Soluţionarea problemei comisului voiajor utilizând algoritmi meta-euristici

Balteanu Andrei 2A4, Tabalae Ioan-Sebastian 2A4

Ianuarie 2022

Abstract

Scopul acestei lucrari este de a soluţiona cunoscuta problemă de optimizare combinatorială a comisului voiajor în varianta asimetrică (*Assymetric Travelling Salesman Problem*) abordând două meta-euristici standard: *Simmulated Annealing* şi *Algorimtul Genetic*. Pentru a concluziona, dar mai ales compara, performanţa şi eficacitatea celor două metode, acestea au fost testate pe 10 instanţe de diferite dimensiuni, dintre care 2 având peste 300 de oraşe. Aşadar, în urma unor serii de teste experimentale observăm existenţa unor rezultate similare în ambele cazuri, deşi Algoritmul Genetic tinde să genereze rezultate mai bune pentru instanţe de dimensiuni mici.

1 Introducere

Problema comisului voiajor(ATSP) este una dintre cele mai cunoscute probleme NP-complete în optimizarea combinatorială. Comis voiajorul trebuie să parcurgă n oraşe pe un drum cât mai scurt, trecând prin fiecare oraş exact o dată revenind la punctul de plecare. Fromularea echivalentă matematică e reprezentată de găsirea unui circuit hamiltonian de cost minim într-un graf complet dat.

Aparţinând clasei de probleme NP-complete, spaţiul de căutare se extinde la cel puţin ordinul exponenţial al numărului de oraşe: totalul de rute posibile pentru n oraşe este de $(n-1)!$. În acest caz algloritmii tradiţionali de optimizare (greedy, branch and bound, etc.) sunt ineficienţi.

În soluţionarea acestei probleme optăm pentru algoritmi meta-euristici, inspiraţi din natură. Cele mai optime sisteme adaptive cunoscute sunt cele biologice, motivând astfel utilizarea unor astfel de algoritmi.

Simmulated Annealing Simulează procesul fizic de încălzire al unui metal şi de scădere treptată al acestuia, scăzând energia sistemului. În algoritmul descris, temperatura scade treptat de la o valoare iniţială (100) la zero. La fiecare iteraţie se selectează aleatoriu o soluţie a cărei calitate este evaluată şi actualizată în funcţie de probabilităţile de temperatură dependente de selectarea soluţiilor mai bune sau mai slabe. Pe măsură ce temperatura scade, algoritmul reduce amploarea căutării sale.

Algoritmul Genetic este o tehnică adaptivă de căutare euristică, bazată pe principiile geneticii şi ale selecţiei naturale. Mecanismul este similar procesului biologic al evoluţiei. Aşadar, algoritmul genetic modifică în mod repetat o populaţie de candidaţi care sunt supuşi la un moment dat unor modificări, precum mutaţia şi schimbul de informaţie genetică dintre aceştia.

Rezultatele extrase în urma experimentului vor ajuta la o comparație în ceea ce privește algoritmul genetic și algoritmul de tip Simulated Annealing. Aceste rezulate vor fi extrase în mod relevant statistic fiecare algoritm fiind rulat de 30 de ori pentru fiecare instanță a problemei.În plus vom observa influența instanțelor asupra rezultatelor, modificările pe care le suferă algoritmii în funcție de numărul de orașe și de repartiția distanțelor dintre orașe.

2 Metode

2.1 Algoritmul genetic

2.1.1 Codificarea soluțiilor

Reperezentarea soluțiilor candidat este considerată a fi factorul central de care depinde succesul sau eșecul unui algoritm gentic. Reprezentarea cea mai des întâlnită este cea din algoritmul genetic clasic și mai exact șirul de biți cu dimensiune și ordine fixe.

În problema de optimizare prezentată există n orașe, fiecărui oraș fiindu-i asociat un număr întreg din intervalul $[0, n-1]$. Astfel, un cromozom va fi reprezentat de o permutare a celor n orașe, codificând un posibil drum. Aceasta permutare va fi condificata pentru a utiliza mai usor și fară erori operatorii specifici algoritmului genetic.

2.1.2 Inițializarea populației

Utilizăm o funcție de randomizare pentru generarea populației inițiale, care cuprinde 100 de indivizi. Lunigmea cromozomilor depinde în mod direct de numărul de orașe din instanță, aceasta fiind egală cu $n - 1$. Acestă inițializare va fi reprezentată de varianta codificată a permutarii.

2.1.3 Selecția și evaluarea

În pasul de selecție al algoritmilor genetici sunt create copii ale cromozomilor care participă la reproducere pentru a crea descendenți. Scopul selecției este acela de a alege pentru supraviețuire cei mai adaptați indivizi din populație. Algoritmul Genetic clasic păstrează populația de aceeași dimensiune astfel prin selecție unii cromozomi pot sa dispară, iar alții să fie copiați de mai multe ori.

