Rezolvarea Problemei Comisului Voiajor cu ajutorul unui Algoritmul Genetic si a tehnicii Simulated Annealing

Haivas Daniel-Andrei

6 Ianuarie 2020

Abstract

Acest raport pune in prim-plan Problema Comisului-Voiajor, problema care va fi rezolvata atat cu un Algoritm Genetic cat si cu un algoritm de tip Simulated Annealing. Totodata, sunt prezentati cei 2 algoritmi alaturi de imbunatatiri ale lor, cele mai importante fiind: schimbarea modului de scadere a temperaturii pentru algoritmul euristic si modificarea adaptiva a probabilatilor specifice algoritmului genetic. In final s-a constatat faptul ca algoritmul de tip Simulated Annealing este mult mai bun decat Algoritmul Genetic.

1 Introducere

1.1 Introducere generală

În acest raport se ia spre analiză cunoscuta Problemă a Comisului Voiajor (fiind data o hartă cu orașe și distantele dintre ele, se dorește aflarea unui parcurs minim care să parcurgă toate orașele o singura data). Ea are o deosebit de mare importanta în viață cotidiana, din acest motiv a devenit extrem de consacrata și cunoscuta.

Aceasta este o problema NP-hard, deci rezolvarea ei printr-un algoritm obișnuit nu este posibila intru-un timp rezonabil. Din acest motiv se vă încercă rezolvarea ei printr-un algoritm genetic și prin metoda Simulated Annealing. Cele doua metode vor fi comparate amănunțit pentru a decide care aduce rezultate mai bune.

Raportul prezinta în prima secțiune, "Introducere", informații introductive referitoare atât la motivarea studierii Problemei Comisului Voiajor, cât și la descrierea acesteia. În capitolul "Metode" sunt prezentate amânunțit Algoritmul Genetic și algoritmul de tip Simulated Annealing, cât și micile retușuri aduse asupra acestora pentru a obține rezultate cât mai apropiate de realitate.

Următoarele doua secvențe, "Experimente" și "Comparații", au că scop observarea rezultatelor, dar și a modului în care rulează cei 2 algoritmi. Totodată,

se măsoară diferențele intre rezultatele obținute, că apoi, în "Concluzii", să se poată trage niste concluzii pertinente asupra eficienței unui algoritm genetic sau a unui algoritm de tip Simulated Annealing în rezolvarea problemei propuse.

1.2 Motivație

Utilitatea ei practica este principalul motiv pentru care s-a decis studierea Problemei Comisului Voiajor. Aplicații ale acestei probleme se pot găsi în: transportul mărfurilor, secventierea ADN-ului, logistica, fabricare de microcipuri etc. Astfel, un algoritm care rezolva aceasta problema într-un mod eficient, este extrem de prețios deoarece câteva schimbări aduse asupra lui îl pot face să rezolve o alta problema din viață cotidiana, precum cele din domeniile enumerate anterior.

Totodată, raportul are ca scop aflarea cărui algoritm este preferabil de utilizat în practica, pentru rezolvarea unor astfel de probleme dintre cei 2 menționați anterior. Algoritmii genetici sunt algoritmi inspirați din domenii precum biologia ce au ajuns să aibă un rol foarte important în domenii precum matematică și informatică. Simulated Annealing este o metoda probabilista de rezolvare a problemelor de optimizare ce a apărut că o îmbunatățire pentru varianta Hill Climbing. Drept urmare, ambele metode au șanse reale să rezolve corect problema propusa.

1.3 Descriere problemă

Problema Comisului Voiajor este definită astfel: Fie G = (V, E) este un graf neorientat în care oricare două vârfuri diferite ale grafului sunt unite printr-o latură căreia îi este asociat un cost strict pozitiv. Cerinţa este de a determina un ciclu care începe de la un nod aleatoriu al grafului, care trece exact o dată prin toate celelalte noduri şi care se întoarce la nodul iniţial, cu condiţia că ciclul să aibă un cost minim. Costul unui ciclu este definit că suma tuturor costurilor ataşate laturilor ciclului.

