

# Problema Rutării Vehiculelor (VRP)

Bancoş Gabriel, Ciurte Iulian, Gavriş Nicolae, Haidu Eduard, Mirth Kevin, Pop Lucas

## Contents

<b>1</b>	<b>Introducere</b>	<b>2</b>
<b>2</b>	<b>Contextul şi înţelegerea proiectului</b>	<b>2</b>
2.1	Cerinţe . . . . .	3
2.2	Ce dorim sa obţinem . . . . .	4
<b>3</b>	<b>State Of The Art</b>	<b>4</b>
<b>4</b>	<b>Implementare</b>	<b>6</b>
4.1	Structura Algoritmului . . . . .	6
4.2	Formularea Matematică . . . . .	7
<b>5</b>	<b>Testare</b>	<b>8</b>
<b>6</b>	<b>Rezultate</b>	<b>8</b>
<b>7</b>	<b>Concluzii</b>	<b>8</b>
7.1	Realizări . . . . .	9
<b>8</b>	<b>Experienţa de muncă în echipă</b>	<b>9</b>

# 1 Introducere

În cadrul proiectului de inteligență artificială echipa noastră formată din Bancoș Gabriel, Ciurte Iulian, Gavriș Nicolae, Haidu Eduard, Kevin Mirth și Pop Lucas am ales să studiem și să aplicăm algoritmi genetici pentru problema rutării vehiculelor. Această documentație detaliază etapele parcurse pentru a dezvolta o posibilă soluție problemei propuse. Am ales această problemă deoarece nu este prea comună în alegerile de proiecte bazate pe algoritmi genetici aceasta având un grad de dificultate ridicat. Ne-am dorit să avem o adevărată provocare pentru a ne dezvolta abilitățile de muncă în echipă din dorința de a învăța și mai ales de a studia cât mai mult cum funcționează algoritmi genetici din spatele acestei probleme. Problema aleasă îmbină două probleme cunoscute, acestea fiind Problema Rucsacului și Problema Comis-Voiajorului. Această documentație detaliază etapele parcurse pentru a dezvolta o posibilă soluție pentru problema propusă.

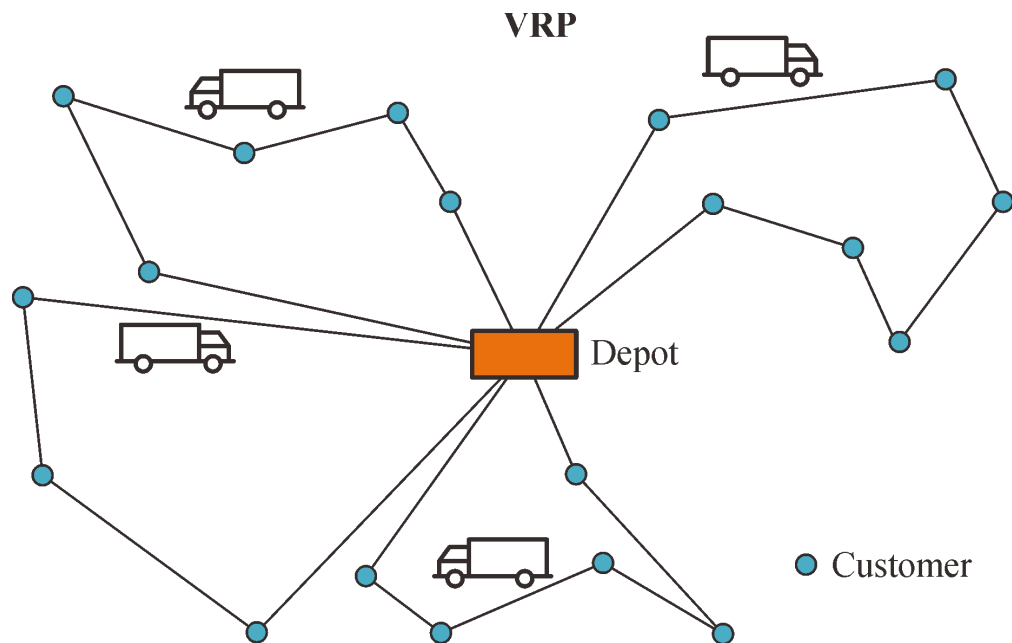
Proiectul nostru propune implementarea unui algoritm genetic pentru a ajunge la o soluție optimă. Algoritmul genetic evoluează o populație de indivizi prin selecție, crossover și mutație pentru a găsi soluția cea mai optimă. Și totodată fiecare individ având o durată de viață care scade în urma selecției.

