Obnoxious p-median problem

Đorđe Drakulić

1. Uvod

Obnoxious p-median problem je varijanta p-median problema koja traži najdalja moguca rešenja, odnosno, za listu korisnika i objekata traži se p objekata koji su što dalje od korisnika.
Pošto je ovaj problem po prirodi NP težak problem, pored brute-force rešenja, biće implementirane optimizacione metode kao što su simulirano kaljenje, VNS itd.

2. Rešavanje problema:

Cilj rešavanja obnoxious p-median problema iz [1] i [2] je zapravo rešavanje:

$$\max \sum_{i \in I} \min_{j \in J,\ y_j=1} d_{ij}$$

gde je:

I – skup klijenata
J – skup objekata
d_{ij} – udaljenost između klijenta i i objekta j
$y_j = 1$ – Objekat j je otvoren

i gde jos važi ograničenje:

$$\sum_{j \in J} y_j = p,\ y_j \in \{0,1\}$$

2.1 Priprema podataka:

Podatke o pojedinom korisniku/objektu cemo čuvati u klasi **Unit** koja sadrži ime, geografsku širinu i geografsku dužinu. Distancu između dve instance Unit ćemo računati pomoću **haversine** funkcije koja izračunava udaljenost između dve tačke u geografskoj površi. Pomoću naknadno napravljene funkcije **generate_units** učitavamo podatke o korisnicima/objektima i pravimo niz instanci Unit na osnovu tih podataka.

2.1 Brute-force:

Brute-force rešenje se osniva na formulaciju iz [1]:

$$\max \sum_{i \in I} \sum_{j \in J} d_{ij}\ x_{ij}$$

gde je dodatno:

x_{ij} - 1 ako je klijent i usvojen objektu j, 0 inače
$x_{ij} \leq y_j$ - dodeli klijente samo otvorenim objektima

Ova matematička formulacija je osnov brute force pristupa rešavanju ovog problema i njen pseudokod je:

```
inicijalizacija rešenja najboljih_p, najbolja_ukupna_udaljenost
for kombinacija_p_otvorenih in sve_kombinacije_p_otvorenih:
        trenutna_distanca =
        izračunaj_ukupnu_distancu(kombinacija_p_otvorenih, klijenti)
        if trenutna_distanca > najbolja_ukupna_udaljenost:
                najboljih_p = kombinacija_p_otvorenih
                najbolja_ukupna_udaljenost = trenutna_distanca
return najboljih_p,  najbolja_ukupna_udaljenost
```

kombinacija_p_otvorenih zadovoljava ograničenje da broj otvorenih objekata treba da bude p i posto prolazimo kroz sve moguće kombinacije p otvorenih, ovaj pristup će proci kroz sva potencijalna rešenja problema.
izračunaj_ukupnu_distancu prolazi kroz svakog klijenta i traži najbliži otvoreni objekat I vraća sumu takvih udaljenosti.

2.2 Optimizacione metode:

Moguće rešenje ovog problema je skup objekata veličine p, pa je zato lako imlementirati optimizacione metode iz [4] kao što su:

1. **VNS -** gde se rešenje menja tako što se redom 1, pa 2, pa na kraju p objekata zatvori, pa neki drugi objekti otvore, pa se na osnovu novodobijenog skupa, izračunava rešenje.
2. **SA -** gde se od početka vrši eksploracija pa polako prelazi na eksploataciju tako što se skup sve manje menja što vise vremena prolazi.
3. **TS -** gde se prvih k objekata čuva kao nedostupnim zarad eksploracije.

2.3 Objekti nad neprebrojivim skupom:

Svi algoritmi navedeni do sad pretpostavljaju da je dat prebrojiv skup objekata od kojih će se birati p najboljih. Moguće je napraviti algoritam koji traži najbolje pozicije p objekata, ali pošto su rešenja objekata međusobno zavisna, nije moguce koristiti standardne swarm intelligence metaheuristike, već nešto što podseća na gradijentni spust. Ideja algoritma je da za objekte prođe kroz jedan po jedan objekat, izračuna ukunu distancu celog rešenja i onda ako je rešenje bolje, taj objekat će se nastaviti kretati u tom smeru, a ako je rešenje gore, objekat će se kretati u nekom drugom pseudonasumicnom smeru. Za kompenzaciju što nemamo predefinisan skup objekata, imaćemo ograničen domen u kojem će se to objekti moći kretati.

