STUDIA UNIVERSITATIS MOLDAVIAE, 2017, nr.2(102)
Seria "Ştiinţe exacte şi economice" ISSN 1857-2073 ISSN online 2345-1033 p.37-42

CZU: 004.43:004.92

PROGRAMAREA APLICAŢIILOR GRAFICE PRIN IMPLEMENTAREA ALGORITMILOR INTELIGENŢI

Maria CRISTEI

Universitatea de Stat din Moldova

În prezent, programarea grafică ocupă un loc important în mediul de dezvoltare a societăţii şi în rezolvarea multor probleme real complexe. Elemente grafice pot fi create folosind majoritatea limbajelor de programare cunoscute în prezent. Unul dintre acestea este Java, care la ora actuală este unul dintre cele mai folosite limbaje de programare. În acest sens, în prezentul articol se descrie elaborarea unei aplicaţii grafice cu date reale, prin implementarea algoritmilor genetici în rezolvarea problemelor real complexe, şi anume: a problemei ,,Comis Voiajorul", ce oferă posibilităţi majore oricărui tip de utilizator dornic de a cunoaşte un traseu, sau distanţa acestuia, folosind harta geografică în timp real. Aplicaţia de concepţie proprie prin implementarea algoritmilor genetici este mult mai puternică în funcţionalităţile pe care le conţine faţă de alte soft-uri existente, acest lucru fiind demonstrat prin analiza comparativă efectuată în cercetare.

Cuvinte-cheie: *algoritmi genetici, aplicaţie informatică, interfaţă grafică.*

PROGRAMMING GRAPHIC APPLICATIONS BY IMPLEMENTING INTELLIGENT ALGORITHMS

Graphical programming currently occupies an important place in the development of society and solving many complex real problems. It is possible to create graphic elements using most programming languages currently known, one of which is Java, which currently is one of the most used programming languages. In this respect, the present article describes the development of a graphical application with real data, by implementing genetic algorithms to solve real complex problems, namely the „Travelling Salesman Problem", which offers great possibilities for every type of user who wants to know a route, or its distance, using the real-time geographic map. The application of own design through the implementation of genetic algorithms is much stronger in the functionality of what it contains in relation to other existing software, this being demonstrated by the comparative analysis carried out in the research.

Keywords: *genetic algorithms, computer application, graphical interface.*

Consideraţii generale privind algoritmii genetici. Algoritmii genetici sunt algoritmi inteligenţi, care se aplică tot mai des la soluţionarea diverselor probleme, în principal a problemelor de complexitate majoră şi se pot obţine rezultate deosebite. Algoritmii genetici sunt metode inspirate din genetică implementate la soluţionarea problemelor complexe de optimizare sau control, ce se bazează pe perfecţionarea unui set iniţial de soluţii. Aceşti algoritmi inteligenţi scot în evidenţă capacitatea omului de a gândi, domeniul de aplicare, posibilităţile de atingere a unor rezultate incredibile şi descoperirea unor lucruri cu adevărat valoroase şi utile. De aceea, studierea şi analiza acestor algoritmi inteligenţi a trezit un mare interes şi entuziasm pentru a descoperi posibilitatea obţinerii unui rezultat rapid şi eficient, prin implementarea lor în rezolvarea problemelor practice.

Actualmente există o mulţime de aplicaţii care uşurează modul de căutare a informaţiei, oferă posibilităţi de a obţine rapid şi efectiv careva date. Printre aceste produse software sunt şi aplicaţiile care oferă date reale despre poziţia geografică a unui oraş, adresa, distanţa dintre mai multe oraşe, ceea ce oferă unui utilizator posibilitatea de a avea acces la toate aceste date.

Astfel, după un studiu aprofundat asupra posibilităţilor oferite de algoritmii inteligenţi, ne-am *propus să elaborăm o aplicaţie informatică grafică, care poate oferi oricărui tip de utilizator date reale despre distanţa şi traseul dintre oraşele preferate selectate direct pe harta geografică care poate fi utilizată fără a avea conexiune la Internet*. Prin această aplicaţie se doreşte evidenţierea implementării algoritmilor inteligenţi în rezolvarea unor probleme complexe, prin reprezentarea rezultatelor într-un mod original, cu efecte vizuale grafice, cu date accesate real, simplu la exterior, dar complex în interior.