Ne propunem din punct de vedere al algoritmului, să ne apropiem cât de mult ne permit abilitățile noastre de adevărata implementare a VRP-ului.

## 2 Contextul și înțelegerea proiectului

Scopul Problemei Rutării Vehiculelor (VRP) este de a găsi o soluție optimă având în vedere ideea de a balansa atât cantitățile de colete din fiecare tir dar și drumul total parcurs de toate camioanele și nu doar alegerea optimă a traseului doar pentru un camion.

Totodată capacitatea fiecărui camion este prestabilită și aceasta nu poate fi depășită sub nicio formă. Problema Rutării Vehiculelor (VRP): VRP-ul este o problemă de optimizare care presupune



identificarea modului de distribuire a obiectelor cu cel mai eficient randament, utilizând mai multe vehicule de la un depozit central la un set de clienți. Obiectivul principal constă în minimizarea costurilor totale, care pot include distanța totală parcursă, livrarea sau alte resurse consumate.

Elementele cheie ale VRP-ului includ: Depozitul central: Acesta este punctul de plecare și de returnare al tuturor vehiculelor. Client: Locațiile unde vehiculul trebuie să livreze. Vehicule: Un număr de vehicule care vor face livrarea.

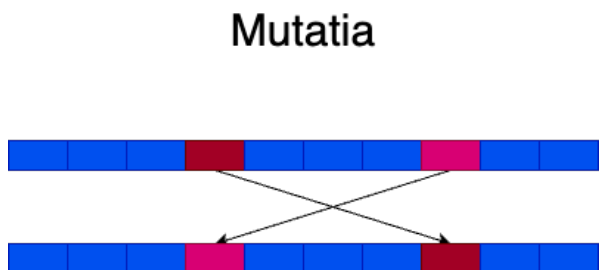
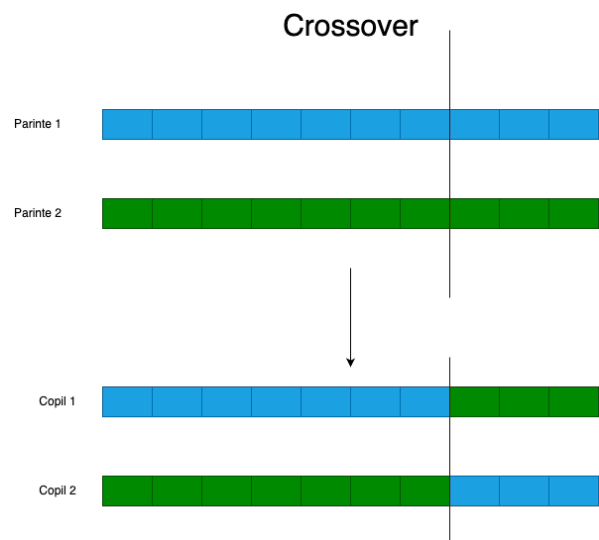
Problema Rutării Vehiculelor cu Capacitate (CVRP) - Este o variantă a VRP-ului care introduce o constrângere suplimentară: fiecare vehicul are o capacitate limită. Aceasta implică limita capacității maxime a fiecărui vehicul.

Modelul CVRP integrează constrângeri de capacitate care limitează greutatea maximă pe care fiecare vehicul o poate transporta fără a pune în pericol fiabilitatea și eficiența livrărilor. Aceste limite ajută la asigurarea utilizării optime a resurselor, prevenirea supraîncărcării vehiculelor și reducerea costurilor asociate.

Livrarea clienților: Fiecare client are o cerere specifică de livrare (cantitate de bunuri).

Obiectivul: Minimizarea costurilor totale prin identificarea rutei optime, respectând constrângerile de capacitate ale vehiculelor.

Algoritmul începe inițial cu o populație de 100 de indivizi, la care le calculăm fitness-ul pentru următorul pas care presupune să alegem pentru fiecare generație 5 indivizi per fiecare părinte folosind metoda turnirului care presupune că cel mai bun individ să treacă în pasul următor, cu mențiunea că acesta are o durată de viață nenulă, deoarece o dată ales, lui îi scade durata de viață cu o unitate. Odată aflați care sunt părinții (câștigătorii turnirurilor) aplicăm o procedură de crossover pentru a combina genele de la cei doi părinți și a ajunge la un nou individ care să fie o posibilă soluție în algoritmul nostru. Mutațiile introduc alte variații în individ, ajutând la menținerea diversității în populație.