Numele problemei provine din analogia cu un vânzător ambulant care pleacă dintr- un oraş, care trebuie să viziteze un număr de oraşe dat şi care apoi trebuie să se întoarcă la punctul de plecare, cu un efort minim (de exemplu timpul minim, caz în care costul fiecărei laturi este egal cu timpul necesar parcurgerii drumului).

Pornind de la aceasta problema, au apărut multe probleme noi care se rezolva în același fel, modelandu-se datele de intrare pe problema Comisului-Voiajor. Spre exemplu, se pot modela că orașe și distanta dintre ele următoarele: clienții și durata de deplasare (în logistica), secvențele de ADN o măsură similarității intre acestea (în biologie) etc.

2 Metode

Pentru rezolvarea Problemei Comisului-Voiajuri se va folosi un Algoritm Genetic cât și un algoritm bazat pe metoda Simulated Annealing.

2.1 Simulated Annealing

Simulated Annealing este o metoda probabilista de rezolvare a problemelor de optimizare, inspirata dintr-un proces din natura. Acest algoritm a apărut că o îmbunatățire pentru varianta Hill Climbing, care avea ca dezavantaj faptul că rămâne blocată în puncte de minim local.

El este similar cu procedeul de încălzire al unui material, urmat de răcirea să lenta astfel încât să i se reducă defectele. Inițializarea și scăderea temperaturii reprezintă un element foarte important în algoritmul de Simulated Annealing. Ideal ar fi ca temperatura să scadă treptat de la valoarea inițială până ajunge la o valoare mai mica decat 0. Ulterior, algoritmul va sta mai mult timp la valori mici, pentru a ne putea apropia cât mai mult de rezultatul optim al problemei date.

Algoritmul de tip Simulated Annealing presupune alegerea aleatorie a unei soluții candidat. Ulterior, se vor alege vecini ai acestei soluții pentru a vedea daca aduc rezultate mai bune. Acest procedeu se oprește la îndeplinirea unei condiții favorabile, iar întreagă procedura explicata vă fi repetata cât timp temperatura nu a scăzut suficient.

2.2 Algoritm Genetic

Algoritmii genetici reprezintă tehnici adaptive de căutare euristică, bazate pe principiile geneticii și ale selecției naturale, enunțate de Darwin ("supraviețuiește cel care e cel mai bine adaptat"). Algoritmul lucrează cu noțiunea de cromozom, care va memora trăsăturile unui individ.

Pentru reprezentarea lor se folosesc șiruri de biți care vor putea fi traduși în valoarea unui posibil punct de minim.Asupra acestor cromozomi se vor aplica anumite operații astfel încât să se ajungă la rezultatul dorit. Procesul biologic al evoluției este similar cu mecanismul descris de un algoritm genetic.

El posedă o trăsătură prin care numai speciile care se adaptează mai bine la mediu sunt capabile să supraviețuiască și să evolueze peste generații, în timp ce acelea mai puțin adaptate nu reușesc să supraviețuiască și cu timpul dispar, ca urmare a selecției naturale.

Probabilitatea ca specia să supraviețuiască și să evolueze peste generații devine cu atât mai mare cu cât gradul de adaptare crește, ceea ce în termeni de optimizare înseamnă că soluția se apropie de optim. Algoritmul genetic presupune apelarea următoarelor proceduri asupra unei populații de la un moment dat:

1. Selecția

   În acest pas se creează copiii, astfel încât se păstrează cei cu cea mai bună adaptare (cel mai mic rezultat) și implicit cu șansa mai mare să ajungă la soluția optimă.

2. Mutația

   Se trece prin fiecare genă a cromozomului și se modifică (din 0 în 1 sau din 1 în 0), în funcție de probabilitatea de mutație aleasă.