Pseudo kod:

```
inicijalizacija nasumicnih p pozicija objekti, ukupna_distanca
for _ in range(broj_iteracija):
    for objekat in objekti:
        treuntna_distanca = 0
        pomeri_objekat(objekat, smer)
        treuntna_distanca  =
            izračunaj_ukupnu_distancu(objekti, klijenti)
        if treuntna_distanca > ukupna_distanca:
            ukupna_distanca = treuntna_distanca
            objekat.zapamti_koordinate()
        else:
            promeni_smer_kretanja(objekat)
    mutiraj(objekti)
for objekat in objekti:
    objekat.vrati_se_na_zapamćenu_poziciju()
return objekti, ukupna_distanca
```

gde je mutiraj(objekti) jako mala šansa da se neki (a može i više) objekti premeste u potpuno drugom pseudonasumicnom mestu.

Moguće je napraviti i ovakve algoritme sa ograničenjima npr. najmanja dozvoljena udaljenost između objekta i klijenta, najmanja dozvoljena udaljenost između dva objekta itd. gde je ovde jos implementirano najmanja dozvoljena udaljenost između objekta i klijenta tako sto se na kraju pretrage pokušava dovesti rešenje u dozvoljeno stanje.

3. Eksperimenti:

Što se tiče skupa klijenata/objekata korisćen je skup gradova i nuklearnih elektrana u Republici Francuskoj [3]:

```
cities_csv.head()
```

	city	lat	lng	country	iso2	admin_name	capital	population	population_proper
0	Paris	48.8567	2.3522	France	FR	Île-de-France	primary	11060000	2148271
1	Bordeaux	44.8400	-0.5800	France	FR	Nouvelle-Aquitaine	admin	994920	994920
2	Marseille	43.2964	5.3700	France	FR	Provence-Alpes-Côte d'Azur	admin	873076	873076
3	Lyon	45.7600	4.8400	France	FR	Auvergne-Rhône-Alpes	admin	522250	522250
4	Toulouse	43.6045	1.4440	France	FR	Occitanie	admin	504078	504078

```
[19]: nuclear_plants_csv.head()
```

[19]:

	Tri	Legal perimeter	Perimètre spatial	Spatial perimeter	Filière	Sector	Power plant	Unit	Combustible	Fuel	...	Unit.1	GPS position (wsg84)	Region	INSEE region code	Depa
0	5	EDF SA	France métropolitaine, sans la Corse ni les il...	Metropolitan France, excluding Corsica and the...	Nucléaire	Nuclear	BELLEVILLE	BELLEVILLE 1	Uranium Enrichi	Enriched Uranium	...	MW	47.508946, 2.875676	CENTRE-VAL DE LOIRE	24	
2	7	EDF SA	France métropolitaine, sans la Corse ni les il...	Metropolitan France, excluding Corsica and the...	Nucléaire	Nuclear	BLAYAIS (LE)	BLAYAIS 1	Multi-oxyde d'uranium et de plutonium	Multi-oxide of uranium and plutonium	...	MW	45.257605, -0.690606	NOUVELLE-AQUITAINE	75	GI
6	18	EDF SA	France métropolitaine, sans la Corse ni les il...	Metropolitan France, excluding Corsica and the...	Nucléaire	Nuclear	BUGEY (LE)	BUGEY 2	Uranium Enrichi	Enriched Uranium	...	MW	45.801148, 5.266072	AUVERGNE-RHONE-ALPES	84	
10	22	EDF SA	France métropolitaine, sans la Corse ni les il...	Metropolitan France, excluding Corsica and the...	Nucléaire	Nuclear	CATTENOM	CATTENOM 1	Uranium Enrichi	Enriched Uranium	...	MW	49.415953, 6.218271	GRAND EST	44	M
14	28	EDF SA	France métropolitaine, sans la Corse ni les il...	Metropolitan France, excluding Corsica and the...	Nucléaire	Nuclear	CHINON B	CHINON B 1	Multi-oxyde d'uranium et de plutonium	Multi-oxide of uranium and plutonium	...	MW	47.228727, 0.168307	CENTRE-VAL DE LOIRE	24	IND