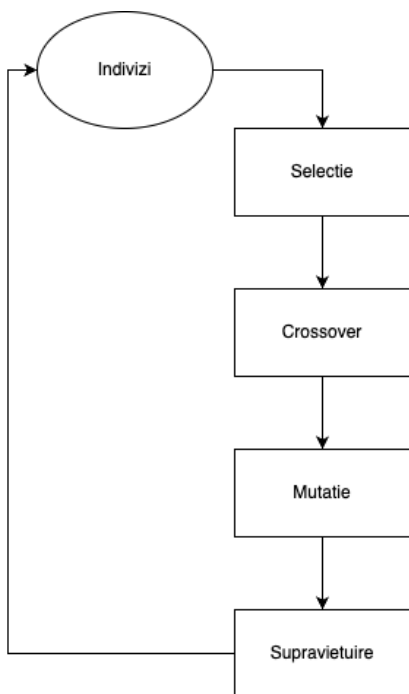
## 2.1 Cerințe

Cerințele acestui proiect presupun ca fiecare individ are scopul de a ajunge la o soluție optima avand ca scop minimizarea costurilor totale si maximizarea utilizarii vehiculelor si resurselor(colete).

Este o problema grea, complexă ceea ce înseamnă ca la fiecare iteratie care trece solutia problema ajunge spre plafonare.

Algoritmii genetici copiază modul cum teoria darwiniană presupune că evoluează speciile pe parcursul a mai multe generații. Mecanismul propus de Charles Darwin implică două proprietăți de bază:

- supraviețuirea indivizilor bine adaptați la mediu
- apariția unor modificări genetice aleatoare



## 2.2 Ce dorim sa obținem

La finalul proiectului dorim să obținem un algoritm funcțional care să îndeplinească cerințele propuse, atât și un aspect eligibil al codului.

La finalul proiectului ne dorim să obținem următoarele obiective:

- Un algoritm funcțional care să îndeplinească cerințele prezentate în cadrul documentației și să rezolve problema.
- Un cod bine structurat și lizibil care să respecte nivelul de programare să fie ușor de înțeles.
- O documentație care să explice clar implemetarea, utilizarea si rezultatele algoritmului propus.

## 3 State Of The Art

Înainte de a intra în amănunte vom începe cu o scurtă descriere despre ce înseamnă **"State Of The Art"**.

State Of The Art reprezintă nivelul actual al dezvoltării într-un anumit domeniu sau tehnologie în cazul nostru "Problema Rutării Vehiculelor(VRP)".

Problema Rutării Vehiculelor(VRP) este una dintre cele mai studiate probleme de combinatorică, fiind aplicată în domeniile de transport, logistică, livrări urbane. Aceasta generalizează binecunoscuta problemă a Comis Voiajorului.

Problema Rutării Vehiculelor(VRP) a apărut pentru prima dată într-un articol de George Dantzig și John Ramsaer din anul 1959. În care prima abordare a fost aplicată livrărilor de benzină. Așadar obiectivul VRP-ului este minimalizarea costurilor totale ale rutei.

### 1. Alogritmii Genetici

Acest studiu [1] explorează aplicarea unui algoritm genetic la problema de rutare a vehiculelor(VRP), în care clienții sunt deserviți de la un singur depozit, având limite de greutate și uneori de distanță. Lucrarea prezintă niște rezultate computaționale pentru un algoritm genetic pur și pentru o varianta hibridă. Aceste abordări sunt competitive cu metodele moderne precum Tabu Search și simulated annealing. Așadar algoritmul genetic propus de către autori demonstrează că poate fi o metodă eficientă pentru VRP.

Conform studiului [8], operatorii de mutație și cel de crossover într-un algoritm genetic contribuie la îmbunătățirea procedurii de mediere a operatorului de mutație. În acest studiu se povestește despre analiza unor diverse mecanisme de selecție a fitness-ului, selecția proporțională, selecția pe baza rangului și selecția prin turnir. Studiul arată că aceste mecanisme combinate cu operatorul de crossover pot modela și pot fi aplicate algoritmilor genetici cu rata de mutație foarte mică sau chiar zero.

