Fakultet elektrotehnike i računarstva
Zavod za elektroniku, mikroelektroniku, računalne i inteligentne sustave

Primjena optimizacije kolonijom mrava na rješavanje problema trgovačkog putnika

Seminarski rad

Predmet: Algoritmi u sustavima upravljanja
Mentor: Doc. dr. sc. Marin Golub, Prof. dr. sc. Leo Budin

Hrvoje Marković

Zagreb, svibanj 2006.

Sadržaj

1. Uvod

Autor se prvi put susreo s pojmom optimizacije kolonijom mrava dok je surađivao s kolegama s Fakulteta prometnih znanosti koji rade na projektu CroGrid. Kolege se bave problemom usmjeravanja vozila (engl. Vehicle Routing Problem) i vrlo su im zanimljivi algoritmi koji omogućuju paralelizaciju. Problem usmjeravanja vozila je zapravo poopćeni problem trgovačkog putnika u kojem umjesto jednog entiteta (trgovačkog putnika), veći broj entiteta (vozila) obavlja obilazak predefiniranog broja odredišta. Cilj je, poput onoga kod problema trgovačkog putnika, promaći put kojim će se obići sva odredišta uz najmanji trošak (npr. najkraći pređeni put). Kako je problem trgovačkog putnika jednostavniji problem, on je odabran kao primjer na kojem će biti prikazana optimizacija kolonijom mrava.

Problem trgovačkog putnika je jedan od najistraženijih i najpoznatijih optimizacijskih problema. Kako pri rješavanju toga problema dolazi do kombinatoričke eksplozije, razvijen je velik broj heurističkih metoda za rješavanje. Često se koriste genetski algoritmi, ali u novije vrijeme i algoritmi teorije rojeva (engl. swarm intelligence) koji pogoduju različitim oblicima paralelizacije (npr. grid computing).

U drugom će poglavlju biti opisana optimizacija kolonijom mrava kao odabrana metoda teorije rojeva, dok će u trećem poglavlju biti definiran problem trgovačkog putnika. U četvrtom će poglavlju biti prikazana programska implementacija odabranog algoritma. U petom će poglavlju biti uspoređeni optimizacija kolonijom mrava i ostali heuristički algoritmi za rješavanje problema trgovačkog putnika. Šesto poglavlje donosi zaključak.

2. Optimizacija kolonijom mrava (ACO)

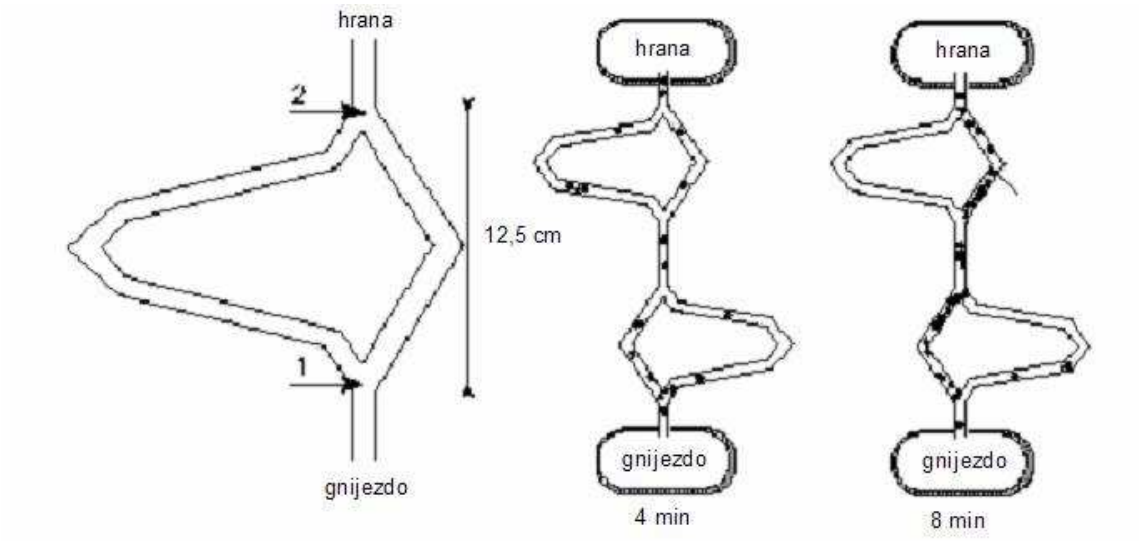
Optimizaciju kolonijom mrava (engl. Ant Colony Optimisation) je prvi uveo Marco Dorigo [1]. Kolonija mrava predstavlja multi-agentski sustav gdje je ponašanje pojedinog (umjetnog) mrava nadahnuto ponašanjem mrava u prirodi.

Primjene obuhvaćaju klasične NP – teške probleme i sežu do usmjeravanja u računalnim i telekomunikacijskim mrežama te dubinske analize podataka (engl. data mining).

2.1. Ponašanje mrava u prirodi

Inspiraciju za optimizaciju kolonijom mrava je dao eksperiment koju je proveo Goss 1989. godine s argentinskim mravima (lat. *Iridomyrmex humilis*).

