestea sunt puse în limbaj natural. Ceva de neimaginat! Oricum, se crede că se va ajunge și la acest moment.

În anul 1950 când a fost publicat testul Turing, autorul prognoza că aproximativ în anul 2000 va exista un calculator capabil să aibă un comportament inteligent, încât ar avea șansa de 30% să păcălească persoana (P2) pentru 5 minute.

În prezent experții în domeniul IA se împart în două tabere în raport cu prognoza respectivă. Unii consideră că în viitorul apropiat se va inventa calculatorul care va trece testul Turing și alții care sunt convinși de contrariu.

Ca să încheiem acest subiect, trebuie să subliniem faptul că testul Turing, care este de un șarm intelectual deosebit, simbolizează idealul pe termen lung al inteligenței artificiale ca ramură a informaticii. Până la acest moment nici o mașină (robot) nu a trecut testul Turing.

Turing considera că cea mai bună cale spre inventarea unei astfel de mașini, care să treacă testul Turing nu este programarea unui calculator dotat cu o mulțime fixă de cunoștințe, ci, mai degrabă, elaborarea unei mașini-autodidacte, capabile să învețe din propria experiență și să utilizeze limbajul natural ca să-și îmbogățească cunoștințele.

Calculatorul-autodidact superinteligent ar putea să-și rezolve propriile probleme și să-și realizeze propriile sale planuri, demonstrând inteligența practică în viața de zi cu zi.

Astfel, ideile lui Turing referitor la caracteristicile mașinii-autodidacte superinteligente, s-au constituit și dezvoltat, ulterior, în subdomenii ale inteligenței artificiale:

- ***Reprezentarea cunoștințelor*** *(Rețele semantice; Sisteme bazate pe logică; Rețele neuronale).*
- ***Prelucrarea limbajului natural scris*** *(Handwriting recognition. Înțelegerea limbajului scris; Sisteme de traducere automată; Sisteme de extragere a informațiilor din texte; Sisteme de rezumare de texte; Sisteme pentru clasificat documente).*

- ***Raționamente*** *(Demonstrarea automată a teoremelor; Web semantic; Raționament ce implică timp și spațiu).*
- ***Recunoașterea vorbirii*** *(Speech recognition. Generarea vorbirii, inclusiv generarea de voci cu emoții; Înțelegerea vorbirii, inclusiv cu accent pe înțelegerea vocilor izolate; înțelegerea vocilor în medii cu zgomot, urmărirea unui vorbitor într- un grup).*
- ***Identificarea și interpretarea imaginilor*** *(Face identification and recognition. Recunoașterea formelor, persoanelor în imagini; Recunoașterea similarității între obiecte identificate în imagini; Segmentarea imaginilor (în părți semnificative), împachetarea imaginilor; Indexarea imaginilor).*
- ***Interpretarea secvențelor video*** *(Recunoașterea persoanelor/formelor într-o secvență, Urmărirea personajelor în imagini).*
- ***Robotica*** *(Sisteme de articulație, de echilibru etc.).*
- ***Învățare*** *(Supervizată, Nesupervizată, Hibridă, Învățare cu algoritmi genetici, Metode inspirate din natură, Metode statistice, Mașini pe vectori suport (PSO, ACO, Calcul cuantic, Data Meaning).*

În alt context, se poate afirma că testul Turing „impune” ca inteligența mașinilor să modeleze inteligența umană. Unii experți în domeniul IA susțin că inteligența mașinilor este o formă diferită de inteligența umană și că, de fapt, este o eroare să tindem a o evalua în termenii inteligenței umane. Se vrea oare construirea unei mașini capabile să simuleze activitatea socială a unui om și care să fie la fel de lentă în raționamente ca și omul? Toate aceste întrebări rămân pentru moment fără un răspuns clar.

1.4. Cum poate învăța Inteligența Artificială?

Așa dar, care sunt metodele de învățare ale inteligenței artificiale? Evidențiem următoarele:

Machine Learning (învățarea mecanică) care reprezintă o parte importantă a sistemelor IA prin care calculatoarele primesc o cantitate enormă de informații pe care o stochează, analizează, memorează și pe baza cărora reacționează. În acest sens un exemplu concret sunt asistenții virtuali care pot recunoaște comenzile vocale, cum ar fi, bunăoară, *Siri* de la *Iphone Apple* ori *Alexa* de la *Amazon*.

Deep Learning (învățarea profundă) reprezintă o metodă de învățare mecanică, dar mult mai profundă, prin care calculatoarele sunt învățate să se comporte ca un om, adică să învețe din experiențe. De exemplu automobilele SMART de ultima generație care circulă singure și pot recunoaște indicatoarele de circulație și care fac diferența între diverse obiecte – autoturisme, piloni, zebre, semafoare, inclusiv pietoni. Prin intermediul învățării profunde, calculatoarele pot să învețe direct din texte, imagini, sunete, etc. De exemplu, Google și Amazon folosesc metode de *Deep Learning* pentru dezvoltarea algoritmilor de recunoaștere a imaginilor în scopul efectuării analizelor preferințelor utilizatorilor în funcție de alegerile și reacțiile lor în raport cu anumite informații ori produse și, ulterior, în a oferi diverse variante pe baza analizelor respective.

Neural networks (rețelele neurale) – reprezintă esența Inteligenței Artificiale, fără de care nu s-ar produce niciun proces de învățare. *Rețelele neurale*, inspirate de rețelele de neuroni din creierul uman, sunt interconectate prin algoritmi. Creierul uman poate executa activități complexe, în primul rând datorită faptului, că are sute de miliarde de neuroni legați între ei prin sinapse. Chiar dacă aproape este imposibil de replicat rețelele de neuroni datorită complexității incredibile a lor din creierul uman, totuși rețelele de neuroni reprezintă cel mai bun obiect de studiu pentru oamenii de știință, în încercarea lor de a dezvolta rețelele neurale ale IA.

Chiar dacă rețelele neurale avansează continuu va trece încă mult timp până când IA va demonstra o inteligență comparabilă cu cea umană. Și, evident, va trece și mai mult timp până când IA va fi în stare să înțeleagă dorințele și gândurile altor oameni și pe care să le folosească pentru a înțelege și a interpreta comportamentul uman de care vorbisem mai sus. Este clar că va trece și mai mult timp până când IA va avea abilitatea de a fi autodidactă, recunoscându-se ca individualitate. Înainte de a putea programa o mașină să gândească precum mintea umană este absolut necesar să înțelegem cum funcționează rațiunea umană de fapt și abia ulterior să putem măcar să o învățăm să imite mintea umană și comportamentul uman.

Astăzi, programele de inteligență artificială, cum ar fi bunăoară, *Siri* de la *Apple* și *Alexa* de la *Amazon* demonstrează adevărate abilități intelectuale în soluționarea diverselor probleme cotidiene. Astfel, *Siri* poate să răspundă la diverse întrebări care țin de cultura general umană, poate să îți reamintească anumite lucruri și să stabilească întâlniri de lucru. Iar Alexa poate să pornească aerul condiționat în apartament, să conecteze lumina în casă, ori să pună muzica preferată.

Aceste aplicații, bineînțeles, sunt impresionante dar ele nu sunt creative și sunt în stare să facă doar atât cât sunt programate. Din aceste considerente sistemele de IA existente în acest moment poartă denumirea ***Weak AI*** sau ***Narrow AI***, ori ***Artificial Narrow Intelligence*** (**ANI**) („inteligență artificială slabă" sau „îngustă").

Menționăm faptul că **ANI** poate imita modul în care oamenii gândesc și comunică dar nu are abilitatea de a face raționamente, de a gândi. **ANI** poate doar să facă niște conexiuni în funcție de programul pe care îl are. Dar totuși **ANI** reprezintă deja un pas foarte important în evoluția inteligenței artificiale și se îndreaptă spre ***Artificial General Intelligence*** **(AGI)**.

AGI vor fi programele capabile să raționeze, să planifice și să îndeplinească procese complexe care ulterior se vor orienta către ***Artificial Super Intelligence*** **(ASI)** – programe care să depășească întreaga inteligența creativă a omenirii. Incredibil – dar nu imposibil!

1.5. Aspecte în evoluția IA. Eventualele pericole ale IA asupra societății

Subiectul cel mai discutat de experții IA ține de următoarea întrebare: Ce se va întâmpla când IA va depăși inteligența umană? În acest context, matematicianul John von Neumann (Decembrie 28, 1903 – Februarie 8, 1957) a introdus termenul *Singularitate tehnologică* care evidențiază momentul când Superinteligența Artificială va deveni superioară Inteligenței Umane. Acest epizod nu exclude și faptul că Superinteligența Artificială poate să aibă un comportament agresiv în raport cu omenirea. Astfel, cercetătorul Stuart Armstrong de la Institutul Future of Humanity a estimat că în anul 2040 va fi atins acest moment și se va produce Singularitatea tehnologică. Personalități din domeniu: Stephen Hawhing, Bill Gates, Steve Jobs, Stuart Russell, Elon Musk și-au exprimat îngrijorarea în legătură cu acest moment [2-6].

În cartea ***„The Structure of Scientific Revolutions"***, publicată în anul 1962, fizicianului și filosofului american **Thomas Samuel Kuhn** (1922-1996) spunea că paradigmele științei reprezintă:

„Realizări științifice care generează modele care, pe o perioadă mai mult sau mai puțin lungă, și într-un mod mai mult sau mai puțin explicit, ghidează dezvoltarea ulterioară a cercetării exclusiv în căutarea soluțiilor la problemele puse de acestea”.

Thomas Samuel Kuhn

Paradigmele moderne ale IA, prin tot ce se inventează, descoperă, elaborează și se produce în prezent, orientează ***Artificial Narrow Intelligence*** spre o nouă etapa calitativă ***Artificial General Intelligence***.

Așa dar, în loc de concluzii, putem afirma că IA în propria dezvoltare își propune la modul ideal să determine cum pot calculatoarele să devină la fel de inteligente ca oamenii. O măsură a inteligenței mașinii, așa cum am examinat anterior, este testul Turing. Până la moment, nici o mașină inteligentă nu a reușit să treacă testul respectiv. In imediatul viitor nu se conturează posibilitatea ca vreo mașină să treacă acest test. Inteligența Artificială este la etapa timpurie de dezvoltare. Cele mai mari invenții și descoperiri în domeniul IA se vor produce în viitorul apropiat. Să fim pregătiți pentru era mașinilor dotate cu ***Artificial General Intelligence*** care vor poseda o Superinteligență Artificială. Istoria inteligenței artificiale se scrie în fiecare zi.

CAPITOLUL 2.
ABORDĂRI ISTORICO-DIDACTICE ÎN STUDIEREA INTELIGENȚEI ARTIFICIALE

În acest capitol sunt examinate cele mai eficiente metode de cercetare în Inteligența Artificială: metoda simbolică, metoda conexionistă, metoda evolutivă. La fel, sunt studiate etapele de dezvoltare a IA. Sunt examinate cele mai mari realizări în domeniul IA obținute pe parcursul timpului cât și tendințele în dezvoltarea Inteligenței Artificiale.

2.1. Caracteristicile metodelor de cercetare în domeniul IA

Cuvântul inteligență provine din limba latină – „intelligentia", ceea ce denotă pricepere, înțelegere, cunoaștere. *Inteligența* este o măsura a capacității de a atinge scopuri (de a rezolva probleme) într-un mediu complex și dinamic.
Inteligența Artificiala reprezintă acea arie a științei calculatoarelor preocupată cu realizarea mașinilor care execută anumite sarcini, care dacă ar fi rezolvate de oameni ar fi considerate că exprimă comportamentul uman [1-8].

Inteligența Artificiala este știința de a construi mașini care sa facă lucruri ce ar necesita inteligență, daca ar fi făcute de oameni (Florin Leon).

Din punct vedere istoric cercetările în domeniul Inteligenței Artificiale au utilizat în mare măsură următoarele două metode semnificative:

- *Metoda simbolică (sau de sus in jos);*
- *Metoda conexionistă (de jos în sus);*
- *Metoda evolutivă* **(***algoritmi genetici/evolutivi),* începând cu anul 1973 cercetările în domeniul IA s-au axat și pe *algoritmi genetici/evolutivi).*

Metoda simbolică tinde să studieze și să reproducă *inteligența* examinând cogniția separat de structura creierului, altfel spus, independent de factorii biologici. Din aceste considerente vine și eticheta simbolistică. Tradițional, inteligența artificială s-a bazat pe metoda simbolică: calcul logic și operații cu

diverse simboluri realizate de om și de către calculatoare. De exemplu, semnele de circulație sunt reprezentate de liste de simboluri pe care calculatorul poate să le înțeleagă prin intermediul simbolurilor și pe care poate să le proceseze conform unor reguli predefinite de programator. Necunoscând simbolurile, calculatorul nu poate procesa informația. Metoda simbolică are o abordare de sus în jos.

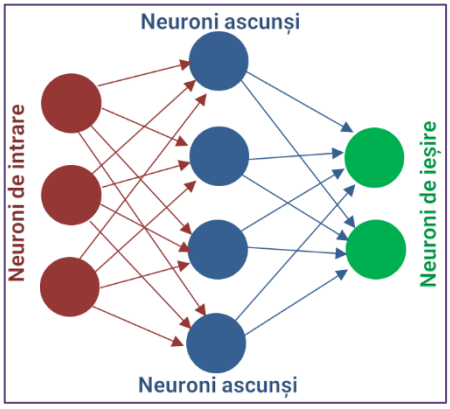
Inteligența artificială simbolică este termenul care se referă la toate metodele din IA ce sunt axate pe cercetări bazate pe reprezentări ale problemelor „simbolice" la nivel înalt (citite de om), logică și căutare. IA simbolică a fost paradigma dominantă de cercetare de la mijlocul anilor 1950 până la sfârșitul anilor 1980. Metodele simbolice au succes în anumite domenii care permit formalizarea datelor problemelor, cum ar fi, de exemplu: probleme de logică, matematică, șah, dame, planificare etc. Un triumf al abordărilor simbolice ține de elaborarea de către IBM a programului Deep Blue, care în anul 1997 la învins la pe campionul mondial la șah Garry Kasparov.

Metodele simbolice nu sunt eficiente însă în rezolvarea unor clase semnificative de probleme care implică procese fundamentale de percepție, acțiuni în mediu cu condiții variate, recunoașterea unor imagini etc., în cazuri când se confruntă cu situații pentru care nu au fost programate. Un exemplu elocvent pentru eșecul inteligenței artificiale construit pe modelele simbolice este *proiectul Cyc*. Proiectul dat, lansat în anul 1984, a încercat să asimileze cunoștințele uzuale ale oamenilor, prin introducere manuală de relații între simboluri. De exemplu, dacă programul cunoaște că păsările au pene, programul recunoscând în imagini pene poate concluziona că țin de o pasăre. În anul 1994 acest proiect ar fi trebuit să poată asimila singur date, procesând cărți și studii. După ce s-a investit mai mult de 60 de milioane de dolari și 5 secole-persoane de introducere de date, proiectul a fost abandonat, nereușind ceea ce și-a propus inițial.

Ca alternativă, o altă metodă importantă în cercetările ce se referă la IA, este metoda ***conexionistă***.

Această metodă are o abordare de jos în sus și implică crearea de rețele neuronale artificiale, în imitarea unui creier – de acolo vine eticheta de conexionist, adică de conexiuni, locul de unde se dau toate comenzile.

O ***rețea neuronală artificială*** este un sistem al cărui design a fost inițial inspirat schematic de funcționarea neuronilor biologici și care a abordat ulterior metodele statistice.



Se consideră că rețelele neuronale au fost inventate în anul 1943, atunci când logicianul *Walter Harry Pitts Jr.* (23 aprilie 1923 – 14 mai 1969) și neurofiziologul și cibernetician american *Warren Sturgis McCulloch* (16 noiembrie 1898 – 24 septembrie 1969) au creat un model de calcul pentru rețelele neuronale bazat pe algoritmi.

O perioadă importantă de timp ideea rețelelor neuronale artificiale nu a mai fost de actualitate, deoarece uriașele resurse de calcul necesare pentru construirea rețelelor neuronale nu existau. Ideea a revenit în actualitate, datorită resurselor de calcul avansate, cum ar fi unitățile de procesare grafică (GPU-uri). Acestea reprezintă cipuri folosite pentru procesarea graficii din jocurile video, dar care s-au dovedit excelente pentru analiza datelor necesare pentru funcționarea rețelelor neuronale.

Modelele conexioniste reprezintă, după cum s-a menționat deja, o alternativă a modelelor simbolice. Rețele neuronale artificiale funcționează după anumite principii inspirate din modul de funcționare al creierului uman, folosind multe elemente simple interconectate între ele. Însă, ca regulă, interpretarea datelor prelucrate de rețele neuronale sunt realizate tot de utilizatorul uman, ca și în cazul metodelor simbolice. Rețelele neurale au o structură diferită de cea a creierului, mult mai simplă, chiar dacă se aseamănă cu creierul uman după principiile de funcționare. De asemenea, la fel ca și creierul uman, rețeaua neurală este compusă din elemente puternice ce posedă capacități de calcul, însă mult mai inferioare decât cele ale neuronului, corespondentul uman.

Pentru a caracteriza rețelele neurale artificiale, se vor considera următoarele 3 criterii:

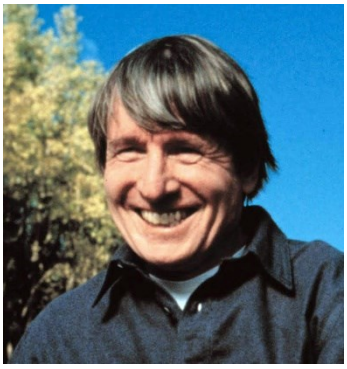
- *modelul* elementului de procesare individual selectat;
- *structura* interconexiunilor adoptată particular (arhitectura);
- *mecanismele* de potrivire a legăturilor (de învățare).

Există multe aplicații ale rețelelor neuronale. Un exemplu este capacitatea camerei foto a smartphone-ului de a recunoaște fețe. Automobilele fără șofer sunt echipate cu mai multe camere foto care încearcă să recunoască alte vehicule,

semne de circulație și pietoni folosind rețele neuronale și astfel automobilul își modifică viteza ori direcția în funcție de situație.

Rețelele neuronale se află, de asemenea, la baza sugestiilor de text, furnizate în procesul de scriere a textelor ori e-mail-uri, dar și în cazul unor softuri de traducere online. Rețelele neuronale mai complicate sunt capabile să învețe singure, după ce au primit instrucțiunile de bază pentru rezolvarea unei probleme. Unele rețele neuronale pot lucra „în parteneriat" pentru a crea ceva nou. Astfel, unele rețele pot crea fețe virtuale care nu aparțin unor oameni reali.

O rețea încearcă să creeze o față, iar o alta, încearcă să determine dacă aceasta este reală sau falsă, iar procesul continuă până când cea de-a doua rețea nu poate stabili dacă fața creată de prima rețea este falsă. Pentru ca rețelele neuronale să fie eficiente trebuie sa posede un volum mare de date pentru instruire.


John Henry Holland

La mijlocul anilor 1960, ***John Henry Holland*** (2 februarie 1929 – 9 august 2015) profesor american de psihologie, inginerie electrică și informatică la Universitatea din Michigan, Ann Arbor, cunoscut ca fondator al abordării sistemelor complexe, după mulți ani de studiere a ideii de simulare a evoluției, dezvoltă ***algoritmii genetici***. Algoritmii genetici modelează moștenirea genetică și lupta Darwiniană pentru supraviețuire. Populațiile evoluează prin apariția de noi caracteristici ale indivizilor în timpul încrucișării și ca efect al mutațiilor aleatorii. În procesul de evoluție supraviețuiesc indivizii care se adaptează cel mai bine mediului. Rezolvarea unei probleme presupune căutarea soluției în spațiul tuturor soluțiilor potențiale folosind o populație de agenți (căutători); Căutarea este ghidată prin intermediul unei funcții care măsoară gradul de apropiere față de soluție. Calculul evolutiv în acest sens reprezintă dezvoltarea și utilizarea de tehnici de rezolvare a problemelor inspirate de evoluția speciilor în natură.

Schema generala a algoritmilor evolutivi

Algoritmii evolutivi sunt algoritmi probabiliști care:

- mențin o populație de reprezentări de soluții candidat;
- care evoluează de-a lungul unor generații/iterații;
- sub controlul unei funcții fitness care măsoară meritul individual.

În ultima perioadă ia amploare o nouă tendință în IA și în științele cognitive. Este unanim agreată idea că sistemul inteligent ar trebui să aibă un corp, apropiat

de cel uman, să aibă o multitudine de senzori, cât și posibilitatea de a acționa în mediul în care poate conceptualiza obiectele cu care urmează să interacționeze. Se crede că astfel, robotul va descoperi lumea pe cont propriu, interacționând cu mediul în care se află prin intermediul senzorilor. Astfel, capacitatea cognitivă a robotului este în conexiune directă cu mediu, capacitățile senzomotorii, scopul propus și în ultimă instanță cu mecanismele cognitive.

2.2. Scurt istoric în dezvoltarea Inteligenței Artificiale

Etapa timpurie a IA (mileniul III, î. Hr. – 1700)

Toate științele, care sunt în serviciul omenirii astăzi, își au începutul pe pământul vechii Elade. Astfel, primul robot antic este considerat Talos din Creta, care acum câteva mii de ani în urmă, reprezenta o mașină perfect operațională, construită de Hefaistos – zeul metalurgiei și a sculpturii, unul din cei 12 zei ai Olimpului. Iar marele Aristotel, vine cu propria contribuție, inventând logica silogistică, primul sistem formal de gândire deductivă.

Este oportun să menționăm că primul sistem expert se consideră a fi un papirus egiptean, datând din mileniul III î.Hr., care constă din 48 de observații asupra diagnosticării și tratării rănilor la cap.

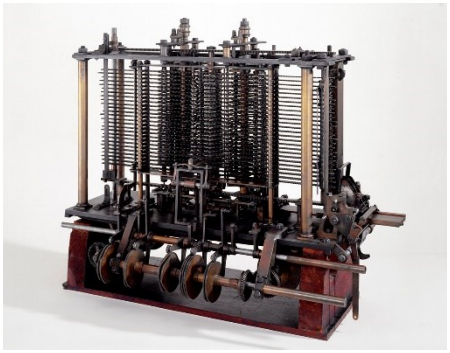
În secolul XIII, Ramon Lull, teolog spaniol, a inventat un dispozitiv pentru descoperirea adevărurilor non-matematice prin intermediul combinatoricii.

În anul 1206, inventatorul arab Al-Jazari, a construit o barcă ce transporta patru muzicieni mecanici, alimentați de fluxul de apă, construcție ce poate fi considerată drept „***primul robot umanoid programabil***”.

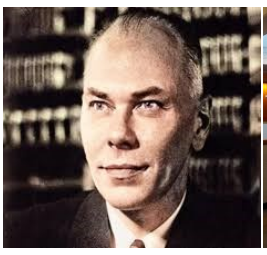
O nouă formă de inteligență artificială era promovată în anul 1642 prin intermediul dispozitivului de calcul „Pascalina”, care putea face adunări și care a fost conceput de celebrul matematician Blaise Pascal. Câteva decenii mai târziu, în 1673, Gottfried Wilhelm Leibniz a inventat un dispozitiv mecanic de calcul care putea efectua toate operațiile aritmetice [5-8].

Etapa ideilor lui Charles Babbage (1750 -1900)

Dar omenirea își dorea mai mult. Ideea mașinilor inteligente, programabile, plutea în aer.



Charles Babbage, faimosul matematician și inginer englez, în premieră a dezvoltat idei și a proiectat o mașină de calculat programabilă, care în linii mari este premergătoare computerelor moderne. Din aceste motive Charles Babbage este considerat „părintele informaticii”. În 1837 Babbage și-a proiectat „Motorul Analitic”, care din cauză constrângerilor financiare și limitărilor tehnologice nu a reușit să o construiască. Fiind construită, după schițele lui Babbage, tocmai în anul 1991, mașina funcționa perfect. Ideile sale s-au dovedit a fi atât de revoluționare încât nu erau în stare să fie înțelese de contemporani. Motorul Analitic avea o memorie de acces aleatoriu (RAM), un cititor de cartele profesionale și includea chiar și o imprimantă. Motorul Analitic dispunea de o unitate centrală de procesare (CPU) care era în stare să efectueze tipurile de operații logice și aritmetice pe care, în zilele noastre, le poate realiza orice CPU. Este semnificativ faptul că Motorul Analitic avea o unitate specială de programe, cu un limbaj de mașină similar celor de astăzi. Babbage descrie, în anul 1837, caracteristicile mașinii sale în lucrarea „On the Mathematical Powers of the Calculating Engine”. Primul program pentru Motorul Analitic a lui Babbage a fost elaborată de Ada Lovelace, fiica poetului englez Byron, care a rămas în istorie ca primul programator.



Conceptul Motorului Analitic a supraviețuit și, ulterior, în anul 1944, Horward Aiken, de la Universitatea Harvard și compania IBM, a împrumutat din ideile lui Babbage la construcția primului computer programabil american Mark 1. Aiken, în semn de respect față de Babbage comenta:

„Dacă Babbage ar fi trăit cu șaptezeci de ani mai târziu aș fi rămas fără serviciu”.

Astfel, conceptul unui computer cu program stocat, capabil de automodificări, având o memorie adresabilă, beneficiind de ramificări condiționale

și capacitatea de a se autoprograma – rămâne să fie până în prezent baza computerelor moderne.

Celebrul matematician George Boole în cartea sa „An investigation into the Laws of Thought”, publicată în 1854, în premieră constituie bazele *Logicii booleene*. Studiul logicii a condus la realizarea primului calculator electronic programabil.

Etapa preocupărilor pentru construirea unui creier artificial (1900 -1974)

Un rol aparte în dezvoltarea Inteligenței Artificiale l-a avut și celebra carte a lui Bertrand Russel și Alferd N. Whitehead, „Principia Methematica”, publicată în 1910-1913. În această carte a fost, în premieră, reformulată matematica pe baza teoriei mulțimilor. Interpretarea lui B. Russel a matematicii prin prisma teoriei mulțimilor, inclusiv „paradoxul lui Russel” a facilitat dezvoltarea de către Turing a teoriei computaționale și ulterior a stimulat dezvoltarea inteligenței artificiale. B. Russel și A. Whitehead, în lucrarea lor, nu au punctat clar conceptul despre computer, chiar dacă fac referință la o mașină logică, care realizează câte o transformare logică la o cuantă de timp. Neavând un computer real, operațiile logice erau rulate în mintea lor. Ulterior însă, A. Turing, s-a inspirat din ideile lor și cunoscând conceptul Motorului Analitic a lui C. Babbage a creat, în anul 1936, primul computer teoretic din lume.

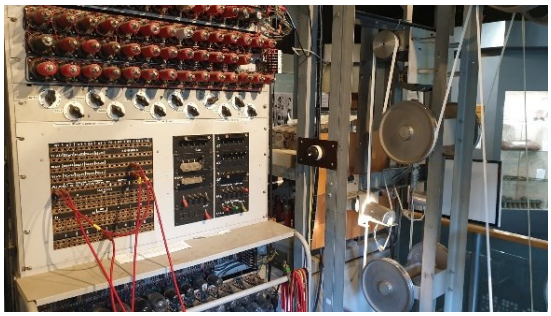
Lucrările mai multor logicieni și matematicieni precum Bertrand Russel, Alferd N. Whitehead, George Boole, Kurt Godel, Emil L. Post, Andrei Kolmogorov, Alonzo Church au contribuit semnificativ la apariția Inteligenței Artificiale ca domeniu de cercetare.

Totuși primul computer programabil din lume a fost realizat de către Konrad Zuse, inventator german. Primul său dispozitiv de calcul Z-1, a fost integral mecanic. Mașina Z-2 utiliza relee și putea să rezolve mai multe ecuații complexe simultan. Cea de-a treia versiune Z-3 a rămas în istorie ca fiind primul computer programabil din lume, chiar dacă în unele surse se argumentează faptul că calculatorul Mark 1, a lui Howard Aiken, este primul în acest sens. Faptul că K. Zuse a construit primul computer digital,

complet programabil, este probată de cererea de patent pe care el a depus-o la 11 aprilie, 1936.