În următorul articol [5] este vorba despre algoritmii genetici, aceștia fiind o familie de modele computaționale. Acești algoritmi pun bazele unei soluții potențiale la o problemă specifică pe o structură de date simplă ce se aseamănă cu un cromozom și se aplică operatori de recombinare pentru a păstra informațiile critice. Algoritmii genetici sunt deseori priviți ca pe niște optimizări de funcții deși gama de probleme la care aceștia sunt aplicați este destul de largă. Algoritmii genetici sunt metode de căutare și optimizare inspirate de selecția naturală și genetică introduse de către John Henry Holland în anii 1970. Algoritmii Genetici modelează spațiul problemei ca o populație pentru ca indivizii, fiecare reprezentând o soluție potențială. Prin iterarea generațiilor, populația evoluează, evoluția fiind măsurată printr-o funcție de fitness care exprimă adaptarea fiecărei soluții. Procedura unui **Algoritm Genetic** este: **generarea inițială** unde este concepută o populație inițială de indivizi (random), **evaluare fitness-ului** unde se calculează valoarea funcției de fitness pentru fiecare individ, **crearea unei populații noi**, **înlocuirea populației** cu noua populație, iar în cele din urmă **Repetarea**. Cele mai bune rezultate cunoscute pentru VRP au fost obținute cu metoda Tabu Search.

Conform cu studiul [6] intitulat "Solving Vehicle Routing Problem by Using Improved Genetic Algorithm for Optimal Solution", autorii aplică un algoritm genetic îmbunătățit pentru a rezolva problema rutării vehiculelor (VRP) în cadrul serviciilor de autobuze ale unei universități. În acest studiu se implementează modelul Capacitated Vehicle Routing Problem (CVRP), având ca obiectiv optimizarea rutelor de autobuz, reducând atât timpul de transport cât și distanța parcursă. Autorii subliniază că, deși reducerea distanțelor este relativ mică, economiile sunt semnificative atunci când se calculează distanțele totale parcurse zilnic sau lunar de toate autobuzele. Așadar prin utilizarea unui algoritm genetic a dus la o soluție optimă care reduce costurile.

Conform cu studiul [3], autorii acestei lucrări prezintă un algoritm genetic cu două straturi (TLGA) pentru optimizarea rutelor vehiculelor electrice cu timpi de încărcare non-liniar. Abordarea optimizează nu doar secvențele de vizite la destinații, ci și stațiile de încărcare disponibile pe traseu și timpul de încărcare a bateriei. Algoritmul depășește metodele tradiționale, abordând ramurile de încărcare neliniare și constrangerile ferestrelor de timp.

## 2. CVRP

În următorul articol [2] este analizată eficiența ratei de mutație în algoritmii genetici care folosesc o reprezentare a șirului de biți de lungime  $n$ , punând sub semnul întrebării recomandarea standard a ratei de mutație  $1/n$ . Autorii constata că ratele de mutație mai mari, de exemplu între  $2/n$  și  $m/n$ , duc la timpi de rulare mai buni. În plus se observă că rata de mutație optimă este foarte sensibilă la mici variații, iar o rată fixă de mutație nu oferă rezultate eficiente în majoritatea cazurilor. Așadar pentru a rezolva această problemă autorii propun o rată de mutație aleatoare precum  $a/n$ , unde  $a$  este selectat dintr-o distribuție Heavy-tailed.

Conform cu [10], studiul propune un algoritm genetic hibrid (HGA) pentru rezolvarea problemelor de rutare a vehiculelor cu capacitate limitată (CVRP), aplicat în domeniul industriei. Algoritmul include trei etape: 1) **generarea populației inițiale**: Combinația near addition method (NAM) și a algoritmului de tip sweep (SA) pentru a crea o populație inițială, 2) **Optimizarea probabilităților de crossover și mutație**: Folosind metoda suprafeței de răspuns (RSM) pentru ajustarea probabilităților prin experimente sistematice, 3) **Îmbunătățirea algoritmului SA**: Incorporarea unui algoritm SA îmbunătățit în procesul de mutație pentru evitarea convergenței premature și pentru a mări diversitatea soluțiilor. Studiul demonstrează aplicabilitatea HGA-ului prin exemple din practică și studii de caz, inclusiv un caz real privind distribuția activă a armatei din Taiwan.

Conform cu [4], articolul "City Vehicle Routing" prezintă o variantă a Problemei de Rutare a Vehiculelor care se aplică în contextul urban. Această variantă a problemei, pe lângă elementele de bază cum ar fi locațiile expeditorilor, punctele de livrare, numărul și capacitatea camioanelor,

ia în considerare și diversele provocări care apar atunci când avem de a face cu mediul urban, precum traficul, reglementările rutiere, poluarea și diversele cerințe ale expeditorilor, firmelor de transport și a clienților în obținerea unei soluții optime.