Hrana je dana mravima u areni s dva puta različitih duljina kao što je to prikazano na slici 1. Na jednom se kraju nalazi mravlje gnijezdo, a na drugom izvor hrane. U svakom smjeru mravi moraju odabrati put kojim će ići.



Slika 1. Eksperiment s argentinskim mravima

Eksperiment pokazuje kako nakon kraće prijelazne faze većina mrava koristi kraći put. Dodatno, pokazano je kako s razlikom u duljini između dva puta raste

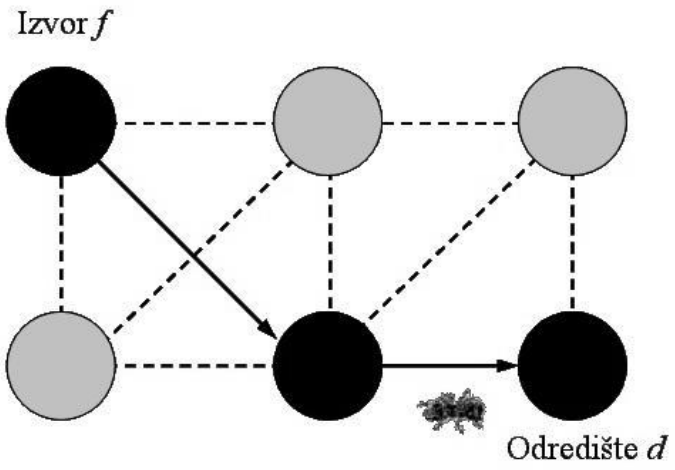
vjerojatnost odabira kraćeg. Takvo se ponašanje mrava objašnjava stigmerijom koja omogućuje povratnu vezu ovisnu o različitim duljinama putova.

Pojam stigmerija označava indirektan način komunikacije posredovanjem lokalnim modifikacijama okoliša. Ona se ostvaruje ostavljanjem kemijskih spojeva koji se nazivaju feromoni.

U točki odabira, mrav donosi vjerojatnosnu odluku kojim putem ići temeljenu na količini feromona na tlu. U sljedećoj iteraciji odabrani put postaje vjerojatniji. Na taj je način ostvarena povratna veza. Kako se na kraćem putu feromoni ostavljaju češće, mravi kroz niz iteracija izabiru kraći put.

2.2. Pronalazak minimalnog puta u grafu

Analogno prirodnim mravima, zadatak je umjetnih mrava pronaći minimalni put između čvorova grafa. Neka je $G = (N, A)$ povezani graf, gdje je N skup svih čvorova, a A skup svih lukova. Broj čvorova je $|N| = n$. Rješenje problema predstavlja put na grafu koji povezuje izvorišni čvor f s odredišnim d, čija je duljina određena brojem lukova na putu. Slika 2 prikazuje minimalni put na grafu.



Slika 2. Pronalazak minimalnog puta na grafu

Sa svakim je lukom (i, j) grafa G povezana varijabla τ_{ij} koja predstavlja umjetni trag feromona. Tragove feromona čitaju i pišu mravi. U svakom čvoru grafa

dolazi do stohastičke odluke koji je čvor sljedeći. k-ti mrav lociran u čvoru i koristi trag feromona τ_{ij} za izračun vjerojatnosti u koji čvor $j \in N_i$ treba otići. N_i je skup susjeda čvora i. Vjerojatnost odabira čvora j je dana izrazom:

$$p_{ij}^{k} = \begin{cases} \dfrac{\tau_{ij}}{\sum_{j \in N_i} \tau_{ij}} & , j \in N_i \\ 0 & , j \notin N_i \end{cases}$$

Dok traži rješenje mrav ostavlja trag feromona na pripadnom luku $\tau_{ij}(t) \leftarrow \tau_{ij}(t-1) + \Delta\tau$. Međutim, tako definirano rješenje može dovesti do konvergencije prema suboptimalnom rješenju tako da se uvodi mehanizam isparavanja. U svakoj se iteraciji eksponencijalno smanjuje količina feromona prema izrazu:

$$\tau \leftarrow (1-\rho)\tau \quad , \rho \in [0,1)$$

Ponašanje algoritma ovisi o broju susjeda svakog čvora. Ukoliko čvorovi grafa imaju više od dva susjeda, algoritam gubi na stabilnosti. Odnosno, postaje kritičan izbor parametara.

Algoritam je moguće poboljšati sagledavanjem svojstava kolonije s jedne strane te sagledavanjem svojstava pojedinog mrava.

2.2.1. Svojstva kolonije

Algoritam je moguće poboljšati korištenjem heuristike. Moguće je heuristički mijenjati količinu ostavljenog feromona tako da je ona proporcionalna kvaliteti rješenja koje se generira. Za pronalazak najkraćeg puta koji obilazi svaki čvor samo jednom, mravi moraju imati barem ograničenu memoriju.