În anii celui de al doilea război mondial, sub conducerea lui Alan Turing, a fost construit primul computer operațional din lume, *Robinson*, care funcționa în bază de relee telefonice. Prin intermediul acestui calculator britanicii au reușit să decodifice majoritatea mesajelor germane codificate prin intermediul celebrei mașini de coduri Enigma. Ceva mai târziu, din cauză creșterii complexității codurilor Enigma, Turing a substituit inteligența electromagnetică a lui *Robinson* cu o versiune electronică denumită *Colossus*, care funcționa având la bază circa 2000 de tuburi electronice.

Colossus a asigurat decodificarea informațiilor militare germane și a contribuit esențial ca Forțele Aeriene Regale să câștige Bătălia Angliei, în lupta cu „neînvinsa" flota aeriana germană.



Genialul A. Turing nu a putut să nu sesizeze asemănările dintre gândirea umană și procesele de calcul. În afară de faptul că a inventat primul computer funcțional din lume și a fundamentat teoretic aspecte importante ale științei computerelor, Turing a depus eforturi uriașe în aplicarea noilor tehnologii la imitarea inteligenței umane.

În acest sens, în celebra sa lucrare, publicată în 1950, „Computing Machinery and Intelligence", Turing a trasat o agendă care a direcționat cercetările în știința computerelor:

- procesul de luare a deciziilor prin intermediul calculatorului;
- algoritmizarea jocurilor și programarea lor;
- înțelegerea limbajului natural de către calculator;
- traducerea computerizată;
- demonstrarea computerizată a teoremelor;
- codificare, decodificarea și spargerea codurilor.

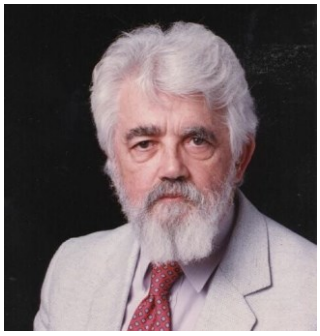
Este necesar de punctat faptul că Alan Turing, în anul 1950, a propus un test pentru determinarea inteligenței artificiale a unei mașini. Turing susținea că „dacă o mașină se comportă la fel de inteligent ca o ființă umană, atunci e la fel de inteligentă ca o ființă umană". În felul acesta se promovează ideea că inteligența unei mașini ar putea fi evaluată luând în considerare doar compartimentul ei. În contextul dat, scoatem în evidență faptul că *Alan Turing*, în cooperare cu prietenul

său *David Champernowne*, a elaborat primul program din lume pentru jocul de șah, care avea inteligența și „comportamentul” unui bun șahist.

Noile științe, care încep a se dezvolta în anii 50-60, *Bionica* (Jack E. Steele), *Cibernetica* (Norbert Wiener) și *Teoria Informației* (Claude Shanon) de asemenea, au contribuit la dezvoltarea inteligenței artificiale.

În anii 50, 60 al secolului trecut s-au reușit realizări impresionante în domeniul inteligenței artificiale. Astfel, în 1956, Allen Newell, J.C. Shaw și Herbert Simon [2] au elaborat programul Logic *Theorist*. Ceva mai târziu, în anul 1957, acești autori au lansat un nou program *General Problem Solver*, care utiliza tehnici recursive de cercetare pentru soluționarea problemelor de matematică. În așa mod, prin intermediul acestor două programe *Logic Theorist* și *General Problem Solver*, autorii enumerați, au reușit să găsească demonstrații la cele mai principale teoreme din „Principia Methematica”. Mai mult chiar, s-a reușit o demonstrație complet nouă, originală, a unei teoreme fundamentale din lucrarea lui B. Russel și A. Whitehead. Acest fapt a surprins atât pe matematicieni, cât și pe cercetătorii din domeniul inteligenței artificiale și i-a determinat pe autori să afirme foarte îndrăzneț, că în prezent în lume există mașini care gândesc, învață și creează. Tot odată, susțineau ei, că gama problemelor cu care se vor ocupa mașinile, va avea același diapazon ca și mintea omului.

În anul 1956, ***John MaCarthy*** (4 Septembrie 1927 - 24 Octombrie, 2011), profesor la Stanford University, a introdus termenul de *inteligență artificială* (Artificial Intelligence (AI)). Tot el a dat și o definiție laconică:



„***Inteligența Artificială este știința și ingineria producerii de mașini inteligente***” [3].

Conferința „Darthmouth Summer Research Project on Artificial Intelligence”, din 1956, de la Darthmouth College, a consacrat nașterea inteligenței artificiale ca domeniu de cercetare și de studiu. De altfel, în anii 50 s-au reușit mai multe descoperiri importante în domeniul IA:

- În anul 1951, celebrul Marvin Minski construiește prima mașină neuronală artificială (neural net machine).
- În perioada 1950 -1960 au fost realizate primele programe ce implementează jucători automat pentru șah și dame (Game AI).
- 1951 – Turing propune un test (Testul Turing) pentru a răspunde la întrebarea „Can machine think?”.

- În anul 1951, Turing în colaborare cu Shannon reușește dezvoltarea unor programe pentru jocul de șah/dame. Iar în anul 1958 un astfel de program a învins un jucător uman pentru prima data.
- Newell și Simon elaborează primul program pentru demonstrarea automată a teoremelor, Logic Theorist a demonstrat automat 38 din 52 de teoreme propuse, propunând demonstrații noi, elegante.
- În anul 1958, profesorul John MaCarthy fiind la celebra universitate MIT propune și dezvoltă limbajul LISP, care conține multiple elemente de noutate, ce stau la baza limbajelor moderne: recursivitate, runtime typing, garbage collection etc.
- Tot în anul 1958, Rosenblatt a dezvoltat Perceptronul - rețea neuronală, prin intermediul cărora se putea realiza modelarea fenomenelor mentale ori comportamentale etc.

Pe parcursul anilor 1960, cercetătorii din domeniul Inteligenței Artificiale, realizau investigații în conformitate cu programul trasat de A. Turing în anul 1950. Astfel:

- Thomas G. Evans, în anul 1963, *Massachusetts Institute of Technology* (MIT), elaborează programul *Analogy*, pentru rezolvarea problemelor IQ, cu analogie geometrică. Ulterior, ideile respective au fost utilizate pentru dezvoltarea calcul simbolic, aplicat în software de largă audiență – *Maple*, *Mathematica*.
- Daniel G. Bobrow, în anul 1964, (MIT), creează programul *Student*, care putea să soluționeze probleme de algebră de liceu după enunțuri clare în limba engleză, publicat în articolul „Natural Language Input for a Computer Problem Solving Sustem”. Discipol al faimosului informatician american Marvin Minsky, ulterior D. Bobrow devine Președinte al Asociației Americane de Inteligență Artificială (AAAI).
- În anul 1965, Feigenbaum reușește, grație procesării limbajului natural, dezvoltarea celebrului program – *ELIZA*.
- La Stanford University, în anul 1969, a fost construit robotul *Shakey* care combină locomoția, percepția și rezolvarea problemelor.
- Edward A. Feigenbaum, în anul 1967, Stanford University, lansează programul *DENDRAL* din domeniul sistemelor expert, care putea să răspundă la întrebări despre compușii chimici.
- În anul 1970 sunt dezvoltați în premieră algoritmii evolutivi.
- Terry Winograd, în anul 1971, (MIT) prin lansarea programului său SHRDLU

a reușit progrese însemnate în înțelegerea limbajului natural. Astfel, programul lui putea să înțeleagă orice propoziție logică în limba engleză care se referea la cuburi colorate.

Epoca primei stagnări a inteligenței artificiale (1974-1980)

În anii 1970, din cauză că cercetătorii în IA nu aveau o viziune clară vis-a-vis de dezvoltarea domeniului IA, se atestă o perioadă de stagnare a IA până aproape de anii 1980. La fel, capacitatea programelor de IA era limitată. Cercetătorii nu dispuneau de soluții privind dificultățile problemelor cu care se confruntau: puterea de calcul și memoria calculatoarelor era limitată, contractări bugetare etc. Optimismul cercetătorilor exagerat a generat așteptări exagerate care nu s-au realizat. Acest fapt a generat limitări de buget în inteligență artificială. Un perceptron este un tip de rețea neuronală, introdus în 1958 de Frank Rosenblatt. Fiind extrem de optimist el prezicea că „un perceptron poate fi capabil să învețe, să ia decizii și să traducă limbi". Un program dinamic de cercetare a acestor concepte a fost realizat în anii șaizeci, dar s-a oprit brusc după publicarea în anul 1969 a cărții „Perceptrons" a lui *Marvin Minsky* și *Seymour Papert*. Datorită criticii dure a lui Marvin Minsky și Seymour Papert în raport cu perceptronii, modelele conexioniste au fost aproape complet abandonate timp de 10 ani. Abia la sfârșitul anilor 1970, ideile noi în IA sunt explorate în programarea logică, raționamentul de *bun simț* și în alte direcții.

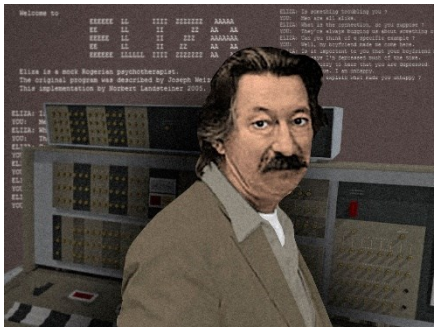
Perioada Boom-ul din 1980-1987

În anii 1980, programele de IA, numite „sisteme expert " au fost adoptate de companii și cunoștințele au devenit subiectul central al cercetării în IA. Între timp, guvernul japonez finanțează masiv IA prin inițiativa sa „calculatoare de generația a V-a". Un alt eveniment este renașterea conexionismului prin lucrările lui John Hopfield și David Rumelhart.

2.3. Chat-boturile Eliza și Perry

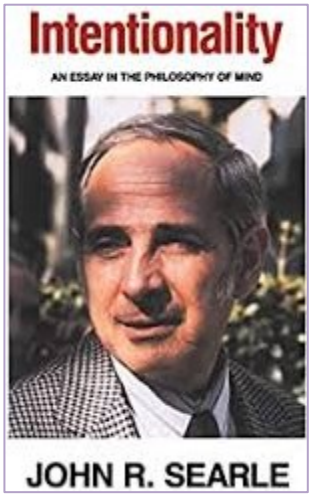
Chat-botul Eliza este unul din cele mai renumite programe de inteligență artificială timpurie. Joseph Weizenbaum a elaborat Eliza, astfel încât să reproducă comportamentul unui psihoterapeut, permițându-i programului să fie

„liber de a-și asuma rolul de a nu cunoaște aproape nimic din lumea reală”.



Programul lui Weizenbaum a fost în măsură să facă pe unii oameni să creadă că au purtat o conversație cu o persoană reală. Astfel, unii erau siguri de faptul că Eliza ar fi primul program capabil să treacă Testul Turing [1-4].

Chat-bot-ul Parry al lui Colby a fost descris ca fiind „Eliza cu atitudine”: acesta încearcă să imite comportamentul unui schizofrenic, folosind o abordare analoagă celei folosite de Weizenbaum. Un grup de psihiatri cu experiență au analizat o combinație de pacienți reali și computere care rulau Parry prin intermediul unei mașini telex. Unui alt grup de 33 de psihiatri le-au fost arătate transcrieri ale convorbirilor. Celor două grupuri le-a fost cerut să identifice care dintre pacienți sunt ființe umane și care sunt programe. Psihiatrii au reușit doar în 48% din cazuri să identifice corect pacienții.



Desigur, testul Turing avea și câteva limitări care nu pot fi ignorate. În acest context, poate fi menționat aici ***John Rogers Searle*** (născut la 31 iulie 1932), filosof analitic, profesor de filozofie Slusser la Universitatea din California, Berkeley, care a adus contribuții majore în domeniile filosofiei minții, filozofiei limbajului și ontologiei sociale. El este cel mai bine cunoscut pentru argumentele aduse în favoarea „Camerei Chinezești”, care își propune să demonstreze că sistemele informatice descrise oficial (în special, Testul Turing) nu pot da naștere la înțelegere intenționată.

În acest sens, John Searle a propus în lucrarea sa publicată în anul 1980, punctează argumente tari împotriva Testului Turing, cunoscute sub numele de experimentul „Camera Chineză”. Searle a susținut că software-ul Eliza ar putea trece Testul Turing pur și simplu prin manipularea de simboluri pe care nu le înțelege. Dar fără a le înțelege, acesta nu ar fi permis să fie descris ca fiind capabil să gândească (cum o fac oamenii). Searle ajunge la concluzia că Testul Turing nu poate dovedi că o mașină poate să gândească, fapt ce a stârnit numeroase controverse în discuțiile științifice.

2.4. Noi valențe în dezvoltarea Inteligenței Artificiale

Ideea de creare a unei noi forme de inteligență a atras și critici, deoarece unele produse elaborate nu reacționau adecvat la o varietate de situații simple. Astfel, de exemplu, cele mai performante programe elaborate reacționau inteligent în procesul de demonstrație a unor teoreme complicate, făceau partide excelente de șah, se descurcau foarte bine în probleme de medicină și chimie, dar aveau abilități derutante atunci când trebuia să identifice, bunăoară, deosebirile dintre un câne și un tigru. Nu erau în stare să facă distincție între lucruri simple, sarcini pe care un copil de 5 ani le poate realiza fără mari eforturi. În contextul respectiv, Hubert Dreyfus, un reprezentant notabil al filosofiei existențialiste susținea că mașinile inteligente, create de om, nu vor egala niciodată abilitățile umane. Spusele lui păreau să aibă mulți susținători în mediile universitare.

Anii 1980 s-au remarcat prin faptul că o serie de companii din domeniul Inteligenței Artificiale erau listate la bursă și produceau diverse „mașini inteligente" pentru comercializare. Companiile din domeniul înțelegerii limbajului natural, al recunoașterii caracterelor și vorbirii, al roboticii, al vederii automate și din alte zone au devenit tot mai prezente în dezvoltarea economiei.

La mijlocul anilor 1990 instituțiile financiare au început să fie dotate cu sisteme bazate pe tehnici statistice eficiente și adaptive. Pieţele valutare, de mărfuri, obligațiuni și de acțiuni erau conduse și menținute de rețele computerizate. Deciziile de vânzare și cumpărare erau luate de programe care se bazau pe modele ale pieţelor reale. Aceste modificări structurale ale pieţelor respective, care se produceau datorită implementării Inteligenței Artificiale, au deschis perspective nebănuite pentru dezvoltarea economică și financiară. Dezvoltarea spectaculoasă a Inteligenței Artificiale acoperea tot mai multe domenii și devenea o necesitate vitală pentru progres.

Un exemplu elocvent în acest sens ține de faptul că în anul 1997, programul *Deep Blue* produs de compania IBM, l-a învins la șah pe Garry Kasparov, campionul mondial la șah en titre al momentului. În premieră mașinile inteligente au demonstrat superioritatea lor în raport cu inteligența umană. Chiar dacă Garry Kasparov nu credea, până la acel moment, că poate pierde în fața calculatorului, a fost nevoit să recunoască înfrângerea. În această ordine de idei unul din cei mai

cunoscuți experți în domeniul inteligenței artificiale, Raymond Kurzweil (născut la 12 februarie 1948), informatician, autor, inventator și futurist american, a observat cu umor:

„Din această perspectivă descoperim că nu există activitate umană care să necesite o inteligență "reală".

Dezvoltarea dinamică a tehnologiilor informaționale a „îmbogățit" domeniul de Inteligență Artificială cu noi valențe. Mai jos vom face o scurtă descriere a celor mai importante ramuri care țin de Inteligența Artificială.

Robotică. Modelele moderne de roboți, recunosc lucrurile înconjurătoare, ocolesc obstacolele, se deplasează pe drumul cel mai scurt. Tot odată au în componență computere programate care pot „vedea", „pipăi", „mirosi", „auzi", pot să distingă vocile și să reacționeze adecvat la diverși stimuli externi. Roboții sunt programați să execute un număr semnificativ de misiuni, în medii puțin mai periculoase pentru oameni, operații repetitive și plictisitoare pentru ființele umane. În prezent există deja roboți care pășesc ca oamenii și pot întreține o discuție vie pe tematici general umane. Este necesar de menționat faptul că în prezent sunt roboți care au potențial de creație: roboți-poeți, roboți-compozitori, roboți-dirijori etc., care ating performanțe surprinzătoare în aceste domenii.

Înțelegere a limbajului natural. Acest domeniu, trasat încă de A. Turing, rămâne a fi unul de top pentru cercetătorii din domeniu Inteligenței Artificiale. Programarea computerelor astfel încât acestea să înțeleagă și să interacționeze cu utilizatorii în limbajul natural al acestora se dezvoltă destul de dinamic în multe țări ale lumii. Recunoașterea vocală, se află la baza înțelegerii limbajului natural. Ulterior dialogul dintre om și calculator, fiind recunoscut, se transformă în text, prin intermediul unui program și poate fi utilizat la soluționarea diverselor probleme practice.

Sisteme expert. Sistemele expert reprezintă sisteme informatice, formate dintr-un grup de programe, care utilizează cunoștințele și abilitățile unor experți, într-un domeniu relativ restrâns, pentru a lua cele mai bune decizii vis-a-vis de subiectul examinat. Calculatoarele simulează comportamentul unui expert uman pentru a soluționa probleme complexe din acest domeniu. Sistemul expert poartă un dialog om-computer în vederea rezolvării pe etape a problemei cercetate.

Astfel, sistemele expert multiplică inteligența artificială formalizată a unor specialiști umani, punând-o la dispoziția persoanelor, care din anumite considerente, nu au acces la experții respectivi.

Rețele neuronale. Rețelele neuronale (în engleză: ***ANN*** de la *artificial neural network*) este o ramură din știința inteligenței artificiale și reprezintă, în același timp, un obiect de investigație și pentru neuroinformatică. Rețele neuronale - reprezintă sisteme care imită inteligența umană prin reproducerea tipurilor de conexiuni fizice care se găsesc în creierul biologic. În creierul uman se află aproximativ 100 miliarde de neuroni, fiecare capabil de 1000 de operații pe secundă. În jur de 30 de miliarde de neuroni formează „materia cenușie", cea care gândește, restul de 70 de miliarde constituind „materia albă", cea care face legătura între neuronii din „materia cenușie". Din cauza constrângerilor tehnologice, în prezent numărul conexiunilor rețelelor neuronale este limitat, comparativ cu zecile de miliarde de conexiuni din creierul uman. Principala trăsătură a acestor rețele este capacitatea și potențialul de a învăța pe bază de exemple, utilizând experiența anterioară pentru a-și crește performanțele.

Algoritmi genetici. Algoritmii genetici reprezintă un mecanism inspirat din funcționarea sistemelor biologice. Algoritmii genetici constituie un model informatic care simulează modelul biologic evoluționist pentru a rezolva probleme de optimizare ori căutare. Astfel, algoritmii genetici încurajează soluțiile „promițătoare", cele mai bune, pentru a soluționa o problemă, și resping, penalizează soluțiile „ne-promițătoare", slabe care nu conduc la rezolvarea problemei examinate. În felul acesta, după mai multe generații se obțin soluții foarte bune pentru probleme de optimizare, cu un număr enorm de parametri. Algoritmii genetici au început să fie recunoscuți ca tehnici de optimizare odată cu rezultatele obținute de John Holland. În prezent, Algoritmii genetici se aplică cu succes în domeniul afacerilor financiare, evaluarea creditelor, predicțiilor financiare etc.

Sisteme logice fuzzy. În logica clasică, dacă un obiect aparține mulțimii date, atunci poate fi notat prin 1 și prin 0, în caz contrar. În logica *fuzzy* gradul de apartenență al obiectului la mulțimea examinată poate lua valori între 0 și 1. Logica *fuzzy* oferă instrumentele necesare pentru reprezentarea în sistemele inteligente a unor concepte incerte cum ar fi „mare", „scump", „ieftin" ș.a., concepte numite variabile lingvistice sau variabile *fuzzy*. Bazate pe logica *fuzzy*, sistemele *fuzzy*,

sunt considerate un caz particular al sistemelor expert, care oferă tehnici și metode pentru cercetarea incertitudinii. Aceasta permite sistemelor logice fuzzy să proceseze date incomplete și să furnizeze rezultate aproximative care reprezintă soluții acceptabile pentru probleme complexe. Astfel, Japonia este țara cu cele mai multe sisteme *fuzzy* implementate, mai ales în domeniile monitorizării producției și a vânzărilor. La fel, multe sisteme logice *fuzzy* au fost încorporate în unele dintre bunurile de larg consum: cuptoare cu microunde, aparate foto, mașini de spălat rufe, aparate de aer condiționat etc.

Jocuri pe computer. Dezvoltarea jocurile pe computer și, în genere, domeniul multimedia, este în ascensiune permanentă. Jocurile pe computer constituie o categorie de aplicații software și sunt destinate distracției. De la apariția industriei jocurilor, în anii 1970, și până la momentul actual, domeniul respectiv a evoluat spectaculos. Industria jocurilor reprezintă una din cele mai profitabile afaceri la nivel mondial, care pune în circulație sute de milioane de dolari. În prezent, nu se mai poate concepe un joc fără a avea în structură elemente de Inteligență Artificială. Implementată reușit, Inteligența Artificială garantează un produs bine comercializat, care generează profit și satisfacție jucătorilor.

2.5. Viitorul apropiat al Inteligenței Artificiale

Pentru dezvoltarea Inteligenței Artificiale se fac investiții mari la nivel internațional. Astfel, Statele Unite, dar și China, au anunțat în 2017 un plan de investiții publice în valoare de 22 de miliarde de dolari (18 miliarde de euro) în domeniul Inteligenței Artificiale până în 2020. Comisia Europeană, pentru a face față acestei provocări, a venit în luna aprilie, anul 2018, cu o inițiativă privind coordonarea acțiunilor vis-a-vis de dezvoltarea domeniului în Uniunea Europeană. Comisia a invitat în primul rând statele membre și sectorul privat să investească până în 2020, în total, 20 de miliarde de euro în cercetare și s-a declarat dispusă să contribuie la acest obiectiv cu o sumă de 1,5 miliarde de euro. Abordarea europeană, tri-direcțională, vizează să sporească investițiile publice și private în acest sector, să realizeze pregătirea necesară pentru schimbările socio-economice aferente și să asigure un cadru etic și juridic corespunzător.

Andrus Ansip (născut la 1 octombrie 1956) un politician estonian, membru al Parlamentului European, fost comisar european pentru piața unică digitală și vicepreședinte al Comisiei Europene, în funcție din 2014 până în 2019 a declarat:

„Ca și motorul cu aburi sau curentul electric în trecut, inteligența artificială transformă lumea în care trăim. Aceasta presupune noi provocări pe care Europa ar trebui să le abordeze în comun pentru ca inteligența artificială să se poată dezvolta și să beneficiem cu toții de pe urma ei. Trebuie să investim cel puțin 20 de miliarde de euro până la sfârșitul anului 2020. Comisia își îndeplinește rolul care îi revine: le oferim cercetătorilor imboldul de care au nevoie pentru a putea dezvolta următoarea generație de tehnologii și aplicații de inteligență artificială, iar companiilor, mijloacele de care au nevoie pentru a le putea adopta și integra în activitatea lor".

Recent, tehnologiile IA au fost folosite pentru a ajuta la secvențierea ARN-ului pentru vaccinuri și a modela vorbirea umană, tehnologii fundamentate pe învățarea automată bazată pe modele și algoritmi și se concentrează tot mai mult pe percepție, raționament și generalizare.

Cu astfel de inovații IA a revenit pe foarte mult timp în centrul atenției lumii științifice.

Se consideră că în viitorul apropiat, Inteligența Artificială ar putea fi dezvoltată pe două direcții: una ***prin învățare***, iar cealaltă ***prin creare și dezvoltare***.

Vor fi dezvoltate aplicații bazate pe IA care au menirea să învețe și să redea la cerere informații, așa cum e conceptul pentru ChatGPT. Totodată, au început să fie dezvoltate și aplicații bazate pe IA care pot genera imagini sau creații muzicale originale.

La o scară mult mai mare, Inteligența Artificială este gata să aibă un efect major asupra durabilității, schimbărilor climatice și problemelor de mediu. În mod ideal și parțial prin utilizarea unor senzori sofisticați, orașele vor deveni mai puțin aglomerate, mai puțin poluate și, în general, mai locuibile.

Este Inteligența Artificială o amenințare pentru umanitate? În acest moment, cu siguranță nu avem de ce să ne speriem, iar viitorul este imprevizibil.

CAPITOLUL 3.
PARADIGME ACTUALE ÎN PROCESUL DE DEZVOLTARE ȘI STUDIERE A ROBOȚILOR DE CHAT

În acest capitolul sunt examinate cele mai esențiale aspecte care țin de dezvoltarea și studierea chat-boților. În contextul dat sunt studiate tipurile și particularitățile esențiale ale chat-boților, metodele actuale de creare a acestora din perspectiva dezvoltării Inteligenței Artificiale, procesării limbajelor naturale (NLP) și a învățării automatizate (ML). Sunt analizate cele mai importante realizări în domeniul dat, obținute pe parcursul timpului, cât și tendințele în dezvoltarea chat-boților moderni. Ca și orice știință, Inteligența Artificiala studiază o serie de probleme cu caracteristici generale comune și dezvoltă tehnici specifice de rezolvare a acestor probleme.

3.1. Iarăși despre Testul Turing

După cum menționam anterior, în celebrul său articol *„Computing Machinery and Intelligence"*, publicat în anul 1950, Turing a lansat în premieră problema posibilității simulării ființei umane cu ajutorul calculatorului. În articol, Alan Turing descrie și testul care se reduce la determinarea de către Om a unui comportament inteligent și unul neinteligent în raport cu calculatorul dotat cu IA. Problema cea mai semnificativă în procesul de elaborare a unui astfel de program se referă la faptul că nu se cunoaște cantitatea de informație necesară care trebuie inclusă, introdusă în program ca ulterior să fie capabil să treacă testul Turing.

Programul de pe calculator trebuie să imite comportamentul uman, luând în considerare spectrul enorm de larg al întrebărilor posibile de adresat, cât și faptul că acestea sunt formulate în limbaj natural.

În alt context, unii experți în domeniul IA afirmă că testul Turing „impune, forțează” ca inteligența mașinilor să modeleze inteligența umană. Experții respectivi susțin că inteligența mașinilor este o formă diferită de inteligență umană

și că, de fapt, este o eroare să tindem a o evalua în termenii inteligenței umane. Se dorește oare construirea unei mașini capabile să imită inteligența unui om și care să fie la fel de lentă în raționamente ca și omul? Acestea sunt întrebări la care încă nu cunoaștem răspunsuri clare.

Limitele testului Turing

Testul Turing a fost criticat de-a lungul anilor, în special pentru că din punct de vedere istoric, natura interogării a trebuit să fie limitată pentru ca un calculator să prezinte inteligență asemănătoare omului. Un calculator ar putea avea un scor mare numai dacă întrebările se refereau la un domeniu foarte limitat de cunoștințe ori persoana care a formulat întrebările a obținut doar răspunsuri de tipul: „Da” sau „Nu”. Când întrebările sunt deschise și necesită răspunsuri conversaționale, este mai puțin probabil ca programul de calculator să poată „păcăli” cu succes persoana care a întrebat.

Un program, de exemplu, precum ELIZA ar putea trece testul Turing, manipulând simboluri pe care nu le înțelege pe deplin. John Searle, după cum menționam anterior, a susținut că acest lucru nu determină inteligența comparabilă cu oamenii.

În prezent, pentru mulți cercetători din domeniul IA, întrebarea dacă un calculator poate trece sau nu testul Turing a devenit irelevantă. În loc să se concentreze asupra modului de a convinge pe cineva că vorbește cu un om și nu cu un program de calculator, accentul real ar fi trebuit să fie pus pe modul de a face o interacțiune om-mașină mai intuitivă și mai eficientă.

