Homework 1: Random search for the Frequency Assignment Problem

Student: José Alejandro Cornejo Acosta
Professor: Dr. Carlos Segura González

Centro de Investigación en Matemáticas.
Subject: Optimización Estocástica

1 Introduction

The Frequency Assignment Problem (FAP) [Lai and Hao, 2015] is a common one that is usually presented in wireless communications systems. Thus, it is a relevant problem since telecommunication networks are used daily. The informal description of the problem is the following: Given a set of frequency channels, and a radio network composed of transmitters and connections between them, FAP aims to assign a frequency channel to each transmitter such that the interference is minimized. Here, interference arises when two close frequency channels are assigned to two close transmitters.
The problem can be formally defined through graph theory: Given a set F of consecutive frequency channels, and a double weight undirected graph $G = (V, E, D, P)$ where V is the set of vertices, E set of edges, and D and P edge weight sets. The purpose is to find a mapping (i.e., a frequency assignment) $f : V \to F$ such that minimizes an objective function. In this case, the objective function is denoted by Equation 1:

$$\min \sum_{(i,j)\in E; |f_i - f_j| \leq d_{ij}} p_{ij} \tag{1}$$

where

- $d_{ij} \in D$ represents the constraint distance between two adjacent vertices i and j.
- $p_{ij} \in P$ represents the penalization when the frequency channels assigned to two adjacent vertices i and j are close enough.

2 Random search for the FAP

In this homework, a random search procedure for the Frequency Assignment Problem is worked. The procedure is very simple and consists in following: given a valid instance for the FAP, and a set frequency channels F, to randomly generate a set of $|X| = 100,000$ feasible candidate solutions. Then, each solution

is evaluated through the objective function of Equation 1. Finally, the candidate solution with minimum objective value is considered as the *best* and it is returned.

In the Frequency Assignment Problem, each candidate solution can be represented as a vector $\vec{x} = \langle x_1, x_2, \cdots, x_{|F|} \rangle$ where $x_i \in F$. Note that x_i values can be repeated more than one. Algorithm 1 shows the general procedure of random search procedure.

Algorithm 1: Random search

Input: A weighted graph $G = (V, E, D, P)$ and the number of frequency channels $|F|$
Output: The best found solution fitness value

```
// Generate 100,000 random solutions
```
$F = \{v \mid v \in \mathbb{Z} \wedge 1 \leq v \leq |F|\}$;
$X = \emptyset$;
for $j = 1$ **to** $100,000$ **do**
  $\vec{x} = \langle x_1, x_2, \cdots, x_{|V|} \rangle$;
  for $i = 1$ **to** $|V|$ **do**
    $x_i =$ any value $v \in_R F$;
  end
  $X = X \cup \{\vec{x}\}$
end
```
// Evaluate solutions and return the best found one
```
$best\_fitness = +\infty$;
foreach $\vec{x} \in X$ **do**
  $best\_fitness = \min\{best\_fitness, evaluator(\vec{x})\}$;
end
return $best\_fitness$;

Algorithm 2: Evaluator

Input: A weighted graph $G = (V, E, D, P)$ and a vector $\vec{x} = \langle x_1, x_2, \cdots x_{|V|} \rangle \in \mathbb{Z}_+^{|V|}$
Output: Fitness value of $\vec{x}$

$fitness = 0$;
foreach $(i, j) \in E$ **do**
  if $|x_i - x_j| \leq d_{ij}$ **then**
    $fitness = fitness + p_{ij}$;
  end
end
return $fitness$;

2.1 Complexity of *Evaluator*

Assuming that we can access in $O(1)$ to x_i, x_j, d_{ij} and p_{ij} for any $(i, j) \in E$, Algorithm 2 evaluates candidate solutions for the Frequency Assignment Problem

in $O(|E|)$. This is because Algorithm 2 consists mainly in a loop that iterates over the set E and all inside operations are performed in constant time.

3 Computational results and experimentation

The random search procedure (Algorithm 1) was executed over the instance `GSM2-272.ctr` for three different numbers of frequency channels $|F| \in \{34, 39, 49\}$. For each value of $|F|$, 100 independent runnings were executed with different *seeds*. The results are summarized through Table 1.