Svojstva kolonije kao cjeline:

1) Dobra rješenja mogu proizaći samo kao rezultat kolektivne interakcije;
2) Svaki mrav koristi samo privatne informacije i lokalne informacije čvora kojeg posjećuje;
3) Mravi komuniciraju samo indirektno;

4) Mravi se sami po sebi ne adaptiraju, naprotiv, mijenjaju način prezentacije problema i percepcije drugih mrava.

2.2.2. Svojstva mrava

Svojstva su mrava kao individue:

1) Mrav traži najisplativije rješenje;
2) Svaki mrav ima memoriju M_k za pohranu informacija o putu i može ju koristiti:
    a. za generiranje smislenog rješenja,
    b. za evaluaciju pronađenog rješenja
    c. za povratak po istom putu;
3) Mrav k može se pomaknuti u bilo koji čvor j u susjedstvu N_i^k;
4) Mravu k može se dodijeliti početno stanje i jedan ili više terminirajućih uvjeta;
5) Mrav stvara rješenje inkrementalno i konstrukcija završava kada je zadovoljen jedan od uvjeta zaustavljanja;
6) Odabir sljedećeg čvora temelji se na vjerojatnosti;
7) Vjerojatnost pojedine odluke mrava temelji se na:
    a. vrijednostima pohranjenim u strukturi čvora $A_{ij} = \lfloor a_{ij} \rfloor$ (tablica ruta) koja se sastoji od kombinacije lokalnih tragova feromona i vrijednosti heurističke funkcije,
    b. privatnoj memoriji mrava,
    c. ograničenjima problema;
8) Pomicanjem iz čvora i u čvor j mrav ažurira trag feromona;
9) Kada je izgradio rješenje mrav se može vratiti istim putem i ažurirati trag feromona na povratku;
10) Kada je izgradio rješenje i vratio se na izvorište, mrav umire i oslobađa resurse.

2.3. Primjene optimizacije kolonijom mrava

Primjene optimizacije kolonijom mrava obuhvaćaju NP-teške probleme. NP-teški problemi su oni optimizacijski problemi kod kojih se složenost ne može izraziti kao polinomska ovisnost. Npr. kod pronalaska minimalnog puta u grafu, graf je eksponencijalno ovisan o dimenziji problema. U ovom slučaju mravi koriste puno manji graf koji je sastavljen od segmenata problema. Za rješenje se nakon mrava može koristiti optimizator za specifičan problem kako bi pronašao lokalni minimum.

Optimizaciju kolonijom mrava je moguće koristiti i kod problema najkraćeg puta gdje se graf dinamički mijenja za vrijeme optimizacije. Fizički graf može biti statičan, ali se mogu mijenjati svojstva (cijene spojeva). To je čest slučaj kod rješavanja npr. mrežnih problema. Kolonija je mrava prigodna i za rješavanje različitih problema koji nastaju kod upotrebe distribuirane računalne arhitekture.

Dodatno, kako je priroda mrava intrinsično distribuirana, ova metoda ima veliku efektivnost kada se koristi u obliku paralelnog i/ili mrežnog procesiranja.

3. Problem trgovačkog putnika (TSP)

Problem trgovačkog putnika (engl. Travelling Salesman Problem) predstavlja prvi problem koji je optimiziran kolonijom mrava (Dorigo 1991./92./96.). On je definiran:

Neka je C skup gradova, a L skup veza koje povezuju elemente od C. S $J_{c_i c_j}$ označena je cijena (duljina) između c_i i c_j, odnosno, duljina između gradova i i j. Problem trgovačkog putnika je optimizacijski problem pronalaska minimalnog Hamiltonova puta na grafu $G = (C, L)$. Hamiltonov je put zatvorena tura Ψ koja obilazi samo jednom svih N^C gradova. Njegova je duljina suma svih cijena $J_{c_i c_j}$ čvorova od kojih se sastoji. Dodatno, udaljenosti ne moraju biti simetrične.

3.1. Rješavanje TSP-a pomoću ACO

Svaki se umjetni mrav, slučajnim odabirom, smješta u neki od gradova. Tijekom svake iteracije mrav odbire sljedeći grad koji će posjetiti. Ovaj se odabir vrši pomoću sljedećeg izraza.

Mrav smješten u gradu i odlazi u grad j odabran među još neposjećenim gradovima prema vjerojatnosti

$$p_k(i,j) = \begin{cases} \dfrac{\tau_{ij}^{\alpha} * d_{ij}^{\beta}}{\sum_{g \in J_k(i)} \tau_{ig}^{\alpha} * d_{ig}^{\beta}} \\ 0 \end{cases}$$

gdje je:

$p_k(i,j)$ vjerojatnost da će mrav k iz grada i otići grad j,

$g \in J_k(i)$ skup gradova koje mrav k još nije obišao,

α relativna važnost traga feromona,

β relativna važnost udaljenosti među gradovima.

