

UNIVERZA V LJUBLJANI  
FAKULTETA ZA MATEMATIKO IN FIZIKO

Finančna matematika – 1. stopnja

Katarina Brilej, Sara Kovačič

**Uporaba metahevrstike GRASP na problemu potujočega  
trgovca**

Projekt OR pri predmetu Finančni praktikum

Mentor: prof. dr. Riste Škrekovski

Ljubljana, 2018

## 1. UVOD

Metahevrstika je algoritemski način reševanja kombinatoričnega optimizacijskega problema, pri katerem na začetku izberemo množico kandidatov za rešitev, in jo iterativno izboljšujemo (glede na neko vnaprej izbrano funkcijo zaželenosti), ter po dovolj korakih vrnemo najboljši element iz te množice. Metahevrstike torej vrnejo približne rešitve, a veliko hitreje kot eksaktni postopki. V projektu bova na problem potujočega trgovca implementirali metahevrstiko GRASP (*greedy randomized adaptive search procedure*). Problem potujočega trgovca bova rešili tudi kot celoštevilski linearni program in primerjali rešitve. Generirali bova nekaj zanimivih grafov in na njih preizkusili algoritem. Rezultate bova primerjali tudi z rezultati iz spleta in rezultati skupine 7, ki bo na problem potujočega trgovca implementirala genetski algoritem.

## 2. PROBLEM POTUJOČEGA TRGOVCA

Problem potujočega trgovca ("travelling salesman problem"/TSP) se glasi:

- **Formulacija v vsakdanjem jeziku:** danih je  $n$  mest in razdalja med poljubnim parom mest (od mesta do mesta lahko potujemo po zgolj eni poti). Najdi najkrajšo (najcenejšo) pot, ki se začne in konča v istem mestu ter obišče vsako mesto natanko enkrat.
- **Formulacija v matematičnem jeziku:** v (neusmerjenem enostavnem) polnem grafu  $K_n$  z uteženimi povezavami (pozitivne vrednosti) najdi najkrajši cikel, ki vsebuje vsa vozlišča. Ciklom, ki vsebujejo vsa vozlišča grafa, pravimo Hamiltonovi cikli.

### 2.1. Celoštevilski linearni program.

Problem potujočega trgovca lahko predstavimo kot *celoštevilski linearni program*. Označimo mesta s števili  $1, \dots, n$ . Strošek (ali razdalja) potovanja iz mesta  $i$  v mesto  $j$  je  $c_{i,j}$ , ( $1 \leq i, j \leq n$ ). Minimiziramo strošek potovanja. Definiramo:

$$X_{i,j} := \begin{cases} 1 & \text{potnik gre iz mesta } i \text{ v mesto } j, \\ 0 & \text{sicer,} \end{cases}$$

$y_i$  ... katero po vrsti obiščemo mesto  $i$

$$\begin{aligned}
\min \quad & \sum_{i=1}^n \sum_{j=1}^n x_{i,j} \cdot c_{i,j} \\
\text{p. p.} \quad & \sum_{i=1}^n x_{i,j} = 1, \text{ za vsak } j \\
& \sum_{j=1}^n x_{i,j} = 1, \text{ za vsak } i \\
& x_{i,j} \in \{0, 1\}, \text{ za vsak } i \text{ in vsak } j \\
& y_i \in \{1, \dots, n\}; \text{ za vsak } i \\
& y_i + 1 - n + n \cdot x_{i,j} \leq y_j; \text{ za vsak } i \text{ in vsak } j > 1
\end{aligned}$$

### 3. GRASP

GRASP (*greedy randomized adaptive search procedure*) je metahevrstika, ki se stoji iz dveh faz: *greedy randomized construction* in *local search*. V prvi fazi na pameten način (odvisno od problema) izberemo izmed vseh možnih rešitev CL (candidate list) množico začetnih približkov RCL (restricted candidates list). To storimo deloma deterministično in deloma stohastično, da zagotovimo, da so začetni približki obetavni, a dovolj razpršeni po celotni množici CL, da bo druga faza pregledala čimvečji del CL. V drugi fazi za vsako izmed teh rešitev  $s \in RCL$  pregledamo elemente  $s' \in CL$  v njeni okolici (kaj je okolica je od problema in načina reševanja odvisno). Če najdemo boljšo rešitev  $s'$ , jo dodamo v RCL ter  $s$  odstranimo. To ponavljamo dokler zaustavitveni pogoj (npr. št. iteracij, zahtevana natančnost) ni izpolnjen.

### 4. NADALJNJE DELO

Problem potujočega trgovca bova predstavili z matriko, kjer element  $a_{i,j}$  matrike  $A$  predstavlja razdaljo (oz. ceno) potovanja po poti od mesta  $i$  do mesta  $j$ . Predpostavimo, da je matrika pozitivna in simetrična in, da 0 izven diagonale v tej matriki pomeni, da povezava med mestoma ne obstaja. Za takšno matriko bova v Pythonu napisali obe fazi algoritma *GRASP* (Greedy randomized construction *in* Local search). Problem potujočega trgovca bova rešili tudi kot celoštevilski linearni program in primerjali rešitve. Generirali bova nekaj zanimivih grafov in na njih preizkusili algoritem. Rezultate bova primerjail tudi z rezultati iz spleta in rezultati skupine 7, ki bo na problem potujočega trgovca implementirala genetski algoritem.

## LITERATURA

- [1] Travelling salesman problem  
[http://en.wikipedia.org/wiki/Travelling\\_salesman\\_problem](http://en.wikipedia.org/wiki/Travelling_salesman_problem)