$\|F\|$	**min**	**max**	μ	σ
34	7.7e+09	8.9e+09	8.6e+09	2.4e+08
39	6.7e+09	7.6e+09	7.2e+09	1.8e+08
49	4.9e+09	5.7e+09	5.4e+09	1.8e+08

Table 1: Results over instance `GSM2-272.ctr`.

From Table 1, "min" and "max" columns represent the objective values of best and worst solutions of the 100 runnings. Columns μ and σ represent the average and standard deviation of the 100 runnings. From Table 1 we can observe that the smaller the value of $|F|$, the worse the solutions are. This is intuitive since for bigger values of $|F|$, the probability of sampling two close elements decreases. However, found solutions are considered of bad quality. Besides, we can see that for all values of $|F|$, dispersion is considerably high. From these observations, we conclude that random search is not good for approaching this problem.

Figure 1 shows a bloxplot of the 100 independent runnings per each value of $|F|$. From this figure, the observations are similar to those obtained from Table 1. That is, the quality of found solutions is better for bigger values of $|F|$. Furthermore, from Figure 1 we can observe that for the three different values of $|F|$, the solutions are not distributed in a symmetric distribution, since the median (orange line) is not located at the middle of the box (for $|F| = 34$), and the range length of quartiles is not the same. We can also see that the solutions are more disperse in the first two quartiles, mainly in the first one. Finally, there are some atypical values for $|F| = 34$ and $|F| = 49$.

3.1 Experimentation details

The random search algorithm was implemented in C++. The source code was attached with this report, but it can also be found in `https://github.com/alex-cornejo/StoOptCIMAT2022`. The `README.md` file contains instructions for compiling and running in a single machine.

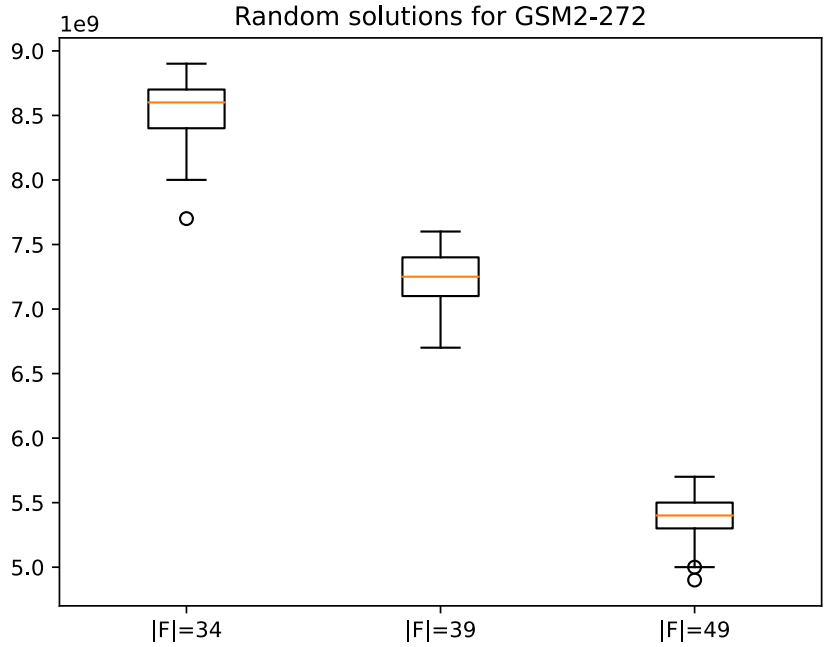

Fig. 1: Boxplot of 100 independent runnings per each value of $|F| \in \{34, 39, 49\}$

For each value of $|F|$, the 100 independent runnings were performed by using parallel computing in the "Laboratorio de Supercómputo del Bajío". To do this, three tasks scrips were created (one per each value of $|F|$). Those scrips are located in `/F34/tasks.txt`, `/F39/tasks.txt` and `/F49/tasks.txt`.

Then, such tasks scrips are running through slurm scrips. The slurm scripts are `slurm_F34`, `slurm_F39` and `slurm_F49`.

Once the slurm scrips are executed through the command `sbatch slurmscript`, the results will be located in different output files in the folders `LSB/F34`, `LSB/F39` and `LSB/F49` respectively. Then, the `plotter.py` file can be executed to join the results and create the boxplot.

References

[Lai and Hao, 2015] Lai, X. and Hao, J.-K. (2015). Path relinking for the fixed spectrum frequency assignment problem. *Expert Systems with Applications*, 42.