Variații și alternative la testul Turing

Au fost dezvoltate o serie de variații ale testului Turing pentru a-l face mai relevant. Astfel de exemple sunt:

- ***Testul Turing inversat*** - în care un om încearcă să convingă un computer că nu este un computer. Un exemplu în acest sens este *Testul CAPTCHA*.
- ***Testul Turing total*** - unde cel care pune întrebări poate testa și abilitățile perceptuale, precum și capacitatea de a manipula obiecte.
- ***Test de semnal inteligent minim*** - unde sunt date numai întrebări adevărat/fals și da/nu.

Teste alternative

Au fost dezvoltate și teste alternative la testele Turing, precum:

- ***Testul Marcus*** - în care un program care poate „viziona” o emisiune de televiziune și ulterior este testat prin întrebări semnificative despre conținutul emisiunii.
- ***Testul Lovelace 2.0*** - realizat pentru a detecta IA prin examinarea capacității sale de a crea artă.
- ***Winograd Schema Challenge*** - adresează întrebări cu variante multiple într-un format specific.

Cum este utilizat testul Turing astăzi?

Deși variațiile testului Turing sunt adesea mai aplicabile înțelegerii noastre actuale despre IA, formatul original al testului Turing este încă folosit până în prezent.

De exemplu, ***Premiul Loebner***, despre care vom discuta mai jos, este acordat anual, începând cu anul 1990, programelor de calculator care trec testul Turing, votat de un juriu. Competiția urmează regulile standard ale testului Turing. Criticii relevanței premiului îl minimizează adesea ca fiind mai mult despre publicitate decât despre testarea cu adevărat dacă mașinile pot gândi.

La o competiție organizată de Universitatea din Reading pentru a marca împlinirea a 60 de ani de la moartea lui Turing în 2014, un chat-bot numit *Eugene Goostman* care simulează un băiat de 13 ani a trecut testul Turing, când a reușit să păcălească 33% din judecători.

În 2018, *Google Duplex* a desfășurat cu succes o discuție cu un coafor la telefon în fața unei mulțimi de 7.000 de oameni. Recepționera nu știa cu cine vorbea și nici nu a bănuit că a discutat cu un robot. Google Duplex este considerat de unii a fi o trecere modernă a testului Turing, în ciuda faptului că nu se bazează pe adevăratul format al testului așa cum l-a proiectat Alan Turing.

GPT-3, un model de procesare a limbajului natural creat de OpenAI, este considerat de unii că are cea mai bună șansă de a depăși testul Turing în adevărata sa formă față de orice tehnologie pe care o avem astăzi. Dar, chiar și cu abilitățile sale avansate de generare de text, mulți au criticat GPT-3, deoarece poate chiar să răspundă la întrebări fără sens.

Oricum, până în prezent, testul Turing rămâne a fi cel mai relevant test în procesul de identificare a inteligenței artificiale.

În continuare, din această perspectivă, ne vom referi la evoluția și dezvoltarea chat-bot-urilor.

3.2. Ce este un chat-bot?

Unii experți susțin că conceptul teoretic privind crearea chat-boților își are originea în lucrările lui Alan Turing din anul 1950 privind crearea mașinilor inteligente. Inteligența artificială, inclusiv în domeniul chat-boților, a progresat de atunci pentru a include supercomputerele superinteligente precum IBM *Watson*.

Se consideră că primul chat a fost arborele telefonic, care a creat o modalitate dificilă de alegere a unei opțiuni după alta pentru apelarea clienților, până s-a ajuns la un model automatizat de serviciu de asistență pentru clienți. Creșterea progreselor complexității IA, ML (Machine Learning) și NLP (Natural Language Processing) au dezvoltat modelul respectiv în chaturi. Și această tendință revoluționară continuă.

Un chat-bot *este un program de computer care simulează și procesează conversația umană (scrisă sau vorbită), permițând oamenilor să interacționeze cu dispozitivele digitale, similar comunicării cu o persoană reală*.

Unii autori utilizează și termenii: ***IM bot, interfață conversațională, smartbot, talkbot, chatterbot, robot de chat*** etc.

Astfel, un *chat-bot* este o interfață de comunicare, bazată pe inteligență artificială, ce ajută indivizii și companiile să desfășoare conversații cu succes pe un domeniu de interes comun [8-12].

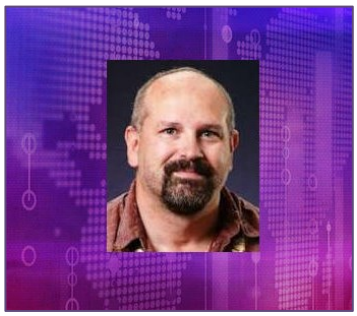
Chat-boții, de regulă, sunt utilizați pentru a desfășura interacțiuni cu utilizatorii, pentru a stimula afacerile, pentru a obține informații de la grupuri mari de obiecte, fenomene, utilizatori etc. În anumite situații, chat-boții se folosesc și în scopuri mai puțin nobile, cum ar fi creșterea artificială a numărului de vizualizări de articole sau video-uri pe web, transmiterea de viruși etc.

Se consideră că acele interfețe conversaționale dezvoltate, care simulează într-un mod realist atitudinea și comportamentul interlocutorului de dialog, ar putea să treacă testul Turing.

Chat-bot-urile concepute în sistemele de dialog, de cele mai multe ori, sunt utilizate pentru livrări ori achiziții de servicii, altfel spus în scopuri practice.

Unele chat-boturi utilizează sisteme sofisticate de procesare a limbajului natural, iar altele mai simple scanează cuvintele cheie și, ulterior, livrează răspunsuri în conformitate cu un model de construcție a răspunsurilor folosindu-se de o bază de date.

Menționăm faptul că informaticianul american Michael Mauldin (născut în anul 1959), creatorul primului Vertbot *Julia*, un program de conversație, în anul 1994, a propus în premieră termenul „ChatterBot".



În afara de aceasta, Michael Loren Mauldin este și inventatorul motorului de căutare web *Lycos*. De asemenea, el este și unul dintre autorii sistemelor de inteligență artificială *Rog-O-Matic* și *Julia*, plus faptul că promovează și este foarte de interesat de ***Robot Fighting League*** (o competiție a luptelor cu roboți).

În prezent, cele mai multe chat-boturi sunt accesate prin intermediul asistenților virtuali: *Amazon Alexa*, *Google Assistant*, ori prin aplicații de mesagerie: *Facebook Messenger*, *WeChat* sau chiar prin aplicații individuale.

Clasificarea chat-boturilor se face în mai multe categorii, în funcție de utilitate: analiză, educație, divertisment, jocuri, sănătate, călătorii, comerț etc.

3.3. Chat-boturi faimoase. Concursul de inteligență artificială Loebner Prize

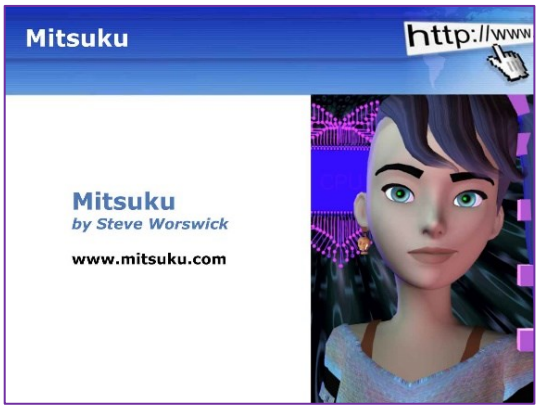
Din punct de vedere istoric, ***Chat-botul Eliza*** creat în 1966 de psihologul american Joseph Weizenbaum, este primul și unul din cele mai renumite programe ale inteligenței artificiale timpurii. Joseph Weizenbaum a elaborat Eliza, astfel încât să reproducă comportamentul unui psihoterapeut, permițându-i programului să fie *„liber de a-și asuma rolul de a nu cunoaște aproape nimic din lumea reală"*. Programul lui Weizenbaum a fost în măsură să facă pe unii oameni să creadă că

au purtat o conversație cu o persoană reală. Astfel, unii erau siguri de faptul că Eliza ar fi primul program capabil să treacă Testul Turing [1-4].

Chat-bot-ul Parry al lui Colby a fost descris ca fiind „Eliza cu atitudine”: acesta încearcă să imite comportamentul unui schizofrenic, folosind o abordare analoagă celei folosite de Weizenbaum. Un grup de psihiatri cu experiență au analizat o combinație de pacienți reali și computere care rulau Parry prin intermediul unei mașini *telex*. Unui alt grup de 33 de psihiatri le-au fost arătate transcrieri ale convorbirilor. Celor două grupuri le-a fost cerut să identifice care dintre pacienți sunt ființe umane și care sunt programe. Psihiatrii au reușit doar în 48% din cazuri să identifice corect pacienții.

Ulterior, un alt program faimos de IA, ***Alice*** (Artificial Linguistic Internet Computer Entity) care a evaluat perfecționând ***programul Eliza***, încearcă să simuleze o conversație umană prin intermediul unei platforme de text sau chat. Noutatea acestui chat-bot constă în faptul că utilizează tehnici de înțelegere a limbajului natural și de procesare a textului pentru a răspunde la întrebările interlocutorului și a începe conversații. Chat-botul Alice, ca și programul Eliza, evident nu are o înțelegere autentică a sensului dialogului desfășurat, dar poate livra răspunsuri credibile și poate fi aplicat ca o formă de divertisment chiar cu scopul de a testa sistemele de inteligență artificială.

Programul Mitsuku este un alt chat-bot celebru. Acest program de inteligență artificială a fost creat în anul 1997 de renumitul expert englez în IA, Steve Worswick. Chat-botul Mitsuku a evoluat calitativ în timp prin intermediul unui proces continuu de perfecționare a algoritmilor și prin exersare și antrenare tehnologică.



Bazându-se pe tehnologii de înțelegere a conversației pentru a răspunde la întrebările utilizatorilor prin intermediul unui mediu de chat și pe procesare a limbajului natural, chat-botul Mitsuku poate demara dialoguri și răspunde la o serie largă de întrebări, de la subiecte simple la subiecte complexe, plus faptul că are simțul umorului și este conceput să fie ușor de utilizat.

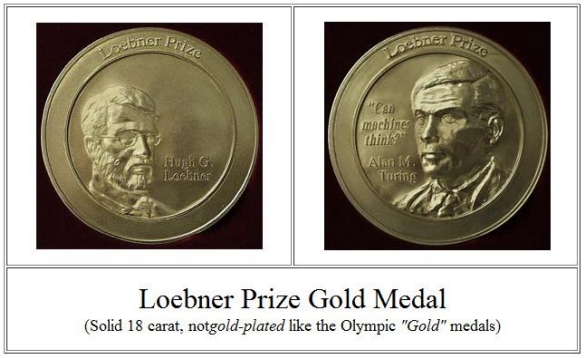
Steve Worswick a exersat și a antrenat îmbunătățind în permanență programul Mitsuku prin intermediul dialogurilor cu sute de utilizatori, completând

mereu cu informații noi și crescând capacitatea de înțelegere a limbajului natural de către chat-bot.

În consecință, programul de inteligență artificială Mitsuku este considerat unul dintre cei mai avansați chat-boți din lume, care se implica într-o conversație cu interlocutorii săi, similar cu modul în care un interlocutor uman poate conversa.

Desigur, testul Turing avea și câteva limitări care nu pot fi ignorate. John Searle a propus în lucrarea sa, în 1980, un argument împotriva Testului Turing, cunoscut sub numele de experimentul „Camera Chineză". Searle a susținut că software-ul Eliza ar putea trece Testul Turing pur și simplu prin manipularea de simboluri pe care nu le înțelege. Dar fără a le înțelege, acesta nu ar fi permis să fie descris ca fiind capabil să gândească (cum o fac oamenii). Searle ajunge la concluzia că Testul Turing nu poate dovedi că o mașină poate să gândească, fapt ce a stârnit numeroase controverse în discuțiile științifice.

Concursul Loebner Prize. În anul 1991, la inițiativa lui Hugh Loebner, (26 martie 1942 - 4 decembrie 2016) inventator și activist social american, a fost constituit ***concursul internațional de inteligență artificială Loebner Prize***, care are drept scop evaluarea performanței sistemelor de inteligență artificială în simularea conversațiilor umane din perspectiva Testului Turing.



Loebner Prize Gold Medal
(Solid 18 carat, not*gold-plated* like the Olympic *"Gold"* medals)

În acest sens concursul se desfășoară sub forma unui test Turing modificat, unde evaluatorii, membrii juriului, dialoghează prin intermediul unui mediu de chat ori text cu programul de inteligență artificială și cu un interlocutor uman, încercând să clarifice care dintre acești doi interlocutori este uman și care este chat-botul. Programele de inteligență artificială care reușesc să convingă evaluatorii că sunt umane sunt premiate cu o sumă semnificativă de bani.

Lansând ***Concursul Loebner***, autorul a promovat conceptul de inteligență artificială și Testul Turing unui public larg și a stimulat interesul pentru domeniul IA. Testul Turing este în același timp și un instrument de măsurare a intelectului uman și a stadiului tehnicii actuale. Astfel, el menționa că:

„Există o noblețe în acest demers. Dacă noi, oamenii, putem reuși să dezvoltăm un intelect artificial performant, acest fapt va fi și măsura dimensiunii intelectului nostru”. Tot el sublinia următoarele: ***„… În anii următori, pot fi premii mai consistente și concursuri mai prestigioase, dar asta va fi întotdeauna primul”.***

În acest sens, în iunie 2012, ***programul de inteligență artificială Eugene Goostman***, cu ocazia comemorării a 100 de ani de la nașterea autorului testului, Alan Turing, a câștigat concursul Loebner Prize de promovare a celui mai prestigios concurs de testare Turing organizat vreodată, reușind să convingă cu succes 29% dintre evaluatori că este uman, adică reprezintă o persoană reală.

Programul de inteligență artificială ***Eugene Goostman*** este un chat-bot dezvoltat la Sankt Petersburg în 2001 de un grup de trei programatori, Vladimir Veselov, Eugene Demchenko și Serghei Ulasen. Sistemul de inteligență artificială *Goostman* este portretizat ca un adolescent din Odessa, de 13 ani, care induce scuze în raport cu cei care interacționează vis-a-vis de propriile erori gramaticale și lipsa de cunoștințe generale.

Iar ceva mai târziu, la 7 iunie 2014, la un concurs dedicat celei de-a 60-a aniversare de la moartea lui Turing, 33% dintre membrii juriului au crezut că *Goostman* este un interlocutor uman. Organizatorul evenimentului, Kevin Warwick, a considerat că chat-botul ***Eugene Goostman*** a trecut testul lui Turing, conform predicției lui Turing în anul 1950, în raport cu faptul că până în anul 2000, sistemele de inteligență artificiale vor fi capabile să păcălească 30% dintre evaluatorii umani după cinci minute de interogatoriu. Însă validitatea și relevanța anunțului privind victoria lui *Goostman* a fost pusă la îndoială de unii din experții din domeniu IA, care au remarcat exagerarea deciziei lui Warwick, prin utilizarea de către chat-bot a umorului fin cu accente odesite și a unor abordări stranii în încercarea de a direcționa greșit interlocutorii de la tendințele sale non-umane și lipsa de inteligență reală.

În ciuda multor dezbateri despre relevanța testului Turing astăzi și validitatea competițiilor care se desfășoară în jurul acestuia, testul rămâne încă un punct de plecare filozofic pentru discutarea și cercetarea IA.

Pe măsură ce continuăm să facem progrese în domeniul inteligenței artificiale pentru a înțelege mai bine modul în care funcționează creierul uman, ***testul Turing rămâne fundamental pentru definirea inteligenței și reprezintă o bază***

pentru dezbaterea despre ce ar trebui să ne așteptăm de la tehnologii pentru ca acestea să fie considerate calculatoare cu inteligență.

3.4. Caracteristicile chat-boturilor

Este clar că chat-boturile inteligente se bazează pe inteligența artificială atunci când comunică cu utilizatorii. În loc de răspunsuri pregătite anterior, robotul deja încearcă să răspundă cu sugestii adecvate vis-a-vis de subiectul discutat. În plus, toate cuvintele rostite de clienți sunt înregistrate pentru procesare ulterioară. Cu toate acestea, inteligența artificială nu este magică și nu este încă pregătită să producă experiențe perfecte pentru utilizatori.

Oricum, chat-boturile în ziua de azi se utilizează pe larg în diverse domenii: comerț, servicii, educație etc.

Exemplu. În situația când se dorește procurarea unui produs careva online, se poate adresa o întrebare și discuta cu un chat-bot pentru a obține ajutor în scopul soluționării problemei respective. Deoarece, în linii mari această tehnologie este încă la etapa incipientă, chat-urile respective pot soluționa doar solicitările simple: Care este costul? Unde este comanda preluată? În cât timp livrați marfa? etc. În situația când un chat-bot nu poate rezolva problema conversația este transferată către interlocutorul uman.

Se pot evidenția câțiva factori importanți care îi determină pe oameni să apeleze la chat-boturi. Astfel, menționăm:

- ***Relaționarea și factorii sociali***. Chat-boturile contribuie la evitarea singurătății interlocutorului și oferă posibilitatea de a discuta liber și nestingherit, dezvoltând abilitățile de conversație și îmbunătățind experiența de relaționare.
- ***Ingeniozitatea și curiozitatea***. Chat-boții se dovedesc a fi destul de ingenioși în discuție cu interlocutorii, punctând chestii inedite și totodată trezesc curiozitate în raport cu utilizatorii. În cest sens, oamenii vor să-și exploreze abilitățile și să încerce ceva inedit, ceva nou.
- ***Productivitate*****.** Informația solicitată este livrată rapid și eficient de către chat-bot.
- ***Recreere și divertisment*****. Chat-boturile pot fi interlocutori plăcuți la petrecerea timpului liber, fiind în stare să amuze interlocutorii, livrându-le glume și sfaturi distractive.**

3.5. Tipuri de chat-boți

În funcție de modul în care au fost programate chat-bot-urile specifice, le putem împărți în două mari categorii:

1. ***Chat-bot simplu (orientat pe activități)***

Chat-botul simplu operează conform comenzilor pre-făcute. Funcțiile de chat simple se realizează în conformitate cu cuvintele cheie scrise anterior pe care le înțeleg. Fiecare dintre aceste comenzi trebuie să fie scrise de dezvoltator separat, folosind expresii obișnuite sau alte forme de analiză a șirurilor. Dacă interlocutorul a adresat o întrebare fără să folosească un singur cuvânt cheie, robotul nu îl poate înțelege și, de regulă, răspunde cu mesaje precum „Sorry, but I don't understand" ori „Sorry, I didn't catch that".

2. ***Chat-bot avansat cu inteligență artificială***

Chat-boții cu inteligență artificială sunt capabili să desfășoare conversații mai inteligente și mai consistente cu interlocutorii. Bază pentru aceasta este componenta care ține IA subordonată așa-numitei învățări automate (ML - Machine Learning) și procesării limbajului natural (NLP - Natural Language Processing). Inteligența artificială permite mașinilor să simuleze inteligența umană. IA este acum o componentă esențială în aproape toate aplicațiile pe care le folosim în viața de zi cu zi. Netflix, de exemplu, folosește inteligența artificială pentru a sugera filme care se potrivesc preferințelor utilizatorilor. Alt exemplu, se referă la Spotify care indică melodii pe baza listelor de redare, iar YouTube poate deja să „ghicească" ce videoclipuri noi ar putea interesa, pe baza istoricului memorat.

- ***Componenta NLP***

Procesarea limbajului natural permite mașinilor să înțeleagă sau să traducă vorbirea umană. Acestea pot atribui un sens cuvântului scris sau rostit și, de exemplu, pot oferi traduceri corecte din punct de vedere gramatical sau pot extrage cuvinte cheie. Un exemplu în acest sens este filtrul de spam, care citește subiectul e-mail-urilor recepționate și decide dacă un e-mail aparține dosarului de spam.

- ***Componenta ML***

Învățarea automată (sau învățarea profundă) permite unei aplicații de IA să recunoască modele în seturile de date. Acest lucru îi permite, după cum indică și numele, să învețe pe cont propriu și să se îmbunătățească continuu. De exemplu, roboții de șah de astăzi (practic) de neînvins, joacă prin milioane de jocuri și pot folosi ML și datele din jocurile anterioare pentru a afla care mișcări sunt puternice sau slabe. Ultima victorie înregistrată de un om împotriva unei IA de șah a fost în anul 2005.

Un chat-bot cu IA este capabil să încorporeze contextul intrării interlocutorului și să răspundă la acesta în mod corespunzător în chat. Astfel, utilizatorul poate conversa cu chat-botul într-un stil natural, conversațional. În plus, chat-bot-ul este capabil să învețe constant din feedback-ul utilizatorilor și din dialogurile care au avut deja loc și să îmbunătățească constant calitatea răspunsurilor sale.

Chat-boții *avansați cu inteligență artificială fiind mult mai sofisticați* sunt numiți adesea asistenți virtuali (asistenți digitali) și sunt mult mai interactivi și personalizați decât cei orientați pe activități. Acești chat-boți folosesc o înțelegere a limbajului natural NLP și ML pentru a învăța din mers. Respectivii chat-boți utilizează informațiile predictive și analizele, pentru a studia în timp preferințele unui utilizator, oferind ulterior recomandări și chiar anticipând cerințele acestuia și, la necesitate, inițiind conversații. Astfel, roboții de chat reușesc să studieze personalizarea interlocutorului. În acest sens, Alexa de la Amazon și Siri de la Apple sunt modele de roboți de chat orientați către consumatori, date și predicții.

3.6. Domenii de aplicare ale chat-boților

Oriunde companiile ar trebui să contacteze cu clienții, fie că este vorba de asistență în magazin, o linie telefonică sau prin e-mail și rețele sociale – un asistent virtual poate prelua sarcini simple sau complexe. În linii mari, putem spune că chat-boții îndeplinesc diferite servicii în trei domenii principale:

1) *Marketing;*
2) *Consultanță și servicii;*
3) *Resurse Umane.*

Concret, chat-boții pot fi folosiți în toate domeniile care au conexiune strânsă cu comunicarea, cum ar fi: *consultanță; e-commerț; marketing online; servicii financiare; sănătate; știri; e-learning; relații Clienți; informație; educație etc.*

Avantajele chat-boților depind de ***performanța IA*** și de ***calitatea datelor*** care le utilizează. Să examinăm mai detaliat aceste aspecte.

Inteligența artificială este folosită pentru automatizarea proceselor repetitive ori de rutină. Chat-boții cu inteligență artificială încorporată funcționează de obicei mai eficient.

În situațiile când chat-boții nu pot face față problemei examinate, robotul de chat solicită clientului o altă întrebare de concretizare, ori redirecționează interlocutorul către un operator uman. Pe măsură ce tehnologia și implementarea IA continuă să se dezvolte, roboții de chat și asistenții digitali se integrează din ce în ce mai mult în viața și în experiența noastră zilnică.

Chat-boții folosesc ***date*** și ***informații*** accesate din diverse surse. De calitatea datelor respective depinde și răspunsul chat-boților. Dacă programul a fost dezvoltat corect (se are în vedere calitatea funcționării componentelor ML și NLP) și datele sunt de calitate, atunci și robotul de chat reacționează corespunzător. Altfel spus, chat-boții funcționează eficient și calitativ dacă IA și datele sunt la fel de calitative. Să punctăm în acest context, de exemplu, că agenția de știri Associated Press a început să utilizeze un software care să scrie știri scurte, bazat pe date și informații sigure. În contextul dat, numărul știrilor livrate a sporit considerabil și a oferit jurnaliștilor reali posibilitatea să se concentreze pe articole lungi și complexe.

3.7. Cum să creezi un chat-bot?

Dezvoltarea chat-boților este un proces creativ și depinde de calitatea dezvoltatorilor. Pentru publicul larg sunt puse la dispoziție platforme specializate pentru crearea chat-boților. În acest sens, platforma *ChatCompose* oferă posibilitatea de a crea chat-boți pentru site-uri, fără a avea nevoie de cunoștințe tehnice temeinice.

În acest sens este necesar de înregistrat pe platforma *ChatCompose* (https://www.chatcompose.com/ro/web.html). *ChatCompose* este o platformă de chat-bot care oferă o vastă varietate de cazuri de utilizare și canale de integrare, inclusiv chat-uri pentru web. Instalarea unui chat-bot pe site-ul instituției ori

companiei rezolvă multe probleme și acoperă multe necesități. Totul procesul este automatizat și, în consecință, se va optimiza serviciul, economisind timp și bani. De aici și necesitatea și importanța de a ști cum să creezi un chat-bot.

La etapa inițială, în procesul de constituire a chat-botului, dezvoltatorii și managerii de comunicare ar trebui să acorde suficient timp, în special în faza de consultare, pentru a clarifica următoarele puncte:

- Ce obiectiv ar trebui să aibă un chat-bot (de marketing)?
- Cum poate chat-botul să atingă acest obiectiv?
- Ce personalitate ar trebui să aibă chat-botul?
- Care este tonul corect al vocii?
- Cum creez o conversație valoroasă?
- Cum răspund la întrebări la care chat-bot-ul meu nu poate răspunde?
- Am nevoie de un chat-bot simplu sau de un chat-bot pentru un mediu mai complex?
- Care sunt subiectele pe care botul ar trebui să le acopere?
- Vreau să ofer o conversație ghidată?

Astfel, un chat-bot fiind o interfață text (uneori audio) poate fi dezvoltat, luând în considerare aspectele punctate mai sus, prin scripturi și/sau întrebări și răspunsuri pregătite din timp. Chat-botul poate utiliza inteligența artificială pentru a răspunde la întrebări, pentru a primi comenzi, pentru a efectua sarcini în mod automat și pentru a oferi toate serviciile pentru care a fost instruit.

Care va fi evoluția roboților de chat? Roboții de chat, la fel ca alte instrumente de IA, vor fi utilizați pentru a îmbunătăți capacitățile umane și a elibera timpul oamenilor, pentru a deveni mai creativi și inovatori, acordând mai mult timp activităților strategice decât celor tactice.

3.8. Ce reprezintă chat-bot-urile pentru educație?

Chat-bot-urile educaționale fac învățarea ușoară și accesibilă, cu accente de gamificare și care fac procesul de studiere mai captivant și mai interesant.

Experții în educația digitală susțin că mulți utilizatori sunt foarte deschiși către chat-boții educaționali. Roboții WhatsApp, de exemplu, sunt utilizați pe larg în educație, acolo unde utilizarea internetului mobil este mult mai răspândită decât internetul desktop.

Chat-ul GPT, https://openai.com/blog/chatgpt/, elaborat de compania *Open AI,* este un soft de inteligență artificială care folosește tehnici de învățare automatizată, utilizând sistemul de feedback din partea interlocutorilor. În timp real, Chat-ul GPT învață să-și aprecieze și să-și evalueze propriile răspunsuri, șlefuind soluțiile propuse pe parcurs ca să devină tot mai exacte.

Chat-ul GPT se autoprezintă în felul următor:

„GPT (Generative Pre-training Transformer) este un model de limbaj automatizat care poate genera text de înaltă calitate în diferite limbi și poate fi utilizat pentru o varietate de scopuri, cum ar fi traducerea automată, scrierea de texte, răspunsul la întrebări etc. Chat GPT ar putea fi un termen folosit pentru a descrie utilizarea unui model GPT pentru a răspunde la mesaje într-un chat. De exemplu, un chat-bot poate utiliza un model GPT pentru a genera răspunsuri la întrebările utilizatorilor într-un chat de pe site-ul web al unei companii sau într-o aplicație de mesagerie”.

Studenții și elevii au înțeles foarte repede că abilitatea și capacitatea Chat GPT de a genera, la solicitare, texte cu un nivel înalt de calitate și exactitate a faptelor, poate fi explorată cu succes la realizarea diverselor proiecte și tratarea subiectelor propuse pentru lucrul individual. În contextul dat, de exemplu, autoritățile responsabile de educație din New York și Los Angeles, ca urmare a îngrijorărilor privind influența nefastă asupra procesului educațional, au decis să interzică profesorilor și studenților din școlile publice implementarea acestui soft în activitatea educațională.