Realizarea aplicaţiei de concepţie proprie presupune implementarea algoritmilor genetici în rezolvarea uneia dintre cele mai cunoscute probleme de optimizare, problema NP-complexă – „problema Comis Voiajorului" (PCV), care nu poate fi rezolvată într-un timp rezonabil decât prin folosirea unor algoritmi nedeterminişti, cu ajutorul cărora se găsesc soluţii foarte bune în perioade de timp foarte scurte.

Problema „Comis Voiajorului" (PCV), matematic, 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 aleator al grafului şi trece exact o dată prin toate celelalte noduri şi care se întoarce la nodul iniţial, cu condiţia ca ciclul să aibă un cost minim. Costul unui ciclu este definit ca suma tuturor costurilor ataşate laturilor ciclului [1,2].

Într-un limbaj non-matematic însă, problema constă în găsirea celui mai scurt drum care trece prin N oraşe, astfel încât acest drum să nu treacă de două ori prin unul şi acelaşi oraş şi să ajungă în oraşul din care a pornit.

Astfel, rezolvarea acestei probleme cu implementarea în aplicaţia de concepţie proprie presupune definirea unui careva număr de oraşe selectate manual direct pe harta geografică, iar în urma unui şir de iteraţii, utilizându-se o varietate largă de operatori genetici, aplicaţia va afişa grafic drumul optimal din oraşul iniţial spre cel final, cu o singură trecere prin acestea, şi revenind în oraşul de plecare (iniţial fiind şi cel final).

Pentru elaborarea informatică a aplicaţiei au fost înaintate un set de cerinţe faţă de aceasta:

- implementarea tuturor operatorilor genetici (de selecţie, crossover şi mutaţie);
- posibilitatea de a seta iniţial un număr arbitrar de indivizi ai populaţiei;
- posibilitatea de a indica un număr de iteraţii pe care algoritmul genetic să-l parcurgă;
- posibilitatea de a indica rata de mutaţie înainte de a începe algoritmul;
- posibilitatea de a indica manual poziţia oraşelor;
- posibilitatea de calcula şi de a afişa distanţa dintre oraşe;
- posibilitatea de a elimina toate oraşele din algoritm pentru a poziţiona altele;
- vizualizarea oraşelor în interfaţă grafică;
- vizualizarea celei mei bune soluţii la fiecare iteraţie a algoritmului.

Realizarea algoritmului genetic pentru rezolvarea PCV. Pentru a realiza un algoritm genetic pentru rezolvarea PCV este nevoie întâi de toate de a formula o codificare robustă a soluţiilor problemei. Această codificare este de dorit să fie realizată astfel, încât orice instanţă posibilă a acestei codificări să reprezinte o soluţie admisibilă a problemei. Mai mult ca atât, codificarea soluţiilor trebuie realizată astfel, încât *funcţia de fitness* să calculeze rapid calitatea individului, la fel şi operatorii algoritmului genetic să opereze rapid cu această codificare.

Pentru problema cercetată, cea mai intuitivă şi simplu de realizat codificare este de a reprezenta drumul ce constituie soluţia problemei, sub formă de şir de numere întregi, unde fiecare număr întreg reprezintă numărul oraşului, de la 1 la N. În aşa formă, oricare două oraşe se consideră adiacente, atunci când numerele acestora sunt adiacente în acest şir sau sunt pe extremităţi. Altfel formulat, orice permutare a N numere reprezintă o codificare a unei soluţii. Scopul algoritmului genetic este de a găsi aşa o permutare, care constituite drumul cel mai minim între oraşe, prin utilizarea metodei indicate de codificare a soluţiilor PCV în cromosoma indivizilor şi prin aplicarea operatorilor de selecţie, de încrucişare (crossover) şi de mutaţie pentru rezolvarea acestei probleme care nu produc duplicate în informaţia genetică a individului şi nu reprezintă o soluţie neadmisibilă a problemei.

Structura implementării algoritmului genetic în rezolvarea PVC arată în felul următor:

- Se creează o populaţie de tururi, definită sub forma unui vector de dimensiunea ***n***;
- Fiecare tur din populaţie reprezintă un vector de dimensiunea ***m***, în care sunt păstrate trecerile prin toate oraşele;
- Se evaluează populaţia iniţială, prin aplicarea operatorilor (selecţie, încrucişare şi mutaţie) definiţi de algoritmii genetici;
- Se aplică operatorul de selecţie asupra populaţiei curente pentru selectarea a doi părinţi, care sunt de tip tur;
- Se aplică operatorul de încrucişare asupra părinţilor selectaţi pentru a obţine un nou urmaş;
- Se aplică operatorul de mutaţie pentru a modifica gena urmaşului;
- Se salvează noul urmaş într-o nouă generaţie;
- Se evaluează noua generaţie prin calcularea valorii *fitness*-ului (calităţii) fiecărui individ;
- Se selectează individul cu cel mai bun *fitness (cel mai puternic)*, care trebuie sa fie sub forma unui tur;
- Se afişează rezultatul final, prin reprezentarea turului selectat.