5 rows × 27 columns

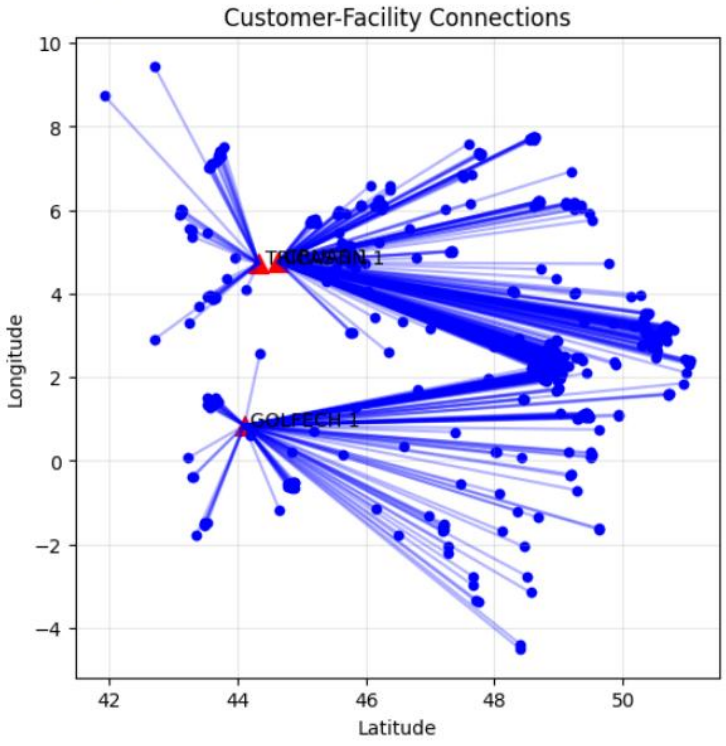
U ovom slučaju ima 627 gradova i 18 nuklearnih elektrana.

3.1 Brute-force:

Koristiće se za proveru ostalih algoritama, za male vrednosti p radi prihvatljivo brzo, ali što se više povećava p, dolazi do kombinatorne eksplozije. U ovom slučaju najgore moguće p jednako je polovini broja nuklearnih elektrana tj. 9 (jer se mora proći kroz n nad p broj kombinacija) za koje mu treba preko 10 minuta da se izvrši.

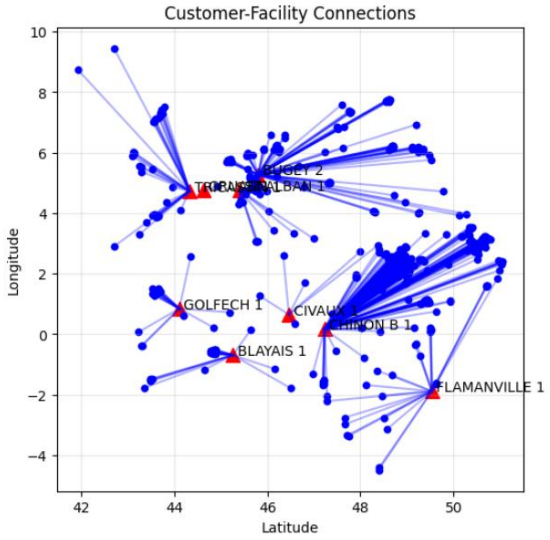
Primer kada je p jednako 3:

```
(CRUAS 1 44.63283 4.750824, GOLFECH 1 44.105751 0.84572, TRICASTIN 1 44.326355 4.731541) 273730.7242838876
Elapsed time: 4.3848961
```



Primer kada je p jednako 9:

```
(BLAYAIS 1 45.257605 -0.690606, BUGEY 2 45.801148 5.266072, CHINON B 1 47.228727 0.168307, CIVAUX 1 46.46218 0.648879, CRUAS 1 44.63283 4.750824, FLAMAN
VILLE 1 49.535986 -1.883342, GOLFECH 1 44.105751 0.84572, ST-ALBAN 1 45.405445 4.755573, TRICASTIN 1 44.326355 4.731541) 138148.3254111005
Elapsed time: 689.3361419
```



3.2 VNS:

Za relativno male p vrednosti, VNS skoro uvek nađe najbolje rešenje, ali što se više p povećava, to ce biti manja verovatnoća da ce naći najbolje rešenje. U odnosu na brute-force radi brzo i za velike vrednosti p i uglavnom daje najbolje rešenje u odnosu na ostale tehnike optimizacije:

