Heurističke metode optimizacije

08

**Fall**

Projekt

Problem raspoređivanja osoblja

Ivan Čapalija

Filip Gobin

Zagreb, Siječanj 2016.

# Opis problema

# Opis primijenjenog algoritma

## Način dobivanja početnog rješenja

# Za dobivanje početnog rješenja izgradili smo pohlepni algoritam koji svakom zaposleniku iterativno dodjeljuje smjene u ovisnosti o trenutnom stanju i zadovoljivosti čvrstih ograničenja.

# Iznimno je teško bilo izgraditi pohlepni algoritam, jer svako rješenje mora zadovoljiti osam čvrstih ograničenja. U svakom koraku algoritma bilo je potrebno voditi računa smije li iduća smjena doći nakon prethodne, jesmo li dosegli ukupni broj smjena, jesmo li premašili maksimalno radno vrijeme, jesmo li premašili maksimalan broj uzastopnih smjena, jesmo li dodijelili dovoljan broj uzastopnih dana odmora, jesmo li premašili broj radnih vikenda i je li trenutni dan ujedno i dan godišnjeg odmora zaposlenika. Kod generiranja početnog rješenja nismo vodili računa o mekim ograničenjima, jer će ona biti optimirana genetskim algoritmom. U nastavku možete pronaći pseudokod pohlepnog algoritma koji je korišten za generiraje početnog rješenja.

# Pseudokod primijenjenog algoritma

Pseudokod algoritma za generiranje početnog rješenja:

#data storage

string[][] data = new string[employeesNum][daysNum];

#iterate employees

for (employee in employees):

#iterate days

for (dayIndex in days):

if (previousDay is not empty):

previousShift = data[employee][day - 1];

else:

dayIndex += employee.minConsecutiveDaysOff;

consecutiveShifts = 0;

#set future shifts number

if (consecutiveShifts == 0):

futureShiftsNum = employee.minConsecutiveShifts;

else if (consecutiveShifts >= employee.minConsectiveShifts &&

consecutiveShifts < employee.maxConsecutiveShifts):

futureShiftsNum = 1;

else:

consecutiveShifts = 0;

continue;

consecutiveShifts += futureShiftsNum;

#suggest a list of valid consecutive shifts

suggestedShifts = suggestShifts(previousShift, futureShiftsNum);

#if there is not a valid list of consecutive shifts skip day

if (suggestedShifts == null):

continue;

#check hard constraints for current state

if (!checkHardConstraints(day)):

continue;

#write suggested shifts to employee

for (shiftIndex in len(suggestedShifts)):

data[employee][dayIndex + shiftIndex] = suggestedShifts[shiftIndex];

#skip days with assigned shifts

dayIndex += len(suggestedShifts) – 1;

# Opis dobivenih rezultata i diskusija

# Kratki zaključak