Conform cu [9], obiectivul CVRP-ului este de a determina rutele optime pentru mai multe vehicule de transport cu capacitate de încărcare a pachetelor limitată, astfel încât costurile totale ale transportului să fie minimizate. De asemenea, este prezentat un algoritm genetic hibrid folosit pentru rezolvarea CVRP-ului, acesta integrând: algoritmii genetici, tehnici de căutare locală, strategii de selecție și de crossover pentru a oferi o diversitate a populației pentru evitarea blocării algoritmului într-un minim local. De asemenea, folosirea unei căutări genetice hibride (HGS) specializată în problema de rutare a vehiculelor cu capacitate (CVRP). Acest algoritm de ultimă generație folosește aceeași metodologie ca și Vidal și Colab (2012), dar include și termenul de SWAP care constă în schimbul a doi clienți între rute diferite. Implementarea SWAP-ului contribuie la performanța căutărilor locale.

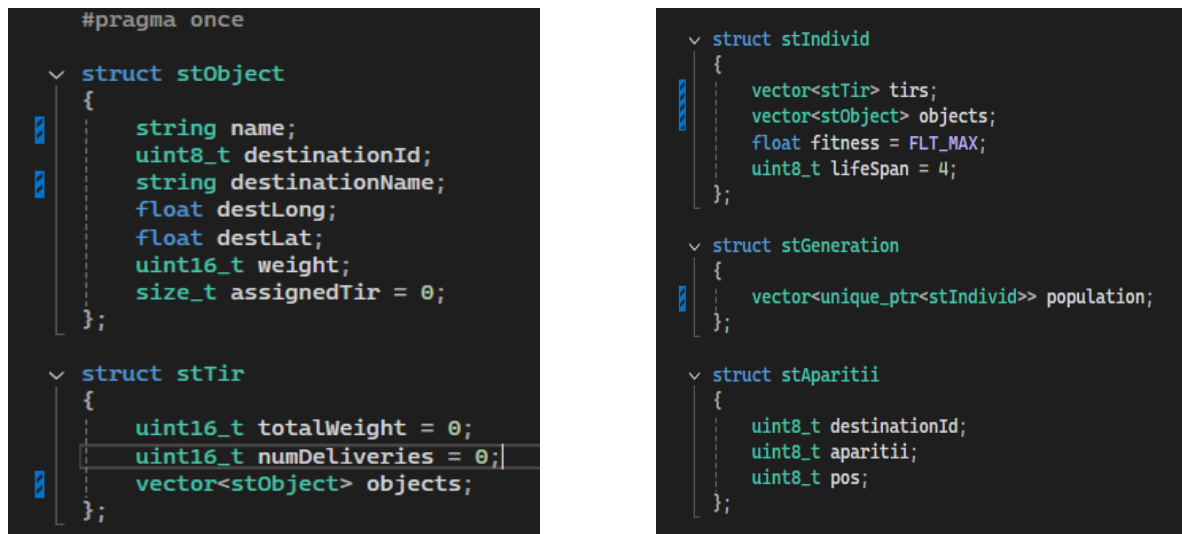
În articolul [7] intitulat "On the Capacitated Vehicle Routing Problem" T.K Ralphs, L Kopman et al propun o metoda de rezolvare a Problemei Rutării Vehiculelor ce implică descompunerea acesteia în doua părți. Cum Problema Rutării vehiculelor este o combinație dintre Problema Comis Voiajorului sau Traveling Salesman Problem (TSP) și Problema Rucsacului sau Bin Packing Problem (BRP), aceștia au ales ca aceste probleme să reprezinte cele două părți ale problemei inițiale și să le rezolve separat, iar apoi soluțiile celor două părți să fie combinate pentru obținerea soluției finale. Algoritmul propus ajută să reducă dimensiunea problemei, dar are nevoie de mai multe optimizări pentru a îmbunătăți timpul de execuție. De asemenea, autorii doresc să extindă acest algoritm pentru a rezolva variante mai complexe ale Problemei Rutării Vehiculelor, cum ar fi cele cu ferestre de timp sau constrângeri de distanță.

## 4 Implementare

Implementarea algoritmului a fost realizată folosind C++ combinat cu metode scrise în C style pentru a avea control asupra codului și o performanță a execuției. S-au utilizat un minim de librării standard precum map, set, random și string, evitând librării complexe pentru a avea un cod cât mai simplu de înțeles și de portat. Modul de abordare ales permite un memory management eficient, alocând memorie atât cât este necesar pentru fiecare metodă și variabilă.