Se aleg candidații pentru următoarea generație pe baza funcției fitness, direct proporțional, prin metoda *roata norocului*. Funcția fitness este utilizată pentru a rezuma, cât de aproape este o soluție dată de atingerea obiectivelor stabilite. Cu cât valoarea funcției fitness este mai bună, cu atât cromozomul are șansă mai mare de a fi selectat. Cifra de merit a funcției fitness în cadrul algoritmului implementat e calculată prin formula :

$$fitness(x) = \frac{1}{(suma\ distanțelor)}$$

2.1.4 Mutația

Se trece prin fiecare genă a cromozomului și se generează un numar aleator din intervalul $(0, 1]$, dacă numărul ales este mai mic decât variabila care indică probabilitatea de mutație atunci acea gena se modifica. Modificam astfel codificarea permutarii, pastrând regulile prin care se obține o permutare validă a orașelor după decodificare pentru calculul rezultatului.

2.1.5 Încrucișarea

Se alege o poziție la întamplare și doi cromozomi, se formează astfel doi copii, unde unul din ei va moșteni genele unui parinte până la poziția de taiere, apoi genele celuilalt, și analog pentru cel de-al doilea copil. Această schemă de încrucișare poate deveni mai variată, însă pentru algoritmul genetic clasic aceasta și-a dovedit utilitatea. Alegerea celor doi cromozomi se bazează pe o schemă de selecție. Se generează o listă de probablități și se ordonează în funcție de această probabilitate indicii cromozomilor. Se aleg apoi câte doi cromozomi, a căror probabilitate este mai mică decât variabila care reprezintă probabilitatea de încrucișare.

2.2 Simmulated Annealing

Simulated Annealing se bazează pe practici metalurgice prin care un metal este încălzit la o temperatură ridicată și răcit. La temperaturi ridicate, atomii se pot schimba imprevizibil, eliminând deseori impuritățile pe măsură ce materialul se răcește într-un cristal pur. Temperatura scade progresiv de la o valoare inițială spre valoarea aleasă ca extremă de oprire, ceea ce permite o bună explorare a spațiului de căutare.

La fiecare pas algoritmul creează o nouă soluție candidat, reprezentată in aceeași manieră ca și în cazul **Algoritmului Genetic**, metode descrise in secțiunile **2.1.1** respectiv **2.1.2** . Se generează aleatoriu o soluție apropiată de cea curentă și evaluând calitatea acesteia în funcție de dependența probabilistică față de temperatură, se decide alegerea soluției actuale.

3 Rezultate experimentale

Pentru fiecare dintre cei doi algoritmi instanțele sunt rulate de un număr de 30 de ori astfel încât rezultatele extrase să fie relevante din punct de vedere statistic. Instanțele pe care le analizăm individual sunt preluate din librăria TSPLIB, diferind prin numărul de orașe și distanțele dintre acestea.

În cadrul Algoritmului Genetic condiția de oprire este reprezentată de numărul de generații la care am ajuns. Pentru rezultatele extrase am ales cea mai bună soluție a ultimei generații, 1000, aceasta fiind și cea mai evoluată. Printre alte date fixate am utilizat probabilitatea de mutație de 0.01, iar probabilitatea de încrucișare cu valoarea 0.2, fiecare generație având un număr de 100 de cromozomi (mărimea populației).

Pentru Simmulated Annealing au fost realizate 1000 de iterații per test. Temperatura a fost scăzută în mod multiplicativ conform formulei $T = T \cdot 0.99$, unde T reprezintă valoarea temperaturii pentru iterația curentă.

Mai jos regăsim tabele ale căror valori reprezintă o sinteză a rezultatelor obținute în urma testărilor efetuate pentru fiecare instanță.

Instanța br17	Dimensiune	Soluția optimă cunoscută	Cea mai bună valoare	Cea mai depărtată valoare	Media	Deviația standard
AG	17	39	39	58	43.15	5.84
SA			39	67	45.83	8.53

Instanța ftv33	Dimensiune	Soluția optimă cunoscută	Cea mai bună valoare	Cea mai depărtată valoare	Media	Deviația standard
AG	34	1286	2091	2396	2275.2	84.5
SA			2022	2876	2502.5	293.81

Instanța ftv38	Dimensiune	Soluția optimă cunoscută	Cea mai bună valoare	Cea mai depărtată valoare	Media	Deviația standard
AG	39	1530	2101	3034	2833.5	125.1
SA			2218	3309	2845.9	243.1

Instanța p43	Dimensiune	Soluția optimă cunoscută	Cea mai bună valoare	Cea mai depărtată valoare	Media	Deviația standard
AG	43	5620	5839	11694	7937.9	2310.8
SA			5888	11861	8291.3	2391.5

Instanța ftv47	Dimensiune	Soluția optimă cunoscută	Cea mai bună valoare	Cea mai depărtată valoare	Media	Deviația standard
AG	48	1776	3046	3634	3399.1	121.36
SA			3195	4190	3596.5	268.21

Instanța ft53	Dimensiune	Soluția optimă cunoscută	Cea mai bună valoare	Cea mai depărtată valoare	Media	Deviația standard
AG	53	6905	12052	13139	12565.6	322.2
SA			12156	13776	12665.6	380.1