Vjerojatnost da će grad biti odabran je funkcija udaljenosti između gradova i količine feromona na putu. Pomoću parametara α i β je također moguće odrediti koji od ova dva faktora ima veću težinu. Jednom kada je ruta završena, odnosno

kada mrav posjeti svaki grad točno jednom, vrši se isparavanje feromona i svaki mrav ostavlja feromone po cijeloj ruti. Količina je feromona dana izrazom:

$$\tau(i,j) = p * \tau(i,j) + \sum_{k=1}^{m} \Delta\tau_k(i,j)$$

gdje vrijedi:

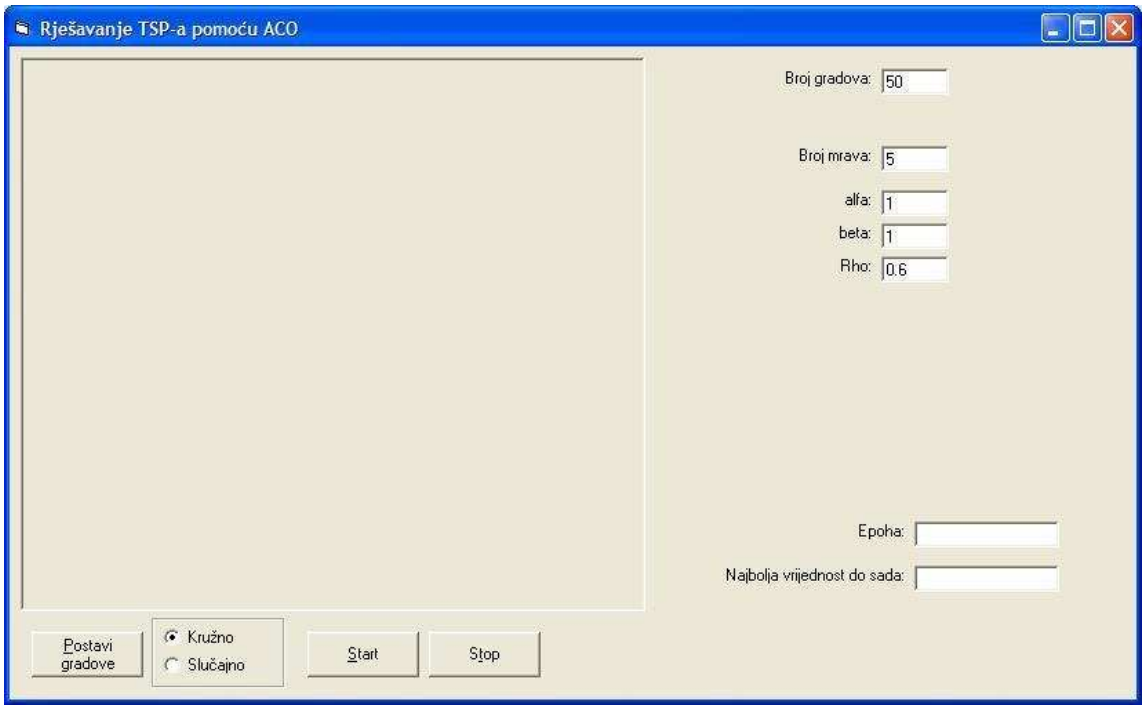
$$\Delta\tau_k = \begin{cases} \frac{1}{L_k}, & (i,j) \in ruta \\ 0, & (i,j) \notin ruta \end{cases}$$

$p * \tau(i,j)$: Umnožak određuje isparavanje feromona. p se naziva konstanta isparavanja. Njezina vrijednost može biti između 1 i 0. Kod nižih vrijednosti feromoni isparavaju brže nego kod viših vrijednosti konstante isparavanja.

$\sum_{k=1}^{m} \Delta\tau_k(i,j)$: Količina je feromona koju ostavlja mrav k definirana s duljinom rute koju je prešao L_k. Intuitivno, kraće će rute imati veće količine feromona.

4. Programska implementacija

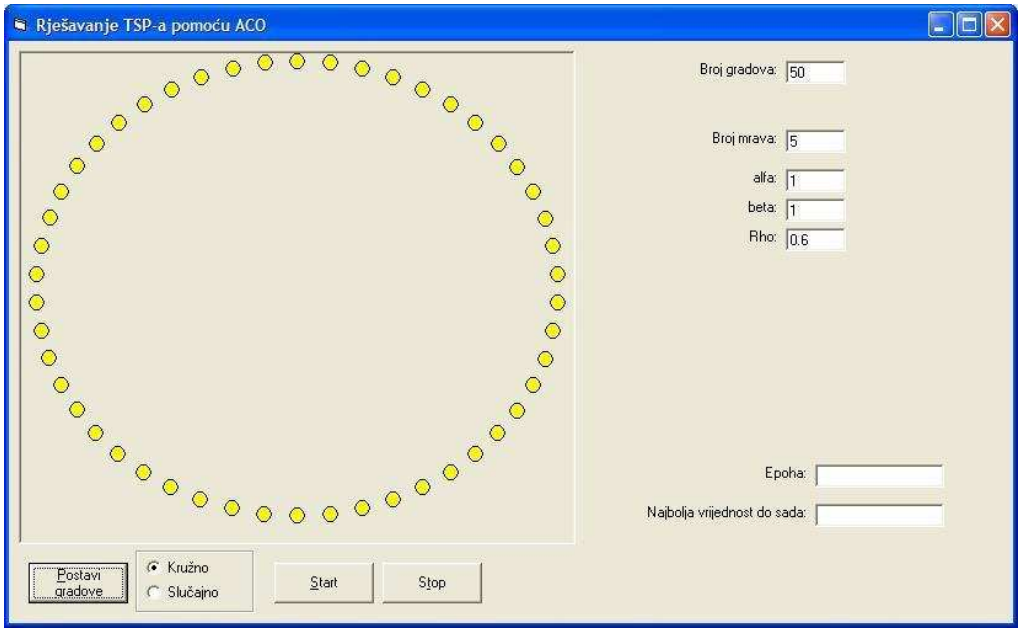
Algoritam opisan u poglavlju 3.1 implementiran je u programskom jeziku Visual Basic pomoću razvojne okoline Visual Studio 6.0. Visual Basic je odabran zbog toga što omogućuje relativno brz razvoj programske podrške uz vrlo jednostavnu izgradnju funkcionalnog sučelja. Program se sastoji od jedne forme na kojoj je moguće definirati parametre problema trgovačkog putnika i parametre kolonije mrava. Formu prikazuje slika 3.