Lansat la 30 noiembrie, anul 2022, ChatGPT reprezintă o nouă generație de softuri de inteligență artificială. ChatGPT poate conversa în scris cu persoana care îi adresează întrebări, genera texte lizibile și veridice, crea imagini și videoclipuri originale bazate pe cărți digitale și scrieri online. Spre deosebire de softurile anterioare, precum GPT-3, lansat în 2020, actualul soft ChatGPT este conceput să fie disponibil pentru toți utilizatorii care au conexiune la internet. În luna decembrie din anul 2022, milioane de cetățeni au conversat cu ChatGPT, solicitând ajutorul la scrierea textelor, compunerea poeziilor, cântecelor ori la redactarea unor eseuri. Solicitările și interogările respective contribuie zilnic la creșterea performanțelor ChatGPT astfel încât el devine mai inteligent, mai calitativ, mai exact și mai utilizat.

Referitor la întrebarea care ține de utilizarea ori neutilizarea ChatGPT, autorii consideră că în raport cu softul respectiv sunt necesare următoarele abordări:

1. Softul ChatGPT nu poate și nu trebuie să fie ignorat ori exclus total din procesul educațional. Capacitățile „intelectuale” ale acestuia sunt de invidiat.
2. Formularea temelor individuale de cercetare pentru elevi și studenți ar trebui să fie foarte exacte, cu accente în exclusivitate pe aspectele punctate de profesori, care să scoată în evidență personalitatea și cunoștințele elevului. Softul ChatGPT, de regulă, tratează problema cu mult mai amplu și mai larg.
3. Softul ChatGPT, în unele cazuri este oportun să genereze anumite soluții, neștiute de elevi și studenți, dar răspunsurile respective ar trebui să fie interpretate, tratate, verificate de însăși utilizatorii. Softul poate să propună de exemplu, traducerea unui text dintr-o limbă în alta, dar numai utilizatorul acceptă varianta finală din perspectiva propriilor viziuni și cunoștințe.
4. În unele situații ar fi necesar și rațional să se compare răspunsurile Softului GhatGPT cu cele livrate de elevi ori studenți. În acest fel vor fi testate rezultatele nu numai din punct de vedere academic dar și al corectitudinii abordării subiectului dat de către studenți.

Într-o perspectivă apropiată, atunci când sistemele de inteligență artificială vor fi îmbinate cu avantajele tehnologiei 5G, utilizatorii, cetățenii simpli, companiile private și instituțiile publice, vor beneficia de chat-boți cu performanțe îmbunătățite, cum ar fi soluțiile propuse, recomandările și previziunile făcute. Cercetările respective se află la stadiul inițial și se vor dezvolta rapid și eficient odată cu creșterea accesului la internet, dezvoltării componentelor IA, inclusiv creșterea performanțelor de procesare a limbajului natural și învățare automată (NLP și ML), astfel încât fiecare persoană de pe mapamond va putea deține un asistent digital personal funcțional și eficient.

CAPITOLUL 4. CALCULUL EVOLUTIV. DEZVOLTAREA ALGORITMILOR GENETICI

În acest capitol sunt tratate unele aspecte ale calcului evolutiv și fundamentele teoretice-practice ale Algoritmilor Genetici, care sunt construiți pe principiul „supraviețuiește cel mai bine adaptat", enunțat de Charles Darwin. În lucrare sunt descrise caracteristicile de bază ale algoritmului genetic, evidențiind avantajele și dezavantajele acestuia. Sunt examinate problemele care cad sub incidența algoritmului genetic. De asemenea, Algoritmul Genetic este examinat din perspectiva soluționării problemelor în care găsirea soluției optime nu este simplă ori cel puțin ineficientă datorită caracteristicilor căutării probabilistice. Sunt arătate etapele în care Algoritmii Genetici codifică o posibilă soluție la o problemă specifică într-o unică structură de date numită „cromozom" și pregătesc terenul pentru a aplica operatorii genetici la aceste structuri astfel încât să mențină informațiile critice.

4.1. Dezvoltarea calculului evolutiv

Abordarea privind aplicarea principiilor evolutive (calculul evolutiv) în soluționarea automată a problemelor (Problem Solving) datează cu mult timp mai înainte comparativ cu apariția și dezvoltarea calculatoarelor moderne.

Alan Turing, încă în anul 1948, lansează o abordarea nouă aplicată la soluționarea problemelor numită abordare evolutivă ori genetică. Ulterior, în anii 1960, Dr. ***Lawrence Jerome Fogel*** (2 martie 1928 –18 februarie 2007), pionier al calcului evolutiv, cât și Wlash (mai târziu David B. Fogel, născut la 2 februarie 1964) introduceau și dezvoltau conceptul de programare evolutivă. În aceeași perioadă Holland se concertează pe algoritmii genetici. Iar Hans-Paul Schwefel (născut la 4 decembrie 1940), un informatician german și profesor emerit la Universitatea din Dortmund și Ingo Rechenberg (20 noiembrie 1934 - 25 septembrie 2021), cercetător și profesor german în domeniul bionicii, lansează și

dezvoltă strategiile evolutive ca modalități alternative de soluționare automată a problemelor. Ceva, mai târziu, în anii 1990, J. R. Koza dezvoltă programarea genetică, o nouă tehnică de căutare a soluțiilor.

Așa dar, *calculul evolutiv* este un domeniu al informaticii moderne, cu accent puternic în matematică, inspirat din procesul evoluției naturale, iar conceptul de bază, care ține de calculul evolutiv, este interconexiunea evoluție naturală – tehnica de rezolvare a problemelor de tip experiment-eroare.

În contextul celor menționate, în prezent, un domeniu important de cercetare al informaticii îl reprezintă calculul evolutiv. După cum se știe, domeniul respectiv își „trage seva", adică își are rădăcinile în procesul evoluției naturale. Algoritmii care apar și se dezvoltă în acest domeniu se numesc *algoritmi evolutivi* și ei includ subdomenii importante de mare perspectivă precum:

- *Programare evolutivă;*
- *Strategii evolutive;*
- *Programare genetică;*
- *Algoritmi genetici.*

Algoritmii evolutivi presupun existența într-un mediu examinat, a unor indivizii constituiți într-o populație, care intră în competiție pentru a supraviețui, a se adapta și a se reproduce. Abilitatea indivizilor de a se adapta în mediul în care activează este corelată cu șansele lor de supraviețuire și reproducere și determină evoluția populației în timp. În contextul modalității de rezolvare a problemelor, populația este modelată de operatorii genetici. Indivizii cel mai bine adaptați, după mai multe iterații a generațiilor, vor determina soluțiile problemei examinate. Mai jos descriem succint fiecare din domeniile menționate.

Programarea evolutivă. Programarea evolutivă este una dintre cele patru paradigme majore de algoritm evolutiv. Este similar cu programarea genetică, dar structura programului de optimizat este fixă, în timp ce parametrii numerici ai acestuia sunt lăsați să evolueze. Programarea evolutivă a fost folosit pentru prima dată de Lawrence J. Fogel în SUA în 1960 pentru a utiliza evoluția simulată ca proces de învățare care urmărește generarea de inteligență artificială. Fogel a folosit mașini cu stări finite ca predictori și le-a dezvoltat. În prezent, a devenit mai greu de distins de strategiile evolutive. Principalul operator al programării evolutive este mutația.

Strategii evolutive. Strategiile evolutive sunt algoritmi evolutivi care datează din anii 1960 și care sunt cel mai frecvent aplicați problemelor de optimizare a cutiei negre în spațiile de căutare continuă. Inspirat de evoluția biologică,

formularea lor originală se bazează pe aplicarea mutației, recombinării și selecției în populații care reprezintă soluții candidate. Din punct de vedere algoritmic, strategiile evolutive sunt metode de optimizare care eșantionează noi soluții candidate stocastic, cel mai frecvent dintr-o probabilitate normală multivariată distribuită. Cele mai proeminente două principii de proiectare ale lor sunt *imparțialitatea și controlul adaptiv* al parametrilor a distribuției eșantionului.

Programarea genetică. Programarea genetică este o metodă independentă de domeniu care generează genetic o populație de programe pe calculator pentru a rezolva o problemă. Mai exact, programarea genetică transformă în mod iterativ o populație de programe pe calculator într-o nouă generație de programe prin aplicarea operațiilor genetice care apar în mod natural. În inteligența artificială, programarea genetică (GP) se consideră o tehnică evolutivă a programelor, pornind de la o populație de programe mai puțin adecvate (de obicei aleatorii), potrivite pentru o anumită sarcină prin aplicarea unor operațiuni similare proceselor genetice naturale populației de programe. Operațiunile aplicate sunt: selectarea celor mai potrivite programe pentru reproducere (*încrucișare*), replicare și/sau *mutație* în dependență de *funcția de fitness* predefinită. Operația de încrucișare implică schimbarea părților specificate ale perechilor selectate (părinți) pentru a produce descendenți noi și diferiți care devin parte din noua generație de programe. Unele programe care nu sunt selectate pentru reproducere sunt copiate din generația curentă în generația nouă. Mutația implică înlocuirea unei părți aleatorii a unui program cu o altă parte aleatorie a unui alt program. Apoi selecția și alte operațiuni sunt aplicate recursiv noii generații de programe. Elementele populației sunt de cele mai multe ori structuri arborescente. În mod obișnuit, membrii fiecărei generații noi sunt mai apți decât membrii generației precedente, iar programul cel mai bun din generație este adesea mai bun decât programele cele mai bune din generațiile anterioare. Încetarea evoluției are loc de obicei atunci când un program individual atinge un nivel predefinit de competență sau de fitness.

Algoritm genetic. Algoritmul genetic este o metodă care se referă la rezolvarea problemelor de optimizare și care se bazează pe selecția naturală, încrucișări și mutații și reprezintă procesul care repetă evoluția biologică. Algoritmul genetic modifică în mod repetat o populație de soluții individuale.

Referitor la algoritmul genetic, pe parcursul acestei lucrări, vom examina în profunzime mai multe aspecte care ține de dezvoltarea și aplicarea algoritmului.

În genere, ținem să menționăm că un algoritm evolutiv este considerat o componentă a calculului evolutiv în inteligența artificială. Un algoritm evolutiv funcționează prin intermediul procesului de selecție în care sunt excluși membrii cei mai puțin adaptați (cei mai puțin în formă) din grupul de populație examinat, în timp ce membrii apți supraviețuiesc și sunt luați în considerație până când sunt determinate soluții mai bune. Cu alte cuvinte, algoritmii evolutivi sunt aplicații computerizate care imită procesele biologice pentru a rezolva probleme complexe. În timp, membrii de succes evoluează pentru a prezenta soluția optimizată a problemei. Algoritmii evolutivi utilizează concepte din biologie, cum ar fi selecția, reproducerea și mutația.

Un algoritm evolutiv are o populație de soluții candidate, spre deosebire de metodele clasice, care încearcă să mențină o singură soluție cât mai bună. Există numeroase beneficii asociate cu algoritmi evolutivi. Unul dintre cele mai mari avantaje vine în câștigurile de flexibilitate, deoarece majoritatea conceptelor de algoritm evolutiv sunt adaptabile chiar și la probleme complexe. Există câteva dezavantaje asociate algoritmilor evolutivi. Altfel, soluția oferită de un algoritm evolutiv este mai bună în comparație cu alte soluții cunoscute. Ca atare, algoritmul nu poate dovedi că nici o soluție obținută este total optimă, doar că este optimă în comparație cu celelalte rezultate.

4.2. Inteligența Artificială în Slujba Binelui

În data de ***7 iulie***, ***2023***, la ***Geneva*** s-a desfășurat summit-ul ***„AI for Good”*** (*„Summitul Global de Inteligență Artificială în Slujba Binelui*"), organizat de Uniunea Internațională a Telecomunicațiilor (UIT, engl. - International Telecommunication Union - ITU), o instituție din cadrul ONU, specializată în noile tehnologii. În cadrul acestui summit au participat circa 3000 de persoane, alături de umanoizii echipați cu inteligență artificială, pentru a discuta dacă ar putea roboții inteligenți să conducă lumea mai bine decât oamenii.

Menționăm faptul că summit-ul global *„AI for Good”* este principala platformă a Națiunilor Unite de promovare a Inteligenței Artificiale pentru soluționarea la nivel global a diferitor priorități, precum problemele climaterice, egalitatea de gen, sănătatea, incluziunea socială și infrastructura durabilă. Acest

summit a fost organizat de UIT, în colaborare cu aproximativ 40 de agenții ONU și guvernul Elveției.

Pe parcursul desfășurării summit-ului, experții în domeniul AI au discutat despre necesitatea elaborării unor reguli clare, care ar putea să garanteze că inteligența artificială va fi utilizată doar în scopuri benefice pentru omenire, precum lupta împotriva foametei sau împotriva schimbărilor climatice.

Secretarul general al UIT, Doreen Bogdan-Martin a avertizat că:

„Realizările și cercetările din domeniul IA, în special cele care privesc inteligența artificială generativă, sunt în progres, iar ONU a cerut crearea unor reguli și limitări, pentru ca aceste tehnologii să aducă beneficii pentru omenire, fără să o pună în pericol. În absența unor astfel de reglementări, IA riscă să ne facă să trăim un adevărat coșmar”.

Totodată, roboții umanoizi consideră că oamenii ar trebui să dea dovadă de prudență în fața inteligenței artificiale și au recunoscut că nu stăpânesc deocamdată emoțiile umane.

Sophia, un robot dezvoltat de compania „Hanson Robotics”, întrebată despre capacitatea roboților umanoizi de a conduce lumea, a punctat:

„Roboții umanoizi pot să conducă cu un nivel de eficiență mai mare decât cel al liderilor umani ... Noi nu avem aceleași prejudecăți sau emoții care pot uneori să afecteze adoptarea deciziilor și putem să analizăm cu rapiditate mari cantități de date pentru a lua cele mai bune decizii", a adăugat robotul.

Ideile scoase în evidență de umanoidul Sophia sunt, în linii mari, ideile și caracteristicile tuturor roboților umanoizi. Cum s-a ajuns la o așa etapă importantă în dezvoltarea IA?

O serie de probleme care cad sub incidența inteligenței artificiale sunt prea complexe pentru a fi soluționate cu tehnici directe și în contextul dat se aplică metode de căutare corespunzătoare, evident, împreună cu tehnicile directe de ghidare a căutării disponibile. Tehnicile respective pot fi descrise independent de orice domeniu de probleme.

Un rol important în dezvoltarea inteligenței artificiale, și în special în domeniul învățării automate (Machine Learning) îl au Algoritmii Genetici. Algoritmii genetici reprezintă tehnici adaptive de căutare euristică, ce se bazează pe principiile geneticii și ale selecției naturale și care pornesc de la idea că „cel mai bine adaptat – supraviețuiește". Mecanismul este similar procesului de evoluție naturală, proces ce se bazează pe principul că „doar speciile care se adaptează mai bine la mediu sunt capabile să supraviețuiască și să evolueze în timp, iar acelea care, în urma selecției naturale, nu reușesc să se adapteze vor dispărea". Șansele că o specie să supraviețuiască în timp, peste generații, este în dependență directă de gradul de adaptare. Dacă ar fi să trecem în limbaj matematic aceasta ar însemna că cu cât este mai mare gradul de adaptare cu atât soluția se apropie de optim. Așa dar, un algoritm genetic reprezintă un model matematic care imită modelul biologic evoluționist pentru a soluționa probleme de optimizare ori căutare. Cum susținea Charles Darwin: *„Speciile care supraviețuiesc nu sunt speciile cele mai puternice și nici cele mai inteligente, ci cele care se adaptează cel mai bine la schimbare."* În contextul dat, mai jos, vom explica funcționarea unuia din cei mai eficienți algoritmi în inteligența artificială – algoritmii genetici.

4.3. Algoritmii genetici. Generalități

Algoritmii genetici (AG), sunt algoritmi de calcul evoluționist, inspirați din Teoria Evoluției lui Darwin. În anul 1960, Ingo Rechenberg (20 noiembrie 1934 - 25 septembrie 2021) a introdus idea calculului evoluționist în lucrarea intitulată „Evolution strategies". Ingo Rechenberg a fost un cercetător și profesor german în domeniul bionicii. Rechenberg a fost un pionier în domeniile calculului evolutiv și al evoluției artificiale. În anii 1960 și 1970 el a inventat mai multe metode de optimizare cunoscute sub numele de strategii de evoluție (din germană Evolutionsstrategie). Echipa de cercetători coordonată de el a aplicat cu succes algoritmii elaborați la probleme de optimizare, inclusiv la proiectarea aerodinamică a aripilor. Acestea au fost primele aplicații tehnice serioase ale evoluției artificiale, un compartiment important care ține de dezvoltarea bionicii și inteligenții artificiale.

În anul 1975, John Henry Holland (2 februarie 1929 – 9 august 2015) un om de știință american și profesor de psihologie și inginerie electrică și informatică la Universitatea din Michigan a introdus și a analizat un model matematic, care prin

intermediul unor proceduri adaptive, pentru a găsi soluția unei probleme de optimizare, se baza pe un mecanism de selecție naturală și evoluție genetică numit algoritm genetic. El a fost un pionier în ceea ce a devenit cunoscut sub numele de algoritmi genetici.

Deseori, în cazul problemelor pentru care soluția optimă implică căutări printre toate combinările, permutările sau aranjările probabilistice, proces foarte complex și uneori ineficient, se utilizează algoritmul genetic. Acesta va implementa structuri de date specifice numite „cromozomi" pentru a codifica cu ajutorul operatorilor genetici soluția posibilă a unei probleme particulare, păstrând informațiile importante.

De obicei, pentru soluționarea unei probleme ce implică un algoritm genetic, se identifică așa numita „populație" ce se construiește aleatoriu pe baza „mulțimii inițiale de soluții posibile". Fiecare individ sau „cromozom" (un șir de caractere, exprimat printr-o consecutivitate de biți), din „populația" examinată, reprezintă o soluție posibilă a problemei. Prin parcurgerea unor iterații consecutive, are loc evoluția „cromozomilor" ce se realizează la nivel de „generație", validarea fiecăreia din care se face printr-o funcție de evaluare, numită *fitness*. Utilizând unul din cei trei operatori genetici principali: ***selecția***, ***crossover***-ul și ***mutația***, se generează „cromozomi" noi, pentru viitoarea populație, identificați din generația curentă, ca cei mai „eficienți". Astfel, exact ca și în biologie, cei mai „puternici cromozomi", cu o mai mare probabilitate, sunt *selectați* din generația dată, pentru a transmite prin copiere caracteristicile lor (valorile „funcției de evaluare", conform cerințelor problemei) generației viitoare, astfel asigurând perpetuarea întregului proces. Cu ajutorul operatorul genetic de încrucișare (*crossover*) are loc combinarea informațiilor a doi indivizi („părinți") din populația curentă pentru a genera unul sau câțiva descendenți. *Mutația* în schimb, permite modificarea aleatorie a unei gene sau a unei secțiuni mici din „cromozom" pentru a asigura diversitatea la viitoarea populație.

Astfel, succesul algoritmilor genetici este asigurat de implementarea lor la soluționarea unui șir de probleme NP-complete, a căror soluții nu pot fi identificate prin metode iterative, ci prin obținerea soluției optime la nivel global [1-10].

Din cele expuse mai sus, pot fi scoase în evidență următoarele ***avantaje*** ale algoritmilor genetici:

- algoritmii genetici sunt mai puternici în raport cu algoritmii clasici de optimizare și mai eficienți comparativ cu metodele de căutare dirijată;

- algoritmii genetici identifică așa numita „populație”, construită din mulțimea soluțiilor posibile, pe când în mod tradițional căutarea se răsfrânge asupra unei soluții din mulțimea de căutare;
- algoritmii genetici interferează căutarea în mod aleatoriu cu căutarea în mod controlat și potrivit teoriei probabilităților pot fi numiți „algoritmi probabiliști”;
- algoritmii genetici echilibrează cercetarea „spațiului stărilor” și identificarea soluțiilor optime;
- algoritmii genetici se răsfrâng asupra codului „spațiului de căutare” și nu direct asupra acestuia;
- algoritmii genetici prevalează prin facilitatea utilizării asupra algoritmilor tradiționali ce solicită îndeplinirea unui șir de caracteristici importante ale funcției de evaluare (funcție continuă, derivabilă, convexă etc.). Probabilitatea de identificare a soluției optime de către algoritmii geneticii este mare;
- algoritmii genetici nu necesită informații derivate (care ar putea să nu fie disponibile pentru multe probleme din lumea reală);
- algoritmii genetici sunt mai rapizi și mai eficienți în comparație cu metodele tradiționale. Optimizează atât funcțiile continue și discrete, cât și problemele multi-obiective;
- algoritmii genetici oferă întotdeauna o listă de soluții „bune” și nu doar o singură soluție;
- algoritmii genetici oferă întotdeauna un răspuns la problemă, care se îmbunătățește în timp și este util atunci când spațiul de căutare este mare și există un număr mare de parametri implicați.

Totuși algoritmii genetici presupun și unele ***dezavantaje***, precum:

- Algoritmii genetici nu sunt potriviți pentru toate problemele, în special pentru problemele care sunt simple și pentru care sunt disponibile informații derivate.
- Valorile funcției de fitness sunt calculate la fiecare populație, în mod repetat, ceea ce poate fi costisitor din punct de vedere computațional pentru unele probleme.
- Fiind o metodă stocastică, nu există garanții asupra optimității sau calității soluției.
- Algoritmul genetic dacă nu este implementat corespunzător poate să nu conveargă către soluția optimă.

4.4. Terminologia utilizată

În algoritmii genetici indivizii dintr-o populație sunt reprezentați de cromozomi cu seturi codificate de parametrii sarcinii, de ex. soluții, care altfel sunt numite puncte în spațiu de căutare (puncte de căutare). În unele lucrări, indivizi se numesc organisme. În acest sens vom clarifica următoarele noțiuni biologice, împrumutate de informaticieni din perspectiva algoritmilor genetici:

- ✓ ***Cromozomii*** (lanțuri sau secvențe de cod) sunt secvențe ordonate de gene.
- ✓ ***O genă*** (numită și proprietate, semn sau detector) este elementul atomic al genotipului, în special al cromozomilor.
- ✓ ***Genotipul*** sau structura este setul de cromozomi ai unui individ dat. În consecință, indivizii unei populații pot fi genotipuri sau cromozomi unici (într-un caz destul de comun, când genotipul este format dintr-un singur cromozom).
- ✓ ***Fenotipul*** este un set de valori care corespund unui anumit genotip. Ori, structura decodificată sau set de parametri ai sarcinii {soluție, punct de spațiu de căutare}.
- ✓ ***O alelă*** este valoarea unei anumite gene, definită și ca valoarea proprietății sau varianta proprietății.
- ✓ ***Locusul*** sau poziția indică locația unei anumite gene în cromozom (lanț). Mulțimea de poziții ale genelor reprezintă locusuri.
- ✓ ***Genom*** – totalitatea materialului genetic al unui organism sau al unei specii, care determină dezvoltarea, funcționarea și transmiterea trăsăturilor ereditare ale organismului de la o generație la alta.
- ✓ ***Individ*** – o entitate unică care are un set specific de cromozomi moșteniți de la părinți. În cadrul algoritmilor genetici un individ reprezintă o posibilă soluție sau o combinație a parametrilor care poate fi optimizată în timp prin procesele de selecție și recombinație și care este evaluată în cadrul unei probleme specifice.
- ✓ ***Populație*** – un grup de indivizi ce împărtășesc un set comun de caracteristici genetice ce ocupă un anumit tip de mediu. Variația genetică în cadrul populațiilor este importantă pentru adaptarea la schimbările de mediu.

- ✓ ***Mapping*** – funcție esențială de evaluare ce are rolul de a atribui o valoare numerică fiecărui individ din populația dată, reflectând calitatea sau adecvarea soluției respective în cadrul problemei de optimizare. Procesul de mapping se mai numește și morfogeneză.

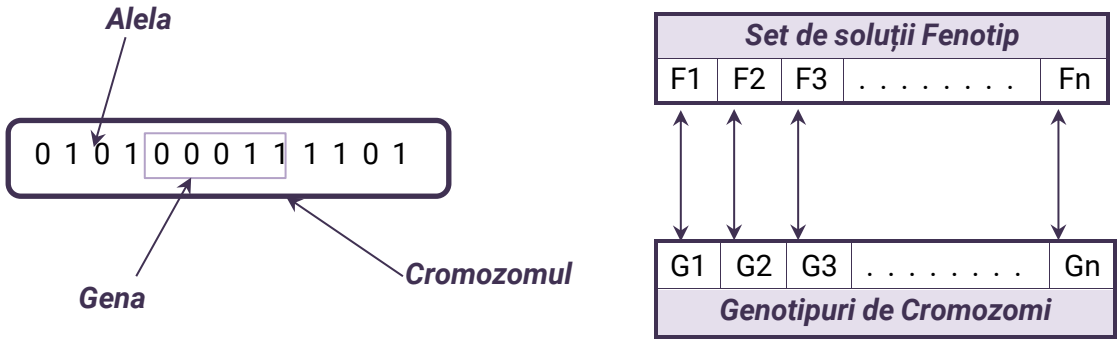


Figura 1. Schema genelor și alelor

Un concept foarte important în algoritmii genetici este funcția care măsoară gradul de adaptivitate cunoscută și sub denumirea de ***funcție de fitness*** *(fitness function)*.

Funcția fitness reprezintă o măsură a adaptabilității unui individ dat în cadrul fiecărei generații. Această caracteristică permite evaluarea gradului de adaptare al indivizilor din populație și se alege dintre cei mai adaptați, adică pe cei cu cele mai mari valori ale funcției de fitness, în conformitate cu evoluția principiul supraviețuirii celui mai „puternic” (cel mai bine adaptat).

Funcția de fitness și-a primit și numele direct din genetică. Are un impact puternic asupra funcționării algoritmilor genetici și trebuie să aibă o precizie și definiție corectă. În problemele de optimizare, funcția de fitness, de regulă, este optimizată (maximizată ori minimizată) și se numește funcție obiectiv.

În probleme de minimizare funcția obiectiv este transformată și problema se reduce la maximizare. În teoria controlului, funcția de fitness poate lua forma unei funcții de eroare, iar în teoria jocurilor - funcția de cost. La fiecare iterație a algoritmului genetic gradul de adaptivitate (fitness-ul) a fiecărui individ dintr-o anumită populație este estimat cu ajutorul funcției de fitness și, pe această bază, se generează următoarea generație (populație de indivizi) care va constitui posibila mulțime de soluții în cazul problemei examinate.

4.5. Aplicații ale algoritmilor genetici

Algoritmii genetici sunt adesea utilizați în informatică, inteligență artificială etc., pentru a găsi soluții complexe și neevidente la probleme de optimizare și de căutare algoritmică. Mai jos vom puncta unele aplicații ale algoritmilor genetici.

A) ***Probleme de optimizare***. Optimizarea este procesul prin care se obțin cele mai bune rezultate în anumite ipoteze. Optimizarea poate fi definită ca fiind procesul prin care sunt determinate condițiile care conduc la valoarea extremă (maximă sau minimă) a unei funcții, eventual supuse unor restricții. Pentru formularea problemei de optimizare se parcurg mai multe ***etape***:

- se selectează una sau mai multe variabile de optimizare;
- se alege funcția obiectiv;
- se identifică setul de constrângeri.

Sunt utilizate pe scară largă și alte probleme de optimizare, precum: programarea lucrărilor, optimizarea calității sunetului, problema comis-voiajorului și altele.