### 4.1 Structura Algoritmului

Mai jos este prezentată structura algoritmului și o scurtă descriere pentru fiecare.



```
#pragma once

struct stObject
{
    string name;
    uint8_t destinationId;
    string destinationName;
    float destLong;
    float destLat;
    uint16_t weight;
    size_t assignedTir = 0;
};

struct stTir
{
    uint16_t totalWeight = 0;
    uint16_t numDeliveries = 0;
    vector<stObject> objects;
};

struct stIndivid
{
    vector<stTir> tirs;
    vector<stObject> objects;
    float fitness = FLT_MAX;
    uint8_t lifeSpan = 4;
};

struct stGeneration
{
    vector<unique_ptr<stIndivid>> population;
};

struct stAparitie
{
    uint8_t destinationId;
    uint8_t aparitie;
    uint8_t pos;
};
```

**Structura obiectului** conține attribute de identificare pentru a îl reprezenta. Având informații despre numele obiectului, destinația acestuia, coordonatele geografice, greutate și tirul în care acesta

a fost plasat.

**Structura tirului** reprezintă tirul propriu-zis. Atributele **totalWeight** și **numDeliveries** ajută la urmărirea sarcinii totale și numărului de livrări realizate de fiecare tir. Lista de **obiecte** permite asocierea directă a obiectelor cu tirurile respective.

**Structura individului** reprezintă o soluție individuală în cadrul algoritmului, fiecare individ având o listă de tiruri pentru a aranja obiectele. Atributele **fitness** și **lifeSpan** sunt folosite pentru evaluarea performanței fiecărui individ în cadrul generației curente. Variabila **fitness** este inițializată cu **FLT\_MAX**, indicând pe măsură ce algoritmul rulează.

**Structura stGeneration** reprezintă o generație de indivizi, adică un set de posibile soluții. Utilizarea unui vector de **unique\_ptr** pentru indivizi permite o gestionare eficientă a memoriei, asigurându-se că fiecare individ este unic și că resursele sunt eliberate automat când nu mai sunt necesare.

**Structura stAparitii** este folosită pentru a ține evidența destinațiilor și a aparițiilor obiectelor în tiruri. Fiecare obiect are asociate un **destinationId**, numărul de apariții și poziția în tir, ceea ce ajută la optimizarea încărcării și livrării.

## 4.2 Formularea Matematică

Mai jos este prezentată formula generală a VRP-ului și anumite constrângeri care se regăsesc și în problema noastră.

### 1. Formula Generală a VRP

#### (a) Formula:

$$\text{Minimizează } \sum_{k=1}^m \sum_{i=0}^n \sum_{j=0}^n c_{ij} x_{ij}^k$$

Unde:

- $m$  este numărul de vehicule disponibile
- $n$  este numărul total de clienți
- $c_{ij}$  este costul de la locația  $i$  la locația  $j$
- $x_{ij}^k$  este o variabilă binară care este 1 dacă vehiculul  $k$  merge de la locația  $i$  la locația  $j$ , altfel este 0.

#### (b) Constrângeri:

- Fiecare client este vizitat o singură dată:

$$\sum_{k=1}^m \sum_{j=0}^n x_{ij}^k = 1 \quad \forall i = 1, \dots, n$$

- Conservarea fluxului pentru fiecare vehicul:

$$\sum_{i=0}^n x_{ij}^k = \sum_{j=0}^n x_{ji}^k \quad \forall k = 1, \dots, m, \forall i = 0, \dots, n$$

- Capacitatea vehiculului:

$$\sum_{i=1}^n d_i \sum_{j=0}^n x_{ij}^k \leq Q_k \quad \forall k = 1, \dots, m$$

Unde  $d_i$  este cererea clientului  $i$  și  $Q_k$  este capacitatea vehiculului  $k$

- Vehiculul pleacă și se întoarce la depozit:

$$\sum_{j=1}^n x_{0j}^k = \sum_{i=1}^n x_{i0}^k = 1 \quad \forall k = 1, \dots, m$$

## 5 Testare

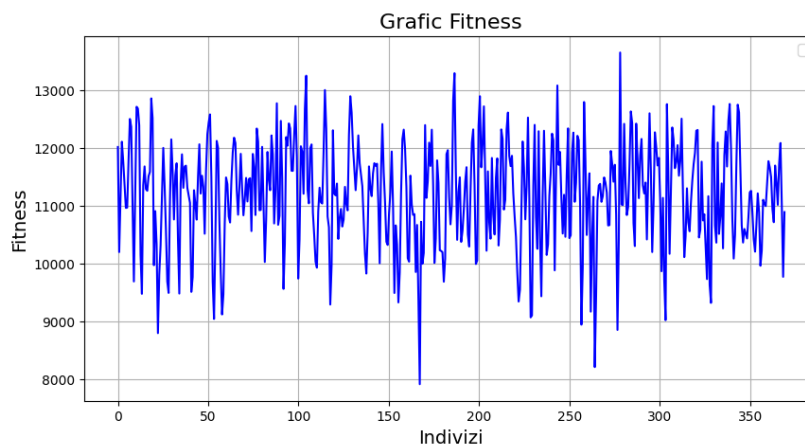
La partea de testări nu am avut părți din cod pentru a testa. Singura "Testare" pe care am făcut-o, a fost să mărim sau să micșorăm rata de mutație, când creșteam rata de mutație fitness-ul devenea unul bun, iar dacă o micșoram se bloca într-un minim local. În cele din urmă s-a optat pentru o rată de mutație statică.