Rezolvarea PCV prin investigarea tuturor soluţiilor posibile la un număr mic de oraşe este costisitoare în timp, iar la un număr mare de oraşe în genere este imposibilă, ceea ce nu este valabil pentru algoritmii genetici. Totuşi, algoritmii genetici fac un număr foarte mare de operaţii pentru a îmbunătăţi fiecare populaţie nouă.

Iar atunci când numărul de indivizi în populaţie este mare şi numărul de oraşe la fel este mare, algoritmul genetic va trebui să lucreze mult pentru a ajunge la o soluţie considerată optimală. Din acest motiv, selectarea unei tehnologii informatice bune sporeşte viteza de lucru a algoritmului genetic şi la configuraţii complexe ale algoritmului această diferenţă de timp nu va fi considerabilă.

Selectarea tehnologiilor informatice pentru realizarea aplicaţiei. Metodologia folosită în realizarea informatică a aplicaţiei elaborate a fost mixtă, bazându-se pe tehnologii software de ultimă oră. Ca limbaj de programare a fost ales Java, deoarece, pe de o parte, suportă programarea orientată pe obiect, ceea ce coincide cu conceptele în algoritmii genetici (individ, populaţie, operator) şi deci aceste concepte reprezintă întocmai nişte obiecte, iar, pe de altă parte, aplicaţiile compilate din limbajul Java sunt executate rapid. Această tehnologie inovatoare s-a remarcat prin simplitate, portabilitate şi robusteţe şi a început să fie utilizată pentru producerea şi dezvoltarea unui număr mare de aplicaţii software, sub diverse forme, aplicaţii grafice interactive atât bidimensionale, cât şi multidimensionale, implementate în diferite domenii.

Mediul de dezvoltare pentru crearea aplicaţiei a fost ales „*NeatBeans*", care este un produs *open source*, complet gratuit pentru uz comercial şi necomercial, şi susţinut de către compania ORACLE.

Decizia noastră a avut ca prim argument utilizarea instrumentelor ce permit implementarea elementelor interfeţei grafice cu utilizatorul, fără necesitate de mari resurse. Din acest punct de vedere, ca *instrumente de dezvoltare* a interfeţei grafice, de desenare şi vizualizare grafică a traseelor performante oferite de algoritm, vom folosi JavaFX, care este cea mai nouă implementare a platformei de dezvoltare pentru interfeţe. JavaFX este un nou motor grafic cu o serie de componente noi de interfaţă, precum grafice, tabele, meniuri şi panouri. Cu ajutorul JavaFX putem crea forme 2D/3D, grafice, putem aplica asupra unui component transformări, stiluri, putem adăuga efecte vizuale, cum ar fi umbre, mecanisme de estompare, reglarea culorilor.

Implementarea unei aplicaţii JavaFX implică proiectarea şi dezvoltarea unui graf de scene cu o structură ierarhică de noduri ce conţin elemente vizuale ale interfeţei grafice cu utilizatorul, care poate trata diferite evenimente legate de acestea şi care poate fi redată. În graful de scene un element reprezintă un nod ce este identificat în mod unic, caracterizat printr-o clasă de stil şi un volum ce-l delimitează. Fiecare nod are un părinte, dar poate avea unul sau mai mulţi copii; este posibil să nu aibă niciunul. Pentru asemenea element poate fi definit un şir de transformări.

Arborele interfeţelor aplicaţiei elaborate este reprezentat grafic în Figura 1, din care se vede că aplicaţia conţine trei interfeţe. Prima interfaţă, intitulată SceneFirst, este creată dintr-un panel pe care sunt incluse 3 componente: text, câmp de tip TextField şi un buton. A doua interfaţă este SecondScene împărţită în două componentei: cea din stânga conţine harta, cea din centru conţine butoanele şi textul rezultat. Iar a treia scenă ThirdScene conţine detalii ale executării algoritmilor inteligenţi.