Instanța ftv64	Dimensiune	Soluția optimă cunoscută	Cea mai bună valoare	Cea mai depărtată valoare	Media	Deviația standard
AG	65	1839	5036	5987	5284.2	232.82
SA			5044	5953	5510.9	251.8

Instanța ftv70	Dimensiune	Soluția optimă cunoscută	Cea mai bună valoare	Cea mai depărtată valoare	Media	Deviația standard
AG	71	1950	5075	6148	5438.6	290.71
SA			5034	6657	5583.9	305.87

Instanța rbg323	Dimensiune	Soluția optimă cunoscută	Cea mai bună valoare	Cea mai depărtată valoare	Media	Deviația standard
AG	323	1326	2045	2945	2830.35	202.2
SA			2245	3240	2895.05	214.71

Instanța rbg403	Dimensiune	Soluția optimă cunoscută	Cea mai bună valoare	Cea mai depărtată valoare	Media	Deviația standard
AG	403	2465	2871	3561	3252.57	213.5
SA			2893	3871	3311.9	254.71

4 Comparații

Din punct de vedere al timpului de execuție, Simmulated Annealing generează rezultate mai rapid, deoarece în comparație cu Algoritmul Genetic acesta nu este supus unei multitudini de procese precum operațiile genetice.

Pe testele ce conțin un cardinal mic de orașe se observă o performanță mai bună a Algoritmului Genetic. Din deviația standard reiese faptul că rezultatele sunt mai apropiate între ele pe parcursul evaluărilor. Pe măsura ce numărul de orașe crește ambii algoritmi generează rezultate destul de similare, sesizând că funcționează aproximativ la fel.

Instanțele au un impact major în rezultatul generat de cei doi algoritmi și datorită modului de dispunere al distanțelor dintre orașe. Aceste instanțe variază mult între ele în ceea ce privește dispunerea distanțelor. Astfel că la unele teste se observă o deviație standard mărită tocmai din cauza acestor dispuneri ale distanțelor.

Considerând că există o singura soluție optimă pentru vizitarea celor n orașe indiferent de instanță putem edifica motivul pentru care Simmulated Annealing generează rezultate general valabile. Fiind un algoritm exploatator și având în vedere faptul că soluția ar fi unimodală, este justificată performanța sa în cadrul testării ATSP.

Cu toate acestea, prin natura problemei aflate în discuție, este trivial faptul că pot exista cel puțin două rute de aceeași distanță. Pentru determinarea unei soluții cât mai favorabile ne interesează atât o exploatare cât și o explorare cât mai eficentă, ce poate fi furnizată de algoritmul genetic, sumarizând astfel și performanța sa în raport cu Simulated Annealing.

5 Concluzii

Observăm cum putem rezolva probleme NP-complete prin intermediul algoritmilor genetici, respectiv Simulated Annealing. Se obțin rezultate bune într-un timp scurt comparativ cu alte metode de rezolvare a problemei precum backtracking sau algoritmi de tip greedy, pentru care gestionarea problemelor de ordin mare ar fi problematică.

Deoarece căutarea se efectuează pe baza unei populații, există mai multe soluții ce pot fi evaluate simultan. Așadar, prin intermediul algoritmilor studiați în această lucrare putem analiza mai multe variante ceea ce duce către optimizarea dorită și motivează optarea pentru algoritmii meta-euristici.

În cazul algoritmului Simulated Annealing am observat o îmbunătățire a rezultatelor prin scăderea mai lentă a temperaturii sau prin mărirea numărului de iterații din bucla interioară, bineînțeles aceste îmbunătățiri aduc cu ele o creștere a timpului de rulare. În ceea ce privește algoritmul genetic, se pot aduce îmbunătățiri prin adăugarea unor noi operatori adaptivi sau îmbunătățirea celor deja existenți precum operatorul de selecție sau operatorul de încrucișare.

References

[1] Algoritmi genetici - noțiuni si definitii
http://students.info.uaic.ro/~vladut.ungureanu/Algoritmi-genetici-ID.pdf
http://www.bel.utcluj.ro/dce/didactic/tice/15_AlgoritmiGenetici.pdf

[2] Algoritmi genetici noțiuni generale
https://en.wikipedia.org/wiki/Genetic_algorithm

https://ro.wikipedia.org/wiki/Algoritm_genetic

[3]Noțiuni algoritmi genetici pentru rezolvarea TSP
https://aip.scitation.org/doi/pdf/10.1063/1.5039131 https://towardsdatascience.com/genetic-algorithm-a-simple-and-intuitive-guide-51c04cc1f9ed

[4] Not, iuni specifice algoritmului genetic s, i simmulated annealing
http://students.info.uaic.ro/~vladut.ungureanu/Algoritmi-genetici-ID.pdf
https://profs.info.uaic.ro/~eugennc/teaching/ga/

[5] Librăria TSPLIB
http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/

[6] Not, iuni Latex
https://www.overleaf.com/learn/latex/tables
https://en.wikibooks.org/wiki/LaTeX/Algorithms