## 6 Rezultate

În imaginea de mai jos este prezentată o rulare a programului unde se pot observa indivizii, fitness-ul acestora și viața fiecărui individ în funcție de selecție, în rularea respectivă au fost generate 20 de generații la o populație de 400 de indivizi.

```
Individ 422 [fitness: 9969.523438, lifeSpan: 4]
Individ 423 [fitness: 10367.905273, lifeSpan: 3]
Individ 424 [fitness: 10200.981445, lifeSpan: 4]
Individ 425 [fitness: 8889.405273, lifeSpan: 3]
Individ 426 [fitness: 10990.759766, lifeSpan: 4]
Individ 427 [fitness: 8981.521484, lifeSpan: 4]
Individ 428 [fitness: 11571.293945, lifeSpan: 4]
Individ 429 [fitness: 11996.889648, lifeSpan: 4]
Individ 430 [fitness: 11709.185547, lifeSpan: 4]
Individ 431 [fitness: 9385.000000, lifeSpan: 4]
Individ 432 [fitness: 9233.738281, lifeSpan: 4]
Individ 433 [fitness: 11569.752930, lifeSpan: 4]
Individ 434 [fitness: 12659.240234, lifeSpan: 4]
Individ 435 [fitness: 10175.412109, lifeSpan: 4]
Individ 436 [fitness: 10943.770508, lifeSpan: 4]
Individ 437 [fitness: 11249.614258, lifeSpan: 4]
Individ 438 [fitness: 10577.923828, lifeSpan: 4]
Best fitness: 7516.750000 [0.05 mutation rate]
```

Iar în imaginea următoare este prezentat graficul rezultat din fitness-ul indivizilor.



## 7 Concluzii

În concluzie acest ultim capitol prezintă rezultatele obținute și obiectivele care au fost atinse. În acest proiect, am reușit să realizăm un algoritm genetic pentru a găsi o soluție cât mai eficientă pentru Problema Rutării Vehiculelor și să atingem obiectivele propuse. Printre realizările noastre se enumără crearea unui algoritm care funcționează bine și oferă soluții apropiate optime pentru organizarea traseelor și livrarea coletelor, dar în același timp am scris și organizat codul într-un mod clar, astfel încât să fie ușor de înțeles. Acest proiect ne-a arătat cum pot fi folosiți algoritmi genetici pentru a rezolva probleme din viața reală, cum ar fi organizarea transporturilor. Chiar dacă soluția noastră mai poate



fi îmbunătățită, considerăm că am reușit să găsim un echilibru între simplitate și eficiență, totodată realizând o lucrare practică și interesantă.

## 7.1 Realizări

Algoritmul prezentat îndeplinește cerințele propuse. Au fost implementate operații de selecție, crossover, mutație și supraviețuire. De asemenea și rezolvarea unei probleme complexe.

## 8 Experiența de muncă în echipă

Munca în echipă este un lucru esențial în dezvoltarea de proiecte, oferind oportunitatea echipei de a-și împărți sarcinile în funcție de cunoștințele fiecărui membru. Totodată munca în echipă scoate la suprafață probleme apărute pe parcursul proiectului, dar de asemenea, ne ajută și la acumularea unor noi cunoștințe.

Echipa noastră a fost formată din 6 membri, fiecărui membru atribuindu-se un anumit task.

Proiectul a fost structurat pe mai multe etape având deadline-uri o data la două săptămâni.

Colaborarea și munca în echipă au fost de cele mai multe ori bune, totodata având și anumite provocări care țineau de structurarea aplicației. În mare parte nu au fost probleme, mai puțin problema sincronizării timpului de întâlnire pentru a lucra.

În concluzie acest proiect ne-a ajutat să ne îmbunătățim munca în echipă, având în vedere că nu știam ce abilități are fiecare membru și totodata descoperirea unor noi domenii de lucru.

Ne-a ajutat să avem o privire de ansamblu diferită față de cea pe care o aveam înainte de a lucra la proiect.