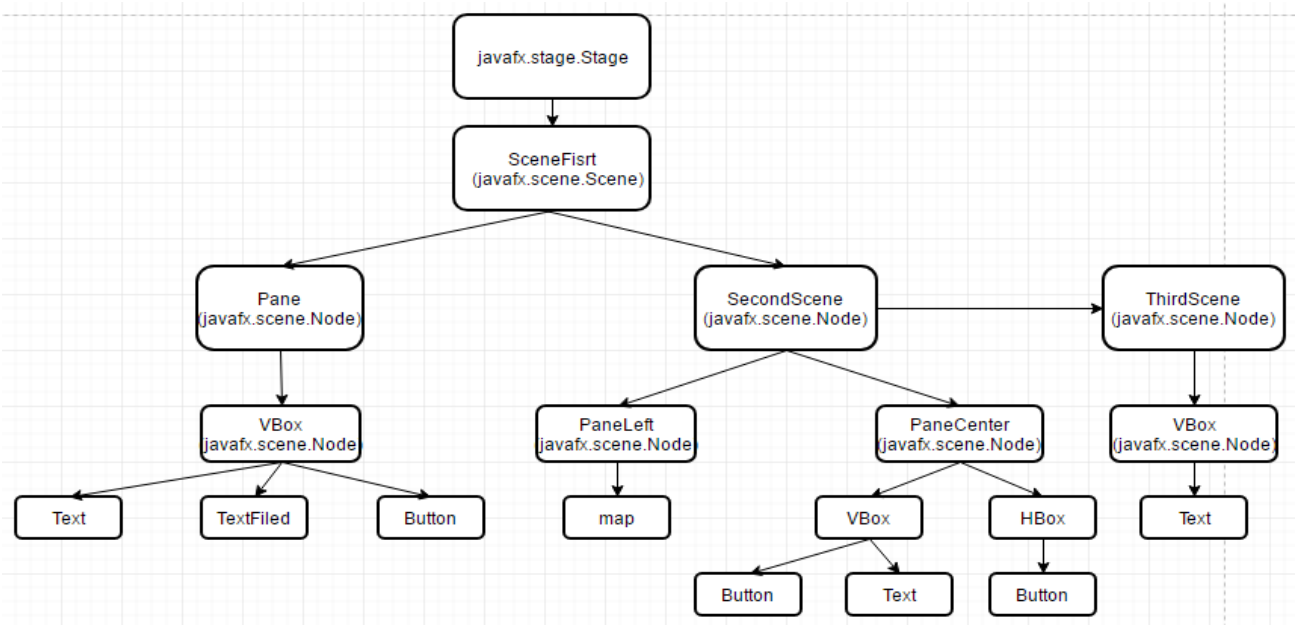

Fig.1. Arborele interfeţelor.

STUDIA UNIVERSITATIS MOLDAVIAE, 2017, nr.2(102)
Seria "Ştiinţe exacte şi economice" ISSN 1857-2073 ISSN online 2345-1033 p.37-42

Rezultate finale obţinute

Configuraţia iniţială a algoritmului genetic se efectuează în fereastra aplicaţiei din Figura 2 şi constituie indicarea mărimii populaţiei, indicarea numărului de iteraţii pe care algoritmul trebuie să-l îndeplinească.

Fig.2. Configuraţia iniţială a algoritmului genetic.

Mărimea populaţiei se indică în prima căsuţă de introducere a textului cu inscripţia „*Dimensiunea Populaţiei*", iar numărul de iteraţii ce va rula algoritmul se indică în a doua căsuţă cu textul iniţial „*Numărul maxim de Generaţii*". După apăsarea butonului „*Trimite*", datele introduse de către utilizator sunt salvate în două variabile şi preluate în următoarea interfaţă, pentru crearea obiectului de tip populaţie cu dimensiunea egală cu numărul introdus de utilizator. În cazul neintroducerii datelor, utilizatorul va primi un mesaj de eroare şi nu-i va permite să continue.

Următoarea interfaţă deschisă (Fig.3) conţine: ***harta în timp real***, extrasă prin intermediul *google maps* cu toate oraşele; ***butoanele***, ca elemente grafice ce oferă susţinere la utilizarea acesteia şi posibilitatea de vizualizare a rezultatului final. Tot ce trebuie să facă utilizatorul este să activeze butonul „*Alege oraşele*" pentru a putea selecta oraşele dorite. Aplicaţia oferă utilizatorului posibilitatea de a selecta manual oraşele dorite şi evidenţierea acestora cu puncte de altă culoare plasate direct pe hartă, după cum este reprezentat în figura ce urmează:

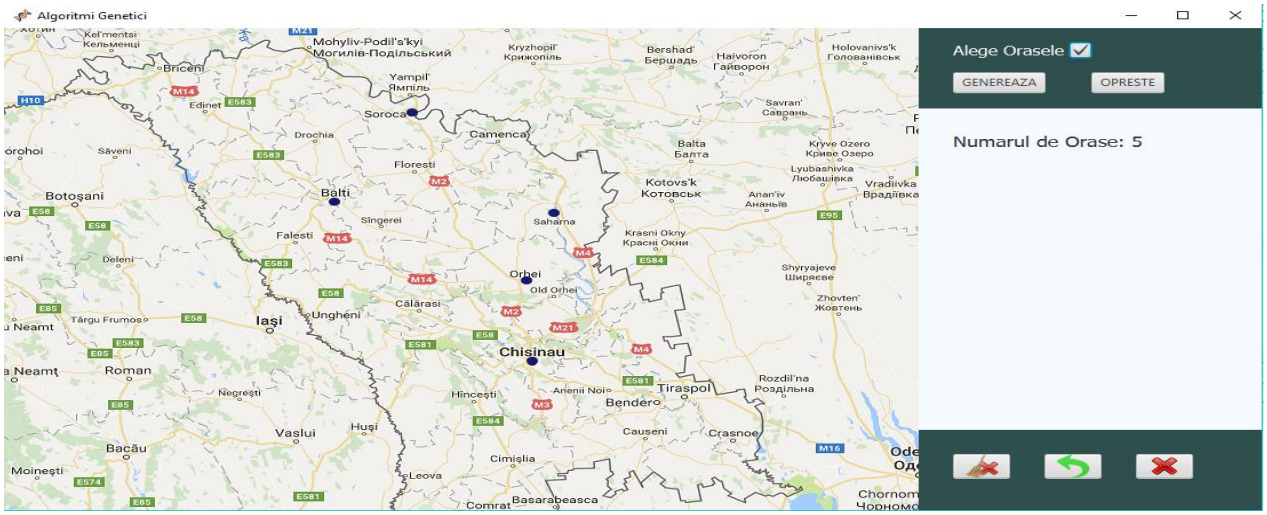

Fig.3. Selectarea oraşelor direct pe harta geografică.

La tastarea butonului „*Generează*" utilizatorul va vedea schimbările traseului până la găsirea celui mai optim drum. La finalizarea algoritmului, lungimea drumului care trece prin toate oraşele selectate în problemă poate fi văzut pe harta aplicaţiei (Fig.4).

STUDIA UNIVERSITATIS MOLDAVIAE, 2017, nr.2(102)
Seria "Ştiinţe exacte şi economice" ISSN 1857-2073 ISSN online 2345-1033 p.37-42

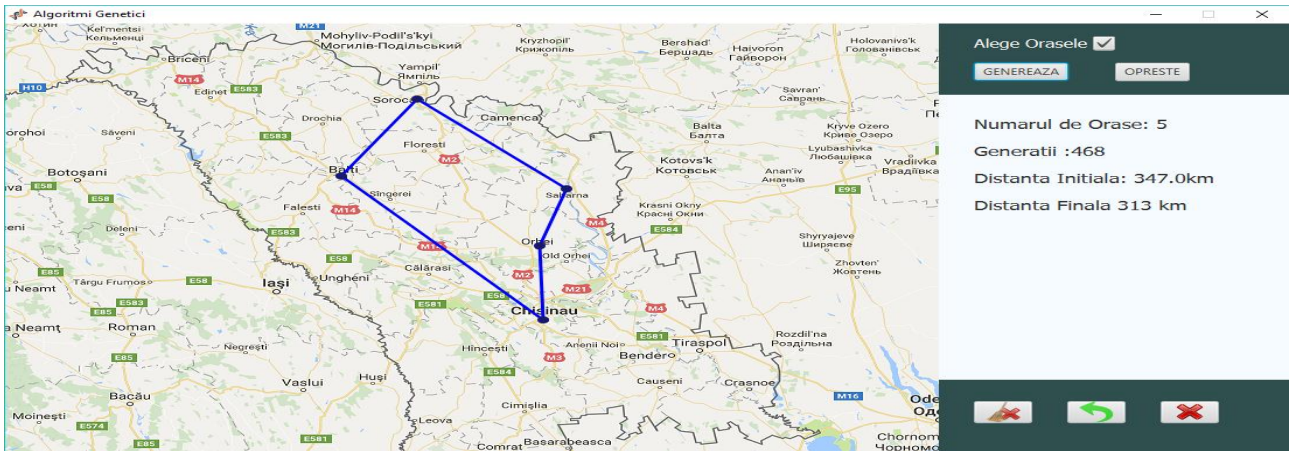

Fig.4. Rezultatul final.