Slika 3. Sučelje programa za rješavanje problema trgovačkog putnika pomoću optimizacije kolonijom mrava

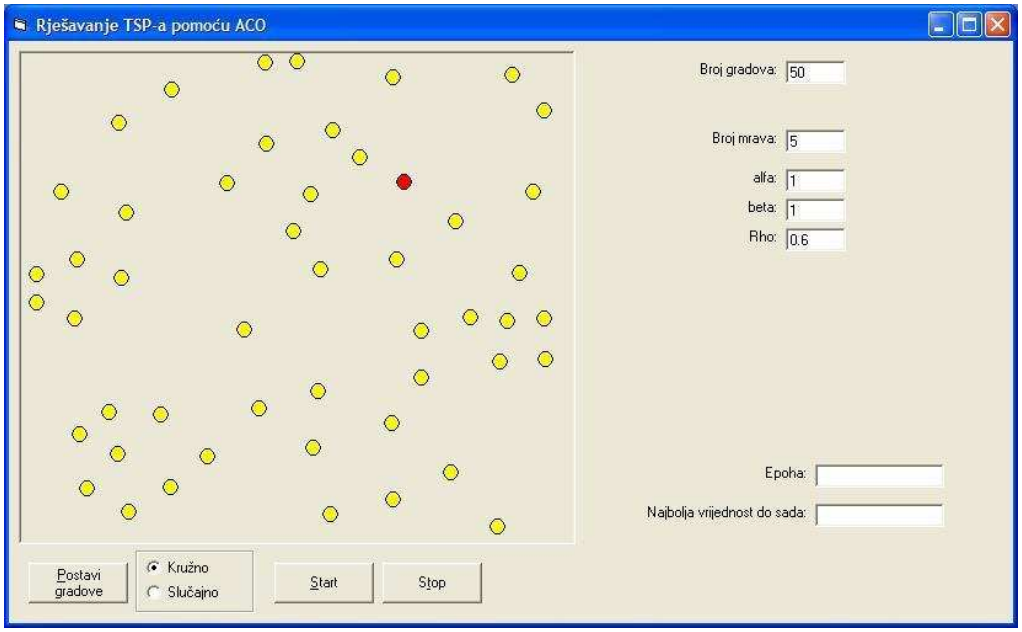
Inicijalno je ponuđeno 50 gradova čiji broj korisnik može promijeniti. Također, moguće je promijeniti i inicijalne vrijednosti za veličinu kolonije (Broj mrava), relativnu važnost traga feromona (alfa), relativnu važnost udaljenosti među gradovima (beta) te konstantu isparavanja (Rho).

Nakon što su definirani ovi parametri potrebno je postaviti gradove. Gradovi se postavljaju pritiskom na gumb *Postavi gradove*. Također, ovisno o odabranoj opciji gradovi se mogu postaviti kružno ili se njihova lokacija može slučajno odabrati. Programski se gradovi inicijalno postavljaju u krug kao što je to prikazano na slici 4.



Slika 4. Inicijalno postavljeni gradovi

Nakon što su gradovi postavljeni u krug, korisnih može promijeniti njihove lokacije. Slika 5 prikazuje postupak promjene lokacije gradova.



Slika 5. Promjena lokacije gradova

Kako bi se promijenila lokacija grada, potrebno je pritisnuti desnom tipkom miša na željeni grad. Željeni grad postaje crvene boje i pomiče se zajedno s kursorom. Ponovnim pritiskom desne tipke miša grad se postavlja na trenutnu lokaciju i postaje žute boje. Na ovaj je način moguće promijeniti lokacije svih gradova.

Pritiskom na gumb *Start* pokreće se postupak optimizacije primjenom kolonije mrava. Tijekom optimizacije, u odgovarajuća se polja ispisuje je broj epoha koje su prošle i najbolja vrijednost puta do sada.