În următorul tabel avem marcate pentru fiecare membru al echipei, capitolele unde au fost aduse cele mai semnificative contribuții de către fiecare membru. Tabelul prezentat este menit să exemplifice activitatea fiecărui membru al echipei. Acest tabel este strict orientativ deoarece membrii echipei au contribuit la mai multe capitole având un nivel de implicare nu la fel de semnificativ.

### Tabel Contribuții Proiect

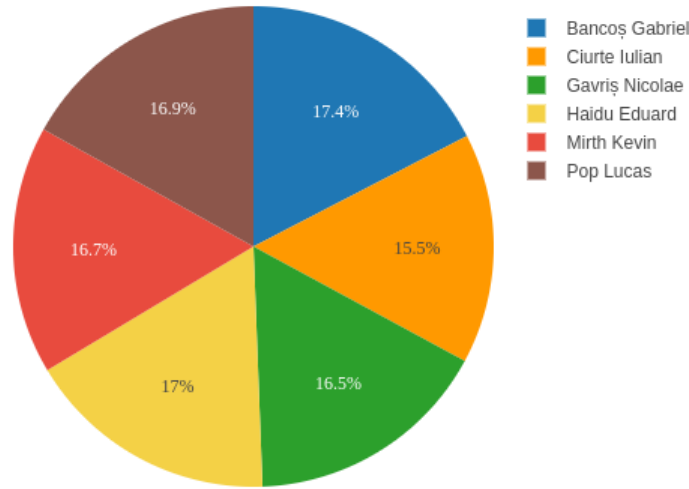
capitol\membru	Bancoș Gabriel	Ciurte Iulian	Gavriș Nicolae	Haidu Eduard	Mirth Kevin	Pop Lucas
Documentație	x	x	x	x	x	x
Prezentare			x	x	x	
Programare	x			x		x
Diagrame		x				

### Tabel Contribuții Documentație

În cea mai mare parte membrii echipei Mirth Kevin și Pop Lucas au făcut documentația, dar ceilalți membrii au contribuit și ei cu câteva idei pe parcurs.

capitol\membru	Bancoș Gabriel	Ciurte Iulian	Gavriș Nicolae	Haidu Eduard	Mirth Kevin	Pop Lucas
Introducere	x	x	x	x	x	x
Contextul proiectului	x		x		x	x
State Of The Art	x	x	x	x	x	x
Implementare					x	x
Testare	x			x	x	x
Rezultate	x			x	x	x
Concluzii		x	x		x	x

**Contribuții**



## References

- [1] Barrie M. Baker and M.A. Ayechev. A genetic algorithm for the vehicle routing problem. *Computers Operations Research*, 30(5):787–800, 2003.
- [2] Benjamin Doerr, Huu Phuoc Le, Régis Makhlara, and Ta Duy Nguyen. Fast genetic algorithms. In *Proceedings of the genetic and evolutionary computation conference*, pages 777–784, 2017.
- [3] Sašo Karakatič. Optimizing nonlinear charging times of electric vehicle routing with genetic algorithm. *Expert Systems with Applications*, 164:114039, 2021.
- [4] Gitae Kim, Yew-Soon Ong, Chen Kim Heng, Puay Siew Tan, and Nengsheng Allan Zhang. City vehicle routing problem (city vrp): A review. *IEEE Transactions on Intelligent Transportation Systems*, 16(4):1654–1666, 2015.
- [5] Seyedali Mirjalili and Seyedali Mirjalili. Genetic algorithm. *Evolutionary algorithms and neural networks: theory and applications*, pages 43–55, 2019.
- [6] Mazin Abed Mohammed, Mohd Khanapi Abd Ghani, Raed Ibraheem Hamed, Salama A Mostafa, Mohd Sharifuddin Ahmad, and Dheyaa Ahmed Ibrahim. Solving vehicle routing problem by using improved genetic algorithm for optimal solution. *Journal of computational science*, 21:255–262, 2017.
- [7] Ted K Ralphs, Leonid Kopman, William R Pulleyblank, and Leslie E Trotter. On the capacitated vehicle routing problem. *Mathematical programming*, 94:343–359, 2003.
- [8] Lothar M Schmitt. Theory of genetic algorithms. *Theoretical Computer Science*, 259(1-2):1–61, 2001.
- [9] Thibaut Vidal. Hybrid genetic search for the cvrp: Open-source implementation and swap\* neighborhood. *Computers & Operations Research*, 140:105643, 2022.
- [10] Chung-Ho Wang and Jiu-Zhang Lu. A hybrid genetic algorithm that optimizes capacitated vehicle routing problems. *Expert Systems with Applications*, 36(2):2921–2936, 2009.