Univerzitet u Novom Sadu

Fakultet tehničkih nauka

Dokumentacija za projektni zadatak

Studenti:	Ivanović Marina, SV 6/2022 Stanojlović Milica, SV 18/2022 Radovanović Ivana, SV 23/2022
Predmet:	Nelinearno programiranje i evolutivni algoritmi
Broj projektnog zadatka:	15
Tema projektnog zadatka:	Ant colony optimization algoritam, problem najkraćeg puta

Uvod

Ant Colony Optimization (ACO) algoritam predstavlja inspiraciju iz ponašanja mrava u prirodi koji koriste feromone za markiranje optimalnih staza između izvora hrane i njihovog mravinjaka. Ovaj algoritam, zasnovan na principima kolonijalnog ponašanja mrava, pokazao se uspešnim u rešavanju problema optimizacije i pretrage u različitim oblastima. U ovom kontekstu, implementiramo ACO algoritam kako bismo rešili problem pronalaženja najkraćeg puta u mreži tačaka.

Opis problema

Cilj je pronaći najkraći put između zadate početne tačke do zadate krajnje tačke u datoteci **data_path_nodes.txt**. Datoteka sadrži informacije o tačkama i njihovim susedima u obliku grafa. Svaka tačka ima svoju jedinstvenu oznaku, koordinate i informacije o susednim tačkama.

Svaka tačka u mreži može se pomeriti u jednu od svojih susednih tačaka, a dužina pređenog puta između dve tačke izračunava se kao euklidska udaljenost između njih. Implementacijom ACO algoritma želimo pronaći optimalan put od početne do krajnje tačke, koristeći virtuelne mrave koji ostavljaju feromone na putanjama koje prelaze.

Rešenje problema je niz oznaka tačaka koje treba obići kako bismo došli od početne do krajnje tačke, uz informaciju o ukupnoj dužini pređenog puta. Programski jezik Python korišćen je za implementaciju ovog algoritma, koristeći biblioteke kao što su NetworkX za rad sa grafom i NumPy za numeričke operacije.

Ovaj program omogućava korisnicima da prilagode parametre algoritma u cilju dobijanja optimalnih rešenja za različite instance problema najkraćeg puta u mreži tačaka.

Implementacija

Struktura programa

Program implementira Ant Colony Optimization (ACO) algoritam za rešavanje problema najkraćeg puta u grafu koji predstavlja mrežu tačaka. Osnovna struktura programa uključuje funkcije za inicijalizaciju feromona, učitavanje podataka iz datoteke, izbor sledeće tačke, ažuriranje feromona i sam ACO algoritam.

```
def aco_algorithm(graph, start_node, end_node, num_ants, num_iterations, alpha, beta, q, decay_factor):
```

Unutar petlje **num_iterations**, iterira se kroz svakog mrava (ant). Za svakog mrava, započinje se od početne tačke i kreće se kroz graf birajući sledeću tačku pomoću funkcije **choose_next_node**. Ažurira se nivo feromona na izabranoj grani i dodaje izabrani čvor u put mrava. Petlja se ponavlja dok mrav ne stigne do krajnje tačke. Računa se dužina puta

koji je mrav prešao. Ažurira se najbolji put i njegova dužina ako je pronađen kraći put, kao i matrica feromona na osnovu puta koji je mrav prešao. Na kraju funkcija vraća najbolji put i dužinu tog puta.

Ovaj proces se ponavlja kroz više iteracija kako bi se postigli bolji rezultati. ACO algoritam koristi feromone i heuristiku kako bi mravi pronašli optimalan put u grafu, a feromoni se ažuriraju na osnovu efikasnosti pronađenih puteva.

Kriterijum optimalnosti

Kriterijum optimalnosti u ovom programu je minimizacija ukupnog pređenog puta od početne do krajnje tačke u mreži. Algoritam teži pronalaženju najkraćeg puta u grafu, uzimajući u obzir euklidske udaljenosti između tačaka.

Način implementacije virtualnih mrava

Program koristi virtuelne mrave za pretragu grafa. Svaki virtuelni mrav prolazi kroz graf, birajući sledeću tačku na osnovu feromona na ivicama i euklidske udaljenosti do dostupnih tačaka. Virtuelni mravi ostvaruju lokalnu pretragu i doprinose ažuriranju feromona na pronađenim putanjama.