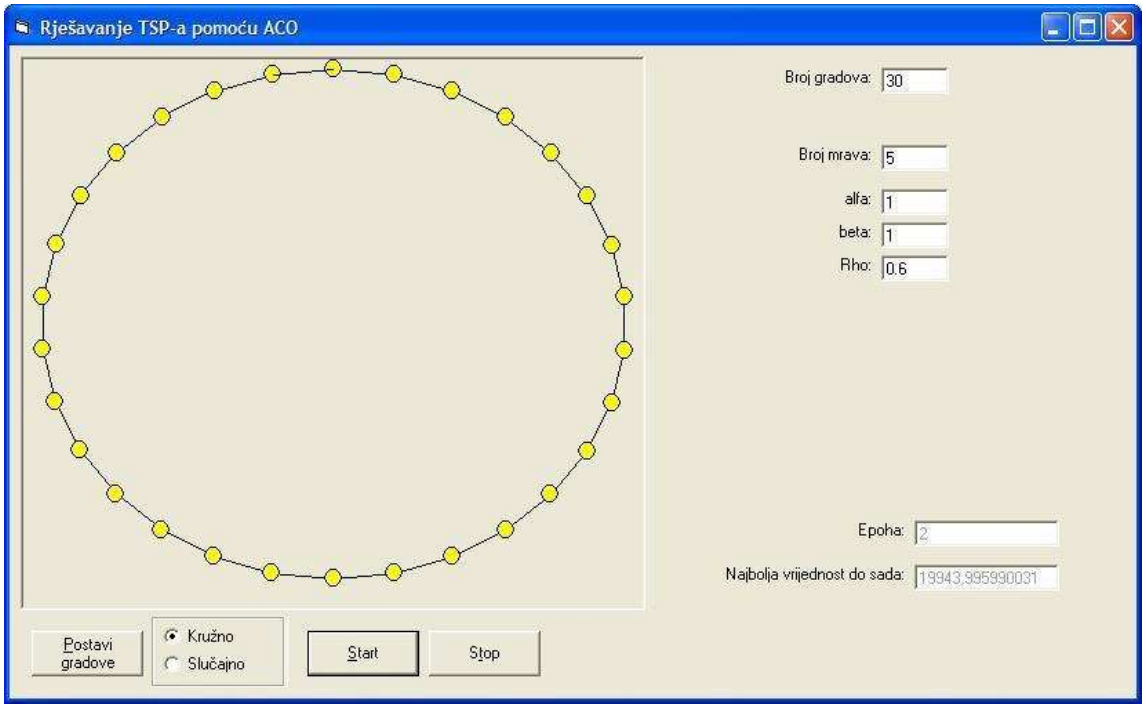
Postupak optimizacije nema definiranih uvjeta zaustavljanja, no njih je moguće definirati kao npr. broj epoha u kojima ne dolazi do smanjivanja najkraće ukupne duljine puta. Optimizacija se u ovoj implementaciji završava pritiskom na gumb *Stop*.

4.1. Primjeri

Kako bi se ispitao rad optimizacije isprobane su različite konfiguracije gradova. Kod pravilnih konfiguracija (npr. kružno postavljeni gradovi) algoritam konvergira vrlo brzo. Kod nepravilnih konfiguracija, konačno rješenje nije deterministički određeno pa se može dogoditi da dva uzastopna izvršavanja algoritma s istim parametrima nad istom konfiguracijom gradova ne rezultiraju jednakim rješenjima.

