SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

TEHNIČKA DOKUMENTACIJA

Zamjena grupa temeljem želja studenata

Grgur Kovač
Ivan Matak

Zagreb, siječanj, 2019.

1. Opis problema

Ulaz u problem zamjene rasporeda su zahtjevi studenata za promjenu grupe, trenutni raspored te ograničenja za svaku od grupa. Rješenje problema je podskup zahtjeva koje ćemo prihvatiti kako bi maksimizirali određenu zadanu funkciju uz to da ne prekršimo zadana ograničenja. Zadana ograničenja su da u nijednoj grupi ne smije biti više ili manje od određenog broja studenata i da niti jedan student ne smije imati preklapanja u rasporedu. Funkcija koju maksimiziramo osmišljena je tako da maksimizira broj broj prihvaćenih zahtjeva te da se ni u jednoj grupi ne nalazi više odnosno manje studenata nego to je preporučeno.

2. Opis algoritma

Za rješavanje problema korišten je genetski algoritam uz pohlepnu inicijalizaciju početne populacije.

2. 1. Prikaz rješenja

Za prikaz rješenja korišten je bit vektor u kojem su pozicije bit vektora zahtjevi studenata. 0 predstavlja neprihvaćen zahtjev, a 1 prihvaćen.

2. 2. Funkcija cilja

Funkcija cilja je višekomponentna i sastoji se od:

- Bodovi A - ukupni zbroj težišnih faktora
- Bodovi B - ukupni zbroj svih nagradnih faktora kod svakog studenta zasebno
- Bodovi C - broj studenata * faktor nagrade za učinjene sve zamjene za studenta
- Bodovi D - suma bodova svih grupa kod kojih je broj studenata u grupi manji od min_preferred. Računa se kao (min_preffered-students_cnt)*(minmax_penalty)
- Bodovi E - suma bodova svih grupa kod kojih je broj studenata u grupi veći od max_preferred. Računa se kao (max_preffered-students_cnt)*(minmax_penalty)

Ukupna ocjena = Bodovi A + Bodovi B + Bodovi C - Bodovi D - Bodovi E

U slučaju da jedinka krši neko od strogih ograničenja vrijednost ukupnog rješenja je **-(broj prekršenih ograničenja)*100000**

2. 3. Način dobivanja početnog rješenja

Prije samog generiranja početnog rješenja radi se pretprocesiranje zahtjeva kako bi se uklonili oni zahtjevi koje nije moguće ispuniti, tj. zahtjeve za ulaz u grupu u kojoj je broj studenata u grupi maksimalan, a ne postoje zahtjevi za izlaz iz te grupe te zahtjeve za ulaz u grupu koja ima minimalan broj studenata, a ne postoje zahtjevi za ulaz u tu grupu što za posljedicu ima smanjeno područje pretraživanja. Za dobivanje početnog rješenja koristi se pohlepni pristup koji nasumično bira pozicije na koje će postaviti bit i uzima to rješenje ako je bolje od prethodnog. Pri inicijalizaciji se koristi i tabu lista koja pamti sve pozicije prijašnjih nasumično biranih bitova te se resetira nakon što se popuni sa 1000 elemenata. Takav proces pohlepnog biranja zahtjeva traje 2 minute. Kod konstrukcije početne populacije dobiveno rješenje se nastavlja nadograđivati pohlepnim algoritmom jos pola minute (za svaku jedinku zasebno). Početna populacija sastoji se od 3 jedinke.

2. 4. Kriterij zaustavljanja

Kriterij zaustavljanja je postavljeno vrijeme (10, 30 ili 60 min).

2. 5. Specifičnosti heuristike

Heuristika je zapravo kombinacija prije opisanog pohlepnog algoritma i evolucijskog algoritma. Za biranje nove generacije koristi se EaMuPlusLambda algoritam selekcije sa elitizmom. Algoritam uzima populaciju predanu kao argument te je evoluira koristeći operator križanja ili mutacije (isključivo). Biramo tri jedinke koristeći se algoritmom EpsilonLexicase koji vraća najbolje jedinke te ih koristi pri stvaranju desetoro djece. Korišteni algoritam križanja odabere 10 nasumičnih indeksa iz prve jedinke te ih kopira u drugu te potom nasumičnih 10 indeksa iz druge koje kopira u prvu. Za algoritam mutacije koristi se prije opisani pohlepni algoritam koji se izvodi 1 sekundu ili nasumična mutacija bitova sa vjerojatnosti (1/broj_zahtjeva).

3. Pseudokod

```
genetski_algoritam():
        pohlepno_odaberi_pocetnu_populaciju()
        krizaj_ili_mutiraj_jedinke(vjerojatnost_krizanja=0.8, vjerojatnost_mutacije=0.2)
        odaberi_sljedecu_generaciju()

krizaj_ili_mutiraj_jedinke(vjerojatnost_krizanja, vjerojatnost_mutaicije):
        B = nasumičan_broj()
        Ako B < 0.8:
                Za svaki par jedinki:
                        križaj()
        Inače:
                Za svaku jedinku:
                        mutiraj(jedinka)

mutiraj(jedinka):
        B = nasumičan_broj()
        Ako B < 0.1:
                jedinka = Nasumično_mutiraj_bitove(jedinka);
        Inače:
                jedinka = pohlepno_mutiraj(jedinka);
```

4. Opis dobivenih parametara i diskusija

Dobiveni parametri:

- Vrijeme pohlepnog algoritma za inicijalnu jedinku: 3min
- Vrijeme pohlepnog algoritma za inicijalnu populaciju: 1min
- Vrijeme pohlepnog algoritma kod mutiranja: 1sek
- Vjerojatnost mutacije jedinke 0.2
- Vjerojatnost križanja jedinki 0.8
- Vjerojatnost nasumične mutacije jednikne tijekom mutacije 0.1
- Broj odabranih jedinki 3
- Broj djece 10

Za sve parametere uspješnost algoritma ne varira jako osim za ekstremne vrijednosti mutacije (npr 0.8) kada algoritam divergira u rješenje koje je izvan dozvoljenih ograničenja.

5. Diskusija o pravednosti zamjene grupa

Nije pošteno da se jedan student prioretizira iznad drugog na temelju broja poslanih zahtjeva. Bilo bi pravednije da se prioretizira na temelju vremena kada je zahtjev primljen.