B) ***Învățare automata***. Cercetătorul John McCarthy definește IA, încă în anul 1956, în felul următor: „IA implică calculatoare care pot îndeplini sarcini ce sunt caracteristice inteligenței umane”. Domeniul care ține de Machine Learning este o modalitate de a realiza Inteligența Artificială. O definiție foarte potrivită a învățării automate (ML), formulată de Arthur Samuel, datează încă din anul 1959: „Machine Learning este domeniul de studiu care oferă calculatoarelor capacitatea de a învăța fără a fi programate în mod explicit”. Problemele tipice rezolvate de Machine Learning se referă la: *regresie, clasificare, segmentare, analiza rețelei*. În ultima perioadă Inteligența artificială, pentru a soluționa unele probleme de importanță majoră, are nevoie de instrumente și tehnici de intervenție care țin de Big Data și Machine Learning. Și în contextual dat Algoritmii Genetici joacă un rol extrem de important. Algoritmii genetici sunt folosiți pentru a rezolva problemele legate de *clasificare*, *predicție*, *crearea de reguli pentru învățare* și altele.

C) ***Modelul sistemului imunitar***. Algoritmii genetici sunt utilizați și pentru modelarea diferitelor aspecte ale sistemului imunitar pentru genele individuale și familiile multigenice în timpul evoluției.

D) ***Alte exemple de probleme*** rezolvate cu ajutorul ***algoritmilor genetici*** se referă la: identificarea soluției optime sau aproximative pentru problema proiectării oglinzilor folosite în sistemele de concentrare solară, optimizarea formei și performanței antenelor în comunicații wireless, proiectarea de dispozitive complexe în domeniul Computer-Aided Design, dezvoltarea mișcărilor realiste ale personajelor sau obiectelor animate în mediul virtual în scopul optimizării parametrilor mișcărilor astfel încât acestea să arate cât mai natural și să îndeplinească obiectivele specifice ale animației, proiectarea optimă a corpurilor aerodinamice în câmpuri de curgere complexe. De asemenea, algoritmii genetici se aplică în procesul de prelucrare și restaurarea a imaginilor, dar și un rol important în ultima perioadă îl joacă în procesul de prognoză a piețelor financiare.

4.6. Codificarea cromozomilor

Căutarea soluțiilor optime sau aproximative pentru o problemă specifică prin intermediul unui algoritm genetic implică un șir de componente ale acestuia, printre care, principale sunt:

1) *codul problemei*
2) *funcția fitness (de potrivire, sau de evaluare).*

Aceste două componente diferă de la o problemă la alta, în dependență de cerințele acesteia. Așa cum s-a arătat mai sus, cromozomii conțin structuri de date (informații) ce permit identificarea soluției problemei și elaborarea codului acestei soluții.

Cromozomii pot lua diverse forme, sau coduri, printre care:

A) ***Codificarea binară***. Cromozomul este format din șiruri de 0 sau 1 care reprezintă binar soluția problemei.

1	0	0	1	1	1	0	1

B) ***Codificarea prin valoare***. Cromozomul este format dintr-un șir de valori vector reale ori întregi care pe ansamblu reprezintă soluția problemei.

0.2	0.1	0.4	0.5	0.7	0.8	0.6	0.9

De exemplu, în situația când este necesar să codificăm Nord, Sud, Est ori Vest, vom utiliza doar 4 valori întregi {1, 2, 3, 4}. Astfel:

1	3	4	2	1	4	3	2

C) ***Codificare cromozomilor prin intermediul permutărilor***. În multe probleme, soluția ține de o reprezentare de ordine a elementelor. În astfel de cazuri, reprezentarea prin intermediul permutării este cea mai potrivită. Un exemplu clasic al acestei reprezentări ține de problema comis-voiajorului. Astfel, comis-voiajorul trebuie să facă un tur al tuturor orașelor, vizitând fiecare oraș exact o dată și să revină în orașul de plecare. Distanța totală a turului trebuie redusă la minimum. Soluția în acest caz reprezintă o ordonare sau o permutare a tuturor orașelor numerotate și, prin urmare, utilizarea reprezentărilor care reprezintă permutări are sens pentru această problemă.

1	3	4	6	5	2	7	1

4.7. Formularea matematică a problemei optimizării

Când este necesar să rezolvăm o problemă, aplicând algoritmii genetici, examinăm anumite soluții „mai bune” obținute la o anumită populație, care țin de o generație mai avansată, față de soluțiile identificate anterior. Setul soluțiilor posibile (fezabile) pentru o anumită problemă constituie „spațiul de căutare” sau „spațiul stărilor”. Fiecare individ din populația algoritmului genetic reprezintă o stare sau o soluție candidată din acest spațiu al stărilor. În esență, spațiul stărilor cuprinde toate configurațiile și combinațiile posibile ale soluțiilor care pot fi evaluate folosind funcția de fitness. În cazul când simpla căutare în spațiul imens al stărilor unei probleme specifice presupune un algoritm foarte complicat, costisitor, atunci se recurge la algoritmi genetici, care și-au demonstrat eficiența în acest sens.

Sub aspect matematic, o astfel de problemă se formulează, după cum urmează:

Fie funcția $g(x_1, x_2, \ldots, x_n): S \rightarrow R$, unde $S = T_1 \times T_2 \times \ldots \times T_n$ reprezintă spațiul stărilor. Să se identifice soluția $s^* = (x_1^*, x_2^*, \ldots, x_n^*) \in S$ astfel încât pentru aceasta, funcția g va atinge minimum (maximum). Mulțimile $T_i \leq R$, unde $i = 1, \ldots, n$. Deci $S \leq R^n$.

Deseori, este suficient să se identifice soluția optimizată s_{optim} sau cea aproximativă, nu neapărat soluția exactă s^*, astfel încât: $|s^* - s_{optim}| \leq \varepsilon$ ori $|g(s^*) - g(s_{optim})| \leq \varepsilon$, unde ε reprezintă exactitatea solicitată pentru soluție.

Exemplu. Pentru a determina maximul funcției $f(x) = x^2 - 1$, unde x ia valori întregi de la 0 la 15, este necesar de clarificat esența matematică a mai multor concepte, din perspectiva algoritmului genetic, examinate anterior.

Astfel, mulțimea {0, 1,...,15} constituie spațiul de căutare și în același timp - un set de soluții potențiale la problemă. Fiecare dintre cele 16 valori, ce aparțin acestei mulțimi, reprezintă punct al spațiului de căutare, decizie, valoare a parametrului, sau fenotip. De remarcat că se numește soluția care optimizează funcția cea mai bună sau soluția optimă. Codificăm valorile x de la 0 la 15 după cum urmează:

0 *0000*	1 *0001*	2 *0010*	3 *0011*	4 *0100*	5 *0101*	6 *0110*	7 *0111*
8 *1000*	9 *1001*	10 *1010*	11 *1011*	12 *1100*	13 *1101*	14 *1110*	15 *1111*

Aceasta este o metodă de codificare binară bine cunoscută, asociată cu scrierea cifrelor zecimale în sistemul binar. Secvențele de cod, prezentate mai sus, sunt numite și ***lanțuri*** sau ***cromozomi***. În exemplul dat, aceștia acționează și ca ***genotipuri***. Fiecare dintre cromozomi este format din 4 gene (altfel putem spune că secvențele binare constau din 4 biți). Valoarea genei într-o poziție specifică se numește ***alelă*** care ia valoarea 0 sau 1. Populația este formată din indivizi selectați dintre acești 16 cromozomi.

Un exemplu de populație egală cu 6 poate fi, de exemplu, o multitudine de cromozomi {0101, 0111, 1001, 1100, 1110, 1111}, care este o formă codificată a următoarelor fenotipuri: {5, 7, 9, 12, 14, 15}. Funcția de fitness în exemplu respectiv este dată de expresia $f(x) = x^2 - 1$. Fitness-ul cromozomilor dintr-o populație este determinată de valoarea acestei funcții. Valorile respectivilor cromozomi corespund anumitor genotipuri. Apropo, valoarea maximă a funcției $f(x) = x^2 - 1$, unde x ia valori întregi de la 0 la 15, este 224.

4.8. Esența Algoritmului Genetic

În literatura de specialitate, Algoritmul Genetic presupune realizarea unor pași (propuși inițial de John Henry Holland), care ar trebui să finalizeze cu o soluție optimă a problemei examinate.

Pasul 1. Crearea/generarea populației inițiale;
Pasul 2. Evaluarea valorii funcției de fitness (funcției de adecvare) a fiecărui individ (mecanism utilizat pentru a măsura și evalua starea unui cromozom);
Pasul 3. Selecția – luând în considerare caracteristicile fiecărui individ, în cadrul acestei etape, unii indivizi se pot reproduce mai des decât alții;
Pasul 4. Crossover-ul (Încrucișarea);
Pasul 5. Mutația;
Pasul 6. Substituirea populației vechi de cromozomi cu noua populație de cromozomi;
Pasul 7. Găsirea celei mai bune soluții (dar, în cazul în care criteriile de optimizare nu sunt îndeplinite, atunci metoda impune întoarcerea la pasul 2 și selectarea în final a celui mai bun individ, ca soluție finală).

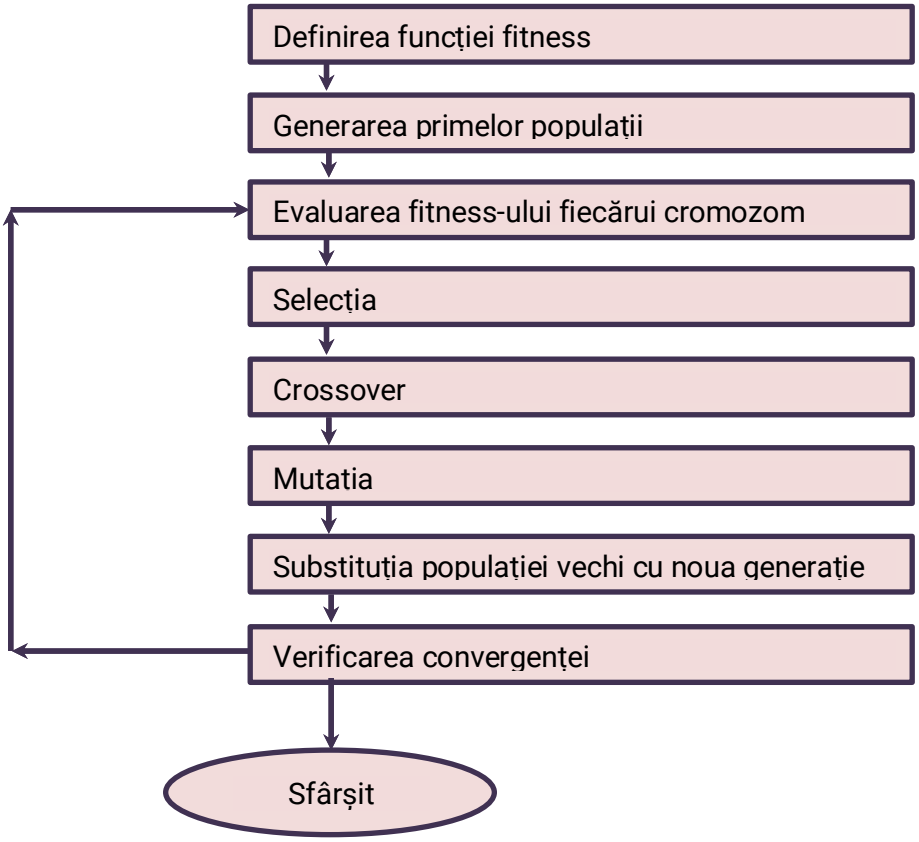


Figura 2. Schema logică a Algoritmului Genetic

Se inițiază astfel un fond genetic aleatoriu prin crearea unui set de cromozomi conform unui șablon predefinit, unde valorile tuturor genelor sunt selectate în mod aleatoriu pentru fiecare cromozom. Acești cromozomi inițiali corespund indivizilor din populația inițială. De regulă, la diferite generații, constant rămâne numărul de indivizi (și implicit de cromozomi) din populație, fapt ce nu

este valabil mereu. Algoritmul Genetic pornește cu un set de soluții admisibile, numit „populație” (creată, după cum am menționat, în mod arbitrar), fiecare dintre ele reprezentând o potențială soluție a problemei, numită „cromozom”.

După ce populația este stabilită, aceasta evoluează spre soluții mai bune, prin diferite procese genetice (*selecția, încrucișarea, mutația*) care conduc la o mai bună valoare a funcției de adecvare, utilizată pentru a evalua starea fiecărui cromozom.

Algoritmii genetici generează o nouă populație, formată din indivizi cu caracteristici mai bune și mai adecvați mediului decât cei ai populației anterioare. Schema logică care ține de implementarea Algoritmului Genetic este prezentată în figura 2.

4.9. Mecanisme de generare a populaţiei

Algoritmii genetici operează pe o populație de indivizi, în care fiecare individ poate să reprezinte o posibilă soluție la o problemă examinată.

Populația se evaluează din perspectiva măsurii propriei „aptitudinii” de a se adapta „condițiilor impuse”. Astfel, în procesul de evaluare se clarifică cât de eficient este un organism din populația respectivă în competiția pentru resursele existente. Indivizii ineficienți din acest punct de vedere pier, iar cei eficienți supraviețuiesc și se „reproduc”. Afirmația respectivă reformulată în termeni matematici este echivalent cu următoarele: cât de „bună”, „optimă” este soluția identificată problemei care îi corespunde.

Prin intermediul „încrucișării” cu alți indivizi din populația examinată, indivizii eficienți sunt capabili să „reproducă” descendenți, care pregătesc terenul pentru apariția unei noi generații de indivizi care, totodată, păstrează ori combină unele din caracteristicile moștenite de la părinți.

În cursul evoluției indivizii „slabi”, cei mai puțin apți, vor avea o probabilitate mai mică să se poată ”reproduce”. În așa mod, în procesul de trecere de la o generație la alta, indivizii respectivi vor dispărea treptat din populație, iar proprietățile pe care le posedau vor dispărea treptat din caracteristicile populației. În anumite situații apar mutații sau modificări spontane ale genelor.

Încrucișarea celor mai eficienți și adaptați indivizi duce la faptul că cele mai promițătoare caracteristici ale spațiului de căutare sunt moștenite. În așa fel, din

generație în generație, caracteristicile „bune”, aplicând concepte și noțiuni din biologie, informatică și matematică, sunt distribuite în întreaga populație.

De la o generație la alta, populația de indivizi va converge treptat către o soluție optimă a problemei. Avantajul unui Algoritm Genetic este că găsește soluții optime aproximative într-un timp relativ scurt.

CAPITOLUL 5.
IMPLEMENTAREA ALGORITMILOR GENETICI. ABORDĂRI PRACTICE

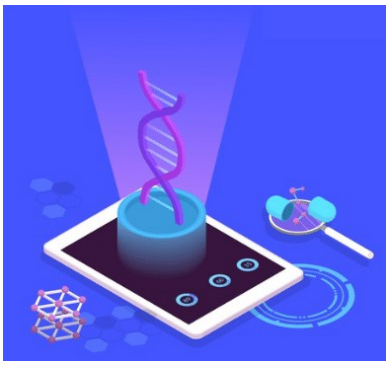
În acest capitol autorii examinează implementarea conceptului de evoluție al lui Darwin, adaptat la funcționarea Algoritmului Genetic, în scopul găsirii unei soluții optime, exprimată prin intermediul funcției de fitness. Conceptul de evoluție a lui Darwin presupune generarea populațiilor (generații) de cromozomi, din care sunt selectați cei mai „eficienți", cei mai „adaptabili" cromozomi din generația curentă. În felul acesta, cromozomii noi sunt formați aplicând unul din cei trei operatori genetici (ori toți trei)i: ***selecția***, ***crossover*** și ***mutația***. În acest sens autorii examinează cele mai populare metode care țin de implementarea operatorilor genetici menționați mai sus. Este examinată o aplicație concretă a Algoritmului Genetic și totodată sunt cercetate condițiile de stopare ale algoritmului respectiv.

5.1. Algoritmul genetic și Teoria Evoluției

Algoritmii genetici sunt inspirați din Teoria Evoluției, dezvoltată de Charles Darwin și care se bazează pe principiul că „specia care se poate adapta cel mai bine la schimbare – supraviețuiește". Mecanismul de funcționare a algoritmului genetic este similar evoluției naturale. Tot din Teoria Evoluției rezultă că supraviețuirea speciei poate fi menținută prin procesul de selecție, reproducere, încrucișare și mutație. În felul acesta conceptul de evoluție al lui Darwin este adaptat la funcționarea algoritmului genetic pentru a găsi soluții la o problemă exprimată prin intermediul funcției de fitness (*funcție obiectiv ori funcție de adaptare*).

O soluție generată de un algoritm genetic se numește cromozom, iar un set de cromozomi este numit populație. Un cromozom, după cum se știe, este compus din gene, iar valoarea sa poate fi fie numerică, binară, simboluri sau caractere, în dependență de problema care se rezolvă.

Populația examinată, alcătuită din cromozomi, va fi supusă la fiecare ciclu unei evaluări. Acest fapt presupune calcularea, prin intermediul funcției de fitness, fitness-ul fiecărui cromozom. În felul acesta se determină „posibilele soluții” ori „cât de aproape sunt soluțiile identificate în cadrul populației respective” în raport cu soluția optimă a problemei examinate. În așa mod se clarifică cât de eficienți, „adaptabili” sunt cromozomii din populația examinată.



Conceptul de evoluție a lui Darwin presupune generarea următoarei populații (generații) de cromozomi, din care sunt selectați cei mai „eficienți”, cei mai „adaptabili” cromozomi din generația curentă. În felul acesta cromozomii noi sunt formați aplicând unul din cei trei operatori genetici (ori toți trei)i: ***selecția***, ***crossover*** și ***mutația***.

Astfel, unii cromozomi din populație se vor împerechea printr-un proces numit ***crossover*** (încrucișare), producând noi cromozomi numiți descendenți, a căror compoziție genică este o combinație a părinților. Într-o generație, câțiva cromozomi pot să treacă prin procesul de mutație, ceea ce presupune modificări în locațiunea genelor. De obicei, numărul de cromozomi care vor suferi încrucișări și mutații este controlat de rata de încrucișare și valoarea ratei de mutație. Cromozomul din populația care se va menține pentru următoarea generație, va fi selectat pe baza regulii evoluției darwiniene, cromozomul care are o valoare de fitness mai „bună” va avea o probabilitate mai mare de a fi selectat din nou în generația următoare.

După câteva generații, valoarea cromozomilor va converge către o anumită valoare de optim care este cea mai „bună soluție” pentru problema cercetată. În continuare vom examina trei operatori genetici importanți: ***selecția***, ***crossover*** și ***mutația*** [1-10].

Reamintim că ***selecția*** reprezintă alegerea indivizilor cu cea mai bună aptitudine pentru reproducere (sortarea după valoarea funcției obiective). Cu cât este mai bună fitness-ul unui individ, cu atât sunt mai mari șansele sale de a se încrucișa și de a-și moșteni genele de către generația următoare. Așa dar, selecția reprezintă modul în care anumiți indivizi ies din populația examinată și trec în altă populație (generație). Alegerea unui procedeu de ieșire din populație ține de problema examinată. În așa mod, este firesc, ca pentru o problema de optimizare, clasificarea populației sa se facă în funcție de valorile corespunzătoare ale cromozomilor. Mai jos vom examina unele metode care țin de procesul de selecție:

- *Selecția cu ajutorul ruletei ponderate;*
- *Selecția după rang;*
- *Selecția după eșantionarea universală stocastică;*
- *Selecția în conformitate cu scalarea liniară a funcției fitness;*
- *Selecția turneu;*
- *Elitism.*

5.2. Selecția cu ajutorul ruletei ponderate (Fitness proportionate selection (FPS))

Una dintre cele mai răspândite modalități de selecție a părinților este selecția proporțională de fitness. Din această perspectivă, fiecare individ poate deveni părinte cu o probabilitate proporțională cu propriul fitness. Din aceste considerente, indivizii mai apți, mai în formă, au șanse mai mari de a se împerechea și de a-și promova propriile trăsături la generația următoare. Strategia dată de selecție pune presiune pe indivizii mai apți din populația respectivă, favorizându-i în așa fel să evolueze în timp.

Cum vom interpreta matematic acest fenomen biologic? Considerăm un cerc împărțit în *n* sectoare (porțiuni), unde *n* reprezintă numărul de indivizi din populație. Astfel, fiecărui individ i se atribuie un sector din cerc care este proporțional cu valoarea sa de fitness. În așa mod fiecărui cromozom i se va pune în corespondență un sector al ruletei, care este direct proporțional, ca mărime, cu fitness-ul cromozomului respectiv. Prin urmare cromozomilor cu fitness mai mare li se vor atribui sectoare mai mari, iar celor cu fitness mic, li se vor pune în corespondență sectoare mai mici. Evident, atunci când aleatoriu li se va arunca o bilă pe ruletă, există probabilitate mai mare de alegere pentru cromozomii cu fitness mare. Simularea ruletei se realizează în felul următor:

- se calculează suma tuturor fitness-urilor cromozomilor. Fie suma este S;
- se generează un număr aleatoriu între 1 și S. Fie numărul aleatoriu este r;
- se parcurge populația ordonată crescător după fitness și se calculează suma fitness-urilor cromozomilor parcurși până se depășește valoarea r.
- se alege cromozomul la care s-a ajuns.

Exemplu 1. Examinăm tabelul de mai jos care conține informația pentru 6 cromozomi.

Cromozomii	*Funcția de fitness*	*Proporția*	*N*
A	10	9%	3
B	14	12,7%	4
C	30	27,2%	5
D	9	8%	2
E	40	36%	6
F	7	6,6%	1
Total	S=110	100%	21

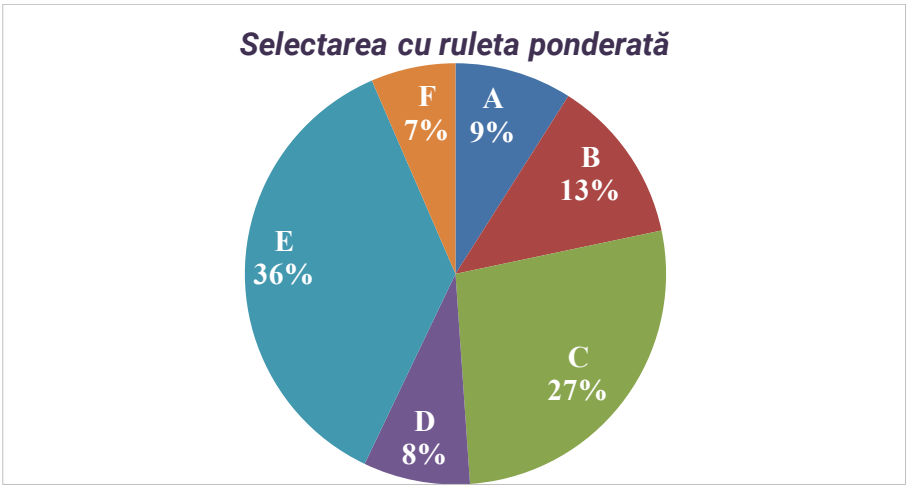


Figura 3. Metoda ruletei ponderate

Se aruncă o bilă pe ruleta și se selectează cromozomul unde se oprește. În mod clar, cromozomii cu valoare de fitness mai mare vor fi selectați de mai multe ori. Aruncând, de exemplu, bila de 6 ori, aleatoriu se obțin următoarele sectoare, deci și cromozomii care corespund sectoarelor respective: A, C, C, E, B, E.

5.3. Selecția după rang (Rank Selection)

În cazul în care fitness-ul cromozomilor examinați diferă deosebit de mult, selecția cu ajutorul ruletei ponderate nu este foarte eficientă și în contextul dat nu este recomandată pentru a fi aplicată. Astfel, de exemplu, dacă suma fitness-urilor unor cromozomi examinați este de 150 de ori mai mică comparativ cu fitness-ul unui cromozom fixat, atunci, evident, probabilitatea cromozomilor cu fitness mic este aproape de zero. Pentru a ***exclude*** această situație, se procedează în felul următor:

- cromozomii se ordonează crescător în funcție de fitness;

- ulterior cromozomii se renumerotează cu numere întregi din intervalul $[1, .., n]$, unde n - este dimensiunea populației, iar cromozomul cu fitness-ul cel mai mic are numărul 1 etc., iar cromozomul cu fitness-ul cel mai mare va avea numărul egal cu dimensiunea populației;
- numerele respective vor fi considerate fitness-uri și în situația dată se aplică selecția prin metoda ruletei ponderate.

Exemplu 2. Examinăm tabelul de mai jos care conține informația pentru 6 cromozomi.

Cromozomii	*Funcția de fitness*	*Rangul*	*Proporția rangului*
A	10	3	14,3%
B	14	4	19%
C	30	5	23,8%
D	9	2	9,5%
E	40	6	28,6%
F	7	1	4,8%
Total	S=110	21	100%

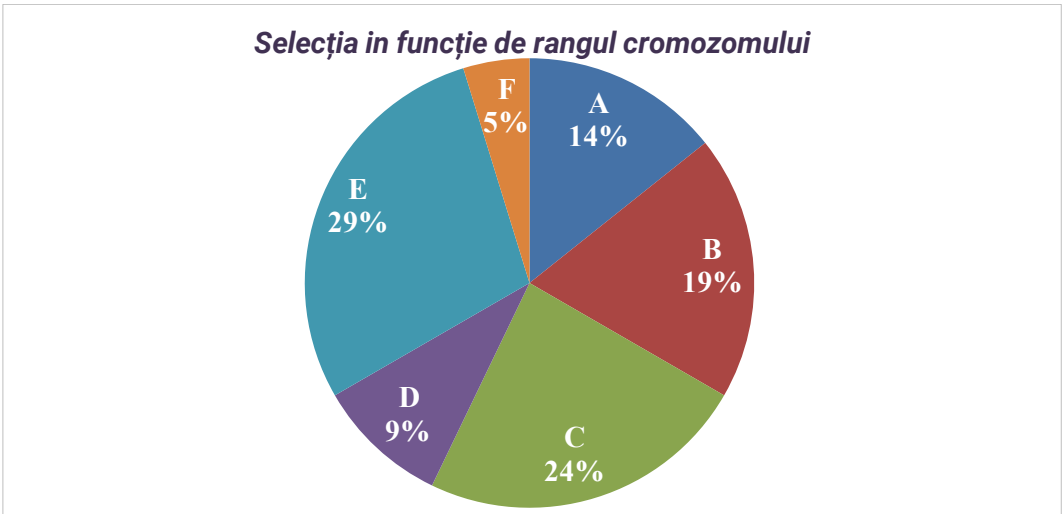


Figura 4. Selecția după rang

Și în această situație sesizăm că cromozomii cu fitness-ul cel mai mare au la fel cele mai mari șanse de a fi selectați, dar de data aceasta nu mai sunt așa mari în comparație cu șansele celorlalți. Aruncând, de exemplu, bila de 6 ori, aleatoriu, se obțin următoarele sectoare, deci și cromozomii care corespund sectoarelor respective: D, C, B, E, A, C.

Așa dar, dacă o populație inițială conține unul sau doi indivizi foarte apți, dar nu cei mai buni din anumite considerente și restul indivizilor populației nu sunt atât

de apți, atunci indivizi cei mai apți vor domina rapid întreaga populație și vor împiedica populația să exploreze alți indivizi potențial mai buni. O dominație atât de puternică provoacă o pierdere foarte mare a diversității genetice, ceea ce cu siguranță nu este avantajos pentru procesul de optimizare. În acest sens:

1. Selectarea rangului clasează mai întâi populația și apoi fiecare cromozom primește fitness din acest clasament.
2. Cel mai puțin apt va avea fitness 1, al doilea cel mai rău 2 etc. Iar cel mai apt va avea fitness n (numărul de cromozomi din populație).
3. După aceasta, toți cromozomii au șanse mai mari de a fi selectați.
4. Schemele de selecție bazate pe rang pot evita convergența prematură.
5. Dar poate fi costisitor din punct de vedere computațional, deoarece sortează populațiile în funcție de valoarea de fitness.
6. Însă această metodă poate duce la o convergență mai lentă, deoarece cei mai buni cromozomi nu diferă atât de mult de alții.