4.1.1. Kružno postavljeni gradovi

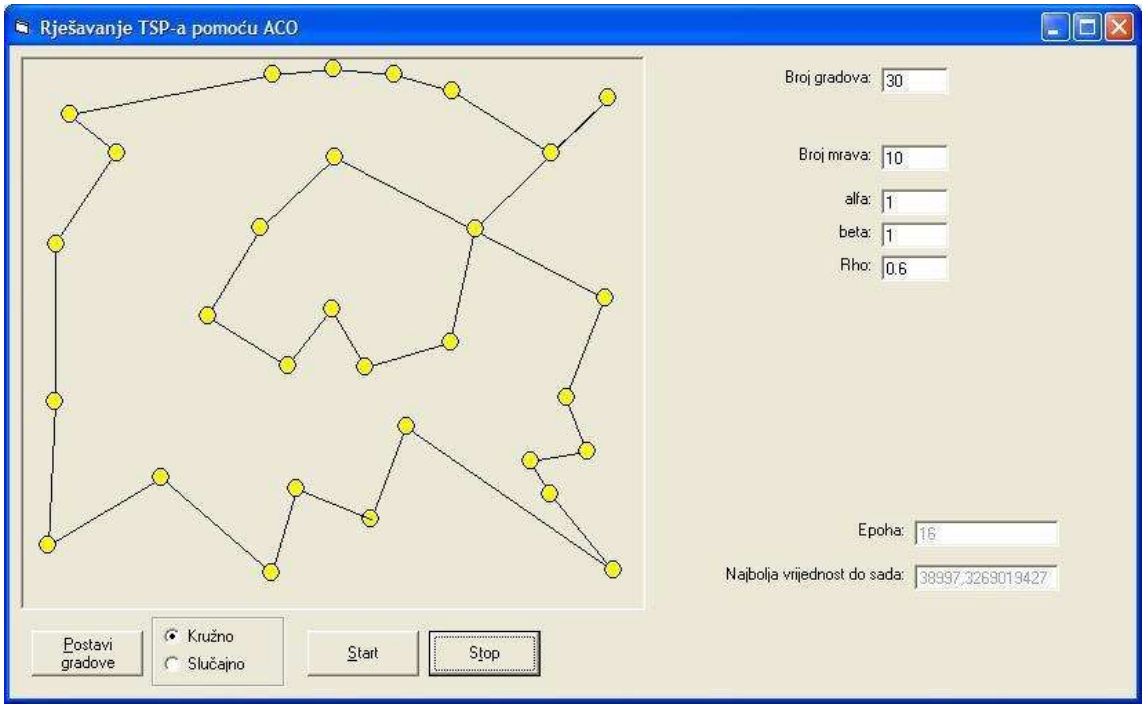
Kod kružno postavljenih gradova nema lokalnih optimuma i zbog toga optimizacija završava vrlo brzo. Elementi matrice feromona su inicijalno svi jednaki (ili 0 ili postavljeni na neku vrijednost), zbog toga vjerojatnost odabira sljedećeg grada inicijalno ovisi samo o udaljenosti među gradovima. Kod rješavanja TSP-a pomoću ACO, uz kružnu konfiguraciju gradova, neisplativo je koristiti veliku koloniju (velik broj mrava), jer bi se kod slijednog izvršavanja samo nepotrebno koristilo procesorsko vrijeme. Slika 6 prikazuje rezultat optimizacije za kružnu konfiguraciju.



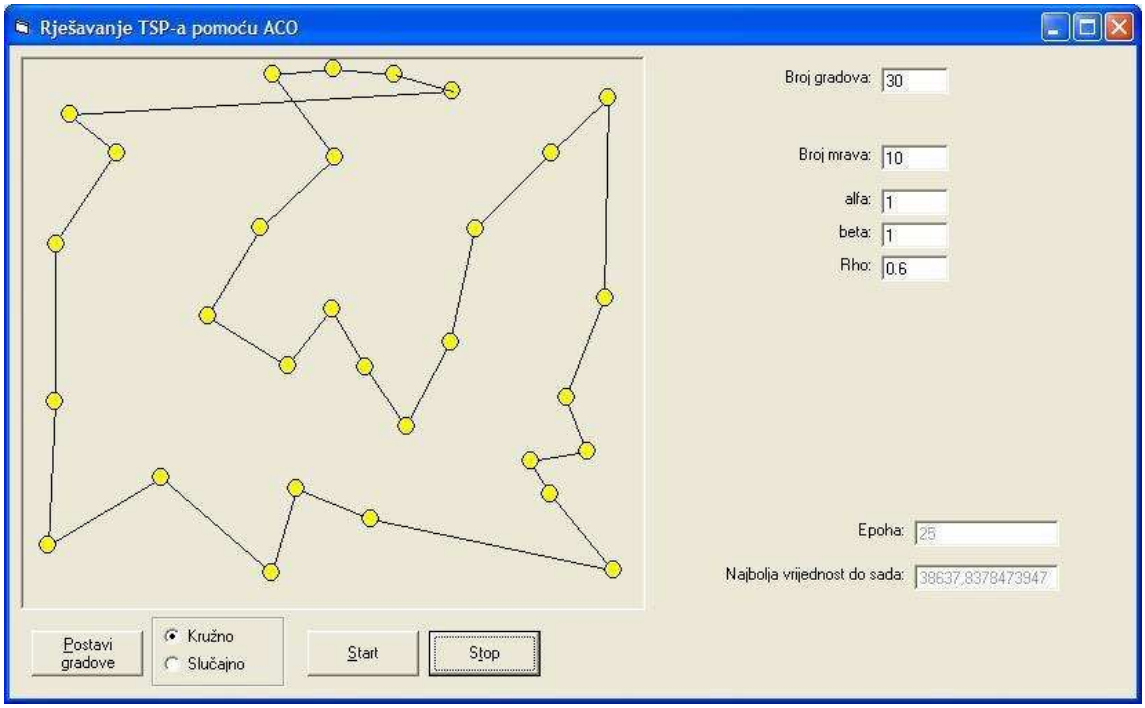
Slika 6. Rješenje TSP-a s kružno postavljenim gradovima

4.1.2. Nepravilno postavljeni gradovi

Kod nepravilno postavljenih gradova ciljna funkcija može imati veći broj lokalnih minimuma i može se dogoditi da algoritam završi u lokalnom optimumu. Naravno, kako optimizacija kolonijom mrava ne predstavlja egzaktnu metodu optimizacije, konačno rješenje ne mora nužno biti optimalno. To je vidljivo iz primjera sa slika 7 i 8. Oba su rješenja dobivena za jednak početni problem i jednake parametre optimizacijskog algoritma.