3. Încrucișarea

   Trebuie ales aleator un punct de tăiere și selectați 2 cromozomi. Procedeul presupune să se formeze 2 copii care vor fi creați din: prima parte (până la taiere) a unui cromozom și a doua parte (după taiere) a celuilalt cromozom.

4. Evaluarea

   Aici se decodifică cromozomul din valoare binară în valoare reală si se decide dacă acest candidat reprezintă o soluție mai bună, prin intermediul funcției fitness.

3 Experimente

3.1 Simulated Annealing

Semnificativ pentru algoritmul de tip Simulated Annealing este temperatura de start ($T_s = 1000$) și temperatura finala ($T_f = 10^{-8}$). Desi se obțineau valori bune într-un timp foarte scurt cu funcția de răcire $T = T * 0.99$ și 300 de iterații în bucla interioara, am decis să aducem niste modificări algoritmului pentru a obține rezultate și mai bune.

Cu cât dimensiunea inputului este mai mare, cu atât temperatura a fost scăzută mai lent cu următoarele funcții:

- $T = T * 0.99$, dacă dimensiunea ≤ 35
- $T = T * 0.999$, dacă $35 <$ dimensiunea ≤ 120
- $T = T * 0.9995$, dacă $120 <$ dimensiunea.

Totodată, relevant pentru eficienta algoritmului este și condiția din bucla while interioara. Bucla se termina când au loc:

- 300 iterații, dacă dimensiunea ≤ 35
- 500 iterații, dacă $35 <$ dimensiunea ≤ 120
- 1000 iterații, dacă $120 <$ dimensiunea.

De asemenea, algoritmul a fost rulat de 30 de ori pentru fiecare instanță, pentru a obține rezultate relevante din punct de vedere statistic.

Intrare	Dimensiune	Solutie	Minim	Maxim	Media	Dev. st.	Timp(s)
br17	17	39	39	39	39	0	91
ftv33	34	1286	1336	1424	1365	22.39	441
p43	43	5620	5622	5629	5625	2.88	851
ry48p	48	14422	14527	14813	14680	84.73	1014
ftv70	71	1950	2054	2146	2104	36.1	1201
kro124	100	36230	38138	39971	38924	377.6	1575
rbg323	323	1326	1388	1418	1399	7.79	4231
rbg358	358	1163	1232	1259	1243	7.41	4872
rbg403	403	2465	2472	2491	2483	6.05	5210
rbg443	443	2720	2744	2765	2754	8.08	5824

3.2 Algoritm Genetic

Semnificativ pentru o rulare eficienta a algoritmului genetic este dimensiunea populației (100), dar și numărul de iterații ale acesuia (10.000). O importanta deosebita pentru acest algoritm o au valorile alese pentru probabilitatea de mutație (PM) și probabilitatea de încrucișare (PC). Cu cât dimensiunea instanței este mai mare, cu atât aceste probabilități au valori mai mici astfel:

- $PC = 0.5$ și $PM = 0.1$, dacă dimensiunea ≤ 35
- $PC = 0.3$ și $PM = 0.015$, dacă $35 <$ dimensiunea ≤ 120
- $PC = 0.25$ și $PM = 0.005$, dacă $120 <$ dimensiunea.

Totodată, au fost folosiți operatori adaptivi care scad la fiecare "pas" PC cu 0.01 iar PM cu 0.001, până când PC ajunge la 0.01 iar PM la 0.001. Acest "pas" reprezintă parcurgerea unui număr de iterații din numărul total de iterații ale algoritmului și este cu atât mai mic cu cât dimensiunea e mai mare:

- 750 iterații, dacă dimensiunea ≤ 35
- 500 iterații, dacă $35 <$ dimensiunea ≤ 120
- 250 iterații, dacă $120 <$ dimensiunea.

De asemenea, algoritmul a fost rulat de 30 de ori pentru fiecare instanță, pentru a obține rezultate relevante din punct de vedere statistic.