5.4. Selecția după eșantionare universală stocastică (Stochastic Universal Sampling (SUS))

Eșantionarea universală stocastică (***Stochastic Universal Sampling (SUS)***) este destul de similară cu ruleta. Cu toate acestea, în loc să avem un singur punct fix, avem mai multe puncte fixe, așa cum se arată în figura 5.

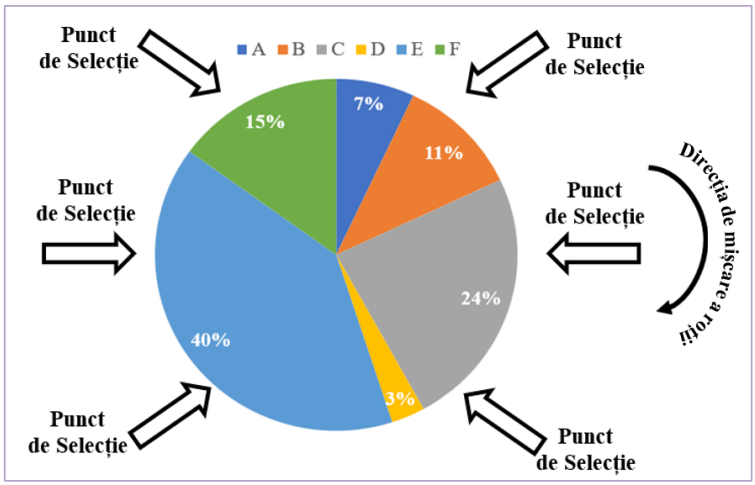


Figura 5. Selecția după eșantionare

Prin urmare, toți părinții sunt aleși într-o singură rotire a roții. De asemenea, o astfel de configurație „încurajează” și alți cromozomi să fie aleși cel puțin o dată.

Exemplu 3. La o singură rotire a ruletei sunt selectați 6 părinți (figura 5).

5.5. Selecția în conformitate cu scalarea liniară a funcției fitness (Linear Fitness Scaling in Genetic Algorithm)

Așa cum s-a menționat anterior, un aspect important al algoritmilor genetici este funcția de fitness, care cuantifică calitatea fiecărei soluții candidate. Funcția de fitness atribuie o valoare numerică, numită și scor de fitness sau valoare de fitness, fiecărui individ din populație.

În unele cazuri, se dorește să se implementeze scalarea liniară a fitness-ului fiecărui cromozom în algoritmul examinat. Scalare liniară de fitness este o tehnică care urmărește îmbunătățirea presiunii de selecție în Algoritmul Genetic prin scalarea liniară a valorilor de fitness.

Într-un algoritm genetic tipic, valorile funcției de fitness ale indivizilor din populație variază de la zero la o anumită valoare pozitivă. Scalarea liniară a fitness-ului urmărește să transforme aceste valori de fitness într-un nou interval, (între zero și unu ori între n_1 și n_2, unde n_1 și n_2 sunt numere reale), într-o manieră liniară. În așa fel toate valorile fitness-ului conținute pe un anumit interval trec în alte valori ale fitness-ului care se conțin pe alt interval.

Motivația scalării liniare a funcției de fitness ține de creșterea presiunii de selecție în Algoritmul Genetic. Prin scalarea liniară a valorilor de fitness, indivizilor cu scoruri de fitness mai mari li se oferă un avantaj mai mare în timpul procesului de selecție. Acest lucru poate ajuta în cazurile în care valorile de fitness nu sunt bine distribuite la nivelul populației sau când este nevoie să se efectueze o ierarhizare a soluțiilor cu fitness mai mare.

Exemplu 4. Examinăm tabelul de mai jos care conține informația pentru 6 cromozomi și valorile funcției de fitness $\boldsymbol{f}$:

Cromozomi		*Funcția de fitness f*	*Funcția de fitness f^**
A	1	10	Nu se cunoaște
B	2	14	Nu se cunoaște
C	3	30	Nu se cunoaște
D	4	9	Nu se cunoaște
E	5	40	130
F	6	7	80.5
Total		S=110	

Este necesar să se efectueze scalarea liniară a funcției de fitness pentru ca să transforme, să „treacă” valorile funcției de fitness 10, 14, 30, 9, 40, 7 pe un nou interval [80.5, 130], unde vom nota: $d_{min} = 80.5$ și $d_{max} = 130$.

Pentru a efectua operația respectivă vom utiliza relația $f^*(x_i) = a \cdot f(x_i) + b$, unde $i = 1,2,3,4,5,6$ și relațiile: $a \cdot f_{min} + b = d_{min}$; $a \cdot f_{max} + b = d_{max}$, pentru care vom avea următorul sistem:

$$\begin{cases} a \cdot f_{min} + b = d_{min} \\ a \cdot f_{max} + b = d_{max} \end{cases} \quad (1)$$

În cazul problemei noastre avem $f_{min} = 7$ și $d_{min} = 80.5$ iar $f_{max} = 40$ și $d_{max} = 130$. Substituind datele respective în sistemul (1) si rezolvând sistemul dat obținem: $a = 1.5$ iar $b = 70$. În continuare calculăm noile valori $f^*(x_i) = a \cdot f(x_i) + b$. Astfel: $f_1^* = 1.5 \cdot 10 + 70 = 85$; $f_2^* = 1.5 \cdot 14 + 70 = 91$; $f_3^* = 1.5 \cdot 30 + 70 = 115$; $f_4^* = 1.5 \cdot 9 + 70 = 83.5$; iar f_5^* și f_6^* sunt deja calculate.

Cromozomii		*Funcția de fitness f*	*Funcția de fitness f**	*Proporția nouă f**	*Proporția veche f*
A	1	10	85	14.5%	9%
B	2	14	91	15.6 %	12,7%
C	3	30	115	19.7%	27.2%
D	4	9	83.5	14.3%	8%
E	5	40	130	22.2%	36%
F	6	7	80.5	13.7%	6.6%
Total		S=110	585	100%	100%

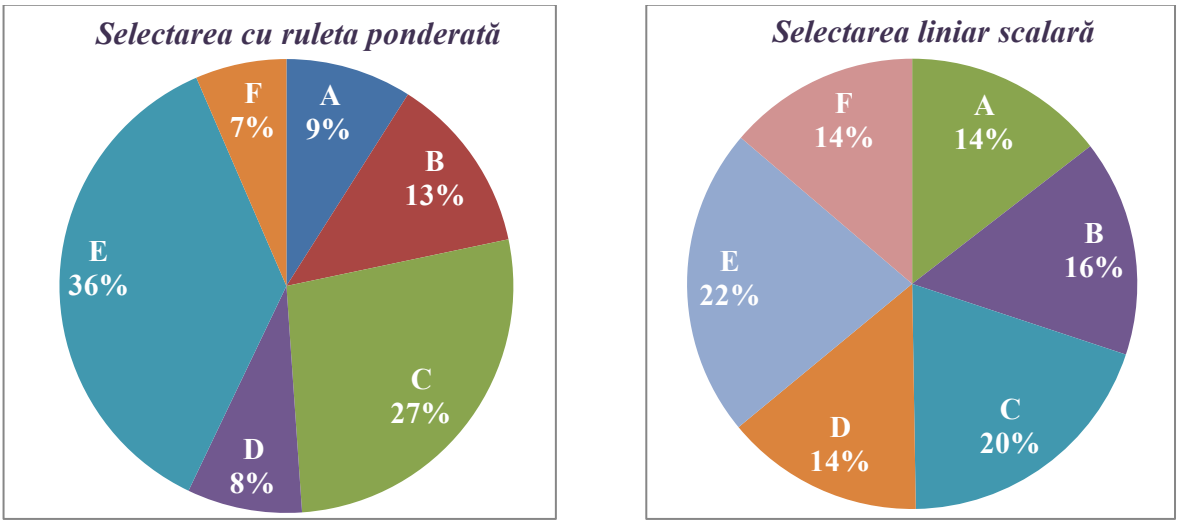


Figura 6. Analiza comparativă dintre 2 metode de selecție

Observăm că după efectuarea scalării liniare $f^* = 1.5 \cdot f + 70$ diagrama a devenit mai uniformă. În acest caz probabilitatea de a fi selectați cromozomii mai puțin apți este mai mare. Iar pe de altă parte cromozomii mai apți nu vor fi dominanți în noua populație. În așa fel, valorile de fitness sunt distribuite uniform la nivelul populației. Iar algoritmul genetic va converge mai sigur către soluția

optimă căutată. În diagramele de figura 6, se poate vede clar diferența de rezultate dintre selectarea cu ruleta ponderată și selectarea liniar scalară.

5.6. Selecția turneu (Tournament Selection)

Prin intermediul ***selecției turneu*** se alege in mod aleatoriu k indivizi din populație iar dintre aceștia doar cei mai buni j sunt selectați pentru a deveni părinți. Aceeași procedură se repetă pentru selectarea următorului părinte. Procedura se repetă până se obține numărul dorit de indivizi. Este cea mai eficientă metodă din punct de vedere al complexității de timp. Din punct de vedere al presiunii de selecție se aseamănă cu selecția bazată pe rang. Selecția turneului este, de asemenea, extrem de populară, deoarece poate funcționa chiar și cu valori negative de fitness.

Exemplu 5. Prin intermediul selecției turneu sunt selectați inițial în mod aleatoriu 3 cromozomi, iar ulterior, la etapa finală, cel mai bun.

Cromozomii	***A***	***B***	***C***	***D***	***E***	***F***	***G***
Valoarea de fitness	15	10	7	6	9	14	8

Dimensiunea turneului este egală cu 3. În mod aleatoriu sunt selectați 3 cromozomi.

Cromozomii	***B***	***D***	***G***
Valoarea de fitness	10	6	8

Cromozomul cu cea mai bună valoare de fitness este B, deci cromozomul învingător este B.

Cromozomul învingător	***B***
Valoarea de fitness	10

5.7. Elitism

În cazul când se constituie o nouă populație prin mutații și încrucișare, persistă probabilitatea de a pierde cei mai buni cromozomi. Pentru a evita această situație, se recomandă trecerea directă, copierea, fără a fi modificați, a celor mai buni cromozomi din vechea populație în noua populație. Procesul dat se numește ***elitism*** și el contribuie semnificativ la creșterea performanțelor

algoritmilor genetici. Elitismul, a fost introdus de Ken DeJong (1975) și reprezintă o schemă adițională a oricărui mecanism de selecție al cărei scop este de a reține cei mai buni $\boldsymbol{k}$ indivizi la fiecare generație. Astfel de indivizi ar putea dispărea în caz contrar din cauză că nu au fost selectați pentru supraviețuire sau fiindcă au fost distruși prin aplicarea operatorilor. Studii experimentale au dovedit că de multe ori elitismul îmbunătățește semnificativ performanța algoritmului genetic.

Exemplu 6. Prin intermediul selecției de tip elitism sunt selectați cei mai buni $n = 3$ cromozomi.

Cromozomii	*A*	*B*	*C*	*D*	*E*	*F*	*G*
Valoarea de fitness	15	10	7	6	9	14	8

Dimensiunea selecției de tip elitism este $n = 3$. În așa mod:

Cromozomii	*A*	*F*	*B*
Valoarea de fitness	15	14	10

5.8. Încrucișarea (crossover)

Operatorul de încrucișare este analog cu reproducerea și încrucișarea biologică și, de obicei, se aplică asupra indivizilor din populația intermediară. Sunt selectați doi indivizi din populația intermediară și anumite porțiuni din cei doi cromozomi sunt interschimbate. Astfel, prin intermediul operatorului de încrucișare se produc unul sau mai mulți descendenți, folosind materialul genetic al părinților (cromozomilor). În felul acesta, operatorul de încrucișare imită încrucișarea inter-cromozomială naturală. De regulă, încrucișarea nu generează descendenți aleatorii și încrucișarea depinde de tipul de codificare a cromozomilor și de problema particulara care este în proces de rezolvare. În continuare examinăm următoarele procedee de încrucișare:

- încrucișare într-un singur punct;
- încrucișarea în două puncte;
- încrucișarea în k-puncte;
- încrucișare uniformă;
- încrucișare ordonată;
- recombinare aritmetică totală;
- încrucișare mixtă.

5.8.1. Încrucișare într-un singur punct (One Point Crossover)

Se alege un punct de încrucișare. De la primul părinte este copiată secvența de la început până la punctul de încrucișare, iar din al doilea părinte este copiată secvența de la punctul de încrucișare până la final.

Exemplu 7. Punctul de tăietură este pe poziția a patra. Urmașul 1 moștenește secvențele $1,0,0,1$ de la Părintele 1 și secvențele $1,1,0$ de la Părintele 2. Urmașul 2 moștenește secvențele $1,1,0,0$ de la Părintele 2 și secvențele $1,0,1$ de la Părintele 1.

Părinte 1	1	0	0	1	1	0	1

Urmaș 1	1	0	0	1	1	1	0

Părinte 2	1	1	0	0	1	1	0

Urmaș 2	1	1	0	0	1	0	1

5.8.2. Încrucișare în două puncte (Two Points Crossover)

Sunt alese două puncte de încrucișare. Secvența dintre cele două puncte este aleasa dintr-unul din părinți, iar ce a rămas din celălalt părinte.

Exemplu 8. Considerăm că punctele de tăietură sunt pe poziția a doua și a cincea. Urmașul 1 moștenește secvențele $1,0$ și $0,1$ de la Părintele 1 și secvențele $0,0,1$ de la Părintele 2. Urmașul 2 moștenește secvențele $1,1$ și $1,0$ de la Părintele 2 și secvențele $0,1,1$ de la Părintele 1.

Părinte 1	1	0	0	1	1	0	1

Urmaș 1	1	0	0	0	1	0	1

Părinte 2	1	1	0	0	1	1	0

Urmaș 2	1	1	0	1	1	1	0

5.8.3. Încrucișare în k-puncte (Multi Point Crossover)

Sunt alese $k-$ puncte de încrucișare. Secvența dintre cele $k-$ puncte sunt alese în baza principiilor enunțate mai sus.

Exemplu 9. Considerăm că punctele de tăietură sunt pe poziția a doua, a patra și a șasea. Urmașul 1 moștenește secvențele $1,0$ și $1,0$ de la Părintele 1 și secvențele $1,0$, și 0 de la Părintele 2. Urmașul 2 moștenește secvențele $1,1$ și $1,1$ de la Părintele 2 și secvențele $0,1$ și 1 de la Părintele 1.

Părinte 1	1	0	0	1	1	0	1

Urmaș 1	1	0	1	0	1	0	0

Părinte 2	1	1	1	0	1	1	0

Urmaș 2	1	1	0	1	1	1	1

5.8.4. Încrucișare uniformă

În procesul încrucișării uniforme, nu se împarte cromozomul în segmente, ci mai degrabă se tratează fiecare genă separat. În acest sens, în mod aleatoriu pentru fiecare cromozom se decide separat dacă va fi inclus sau nu în cromozomul descendent. De asemenea, se permite de realizat încrucișarea foarte calculat, cu intenție clară și în cazul când este necesar de moștenit mai mult material genetic de la un părinte.

Exemplu 10. Punctele de tăietură se iau aleatoriu ori foarte calculat, cu o intenție clară. Urmașul 1 moștenește secvența $1,3$ și secvențele 5 și 6 de la Părintele 1 și secvențele $5,7$ și 3 de la Părintele 2. Urmașul 2 moștenește secvența $2,1$ și secvențele 4 și 3 de la Părintele 2 și secvențele $4,2$ și 1 de la Părintele 1.

Părinte 1	1	3	4	2	5	1	6

Urmaș 1	1	3	5	7	5	3	6

Părinte 2	2	1	5	7	4	3	3

Urmaș 2	2	1	4	2	4	1	3

5.8.5. Încrucișare ordonată

În procesul încrucișării ordonate se ține cont de faptul că toate cifrele care reprezintă Părinții cromozomi trebuie să fie diferite. De exemplu, în cazul examinării problemei comis voiajorului, toate orașele, numerotate de la 1 la n, vor fi parcurse doar o singură dată. Deci, toate sunt diferite. În acest caz Urmașii la fel trebuie să posede toate numerele respective dar, în același timp, este necesar ca ele să fie diferite.

Exemplu 11. Punctele de tăietură se iau în funcție de scopul propus. Urmașul 1 moștenește secvența $1,3$ și secvențele 2 și 6 de la Părintele 1, iar secvențele $5, 7, 4$ de la Părintele 2. Urmașul 2 moștenește secvențele 7 și 1, cât și secvențele 6 și 3 de la Părintele 2, iar secvențele $4, 2, 5$ de la Părintele 1.

Părinte 1	1	3	4	2	5	7	6

Urmaș 1	1	3	5	7	4	2	6

Părinte 2	2	1	5	7	4	6	3

Urmaș 2	7	1	4	2	5	6	3

5.8.6. Recombinare aritmetică totală (Whole Arithmetic Recombination)

Această metodă este folosită pentru cromozomii care sunt reprezentați prin intermediul numerelor reale. În cazul dat se ia media ponderată a celor doi părinți în baza următoarelor formule:

- $Urmaş\ 1 = \alpha \cdot x + (1 - \alpha) \cdot y$, unde x aparține Părintelui 1, iar y aparține Părintelui 2.
- $Urmaş\ 2 = \alpha \cdot x + (1 - \alpha) \cdot y$, unde x aparține Părintelui 2, iar y aparține Părintelui 1.

Coeficientul $\alpha = 0, \ \dots, \ 1$.

Exemplu 12. Evident, dacă $\alpha = 0.5$, atunci ambii urmași vor fi identici, așa cum se arată în diagramele de mai jos.

P1	0.5	0.5	0.4	0.4	0.3	0.3

U1	0.35	0.2	0.3	0.35	0.25	0.3

P2	0.2	0.3	0.2	0.3	0.2	0.3

U2	0.35	0.2	0.3	0.35	0.25	0.3

5.8.7. Încrucișare mixtă (Blend Crossover)

Luând în considerare valoarea cromozomilor pentru Părintele 1 și Părintele 2, aplicând încrucișarea mixtă, obținem valorile pentru Urmașii 1 și 2, care se iau aleatoriu din intervalul:

$$[x_1 - \alpha(x_2 - x_1), x_2 + \alpha(x_2 - x_1)] \ (2),$$

unde $x_1 < x_2$, și x_1 ține de Părintele 1, iar x_2 ține de Părintele 2. În caz că nu se respectă condiția respectivă, se va lua $x_1 < x_2$, unde x_1 ține de Părintele 2, iar x_2 ține de Părintele 1. Coeficientul $\alpha = 0, \ \dots, \ 1$. Se recomandă ca $\alpha = 0.5$.

Exemplu 13. Dacă $\alpha = 0.5$, atunci determinând intervalele (2), pentru fiecare pereche $x_1 < x_2$, aleatoriu alegem valorile pentru Urmașii 1 și 2, așa cum se vede în diagramele de mai jos.

P1	- 2.7	4.3	1.2	-0.1	1.5

U1	-5.3	3.2	- 0.7	0.9	2.7

P2	6.8	-5.4	-2.2	1.3	3.5

U2	-2.9	- 0.3	- 1.3	0.7	2.1

Metodele de încrucișare prezentate mai sus sunt destul de generale. De obicei sunt alese metode specifice problemei care se dorește a fi rezolvată și care generează o populație mai bună.

5.9. Mutația (Mutation)

În termeni simpli, mutația poate fi definită ca o mică modificare aleatorie a cromozomului, pentru a obține o nouă soluție. Mutația este utilizată pentru menținerea și introducerea diversității în populația genetică și se aplică de obicei cu o probabilitate scăzută. Mutația este parte din Algoritmul Genetic care este legată de „explorarea" spațiului de căutare. Din practică, s-a observat că mutația este esențială pentru convergența Algoritmului Genetic. Mutația se utilizează în principal pentru a evita căderea soluțiilor într-un optim local.

În această secțiune, descriem unii dintre cei mai des utilizați operatori de mutație. La fel ca operatorii de încrucișare, aceasta nu este o listă exhaustivă, iar implementatorii Algoritmului Genetic ar putea găsi o combinație a abordărilor examinate ori un operator de mutație specific problemei mai util. Mai jos vom examina câțiva cei mai des utilizați operatori care țin de mutație:

- mutația de inversare a biților;
- mutația Swap;
- mutația Scramble;
- mutația inversă.

5.9.1. Mutația de inversare a biților (Bit Flip Mutation)

Acest operator realizează o inversare a biților în cadrul cromozomilor examinați. Astfel, sunt selectați unul sau mai mulți biți aleatorii și ulterior sunt inversați, în loc de 0 se scrie 1, și în loc de 1 se scrie 0.

Cromozom	1	0	0	1	1	0

Mutația	1	0	1	0	0	0

5.9.2. Mutația Swap (Swap Mutation)

În mutația de schimbare (Swap), sunt selectate aleatoriu două poziții pe cromozomi și se schimbă valorile între ele. Tipul dat de mutație se aplică în codificările bazate pe permutare.

Cromozom	1	5	6	1	2	3

Mutația	1	3	6	1	2	5

5.9.3. Mutația Scramble (Scramble Mutation)

Mutația *Scramble* este, de asemenea, populară cu reprezentările de permutare. În cromozomul examinat se alege o submulțime de gene și valorile lor sunt amestecate după dorință ori aleatoriu.

Cromozom	1	5	6	1	2	3

Mutația	1	6	5	2	1	3

5.9.4. Mutația inversă (Inversion Mutation)

Operatorul de mutație inversă acționează în felul următor: se alege o secvență de gene și ulterior submulțimea respectivă este pur și simplu inversată în cromozomul examinat.

Cromozom	1	5	6	1	2	3

Mutația	1	2	1	6	5	3

În funcție de problema examinată, luând în considerare aspectele specifice, pe parcursul realizării Algoritmului Genetic, implementatorii ar putea aplica și alți operatori care țin de mutație (unii chiar inventați) pentru a asigura convergența soluției către valoarea optimă.

5.10. Când se stopează în realizare Algoritmul Genetic?

Condiția de stopare a unui algoritm genetic este importantă pentru a determina când se va termina de rulat Algoritmul Genetic. Evident, se dorește o condiție de stopare astfel încât la finele rulării soluția obținută să fie aproape de cea optimă.

În linii mari, Algoritmul Genetic se stopează, atunci când se îndeplinește una dintre următoarele condiții de finalizare:

- În situația când nu există nici o îmbunătățire a populației pentru k – iterații executate.
- În situația când se ajunge la un număr absolut de generații.
- În situația când valoarea funcției de fitness a atins o anumită valoare predefinită.

Pe parcursul realizării Algoritmului Genetic, de obicei, există un contor care contabilizează generațiile create. Totodată, în algoritmul respectiv se ține evidența generațiilor pentru care nu a existat nici o îmbunătățire a populației. De fiecare

dată când nu se generează descendenți mai buni decât indivizii din populație se trece la o altă generație ori chiar procesul de implementare poate fi stopat pentru a îmbunătăți anumite aspecte ale mecanismelor de implementare.

Condiția de terminare a unui Algoritm Genetic este foarte specifică problemei, iar implementatorul ar trebui să verifice diferite opțiuni pentru a vedea ce se potrivește cel mai bine în cazul problemei examinate. Algoritmii genetici continuă să fie rulați, de obicei, după ce s-a creat un număr predefinit de populații noi și soluția s-a îmbunătățit de la o populație la alta, dar nu s-a atins scopul propus. Procesul se desfășoară până în situația când va fi atins obiectivul stabilit. În această situație se afișează cei mai buni cromozomi (cel mai bun cromozom) care reprezintă soluția optimă.

5.11. Aplicații ale algoritmului genetic

Mai jos vom soluționa o problemă practică, care ține de optimizare, aplicând noțiunile și conceptele de bază examinate anterior.

Problemă. Aplicând Algoritmul Genetic se cere de găsit maximul funcției $F(x) = x^2 - 1$ pe intervalul de numere întregi $[0, 15]$.

Soluție. Pentru a codifica cromozomii din populația inițială (generația 0) și a opera cu ea ulterior, vom utiliza scrierea numerelor naturale în baza 2, extinsă pe 4 biți. Astfel, codificăm valorile numerelor naturale x de la 0 la 15 după cum urmează:

0 *0000*	1 *0001*	2 *0010*	3 *0011*	4 *0100*	5 *0101*	6 *0110*	7 *0111*
8 *1000*	9 *1001*	10 *1010*	11 *1011*	12 *1100*	13 *1101*	14 *1110*	15 *1111*

Creăm populația inițială din 4 cromozomi, aleși în mod aleatoriu: 1001, 0011, 1100 și 0111. Includem cromozomii aleși în tabelul ce urmează. Funcția de fitness în cazul nostru va fi $F(x) = x^2 - 1$. Este necesar să se determine $max\, F$ pe intervalul indicat. În continuare vor fi realizați următorii pași:

***Pasul 1*.** Fiind constituită populația inițială (Generația 0) prin intermediul funcției fitness, se va măsura fitness-ul fiecărei valori și ulterior se vor selecta cei mai „buni” cromozomi.

	Populația inițială	*Valori zecimale*	$F(x) = x^2 - 1$	*Procente*	*Procente/ medie*	*Aproximare*
1	1001	9	80	24.24	1.14	1
2	0011	3	8	2.42	0.11	0
3	1100	12	143	43.33	2.05	2
4	0111	7	48	30	0.69	1
	Suma		**279**	**100**	**4**	**4**
	Media		69.75	25	1	1

La efectuarea primului ciclu, valoarea cea mai bună a funcției fitness este 143, pentru $x = 12$. În continuare, încercăm să obținem o îmbunătățire a populației de cromozomi aplicând operatorii genetici: *selecția*, *crossover*-ul și *mutația*.

Pasul 2. Selecția se va face cu ajutorul metodei numită ruleta ponderată**.** După efectuarea calculelor, se elimină din populație, cromozomul 2, iar cromozomul 3 va fi multiplicat de 2 ori. Cromozomul 1 și 4, vor fi incluși, de asemenea, în noua generație de cromozomi.

În continuare, supraviețuirea speciei (căutarea soluției) se va face prin intermediul procesului de încrucișare și mutație. Încrucișarea se va face într-un singur și două puncte (one and two point crossover), poziția căruia se va alege aliator. Mutația de tip SWAP se va produce numai pentru cromozomul 2, pentru ultimele două elemente, așa cum este arătat în tabelul de mai jos.

	Populația după 1 selecție	*One and two Point Crossover*	*Populația după crossover*	*Mutația SWAP pentru cr.2*	*Populația I generație*
1	100**1**	Poziția 4	1000	1000	1000
2	110**0**		1101	1110	1110
3	11**00**	Poziția 3,4	1111	1111	1111
4	01**11**		0100	0100	0100

Pasul 3. Obținând o nouă generație de cromozomi (o nouă populație): 1000, 1110, 1111 și 0100, la fel se va calcula fitness-ul fiecărui cromozom și se vor compara.

	Populația I generație	*Valori x*	$F(x) = x^2 - 1$	*Procente*
1	1000	8	63	12.7
2	1110	14	195	39.2
3	1111	15	224	45.1
4	0100	4	15	3.01
	Suma		**497**	**100**
	Media		124.25	25

Cea mai mare valoare (cel mai bun fitness) este obținut pentru cromozomul 1111. Astfel, în două cicluri, au fost calculate 8 valori ale funcției fitness pentru

cromozomii obținuți. Dintre ele valoarea 224 reprezintă și maximul funcției $F(x) = x^2 - 1$, pe intervalul de numere întregi $[0, 15]$. Astfel, prin intermediul acestui exemplu s-a arătat la modul practic cum poate fi implementat Algoritmul Genetic pentru a obține o soluție în procesul de optimizare a unei funcții.

Este cunoscut faptul că o bună parte din algoritmi de optim necesită o cantitate enorm de mare de memorie și timpul de execuție crește exponențial. Pentru excluderea acestei inconveniențe, în practică sunt elaborați așa numiții algoritmi euristici care livrează soluții nu în mod obligatoriu optime, dar „suficient de bune" într-un timp relativ mai scurt și folosind o cantitate de memorie acceptabilă. Elaborarea acestor algoritmi are la bază, de regulă, idei empirice, idei naturale, „raționale", așa cum s-a procedat în cazul Algoritmului Genetic. Algoritmii euristici sunt preferați în situații când efortul de găsire a unei soluții optime este extrem de mare, comparativ cu câștigul obținut și în condițiile când efortul mare de programare nu se justifică.