Slika 7. Prvo rješenje TSP-a s nepravilno postavljenim gradovima



Slika 8. Drugo rješenje TSP-a s nepravilno postavljenim gradovima

5. Usporedba ACO i drugih algoritama za rješavanje TSP-a

Dorigo i Gambardella [1] su usporedili optimizaciju kolonijom mrava s nekim drugim heurističkim algoritmima za optimizaciju. Kako njihovi rezultati predstavljaju najopsežniju usporedbu koju je autor pronašao oni su prikazani ovdje. Tablica 1 predstavlja usporedbu optimizacije kolonijom mrava (ACO), genetskih algoritama (GA), evolucijskog programiranja (EP), simuliranog kaljenja (SA) i hibridnu kombinaciju kaljenja i genetskih algoritama (AG) koju su uveli Lin, Kao i Hsu (1993.).

Tablica 1. Usporedba različitih heurističkih algoritama za rješavanje poznatih konfiguracija problema trgovačkog putnika (tablica je preuzeta iz [2])

Naziv problema	ACO	GA	EP	SA	AG	Optimalno rješenje
Oliver30 (problem 30 gradova)	**420** **(423,74)** [830]	421 (N/A) [3 200]	**420** **(423,74)** [40 000]	424 (N/A) [24 617]	**420** (N/A) [12 620]	**420** **(423,74)**
Eil50 (problem 50 gradova)	**425** (427,96) [1 830]	428 (N/A) [25 000]	426 **(427,86)** [100 000]	443 (N/A) [68 512]	436 (N/A) [28 111]	**425** (N/A)
Eil75 (problem 75 gradova)	**535** **(542,31)** [3 480]	545 (N/A) [80 000]	542 (549,18) [325 000]	580 (N/A) [173 250]	561 (N/A) [95 506]	**535** (N/A)
KroA100 (problem 100 gradova)	**21 282** (21 285,44) [4 820]	21 761 (N/A) [103 000]	N/A (N/A) [N/A]	N/A (N/A) [N/A]	N/A (N/A) [N/A]	**21 282** (N/A)

U tablici 1 je prikazana je najkraća cjelobrojna duljina rute, ispod je u zagradama prikazana najkraća duljina rute dobivena kao realni broj, a ispod toga, u uglatim je zagradama dan broj ciklusa koji su bili potrebni za pronalazak najkraće cjelobrojne rute. Rezultati za EP su od Fogel (1993.), a oni za genetske algoritme su od Bersini, Oury i Dorigo (1995.) za KroA100 problem i od Whitley, Starkweather i Fuquay (1989.) za Oliver30, Eil50 i Eil75 problem. Rezultati za SA i AG su od Lin, Kao i Hsu (1993.). Specifične definicije problema trgovačkog putnika Oliver30, Eil59,

Eil75 i KroA100 su uključene u TSPLIB [2]. Najbolje je rješenje za svaki problem označeno masnim slovima. Zanimljivo je primijetiti kako su složenosti svih algoritama n^2t, osim za EP čija je složenost nt (gdje je n broj gradova, a t broj generiranih ruta). Iz toga je jasno kako ACO i EP uvelike nadmašuju GA, SA i AG.

Moguće je vidjeti kako za prikazane probleme optimizacija kolonijom mrava daje najbolje rezultate. Jedino kod najjednostavnijeg problema s 30 gradova (Oliver30) evolucijsko programiranje daje rezultate koji su jednaki onima od kolonije mrava.

6. Zaključak

U radu je opisana optimizacija kolonijom mrava koja se pokazala kao vrlo intuitivna i jednostavna, a opet efikasna optimizacijska metoda. Također, definiran je i problem trgovačkog putnika koji još uvijek zaokuplja akademsku zajednicu.

Glavni je dio seminarskog rada programska implementacija optimizacije kolonijom mrava za rješavanje problema trgovačkog putnika. Tijekom implementacije autor je seminara došao do zaključka kako ponašanje kolonije mrava uvelike ovisi o tome kako su raspoređeni gradovi. Ukoliko su gradovi pravilno raspoređeni tada algoritam konvergira vrlo brzo, neovisno o broju gradova. U tom je slučaju nepotrebno i neisplativo koristiti velike kolonije. Ukoliko se radi o nepravilnom rasporedu gradova tada optimalnost algoritma postaje upitna i konačno rješenje nije jednoznačno određeno.

Unatoč tome, u radu je prikazano kako optimizacija kolonijom mrava daje bolje rezultate od drugih često korištenih heurističkih optimizacijskih metoda.

7. Literatura

1. Dorigo, M., Gambardella, L.M., Ant Colonies for the Traveling Salesman Problem, *BioSystems*, 43:73-81, 1997.

2. TSPLIB: http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95

3. Dorigo, M., Maniezzo, V., Colorni, A., Ant System: Optimization by a colony of cooperating agents, *IEEE Transactions on Systems, Man, and Cybernetics-Part B*, 26(1):29-41, 1996.

4. http://iridia.ulb.ac.be/~mdorigo/ACO/ACO.html