Strategija odabira putanje mrava

Strategija odabira putanje mrava se zasniva na kombinaciji feromona na ivicama i euklidskih udaljenosti između tačaka. Virtuelni mrav bira sledeću tačku sa verovatnoćom koja zavisi od feromona na ivici i udaljenosti do dostupnih tačaka. Ovaj pristup omogućava raznolikost putanja koje mravi istražuju.

Odabir parametara algoritma

Parametri algoritma, kao što su broj mrava, broj iteracija, alfa, beta, q i faktor raspada, određuju ponašanje ACO algoritma. Korisnik ima mogućnost podešavanja ovih parametara u skladu sa specificnim zahtevima problema kako bi se postigao optimalan rezultat.

Rezultati algoritma

Algoritam daje rezultate u vidu najkraće pronađene putanje i ukupne dužine tog puta od početne do krajnje tačke. Rezultati se ispisuju na standardnom izlazu i omogućavaju korisnicima da evaluiraju efikasnost algoritma u rešavanju konkretnih instanci problema najkraćeg puta.

Primenom ovog algoritma možemo dobiti različita rešenja iz različitih pokretanja programa. To je karakteristično za probabilističke algoritme poput ACO.

Različita rešenja mogu proizaći iz sledećih razloga:

1. **Slučajni izbori:** Mravi prave odluke na osnovu verovatnoća. Tokom svake iteracije, mrav bira svoj naredni korak na osnovu feromonskih tragova i heurističkih informacija, što uključuje i element slučajnosti. Različiti izbori tokom izvođenja mogu dovesti do različitih puteva.
2. **Evaporacija feromona:** Algoritam primenjuje stopu isparavanja feromona, što znači da se vrednosti feromona smanjuju tokom vremena. Ovo može dovesti do različitih feromonskih tragova tokom različitih izvođenja, što utiče na izbore mrava.
3. **Inicijalizacija feromona:** Početna vrednost feromona može biti postavljena na različite vrednosti, što takođe može uticati na rezultirajuće puteve.
4. **Parametri algoritma:** Parametri poput broja mrava, faktora isparavanja, i faktora uticaja feromona i heuristike mogu uticati na ponašanje algoritma. Različite vrednosti parametara mogu dovesti do različitih rešenja.

Ako želimo dobiti više različitih najboljih puteva iz različitih pokretanja, možemo ponoviti izvođenje algoritma više puta i sačuvati sve pronađene najbolje puteve. A zatim analizirati ova rešenja kako bismo dobili različite alternative za traženi najkraći put.

- Best path length: 16930.61900579143
- Best path length: 18154.039202588105
- Best path length: 22708.227941001696

Za parameter su uzete vrednosti:

```
num_ants = 10
```

Broj mrava koje koristi ACO algoritam. Veći broj mrava istražuje veći deo prostora rešenja.

```
num_iterations = 5
```

Broj iteracija koje ACO algoritam izvršava. Svaka iteracija omogućava mravima da konstruišu staze i ažuriraju nivoe feromona.

```
alpha = 3.0
```

Parametar uticaja feromona. Veći alpha naglašava uticaj feromona u odlučivanju mrava.

```
beta = 1.0
```

Parametar uticaja heuristike. Veći beta daje više težine heurističkim informacijama (npr. udaljenost između čvorova) u odlučivanju mrava.

```
q = 2.0
```

Parametar za unos feromona. Ovaj parametar određuje količinu feromona koju mrav ostavlja dok prolazi stazom. Povećanje vrednosti q dovodi do većeg ostavljanja feromona.

```
decay_factor = 0.95
```

Faktor raspada feromona. Kontroliše brzinu kojom feromoni isparavaju tokom vremena. Veća vrednost znači sporiji raspada feromona.

Zaključak

U dokumentaciji smo detaljno opisali strukturu programa, kriterijum optimalnosti i implementaciju virtuelnih mrava koristeći ant colony optimization algoritam. Strategija odabira putanje mrava bazirana je na feromonima, a parametri algoritma su pažljivo odabrani kako bi se postigao optimalan rezultat. Rezultati su analizirani i prikazani, potvrđujući efikasnost algoritma u rešavanju problema najkraćeg puta. Implementacija je izvršena u programskom jeziku Python, a dokumentacija pruža sve potrebne informacije za razumevanje, reprodukovanje i evaluaciju rešenja.