Projektni zadatak iz kolegija

Heurističke metode optimizacija

Optimizacija zamjena grupa studenata

s tabu pretraživanjem

Danijel Butković , 0246041206

Dino Perić , 0318003179

# Opis problema

Na FER-u studenti koriste zamjenu grupa kako bi promijenili grupu u kojoj bi htjeli slušati određenu aktivnost na predmetu (npr. predavanja na Fizici). Za što uspješniju zamjenu grupa potrebno je napraviti algoritam koji će temeljem prikupljenih zahtjeva studenata napraviti što veći broj zamjena studenata.

# Opis primijenjenog algoritma tj. heuristike (prikaz rješenja, funkcija cilja/prikladnosti, način dobivanja početnog rješenja, kriterij zaustavljanja i veličina iteracije, elementi dizajna specifični za odabranu heuristiku…)

Algoritam koji je korišten za rješavanje problema je pretraga lokalnog susjedstva kombinirana s tabu listom. Rješenje prikazujemo s binarnim vektorom koji je jednake duljine kao i broj zahtjeva. Vektor predstavlja jedan redak unutar requests file-a. Ako se u vektoru na nekom indeksu n „1“ to znači da je odobren zahtjev pod tim rednim brojem.

Početno rješenje se dobiva tako da uzmemo binarni vektor(u kojem su svi elementi 0) te nasumično odaberemo n zahtjeva koje ćemo odobriti. Prilikom testiranja smo išli greedy pristupom, odnosno iterirali smo kroz vektor i odobrili najveći mogući broj zahtjeva.

Lokalno susjedstvo generiramo tako da uzmemo pola nasumično odabranih indeksa unutar binarnog vektora te na njima radimo bitflip te provjeravamo može li se taj zahtjev odobriti ili oduzeti. Uzimamo u obzir samo one susjede gdje je bitflip moguć.

Nakon generiranja susjeda računamo score za svakog od njih i biramo najboljeg. Indeks zamijenjenog bita najboljeg susjeda dodajemo u taboo listu.

Kriterij zaustavljanja izvođenja programa je vremensko ograničenje jer se do savršenog rješenja u realnoj situaciji nikad neće doći..

Elementi koji su specifični za odabranu heuristiku:

* Binarni vektor – predstavlja listu zahtjeva te sadržava samo 0 ili 1 na svakoj poziciji.

1 predstavlja da je zahtjev odobren, 0 predstavlja da je zahtjev odbijen.

* Tabu lista – koristi se za izbjegavanje lokalnog optimuma na način da bit koji se promjeni

u binarnom vektoru ne smije mijenjati sljedećih *n* iteracija

* Generiranje susjedstva – pretraživanje susjeda u nadi da ćemo naći bolje rješenj

# Pseudokod primijenjenog algoritma

Generiraj početni vektor  
najboljiSusjed = početniVektor  
tabooLista = []  
Dok ima vremena:  
 Generiraj susjedstvo za najboljiSusjed:  
 Odaberi nasumične indekse na kojima će se vršiti bit flip   
 Za indeks u vektoru:  
 Ako indeks nije u tabooLista:  
 Ako je moguće izvršiti zamjenu:  
 spremi susjeda  
 Za svakog susjeda Izračunaj ocjenu  
 pronađeniSusjed = PronađiSusjedaSNajboljomOcjenom();  
 ako Score(pronađeniSusjed) > Score(najboljiSusjed):  
 najboljiSusjed = pronađeniSusjed  
 Dodaj index zamijenjenog bita u tabu listu

Kraj

# Opis dobivenih parametara i diskusija (npr. utjecaj određenih parametara heuristike na kvalitetu rješenja, vrijeme izvođenja algoritma)

Rezultati

Award activity: ['0', '2', '4']

Award student: 1

Minmax penalty: 1

1. **Sample 3**

Iterations: 498  
Time elapsed: 600.5764336585999  
Best score: -7546  
Number of requests given: 697

Iterations: 1389  
Time elapsed: 1800.8971588611603  
Best score: -4008  
Number of requests given: 1531

Iterations: 2330  
Time elapsed: 3600.796347141266  
Best score: -2991  
Number of requests given: 1808

1. **Sample 4**

Iterations: 1781  
Time elapsed: 600.2963931560516  
Best score: -2253  
Number of requests given: 907

Iterations: 5351  
Time elapsed: 1800.2538838386536  
Best score: -2200  
Number of requests given: 891

Iterations: 10947  
Time elapsed: 3600.0730998516083  
Best score: -2211  
Number of requests given: 899

1. **Sample 5**

Iterations: 29199  
Time elapsed: 3600.059859275818  
Best score: -43  
Number of requests given: 526

Iterations: 14861  
Time elapsed: 1800.046605348587  
Best score: -9  
Number of requests given: 530

Iterations: 5030  
Time elapsed: 600.0781738758087  
Best score: -3  
Number of requests given: 548

1. **Sample 6**

Minmax penalty: 2  
Iterations: 18050  
Time elapsed: 3600.1824045181274  
Best score: -520  
Number of requests given: 785

Minmax penalty: 2  
Iterations: 8995  
Time elapsed: 1800.0812602043152  
Best score: -518  
Number of requests given: 787

Minmax penalty: 2  
Iterations: 2705  
Time elapsed: 600.207649230957  
Best score: -476  
Number of requests given: 775

1. **Sample 7**

Minmax penalty: 2  
Iterations: 1438  
Time elapsed: 600.4533398151398  
Best score: -3798  
Number of requests given: 1294

Iterations: 4083  
Time elapsed: 1800.3956537246704  
Best score: -3877  
Number of requests given: 1287

Iterations: 8246  
Time elapsed: 3600.1491680145264  
Best score: -3935  
Number of requests given: 1252

Iz gornjih podataka je vidljivo da vrijeme izvođenja utječe na krajnje rješenje na način da će maksimizirati funkciju cilja (Score) s većim brojem iteracija.

Kod sample 6 i 7 smo pokušali promijeniti minmax penalty i dobili smo lošije rezultate sa duljim izvođenjem programa.

# Diskusiju o „fairness-u“ zamjene grupa (npr. treba li algoritam prioritizirati studente koji traže samo jednu zamjenu ili studente koji traže više zamjena; je li pošteno da studentu s traženom jednom zamjenom nije provedena zamjena, dok su studentu koji je tražio 5 zamjena, provedene 3 ili čak svih 5 zamjena; je li primjerice bolje, u slučaju ako se može provesti samo 1/5 zamjena, provesti tu zamjenu ili ne provoditi zamjene uopće). Također, u raznim slučajevima, prokomentirajte kako biste definirali nagrade/kazne (težinske faktore), s ciljem unaprjeđenja zamjene grupa na FER-u. Primjerice, osvrnite se na to kako bodovima nagraditi ili kazniti zamjenu grupa za studente koji su imali 1/1 zamjenu [broj\_provedenih\_zamjena/broj\_traženih\_zamjena], 0/1, 0/n, 1/n, 3/n, n/n)

Algoritam ne preferira niti jednog studenta te ne radi razliku između studenta s jednom zahtjevom ili studenta s 5 zahtjeva za zamjenom grupa. Zahtjevi se uzimaju slučajnim odabirom tako da svi imaju jednaku šansu da im se zahtjev odobri. To bi trebalo raditi fairness između odabira zahtjeva.