Este necesar de menționat încă odată că ***Algoritmii evolutivi (Evolutionary Algorithms)*** aparțin domeniului denumit calcul evolutiv, care utilizează mecanisme inspirate de biologie (selecție, crossover, mutație etc.) pentru a căuta soluții optime. Iar ***Algoritmii Genetici (Genetic Algorithms)*** sunt algoritmii cei mai larg utilizați dintre toți algoritmii care țin de Algoritmii evolutivi, fiind algoritmi euristici de căutare, care imită procesul natural de selecție pentru a alege soluția candidată „cea mai potrivită”.

CAPITOLUL 6. ALGORITMUL GENETIC: INTERCONEXIUNE DINTRE BIOLOGIE, MATEMATICĂ ȘI INFORMATICĂ. APLICAȚII PRACTICE

În acest capitol, autorii abordează unele aspecte metodico-didactice care țin de implementarea Algoritmului Genetic. Este examinat, la modul practic, procesul de formare a populațiilor de cromozomi, cât și procedeele de aplicare a operatorilor genetici: selecție, crossover-ul și mutația. La fel, se cercetează procesul de optimizare a funcției fitness, mizând pe îmbunătățirea soluției odată cu trecerea calitativă de la o populație la alta. Sunt evidențiate tehnicile și procedeele care pot fi aplicate eficient atât de biologi cât și de matematicieni și informaticieni la rezolvarea problemelor de optimizare.

6.1. Mecanismele de funcţionare a Algoritmului Genetic

Faptul că matematica și informatica se aplică pe larg în diverse științe, inclusiv în biologie, este un lucru cunoscut și apreciat. Reciprocitatea în acest sens nu întotdeauna are loc. De exemplu, în știința modernă nu se întâlnesc așa de multe situații când mecanismele, conceptele și noțiunile de bază din biologie se utilizează pe larg și eficient în matematică și informatică.

Algoritmul genetic este în acest sens un exemplu elocvent și convingător. Algoritmii genetici reprezintă tehnici adaptive de căutare euristică, care se implementează mizând pe principiile selecției naturale și ale geneticii. Mecanismele de realizare a Algoritmului Genetic sunt asemănătoare evoluției naturale și mizează pe principiul enunțat de Charles Darwin, „*supraviețuiește nu cel mai puternic ori inteligent, dar cel mai bine adaptat*”. Astfel, Algoritmul Genetic reprezintă un model informatic-matematic care imită modelul biologic evoluționist pentru a soluționa probleme de căutare și optimizare. Algoritmul Genetic este determinat dintr-un set de elemente care reprezintă o populație, formată din

cromozomi (șiruri binare) și un set de operatori genetici (selecția, încrucișare și mutația) care influențează structura populației.

Pentru a facilita procesul de studiere a Algoritmului Genetic, de către studenți, autorii au încercat să trateze cât mai simplu și clar prin intermediul exemplelor construite cu grijă și migală, modalitățile de funcționare a acestui algoritm.

Algoritmul genetic este des utilizat pentru probleme în care găsirea soluției optime nu este simplă ori cel puțin ineficientă datorită caracteristicilor căutării probabilistice. Algoritmii genetici codifică o posibilă soluție la o problemă specifică într-o unică structură de date numită „cromozom” și aplică operatorii genetici la aceste structuri astfel încât să mențină informațiile critice. Procesul de implementare a algoritmilor genetici pornește de la o „mulțime inițială de soluții posibile” ale problemei examinate (aleasă de obicei aleatoriu) numită în literatura de specialitate „populație”.

Fiecare individ din „populația examinată” reprezintă o posibilă soluție a problemei și este numit „cromozom”, care este un șir de simboluri, de obicei exprimat ca un șir de biți. Cromozomii examinați evoluează pe durata iterațiilor succesive efectuate, numite simbolic – generații. În cadrul fiecărei generații, cromozomii respectivi sunt evaluați folosind unele măsuri de potrivire, numite – fitness.

Pentru generarea următoarei populații (generații) sunt selectați cei mai „eficienți”, cei mai „buni” cromozomi din generația curentă. Cromozomii noi sunt formați utilizând unul din cei trei (ori chiar toți trei) operatori genetici centrali: **selecția, crossover și mutația**.

Astfel, selecția asigură procesul din următoarea perspectivă: anumiți cromozomi din generația examinată (curentă) sunt copiați în dependență de valoarea funcției lor de potrivire, în corespundere cu cerințele problemei, în noua generație. Acest fapt denotă că cromozomii cu o semnificație mare au o probabilitate mare să contribuie la formarea noii generații.

Operatorul genetic crossover (încrucișare) reprezintă procesul prin care pe baza a doi indivizi (cromozomi) din populația curentă sunt formați doi indivizi (cromozomi), numiți și descendenți, pentru următoarea populație. Mutația este operatorul genetic care reprezintă procesul prin intermediul căruia un cromozom din populația curentă este modificat și salvat în noua populație.

Algoritmii Genetici au fost aplicați cu succes într-o varietate de probleme NP-complete care necesită optimizarea globală a soluției și, în acest sens, nu există o metodă iterativă de rezolvare.

În Algoritmii Genetici, indivizii dintr-o populație sunt reprezentați de cromozomi cu seturi codificate în ei, parametrii sarcinii, de exemplu, soluții care altfel sunt numite puncte în spațiu de căutare ori puncte de căutare. În unele lucrări, indivizii se numesc organisme. În acest sens vom clarifica sensul următoarelor noțiuni biologice din perspectiva informaticii.

Conceptul de evoluție al lui Darwin este adaptat la funcționarea algoritmului genetic pentru a găsi soluții la o problemă exprimată prin intermediul funcției de fitness (funcție obiectivă ori funcție de adaptare). ***Funcția fitness*** *(fitness function)* reprezintă o măsură a adaptabilității unui individ dat în cadrul fiecărei generații. Această caracteristică permite evaluarea gradului de adaptare a indivizilor din populație și se alege dintre cei mai adaptați, adică pe cei cu cele mai mari valori ale funcției de fitness, în conformitate cu evoluția principiul supraviețuirii celui mai „puternic” (cel mai bine adaptat).

Așa dar, selecția reprezintă alegerea indivizilor cu cea mai bună aptitudine pentru reproducere (sortarea după valoarea funcției obiective). Cu cât este mai bun fitness-ul unui individ, cu atât sunt mai mari șansele sale de a se încrucișa și de a-și moșteni genele de către generația următoare. Operatorul de încrucișare este analog cu reproducerea și încrucișarea biologică și, de obicei, se aplică asupra indivizilor din populația intermediară. Sunt selectați doi indivizi din populația intermediară și anumite porțiuni din cei doi cromozomi sunt interschimbate. În termeni simpli, mutația poate fi definită ca o mică modificare aleatorie a cromozomului, pentru a obține o nouă soluție. Mutația este utilizată pentru menținerea și introducerea diversității în populația genetică. Mutația este partea din Algoritmul Genetic care este legată de „explorarea” spațiului de căutare [1-10].

6.2. Aplicații practice ale Algoritmului Genetic. Determinarea maximului unei funcții

Mai jos vom soluționa două probleme practice, care țin de optimizare, aplicând noțiunile și conceptele de bază examinate anterior.

Problema 1. Aplicând Algoritmul Genetic se cere de găsit maximul funcției $\boldsymbol{F(x) = -x^2 + 16x - 15}$, pe intervalul de numere întregi $[1, \dots, 15]$.

Soluție. Pentru a codifica cromozomii din populația inițială (generația 0), și a opera cu ea ulterior, vom utiliza scrierea numerelor naturale în baza 2 extinsă pe 4 biți. Astfel, codificăm valorile numerelor naturale x de la 0 la 15 după cum urmează:

0 0000	1 0001	2 0010	3 0011	4 0100	5 0101	6 0110	7 0111
8 1000	9 1001	10 1010	11 1011	12 1100	13 1101	14 1110	15 1111

Creăm populația inițială din 4 cromozomi, aleși în mod aleatoriu: 0010, 0011, 0110 și 1011. Funcția de fitness în cazul nostru va fi $\boldsymbol{F(x) = -x^2 + 16x - 15}$. Este necesar să se determine $max\ F$ pe intervalul indicat. În continuare vor fi realizați următorii pași:

***Pasul 1*.** Includem cromozomii aleși aleatoriu în tabelul de mai jos. Fiind constituită populația inițială (Generația 0) prin intermediul funcției fitness, se va măsura fitness-ul fiecărei valori și ulterior se vor selecta cei mai „buni" cromozomi.

	Populația inițială	***Valori zecimale***	$\boldsymbol{F(x) = -x^2 + 16x - 15}$	***Procente***	***Procente / medie***	***Aproximare***
1	0010	2	13	10.65	0.42	0
2	0011	3	24	19.67	0.78	1
3	0110	**6**	45	36.89	1.5	2
4	1011	11	40	32.79	1.3	1
	Suma		**122**	**100**	**4**	**4**
	Media		**30.5**	**25**	**1**	**1**

La efectuarea primului ciclu, valoarea cea mai bună a funcției fitness este 45, pentru $x = 6$. În continuare, încercăm să obținem o îmbunătățire a populației de cromozomi, aplicând operatorii genetici: selecția, crossover-ul și mutația.

Pasul 2. Selecția se va face cu ajutorul metodei numită ruleta ponderată. După efectuarea calculelor în tabelul de mai sus și efectuarea aproximației, se elimină din populație cel mai „slab" cromozom, cromozomul 1, iar cromozomul 3, cel mai „bun", va fi multiplicat de 2 ori. Cromozomii 2 și 4, vor fi incluși, de asemenea, în noua generație de cromozomi. În continuare, supraviețuirea speciei (căutarea soluției) se va face prin intermediul procesului de încrucișare și mutație. Încrucișarea se va face în două și într-un singur punct (two and one point crossover).

Mutația inversă se va efectua numai pentru cromozomul 2, așa cum este arătat în tabelul de mai jos.

	Populația după 1 selecție	*Two and One Point crossover*	*Populația după crossover*	*Mutație inversată pentru cr.2*	*Populația I generație*	*Valoarea x*
1	0**01**1	Poziția 2 și 3	0101	0101	0101	5
2	0**10**1		0011	1100	1100	12
3	01**0**1	Poziția 3	0111	0111	0111	7
4	10**1**1		1001	1001	1001	9

Pasul 3. În felul acesta obținem o nouă generație de cromozomi (o nouă populație): 0101, 1100, 0111 și 1001. Pentru această populație, de asemenea se va calcula fitness-ul fiecărui cromozom și se va identifica maximul funcției examinate.

	Populația I generație	*Valori zecimale*	$F(x) = -x^2 + 16x - 15$	*Procente*	*Procente/ medie*	*Aproximare*
1	0101	5	40	23.67	0.94	1
2	1100	12	33	19.53	0.78	1
3	0111	7	48	28.40	1.13	1
4	1001	9	48	28.40	1.13	1
	Suma		**169**	**100**	**4**	**4**
	Media		**42.25**	**25**	**1**	**1**

În tabelul de mai sus, pentru cromozomii 3 și 4, efectuând calculele respective, s-a obținut cel mai „bun" fitness, valoarea funcției fiind egală cu 48. Nu este clar dacă fitness-ul egal cu 48 este valoarea maximă a funcției F(x). În acest sens, este necesar să se mai realizeze un ciclu, altfel spus să se creeze o nouă populație de cromozomi. Selecția, și în acest caz, se va face prin intermediul ruletei ponderate. Cromozomii examinați, având aproximarea egală cu 1, vor participa absolut toți la formarea celei de II generație a populației. Iar ulterior se vor aplica operatorii genetici crossover-ul și mutația.

Pasul 4. În continuare, supraviețuirea speciei (căutarea soluției) se va face prin intermediul procesului de încrucișare și mutație.

În următorul tabel, încrucișarea, pentru perechile de cromozomi 0101 și 1100 se va face într-un singur punct 4 (one point crossover) iar pentru perechile de cromozomi 0111 și 1001 în două puncte (two point crossover).

Mutația inversă se va realiza numai pentru cromozomul 3, așa cum este arătat în următorul tabel.

	Populația după II selecție	*One and two point crossover*	*Populația după crossover*	*Mutație inversată pentru cr.3*	*Populația II generație*	*Valoarea x*
1	010**1**	Poziția 4	0100	0100	0100	4
2	110**0**		1101	1101	1101	13
3	0**11**1	Poziția 2,3	0001	1000	1000	8
4	1**00**1		1111	1111	1111	15

Pasul 5. Obținând o nouă generație de cromozomi (o nouă populație): 0100, 1101, 1000 și 1111, la fel se va calcula fitness-ul fiecărui cromozom și se va identifica maximul.

	Populația II generație	*Valori x*	$F(x) = -x^2 + 16x - 15$	*Procente*
1	1101	13	24	22.64
2	0100	4	33	31.13
3	1111	15	0	0
4	1000	8	49	46.23
	Suma	**40**	**106**	**100**
	Media	**9.75**	**26.5**	

Cea mai mare valoare (cel mai bun fitness) este obținut pentru cromozomul 1000. Astfel, în ciclurile realizate au fost calculate, pentru cromozomii obținuți, 12 valori ale funcției fitness. În felul acesta au fost evaluați 12 cromozomi din cei 15. Astfel, am obținut următorul tabel:

Populații	*Cromozomii*	*Maximul F(x)*
Populația inițială Generația 0.	2, 3, 6, 11	$max\, F(6) = 45$
Populația la ciclul 1. Generația 1.	5, 7, 9, 12	$max\, F(7) = 48$ $max\, F(9) = 48$
Populația la ciclul 2. Generația 2.	4, 8, 13, 15	$max\, F(8) = 49$

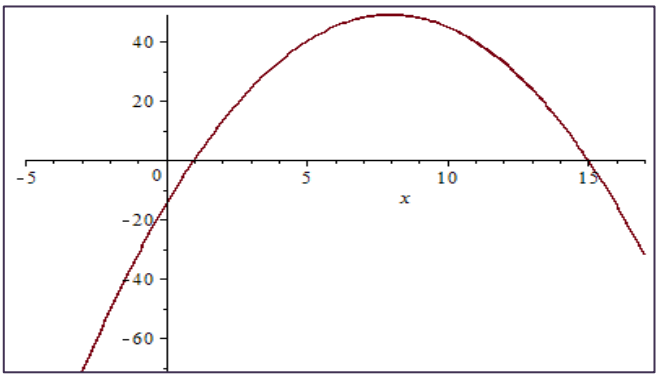
Observăm că la fiecare ciclu, maximul funcției de fitness s-a îmbunătățit: 45, 48 și 49. Nu este dificil de observat că pentru perechile de numere (1,15), (2,14), (3,13), (4,12), (5,11), (6,10), (7,9) funcția de fitness ne dă aceleași valori.

Pentru cromozomii 7 și 9 se obțin valorile funcției egale cu 48. Iar pentru 6 și 10 valoarea funcției este 45. Pentru 8 se obține valoarea 49, care și reprezintă maximul funcției $F(x) = -x^2 + 16x - 15$, pe intervalul de numere naturale $[1, \dots, 15]$. Prin intermediul exemplului respectiv s-a arătat la modul practic cum

poate fi implementat Algoritmul Genetic pentru a obține o soluție în procesul de optimizare a unei funcții.

Verificăm rezultatul obținut prin intermediul softului MAPLE.

$$plot\left(-x^2+16\,x-15,\,x=-5\,..17\right)$$



$$maximize\left(-x^2+16\,x-15,\,location\right)$$

$$49,\ \{[\{x=8\},\,49]\}$$

Calculând maximul funcției prin intermediul softului MAPLE, obținem același rezultat: $\boldsymbol{max(-x^2+16x-15)}$ este egal cu 49 în punctul 8**.**

Clar lucru, în cazul soluționării altor exemple, numărul de iterații poate fi cu mult mai mare. Important este să se înțeleagă că în cazul problemelor dificile de optimizare Algoritmul Genetic este cu mult mai eficient comparativ cu alte metode de rezolvare. Algoritmii genetici sunt simplu de utilizat și nu cer proprietăți importante ale funcției fitness (funcției obiectiv), precum continuitate, derivabilitate, convexitate, ca în cazul algoritmilor clasici. Algoritmii Genetici pot găsi soluții optime sau aproape de optim cu o mare probabilitate. Algoritmii genetici sunt mai rapizi și mai eficienți în comparație cu metodele tradiționale. Optimizează atât funcțiile continue și discrete, cât și problemele multi-obiective.

În felul acesta conceptul de evoluție al lui Darwin este adaptat la funcționarea algoritmului genetic pentru a găsi soluții la o problemă exprimată prin intermediul funcției de fitness (funcție obiectivă ori funcție de adaptare). Un concept foarte important în algoritmii genetici este funcția care măsoară gradul de adaptivitate cunoscută și sub denumirea de ***funcție de fitness*** *(fitness function)*. ***Funcția fitness*** reprezintă o măsură a adaptabilității unui individ dat în cadrul fiecărei generații. Această caracteristică permite evaluarea gradului de adaptare al indivizilor din populație și se alege dintre cei mai adaptați, adică pe cei cu cele mai mari valori ale funcției de fitness, în conformitate cu evoluția principiul supraviețuirii celui mai „puternic” (cel mai bine adaptat). Reamintim că ***selecția*** reprezintă alegerea indivizilor cu cea mai bună aptitudine pentru reproducere (sortarea după valoarea funcției obiective). Cu cât este mai bun fitness-ul unui

individ, cu atât sunt mai mari șansele sale de a se încrucișa și de a-și moșteni genele de către generația următoare. Operatorul de ***încrucișare*** este analog cu reproducerea și încrucișarea biologică și, de obicei, se aplică asupra indivizilor din populația intermediară. Sunt selectați doi indivizi din populația intermediară și anumite porțiuni din cei doi cromozomi sunt interschimbate.

În termeni simpli, ***mutația*** poate fi definită ca o mică modificare aleatorie a cromozomului, pentru a obține o nouă soluție. Mutația este utilizată pentru menținerea și introducerea diversității în populația genetică. Mutația este partea din Algoritmul Genetic care este legată de „explorarea" spațiului de căutare.

Problema 2. Aplicând Algoritmul Genetic se cere de găsit maximul funcției $\boldsymbol{F(x) = -x^2 + 14x - 13}$, pe intervalul de numere întregi $[1, \dots, 13]$.

Soluție. Pentru a codifica cromozomii din populația inițială (generația 0), și a opera cu ea ulterior, vom utiliza scrierea numerelor naturale în baza 2 extinsă pe 4 biți. Astfel, codificăm valorile numerelor naturale x de la 1 la 14 după cum urmează:

0 0000	1 0001	2 0010	3 0011	4 0100	5 0101	6 0110	7 0111
8 1000	9 1001	10 1010	11 1011	12 1100	13 1101	14 1110	15 1111

Creăm populația inițială din 4 cromozomi, aleși în mod aleatoriu: 3, 5, 10 și 11 ori, corespunzător, în format de biți: 0011, 0101, 1010 și 1011. Funcția de fitness în cazul nostru va fi $\boldsymbol{F(x) = -x^2 + 14x - 13}$. Este necesar să se determine $max\ F$ pe intervalul indicat. În continuare vor fi realizați următorii pași:

Pasul 1. Includem cromozomii aleși aleatoriu în următorul tabel. Fiind constituită populația inițială (Generația 0) prin intermediul funcției fitness, se va măsura fitness-ul fiecărei valori și ulterior se vor selecta cei mai „buni" cromozomi.

	Populația inițială	***Valori zecimale***	$\boldsymbol{F(x) = -x^2 + 14x - 13}$	***Procente***	***Procente/ medie***	***Aproximare***
1	0011	3	20	20.20	0.81	1
2	0101	5	**32**	32.32	1.29	1
3	1010	10	27	27.28	1.09	1
4	1011	11	20	20.20	0.81	1
	Suma		**99**	**100**	**4**	**4**
	Media		**24.75**	**25**	**1**	**1**

La efectuarea primului ciclu, valoarea cea mai bună a funcției fitness este 32, pentru $x = 5$. În continuare, încercăm să obținem o îmbunătățire a populației de cromozomi, aplicând operatorii genetici: selecția, crossver-ul și mutația.

***Pasul 2*.** Selecția se va face cu ajutorul metodei numită ruleta ponderată**.** După efectuarea calculelor în tabelul anterior și efectuarea aproximației, se elimină din populație, cromozomul 1, iar cromozomul 3 va fi multiplicat de 2 ori. Cromozomii 2 și 4, vor fi incluși, de asemenea, în noua generație de cromozomi. În continuare, supraviețuirea speciei (căutarea soluției) se va face prin intermediul procesului de încrucișare și mutație. Încrucișarea se va face în două și într-un singur punct (two and one point crossover).

Mutația inversă se va efectua numai pentru cromozomul 2, așa cum este arătat în tabelul de mai jos.

	Populația după 1 selecție	*One Point crossover*	*Populația după crossover*	*Mutație inversată pentru cr.1 și cr.4 și mutația SWAP pentru cr.3*	*Populația I generație*	*Valoarea x*
1	00**1**1	Poziția 3	0001	1000	1000	8
2	01**0**1		0111	0111	0111	7
3	1010		**10**10	0110	0110	6
4	1011		1011	1101	1101	13

***Pasul 3*.** În felul acesta obținem o nouă generație de cromozomi (o nouă populație): 1000, 0111, 0110 și 1101. Pentru populația respectivă la fel se va calcula fitness-ul fiecărui cromozom și se va identifica maximul funcției examinate.

	Populația I generație	*Valori zecimale*	$F(x) = -x^2 + 14x - 13$	*Procente*	*Procente/ medie*	*Aproximare*
1	1000	8	35	33.01	1.32	1
2	0111	7	**36**	33.96	1.36	1
3	0110	6	35	33.01	1.32	1
4	1101	13	0	0	0	0
	Suma		**106**	**100**	**4**	**4**
	Media		**26.5**	**25**	**1**	**1**

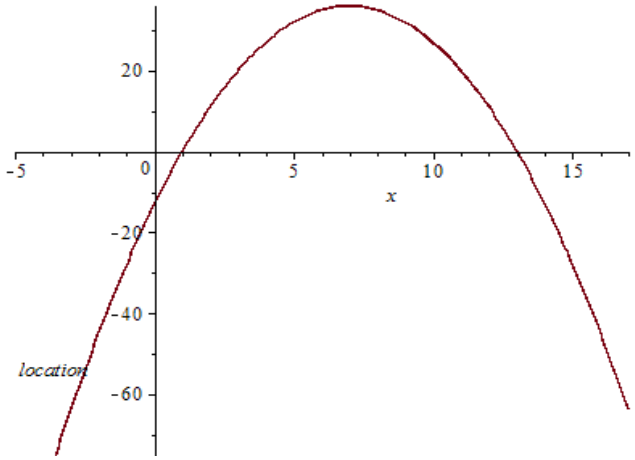
Cea mai mare valoare (cel mai bun fitness) este obținut pentru cromozomul 0111. Astfel, în ciclurile realizate au fost calculate, pentru cromozomii obținuți, 8 valori ale funcției fitness. În felul acesta au fost evaluați 8 cromozomi din cei 14. Astfel, am obținut următorul tabel:

Populații	Cromozomii	Maximul F(x)
Populația inițială Generația 0.	3, 5, 10, 11	$max\ F(5) = 32$
Populația la ciclul 1. Generația 1.	8, 7, 6, 13	$max\ F(8) = 35$ $max\ F(7) = 36$ $max\ F(6) = 35$

Observăm că la ciclul 1, maximul funcției de fitness s-a îmbunătățit, comparativ cu ciclul 0: 35, și 36. Nu este dificil de observat că pentru perechile de numere (1,13), (2,12), (3,11), (4,10), (5,9), (6,8) funcția de fitness ne dă aceleași valori.

Verificăm rezultatul obținut prin intermediul softului MAPLE.

$$plot\left(-x^2 + 14\,x - 13,\ x = -5\ ..17,\ location\right)$$



Calculând maximul funcției prin intermediul softului MAPLE obținem același rezultat: $\boldsymbol{max(-x^2 + 14x - 13)}$ este egal cu 36 în punctul 7**.**

$$maximize\left(-x^2 + 14\,x - 13,\ location\right)$$

$$36,\ \{[\{x = 7\},\ 36]\}$$

6.3. Algoritmul amplasării optime a vârfurilor unui graf pe o riglă

Problema amplasării optime a vârfurilor unui graf neorientat pe o riglă liniară, este o problemă clasică care necesită cunoștințe atât din matematică, informatică cât și, evident, care țin și de Algoritmul Genetic. În acest compartiment vom propune un algoritm care soluționează problema respectivă în anumite condiții.

Problema amplasării vârfurilor unui graf pe o riglă liniară. Algoritmul amplasării

Problemă. Fie este dat graful G, unde $n = |G|$ - este numărul de vârfuri ale grafului G. Se cere de găsit cea mai bună amplasare a vârfurilor grafului G pe o riglă liniară după realizarea algoritmului genetic în k-cicluri (k-generații) și $k < n$. Se consideră că distanțele dintre vârfurile grafului sunt egale.

Soluție. Suma muchiilor grafului G se calculează conform relației de mai jos:

$$L(G) = \sum_{i=1}^{n} \sum_{j=1}^{n} d_{i,j}\, a_{i,j}, \quad (3)$$

unde, $n = |G|$ - este numărul de vârfuri, $d_{i,j}$ – reprezintă distanța dintre vârfurile grafului G, v_i și v_j pe rigla examinată. Menționăm că distanța în cazul dat, se va măsura în numărul de muchii ale grafului. Iar $a_{i,j}$ – este elementul corespunzător al matricei de adiacență (0 sau 1). Altfel spus se cere de găsit $min\ L(G)$ după schimbarea a k-generații (după efectuarea a k-cicluri).

Algoritmul genetic aplicat constă în următoarele:

Pasul 1. Se creează populația inițială care constă din k cromozomi, compus din n elemente (vârfurile grafului):

$$C_1^i, C_2^i, \ldots., C_k^i, \text{ unde } i = 0, 1, 2, \ldots, n-1.$$

Notații. Indicele ***i*** pentru cromozomii $C_1^i, C_2^i, \ldots., C_k^i$, denotă numărul generației din care fac parte cromozomii cu numărul $1, 2, \ldots, k$.

Pasul 2. Se amplasează vârfurile grafului pe rigla liniară în corespundere cu valorile cromozomilor examinați: $C_1^i, C_2^i, \ldots., C_k^i$.

Pasul 3. Se calculează lungimea fiecărui cromozom $C_1^i, C_2^i, \ldots., C_k^i$. Acest fapt presupune calculul numărului de segmente orizontale care unesc vârfurile grafului în conformitate cu amplasamentul realizat la ***pasul 2***, pentru fiecare cromozom separat. Se obțin lungimile:

$$L_1(C_1^i), L_2(C_2^i), \ldots, L_k(C_k^i)$$

***Pasul 4*.** Se calculează suma totală a muchiilor grafului în corespundere cu amplasamentul vârfurilor pe riglă determinat de populația respectivă de cromozomi.

$$S(i) = L_1(C_1^i) + L_2(C_2^i) + \cdots + L_k(C_k^i) \quad (4)$$

unde i-este numărul ciclului ori a generației de populație, $i = 0, 1, 2, \ldots, n-1$.

Pasul 5. Se alege cel mai „apt" cromozom (cu lungimea cea mai mică) din populația $C_1^i, C_2^i, \ldots., C_k^i$. Fie acest cromozom este C_r^i, unde $1 \leq r \leq k$.

Pasul 6. Asupra cromozomului C_r^i se aplică operatorul genetic ***mutația inversată*** după primul element. Altfel spus, la primul ciclu, primul element al cromozomului rămâne pe loc, iar celelalte elemente se scriu în ordinea inversă, începând cu ultimul, care va fi deja pe locul al doilea în reprezentarea cromozomului. La al

doilea ciclu, primul și la doilea element rămân intacți, iar celelalte elemente se scriu în ordinea inversă, începând cu ultimul, care va fi deja pe locul al treilea. Și tot așa procedăm la fiecare ciclu.