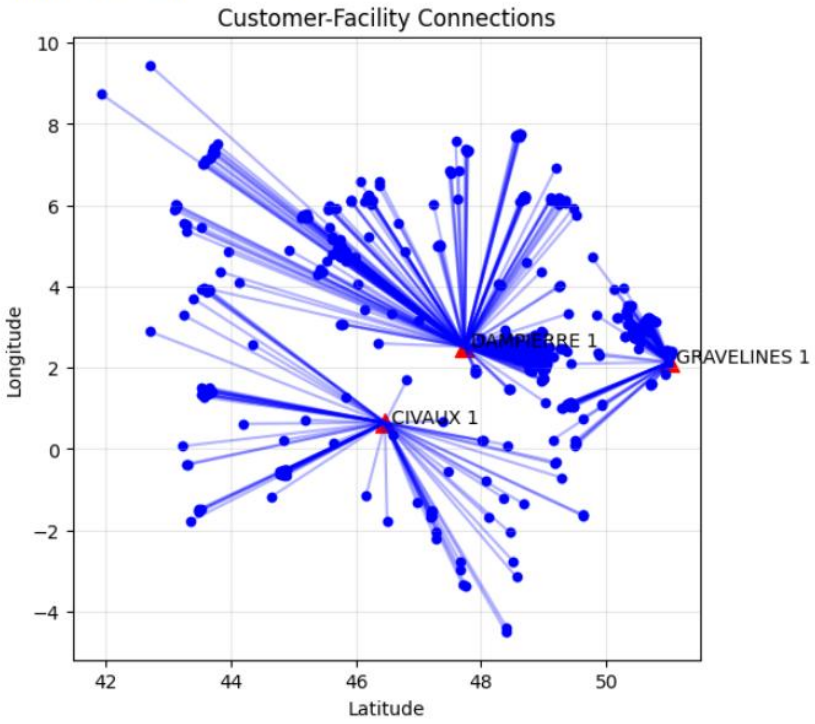
```
[PENLY 1 49.976144 1.210236, FLAMANVILLE 1 49.535986 -1.883342, ST-LAURENT B 1 47.720248 1.580217, DAMPIERRE 1 47.732638 2.517824, BELLEVILLE 1 47.50894
6 2.875676, CHOOZ B 1 50.090344 4.789588, BUGEY 2 45.801148 5.266072, GOLFECH 1 44.105751 0.84572, PALUEL 1 49.858754 0.634759] 111433.39686784256
Elapsed time: 23.682133899999826
```

3.3 SA:

Na osnovu eksperimenata, simulirano kaljenje radi najgore. Cak i za malo p, mnogo je veca sansa da ne nađe najbolje rešenje u odnosu na VNS, ali radi primetno brže za veliko p u odnosu na VNS:

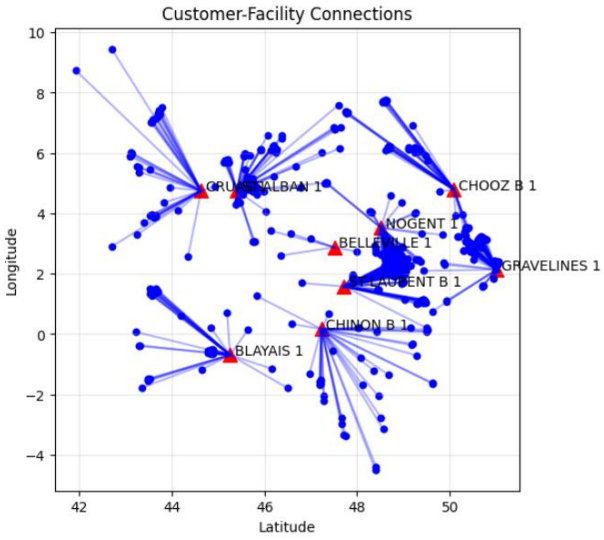
Za p jednako 3 nije uspeo da nađe najbolje rešenje:

```
[CIVAUX 1 46.46218 0.648879, GRAVELINES 1 51.012846 2.139287, DAMPIERRE 1 47.732638 2.517824] 252824.6458790651
Elapsed time: 1.4604916000007506
```



Za p jednako 9 ne radi puno sporije:

```
[GRAVELINES 1 51.012846 2.139287, NOGENT 1 48.514581 3.524182, ST-LAURENT B 1 47.720248 1.580217, CHINON B 1 47.228727 0.168307, CHOOZ B 1 50.090344 4.7
89588, CRUAS 1 44.63283 4.750824, BLAYAIS 1 45.257605 -0.690606, BELLEVILLE 1 47.508946 2.875676, ST-ALBAN 1 45.405445 4.755573] 106567.2504239508
Elapsed time: 4.1748677999985375
```



3.4 TS:

Tabu pretraga u ovom slučaju se ponaša jako slično simuliranom kaljenju, pa zato ima slicne osobine sto se tiče njenog rešenja pretrage. Za male vrednosti p, veca je šansa da nađe najbolje rešenje u odnosu na simulirano kaljenje, mada za veće p nije puno bolje od simuliranog kaljenja sa zanemarivo istom brzinom.