Intrare	Dimensiune	Solutie	Minim	Maxim	Media	Dev. st.	Timp(s)
br17	17	39	39	40	39.3	0.19	251
ftv33	34	1286	1351	1422	1388	28.21	523
p43	43	5620	5661	5742	5708	31.78	611
ry48p	48	14422	21309	25647	23130	1598.3	689
ftv70	71	1950	3663	4216	3987	218.96	1525
kro124	100	36230	72856	88452	79542	6154.2	2112
rbg323	323	1326	3870	4146	4040	94.38	21785
rbg358	358	1163	4074	4521	4328	166.6	28465
rbg403	403	2465	5201	5444	5303	99.33	32967
rbg443	443	2720	5902	6475	6240	170.15	39756

4 Comparații

Din privință rezultatelor observam că algoritmul de tip Simulated Annealing s-a descurcat foarte bine, acesta ajungând să obțină rezultate foarte apropiate de soluțiile optime. Totodată, se poate vedea că timpul creste semnificativ pe măsură ce intervin schimbările impuse pentru a obține rezultate mai bune. Totuși, timpul total nu atât de mare, fapt ce ne demonstrează că modificările aduse algoritmului au meritat.

Din cel de-al doilea tabel se poate observa că rezultatele date de algoritmul genetic nu sunt prea bune, acesta obținând valori de 2 - 3 ori mai mari decât soluția optima. De asemenea, timpul este foarte mare, mai ales pentru dimensiuni mare ale instanțelor.

Făcând comparație intre cele doua tabele, se poate observa că rezultatele din tabelulu algoritmului de tip Simulated Annealing sunt mult mai bune decât valorile obținute de către Algoritmul Genetic (atât soluțiile obținute cât și timpul de rulare).

5 Concluzii

Atât modifcarile aduse algoritmului de tip Simualted Annealing cât și Algoritmului Genetic au reprezentat îmbunătățiri, rezultatele obținute când acestea erau valabile fiind cele mai bune. În cazul algoritmului Simulated Annealing, o scădere a temperaturii lenta determina existenta mai multor soluții candidat, iar creșterea numărului de iterații din bucla interioara determina verificarea mai multor vecini ai unei anumite soluții.

Desi rezultatele algoritmului genetic nu sunt foarte bune, lipsa acestor modificări ar fi dus la rezultate mult mai slabe. Pentru dimensiuni mai mici, valorile probabilităților trebuie să fie mai mari deoarecere lungimea unui cromozom fiind mica, șansă de a realiza o mutație sau o încrucișare este foarte scăzută. De asemenea, "pasul" pentru scăderea probabilităților este mai mare la dimensiuni mai mici, pentru că se dorește păstrarea unor probabilități crescute pentru mai mult timp.

Așadar, în urma îmbunătățirii și analizei celor 2 algoritmi luați drept studiu în acest raport, putem concluziona că cel mai bun pentru rezolvarea Problemei Comisului Voiajor este un algoritm de tipul Simulated Annealing, acesta având rezultate mai bune într-un timp mai scurt, comparativ cu Algoritmul Genetic.

References

[1] https://profs.info.uaic.ro/~eugennc/teaching/ga/

[2] https://gitlab.com/eugennc/teaching/-/tree/master/GA

[3] http://www.geatbx.com/docu/fcnindex-01.html#P89_3085

[4] http://www.cplusplus.com/reference/vector/vector/

[5] https://ro.wikipedia.org/wiki/Algoritm_genetic

[6] https://www.sciencedirect.com/topics/materials-science/simulated-annealing

[7] http://webspace.ulbsibiu.ro/daniel.morariu/html/StudentDoc/ML/IA-laborator5.pdf

[8] http://www.bel.utcluj.ro/dce/didactic/tice/15_AlgoritmiGenetici.pdf

[9] https://www.researchgate.net/post/Simulated_Annealing_vs_genetic_algorithm

[10] https://www.hindawi.com/journals/cin/2016/1712630/?fbclid=IwAR0XFMS4A9Dkl5GGp61O4c1wC4zOtm-s_h7tB_KhRr_1UuwBa36i3FvZm3o