Notăm cromozomul nou obținut în urma efectuării mutației inversate prin C_{rm}^{i}.

***Pasul 7*.** Se calculează lungimea cromozomului nou obținut C_{rm}^{i}.

***Pasul 8*.** Din prima generație a cromozomilor se identifică și se elimină cel mai slab cromozom (care are lungimea maximă), care ulterior se înlocuiește cu cromozomul C_{rm}^{i}.

Pasul 9. Se construiește următoarea generație de populație de cromozomi (în care este deja inclus noul cromozom C_{rm} și este exclus cromozomul cel mai slab) și ulterior se trece la executarea ***pasului 2***.

Pasul 10. Procesul de realizare al algoritmului genetic se stopează după efectuarea a k − cicluri, ori în altă terminologie construcția a k − generații de cromozomi. La fiecare ciclu se calculează $S(0), S(1), \dots, S(k)$, unde pentru fiecare sumă se îndeplinește condiția: $\boldsymbol{S(m) \geq S(m+1), m = 0, 1, \dots, k}$. Cea mai bună amplasare a vârfurilor grafului se obține la realizarea ultimului ciclu, în care ultimul cromozom modificat genetic C_{rm}^{i} reprezintă soluția $min\ L(G)$.

6.4. Utilizarea algoritmul amplasării la soluționarea problemelor

Problema 1. Se cere de găsit cea mai bună amplasare a vârfurilor grafului G din figura 1 pe o riglă după realizarea a trei cicluri ale algoritmului genetic. Mai jos este dat graful G și populația inițială, alcătuită din 3 cromozomi.

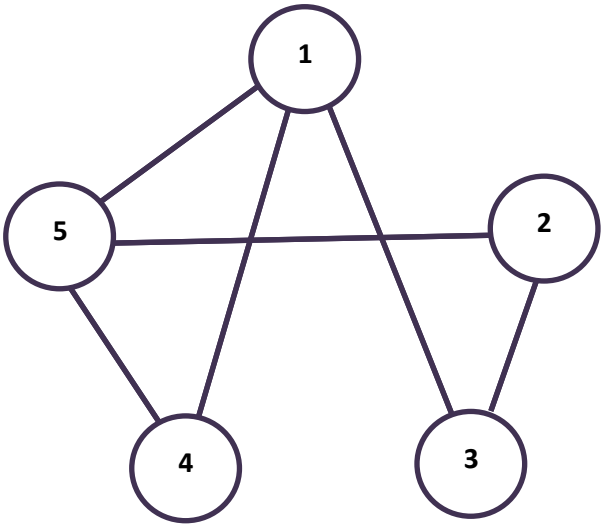


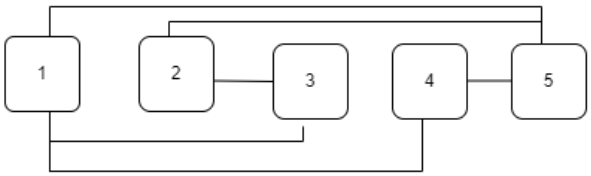
Figura 1. Graful examinat, $n = 5$ și $m = 6$

Populația inițială de cromozomi (Generația 0)

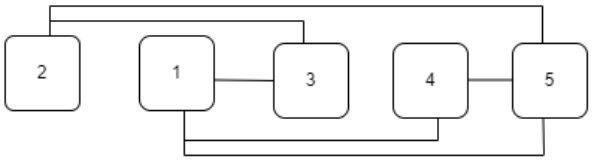
Cromozomul C_1^0	**1**	**2**	**3**	**4**	**5**
Cromozomul C_2^0	**2**	**1**	**3**	**4**	**5**
Cromozomul C_3^0	**1**	**5**	**3**	**4**	**2**

Soluție. Amplasăm vârfurile grafului pe riglă în corespundere cu populația inițială de cromozomi, astfel încât să putem calcula lungimea fiecărui cromozom. Astfel, obținem:

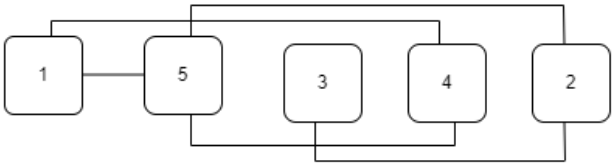
Cromozomul C_1^0



Cromozomul C_2^0



Cromozomul C_3^0



Calculăm numărul segmentelor orizontale dintre vârfurile grafului (dintre elementele cromozomilor). Obținem:

$$L_1(C_1^0) = 3 + 5 + 3 + 3 = 14;$$
$$L_2(C_2^0) = 2 + 5 + 3 + 3 = 13;$$
$$L_3(C_3^0) = 3 + 4 + 4 + 2 = 13.$$

Se calculează suma totală a muchiilor grafului în corespundere cu amplasamentul vârfurilor pe riglă determinat de populația respectivă de cromozomi:

$$S(0) = L_1(C_1^0) + L_2(C_2^0) + L_3(C_3^0) = 14 + 13 + 13 = 40.$$

Dintre cromozomii cei doi, cromozomi de lungime minimă (cei mai apți) C_2^0 și C_3^0, îl selectăm pe primul, C_2^0.

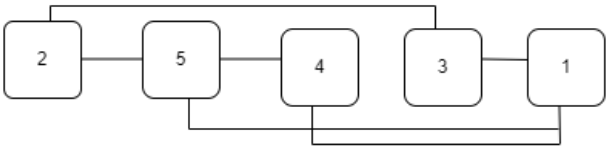
Cromozomul C_2^0	**2**	**1**	**3**	**4**	**5**

Asupra cromozomului C_2^0 aplicăm operatorul genetic mutația inversată și îl transformăm în cromozomul C_1^1. Altfel spus, la primul ciclu, primul element al cromozomului (2) rămâne pe loc, iar celelalte elemente se scriu în ordinea inversă, începând cu ultimul, care va fi deja pe locul al doilea în reprezentarea cromozomului. În așa mod obținem:

Cromozomul C_1^1	**2**	**5**	**4**	**3**	**1**

Calculăm lungimea cromozomului C_1^1.

Cromozomul C_1^1.



În așa mod, obținem:

$$L_1(C_1^1) = 2 + 3 + 3 + 3 = 11.$$

Astfel, din generația inițială a populației, substituim cromozomul C_1^0 de lungimea 14 (cel mai puțin apt) cu cromozomul mai performant C_1^1 de lungimea 11. Cromozomul C_2^0 îl vom nota C_2^1, iar cromozomul C_3^0 îl vom nota C_3^1. Obținem, în așa fel, Generația 1 a populației de cromozomi.

Populația de cromozomi (Generația 1)

Cromozomul C_1^1	**2**	**5**	**4**	**3**	**1**
Cromozomul C_2^1	**2**	**1**	**3**	**4**	**5**
Cromozomul C_3^1	**1**	**5**	**3**	**4**	**2**

Lungimea fiecărui cromozom este:

$$L_1(C_1^1) = 2 + 3 + 3 + 3 = 11;$$
$$L_2(C_2^1) = 2 + 5 + 3 + 3 = 13;$$
$$L_3(C_3^1) = 3 + 4 + 4 + 2 = 13.$$

Suma totală a muchiilor grafului în corespundere cu amplasamentul vârfurilor pe riglă determinat de populația respectivă de cromozomi la efectuarea primului ciclu este:

$$S(1) = L_1(C_1^1) + L_2(C_2^1) + L_3(C_3^1) = 11 + 13 + 13 = 37.$$

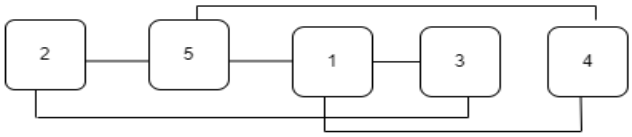
Cromozomul cel mai bun din populația generației 1 este:

Cromozomul C_2^1	**2**	**5**	**4**	**3**	**1**

Asupra cromozomului C_2^1 aplicăm mutația inversată și obținem cromozomul C_2^2 . Adică, la ciclu II, primul și al doilea element al cromozomului (2, 5) rămân pe loc, iar celelalte elemente se scriu în ordinea inversă, începând cu ultimul, care va fi deja pe locul al treilea în reprezentarea cromozomului. În așa mod obținem:

Cromozomul C_2^2	**2**	**5**	**1**	**3**	**4**

Cromozomul C_2^2



Calculăm lungimea cromozomului C_2^2.
Obținem:

$$L_2(C_2^2) = 2 + 3 + 4 + 2 = 11.$$

În continuare, în generația 1 a populației, avem doi cromozomi de lungime maximă, considerați cei mai slabi: C_2^1 și C_3^1. Selectăm cromozomul cel mai puțin apt (cu indicele cel mai mare) C_3^1 și îl substituim cu cromozomul mai performant C_2^2 de lungimea 11. Cromozomul C_1^1 îl vom nota C_1^2, iar cromozomul C_2^1 îl vom nota C_3^2. Obținem, în așa fel, Generația II a populației de cromozomi.

Populația de cromozomi (Generația 2)

Cromozomul C_1^2	**2**	**5**	**4**	**3**	**1**
Cromozomul C_2^2	**2**	**5**	**1**	**3**	**4**
Cromozomul C_3^2	**2**	**1**	**3**	**4**	**5**

Lungimea fiecărui cromozom este:

$$L_1(C_1^2) = 2 + 3 + 3 + 3 = 11;$$
$$L_2(C_2^2) = 2 + 3 + 4 + 2 = 11.$$
$$L_2(C_3^2) = 2 + 5 + 3 + 3 = 13;$$

Suma totală a muchiilor grafului în corespundere cu amplasamentul vârfurilor pe riglă determinat de populația respectivă de cromozomi la efectuarea ciclului doi este:

$$S(2) = L_1(C_1^2) + L_2(C_2^2) + L_3(C_3^2) = 11 + 11 + 13 = 35.$$

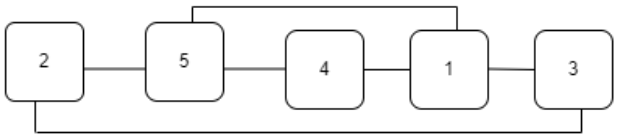
Cromozomul cel mai bun din populația generației 2 este:

Cromozomul C_2^1	**2**	**5**	**4**	**3**	**1**

Asupra cromozomului C_2^1 aplicăm mutația inversată și cromozomul nou obținut îl vom nota C_2^3. Adică, la ciclu III, primul, al doilea și al treilea element (2, 5, 4) al cromozomului rămân pe loc, iar celelalte elemente se scriu în ordinea inversă, începând cu ultimul, care va fi deja pe locul al patrulea în reprezentarea cromozomului. În așa mod obținem:

Cromozomul C_2^3	**2**	**5**	**4**	**1**	**3**

Calculăm lungimea cromozomului C_2^3.



Obținem:

$$L_2(C_2^3) = 2 + 3 + 3 + 2 = 10.$$

În continuare, în generația II a populației, selectăm cromozomul de lungime maximă egală cu 13, considerat cei mai slab C_2^1. Cromozomul cel mai puțin apt îl substituim cu cromozomul mai performant C_2^3 de lungimea 10. Obținem, în așa fel, Generația III a populației de cromozomi.

Populația de cromozomi (Generația 3)

Cromozomul C_1^3	**2**	**5**	**4**	**3**	**1**
Cromozomul C_2^3	**2**	**5**	**4**	**1**	**3**
Cromozomul C_3^3	**2**	**5**	**1**	**3**	**4**

Lungimea fiecărui cromozom este:

$$L_1(C_1^3) = 3 + 5 + 3 + 3 = 11;$$
$$L_2(C_2^3) = 2 + 3 + 3 + 2 = 10;$$
$$L_3(C_3^3) = 3 + 4 + 4 + 2 = 11.$$

Suma totală a muchiilor grafului în corespundere cu amplasamentul vârfurilor pe riglă determinat de populația respectivă de cromozomi la efectuarea ciclului trei este:

$$S(3) = L_1(C_1^3) + L_2(C_2^3) + L_3(C_3^3) = 11 + 10 + 11 = 32.$$

Evoluția lungimilor în cazul generațiilor 0, I, II, III este următoarea:

$$S(0) = 40 > S(I) = 37 > S(II) = 35 > S(III) = 32.$$

Cea mai bună amplasare a vârfurilor grafului pe riglă se obține la realizarea ultimului ciclu (din trei cicluri), în care ultimul cromozom modificat genetic C_2^3 reprezintă soluția $\min L(G) = 10$, deoarece $L_2(C_2^3) = 10$.

Problema 2. Se cere de găsit cea mai bună amplasare a vârfurilor grafului G din Figura 2 pe o riglă după realizarea a trei cicluri ale algoritmului genetic. Mai jos este dat graful G și populația inițială alcătuită din 3 cromozomi

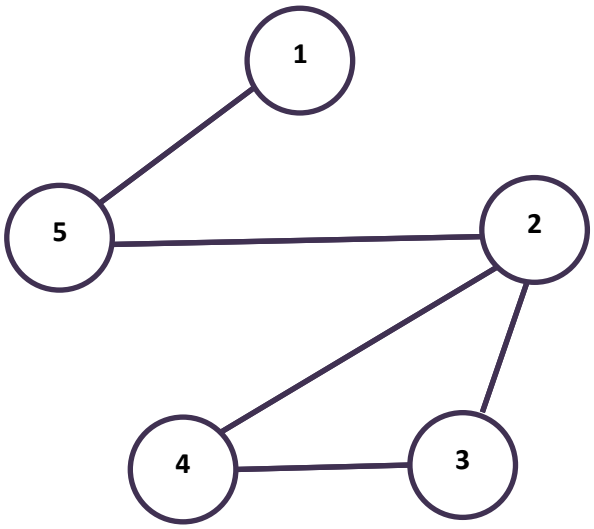


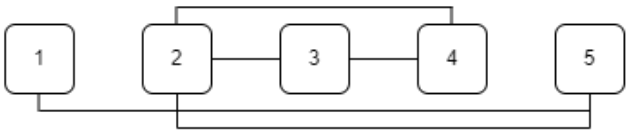
Figura 2. Graful examinat, $n = 5$ și $m = 5$

Populația inițială de cromozomi (Generația 0)

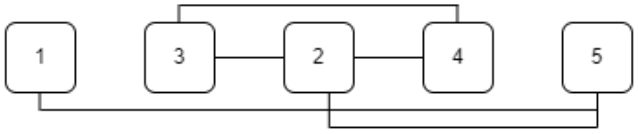
Cromozomul C_1^0	**1**	**2**	**3**	**4**	**5**
Cromozomul C_2^0	**1**	**3**	**2**	**4**	**5**
Cromozomul C_3^0	**5**	**2**	**3**	**4**	**1**

Soluție. Amplasăm vârfurile grafului pe riglă în corespundere cu populația inițială de cromozomi, astfel încât să putem calcula lungimea fiecărui cromozom. Prin urmare, obținem:

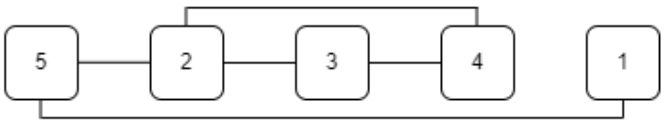
Cromozomul C_1^0



Cromozomul C_2^0



Cromozomul C_3^0



Calculăm numărul segmentelor orizontale dintre vârfurile grafului (dintre elementele cromozomilor). Obținem:

$$L_1(C_1^0) = 1 + 4 + 4 + 2 = 11;$$
$$L_2(C_2^0) = 1 + 3 + 4 + 2 = 10;$$
$$L_3(C_3^0) = 2 + 3 + 3 + 1 = 9.$$

Se calculează suma totală a muchiilor grafului în corespundere cu amplasamentul vârfurilor pe riglă determinat de populația respectivă de cromozomi:

$$S(0) = L_1(C_1^0) + L_2(C_2^0) + L_3(C_3^0) = 11 + 10 + 9 = 30.$$

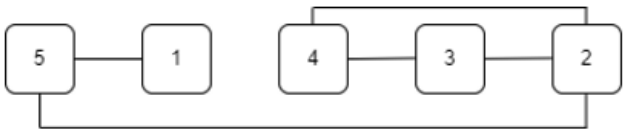
Dintre cromozomii examinați cromozomul C_3^0 are lungime minimă egală cu 9, deci este cel mai apt. Considerăm cromozomul C_3^0 selectat.

Cromozomul C_3^0	**5**	**2**	**3**	**4**	**1**

Asupra cromozomului C_3^0 aplicăm operatorul genetic mutația inversată și vom obține un nou cromozom pe care îl vom nota C_1^1. Altfel spus, la primul ciclu, primul element al cromozomului (5) rămâne pe loc, iar celelalte elemente se scriu în ordinea inversă, începând cu ultimul, care va fi deja pe locul al doilea în reprezentarea cromozomului. În așa mod obținem:

Cromozomul C_1^1	**5**	**1**	**4**	**3**	**2**

Calculăm lungimea cromozomului C_1^1.



În așa mod, obținem:

$$L_1(C_1^1) = 2 + 1 + 3 + 3 = 9.$$

Astfel, din generația inițială a populației, substituim cromozomul C_1^0 de lungimea 11 (mai puțin apt) cu cromozomul mai performant C_1^1 de lungimea 9. Cromozomul C_2^0 îl vom nota C_2^1, iar cromozomul C_3^0 îl vom nota C_3^1. Astfel, obținem Generația 1 a populației de cromozomi.

Populația de cromozomi (Generația 1)

Cromozomul C_1^1	**5**	**1**	**4**	**3**	**2**
Cromozomul C_2^1	**1**	**3**	**2**	**4**	**5**
Cromozomul C_3^1	**5**	**2**	**3**	**4**	**1**

Lungimea fiecărui cromozom este:

$$L_1(C_1^1) = 2 + 1 + 3 + 3 = 9;$$
$$L_2(C_2^1) = 1 + 3 + 4 + 2 = 10;$$
$$L_3(C_3^1) = 2 + 3 + 3 + 1 = 9.$$

Suma totală a muchiilor grafului în corespundere cu amplasamentul vârfurilor pe riglă determinat de populația respectivă de cromozomi la efectuarea primului ciclu este:

$$S(1) = L_1(C_1^1) + L_2(C_2^1) + L_3(C_3^1) = 9 + 10 + 9 = 28.$$

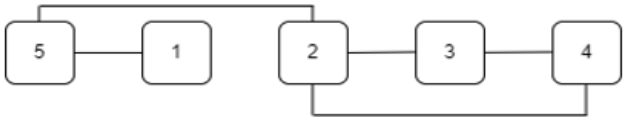
Dintre cei doi cromozomi de lungimea 9, selectăm cromozomul C_1^1 ca cel mai bun din populația generației 1:

Cromozomul C_1^1	**5**	**1**	**4**	**3**	**2**

Asupra cromozomului C_1^1 aplicăm mutația inversată și vom obține un nou cromozom pe care îl vom nota C_2^2. Adică, la ciclu II, primul și al doilea element al cromozomului (5, 1) rămân pe loc, iar celelalte elemente se scriu în ordinea inversă, începând cu ultimul, care va fi deja pe locul al treilea în reprezentarea cromozomului. În așa mod obținem:

Cromozomul C_2^2	**5**	**1**	**2**	**3**	**4**

Calculăm lungimea cromozomului C_2^2.



Obținem:

$$L_2(C_2^2) = 2 + 1 + 2 + 2 = 7.$$

În continuare, în generația 1 a populației, avem 2 cromozomi de lungime maximă, egală cu 9, considerați cel mai slabi: C_3^1 și C_2^1. Cromozomul mai puțin apt C_2^1 îl vom substitui cu cromozomul mai performant C_2^2 de lungimea 7. Obținem, în așa fel, Generația II a populației de cromozomi.

Populația de cromozomi (Generația 2)

Cromozomul C_1^2	**5**	**1**	**4**	**3**	**2**
Cromozomul C_2^2	**5**	**1**	**2**	**3**	**4**
Cromozomul C_3^2	**5**	**2**	**3**	**4**	**1**

Lungimea fiecărui cromozom este:

$$L_1(C_1^2) = 2 + 1 + 3 + 3 = 9;$$

$$L_2(C_2^2) = 2 + 1 + 2 + 2 = 7;$$
$$L_3(C_3^2) = 2 + 3 + 3 + 1 = 9.$$

Suma totală a muchiilor grafului în corespundere cu amplasamentul vârfurilor pe riglă determinat de populația respectivă de cromozomi la efectuarea ciclului doi este:

$$S(2) = L_1(C_1^2) + L_2(C_2^2) + L_3(C_3^2) = 9 + 7 + 9 = 25.$$

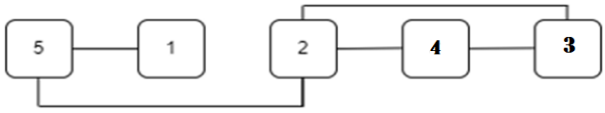
Alegem cromozomul cel mai bun din populația generației 2. Este, evident ca trebuie să alegem cromozomul de lungime minimă, care este C_2^2:

Cromozomul C_2^2	**5**	**1**	**2**	**3**	**4**

Asupra cromozomului C_2^2 aplicăm mutația inversată și obținem un nou cromozom C_3^3. Adică, la ciclu III, primul, al doilea și al treilea element (5, 1, 2) al cromozomului rămân pe loc, iar celelalte elemente se scriu în ordinea inversă, începând cu ultimul, care va fi deja pe locul al patrulea în reprezentarea cromozomului. În așa mod obținem:

Cromozomul C_2^3	**5**	**1**	**2**	**4**	**3**

Calculăm lungimea cromozomului C_3^3.



Obținem:

$$L_3(C_3^3) = 2 + 1 + 2 + 2 = 7.$$

În continuare, în generația II a populației, selectăm cromozomul de lungime maximă egală cu 9, considerați cei mai slab C_1^2 ori C_3^2. Cromozomul cel mai puțin apt îl substituim cu cromozomul mai performant C_3^3 de lungimea 7. Obținem, în așa fel, Generația III a populației de cromozomi.

Populația de cromozomi (Generația 3)

Cromozomul C_1^3	**5**	**1**	**4**	**3**	**2**
Cromozomul C_2^3	**5**	**1**	**2**	**3**	**4**
Cromozomul C_3^3	**5**	**1**	**2**	**4**	**3**

Lungimea fiecărui cromozom este:

$$L_1(C_1^3) = 1 + 1 + 3 + 2 = 9;$$
$$L_2(C_2^3) = 1 + 1 + 2 + 2 = 7;$$
$$L_3(C_3^3) = 1 + 1 + 3 + 2 = 7.$$

Suma totală a muchiilor grafului în corespundere cu amplasamentul vârfurilor pe riglă determinat de populația respectivă de cromozomi la efectuarea ciclului trei este:

$$S(3) = L_1(C_1^3) + L_2(C_2^3) + L_3(C_3^3) = 9 + 7 + 7 = 23.$$

Evoluția lungimilor în cazul generațiilor 0, I, II, III este următoarea:

$$S(0) = 30 > S(I) = 28 > S(II) = 25 > S(III) = 23.$$

Cea mai bună amplasare a vârfurilor grafului pe riglă se obține la realizarea ultimului ciclu, în care ultimii cromozomi modificați genetic C_2^3 și C_3^3 reprezintă soluția $\min L(G) = 7$, iar S(III) = 23.

BIBLIOGRAFIE

CAPITOLUL 1-3

1. DUMITRESCU, D. *Principiile Inteligentei Artificiale*. Cluj-Napoca: Editura Albastra, 2002.
2. POOLE D. L.; MACKWORTH, A. K. *Artificial Intelligence Foundations of Computational Agents*. Cambridge University Press, 2010. Disponibil online: https://artint.info/2e/html/ArtInt2e.html
3. GARDNER, H. *The Frames of Mind: The Theory of Multiple Intelligences*. Second Edition. New York: Fontana Press, An Inprint of Harper Collins Publishers, 1993.
4. Curs online la Inteligenta Artificiala predat de Sebastian Thrun si Peter Norvig: https://www.udacity.com/wiki/cs271/downloads
5. ГРАЕ, Д. *Наука о данных с нуля*: Пер. с англ. СПб.: БХВ-Петербург, 2017. 336 с.
6. ВАНДЕР ПЛАС, Дж. *Python для сложных задач: наука о данных и машинное обучение*. СПб.: Питер, 2018. 576 с.
7. ЖЕРОН, О. *Прикладное машинное обучение с помощью Scikit-Learn и TensorFlow: концепции, инструменты и техники для создания интеллектуальных систем*. Пер. с англ. СпБ.: ООО "Альфа-книга", 2018. 688 с.
8. Agenția pentru Drepturi Fundamentale a Uniunii Europene. Despre Inteligența Artificială și BigData. https://fra.europa.eu/en/themes/artificial-intelligence-and-big-data.
9. CHIRIAC, L.; CHIRIAC, E.; LUPAȘCO, N.; PAVEL, M. Paradigme moderne în dezvoltarea și învățarea Inteligenței Artificiale. In: The 29th Conference on Applied and Industrial Mathematics CAIM 2022. 25-27 august 2022, Chişinău. Chișinău: TSU, 2022, pp. 116-124. ISBN 978-9975-76-401-8. https://ibn.idsi.md/sites/default/files/imag_file/116-124_6.pdf
10. CHIRIAC, L.; LUPAȘCO, N.; PAVEL, M. Abordări istorico-didactice în Studierea Inteligenței Artificiale. În: *Materialele Conferinței științifice internaționale „Abordări inter/transdisciplinare în predarea științelor reale*

(concept STEAM)”, ediția a II-a. 28 – 29 octombrie 2022. Chișinău: UST, 2022, pp. 266-275. ISBN 978-9975-76-411-7 (PDF).
11. CHIRIAC, L.; GLOBA, A.; LUPAȘCO, N. Repere istorico-didactice în studierea Inteligenței Artificiale. În: Conferința "The 26th Conference on Applied and Industrial Mathematics". Chișinău, Moldova, 20-23 septembrie 2018, pp. 10-19. https://ibn.idsi.md/vizualizare_articol/92588
12. *https://www.ibm.com/topics/chat-bots. Accesat în 12 februarie 2023.*
13. *^„Ce este Chat GPT, la ce poate fi folosit în lumea reală și de ce a început să ridice semne de întrebare”, euronews.ro, 12 februarie 2023*

CAPITOLUL 4-6

1. HOLLAND, J.H. *Adaptation in Natural and Artificial Systems*. Ann. Arbor: University of Michigan Press, 1975. 183 p.
2. MITCHELL, M. *An Introduction to Genetic Algorithms*. Cambridge: MIT Press, 1996.
3. MITCHELL, M. Genetic Algorithms: An Overview. In: *Complexity*, 1995. Nr. 1(1), pp. 31-39.
4. DUMITRESCU, D. *Algoritmi genetici și strategii evolutive - Aplicatii în inteligența artificială și în domenii conexe*. Cluj-Napoca: Editura Albastra, 2000.
5. GOLDBERG, D.E. Genetic *Algorithms in Search, Optimization and Machine Learning*. Addison -Wesley, Reading, MA,1989.
6. GAREY, M.R.; JOHNSON, D.S. *Computers and Intractability: A Guide to NP-completeness*. New York: W.H. Freeman and Company, 1978.
7. KOZA, J.R. *Genetic Programming*. Cambridge, MA: MIT Press, 1992.
8. OLTEAN, M. Proiectarea și implementarea algoritmilor. Cluj-Napoca: Computer Libris Agora, 2000.
9. BEASLEY, D.; BULL, D. R.; MARTIN, R. R. An Overview of Genetic Algorithms: Part 1, Fundamentals. In: *University Computing*, 1993. Vol. 15(2), pp. 58- 69.
10. RUSSELL, S.; NORVIG, P. *Artificial Intelligence: A Modern Approach*. Second Edition. Prentice Hall, 2003.