După cum se observă din Figura 4, aplicaţia ne returnează date suplimentare, cum ar fi: numărul de oraşe alese, distanţa iniţială a traseului fără schimbări, distanţa finală care este cea mai bună şi numărul de generaţii necesare pentru găsirea celui mai bun traseu. Pe lângă aceste date, aplicaţia conţine şi alte funcţionalităţi, precum: ***butonul*** *„Opreşte*", necesar pentru întreruperea algoritmului în cazul în care numărul de generaţii este destul de mare, iar traseul nu se modifică; ***butoanele din partea de jos*** a aplicaţiei cu care se poate curăţa fereastra, reveni la fereastra de pornire sau ieşire.

Analiză comparativă. Există pe piaţă *soft*-uri asemănătoare ce posedă funcţionalităţi de calcul al distanţei dintre oraşe. Dintre acestea face parte şi *Google Maps, distancecalculator.net* sau *distancefromto.net*. Pe acest fundal, după o analiză comparativă dintre *soft*-urile enumerate şi aplicaţia de concepţie proprie au fost identificate careva deosebiri ce scot în evidenţă unicitatea funcţionalităţilor oferite de aplicaţia elaborată. Rezultatele analizei comparative au demonstrat că aplicaţia elaborată este mult mai facilă şi lucrează mult mai eficient, datorită faptului că traseul parcurgerii oraşelor se schimbă în conformitate cu un nou oraş adăugat. Pentru ca în rezultat utilizatorul să obţină drumul de lungime minimă, nu i se limitează numărul de oraşe selectate şi este simplu în utilizare. Rezultatele obţinute în urma studiului realizat sunt prezentate în următorul tabel.

Tabel

Comparaţie între aplicaţiile existente şi aplicaţia elaborată

	Aplicaţia proprie	**Google Maps**	**Distance Calculator**
Calcularea distanţei dintre mai mult de 2 puncte	+	+	+
Alegerea punctelor direct pe hartă	+	+	-
Adăugarea unor noi puncte, după care vizualizarea rezultatului	+	-	+
Modificarea traseului în dependenţă de punctele adăugate	+	-	-
Vizualizarea traseului	+	+	+
Disponibilitate offline	-	-	-
Simplitate la alegerea punctelor de destinaţie	+	-	+
Distanţa obţinută în km	+	+	+
Interfaţă simplă şi uşor de utilizat	+	+	+

Concluzii

Abilitatea algoritmului şi aplicarea cu succes în problemele complexe din lumea reală întăreşte tot mai mult concluzia că algoritmii genetici sunt o tehnică puternică şi robustă de optimizare. Puterea algoritmilor genetici constă în uşurinţa cu care sunt implementaţi, oferind posibilităţi desăvârşite în căutarea unei soluţii

optime a unei probleme, deschid noi orizonturi de rezolvare şi aplicare în lumea reală. Anume aceşti algoritmi inteligenţi au deschis porţile spre noi idei şi realizări unice şi utile.

Problemele practice destul de complexe rezolvate pe baza algoritmilor inteligenţi, discuţiile şi opiniile persoanelor interesate de domeniul dat, care au descoperit lucruri uimitoare, ne-au trezit interesul de a aplica aceşti algoritmi în domeniul graficii pe calculator şi de a crea o aplicaţie grafică practică utilizând tehnologii informatice de ultimă oră. Prin aplicaţia elaborată s-a îndeplinit cu succes evidenţierea beneficiului oferit de aceşti algoritmi la găsirea unor soluţii optime şi reprezentarea într-un mod grafic a datelor obţinute în urma implementării acestuia. Aplicaţia descrie un model facil de obţinere a unor date reale, şi anume: distanţa de la un oraş la altul cu condiţia de a trece o singură dată printr-un oraş şi cu posibilitatea de a vizualiza traseul. Datele obţinute sunt preluate în timp real, calculate corect şi verificate prin comparaţie cu motorul *google maps*.

Referinţe:

1. NEAGOE, V. *Algoritmi Genetici*. http://www.victorneagoe.com/university/prai/lab5a.pdf
2. MIHAESCU, C. *Problema Comis-Voiajorului.* Craiova, Dolj, România. 19 p.

Prezentat la 14.06.2017