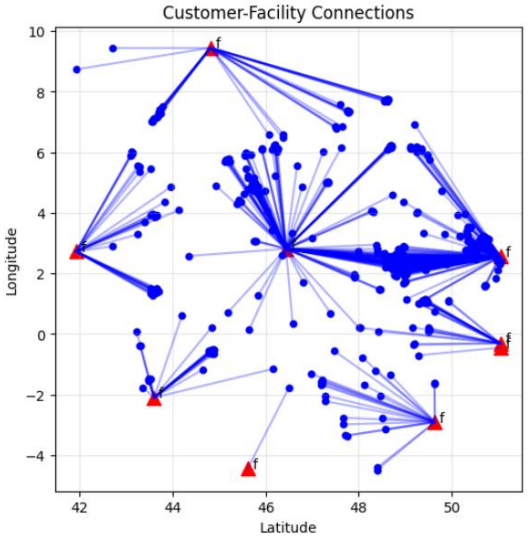
Primer za p jednako 9:

```
[BUGEY 2 45.801148 5.266072, PENLY 1 49.976144 1.210236, CRUAS 1 44.63283 4.750824, CIVAUX 1 46.46218 0.648879, ST-LAURENT B 1 47.720248 1.580217, BELLE
VILLE 1 47.508946 2.875676, NOGENT 1 48.514581 3.524182, PALUEL 1 49.858754 0.634759, FLAMANVILLE 1 49.535986 -1.883342] 106567.2504239508
Elapsed time: 6.975443899998936
```

3.5 Arbitrary positioned facilities:

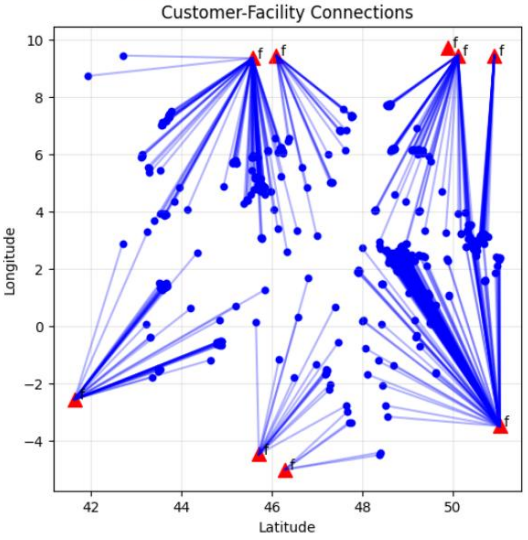
Nema skup mogućih rešenja, već nad geografskom površi traži najbolje koordinate elektrana. Uglavnom daje bolja rešenja čak i od brute-force rešenja, ali ima tendenciju da postavlja rešenja pri ivicama dozvoljenog domena, jer je tu uglavnom najbolje rešenje. Na osnovu eksperimenata radi sličnom brzinom kao VNS:

```
[f 46.43635172509847 2.8163748100772397, f 51.04129999999999 2.5603670329058628, f 51.04129999999999 -0.30548524538523364, f 44.81534119233406 9.4503, f
41.9267 2.72253402304519, f 51.04129999999999 -0.4476831668365744, f 45.616178393805946 -4.432461436933945, f 43.601530475781296 -2.0985675982845517, f
49.636571174337924 -2.9052870694258397] 132894.02452670582
Elapsed time: 25.273775400000886
```



Primer rada ovog algoritma sa dodatim ograničenjem minimalne dozvoljene udaljenosti između objekta i klijenta:

```
Elapsed time: 25.5950239
[f 45.71312407051188 -4.464551997218639, f 46.28885100025566 -4.9991237536050654, f 50.11908386163968 9.4503, f 51.04129999999999 -3.4807247956349903, f
46.10013665298672 9.4503, f 45.57202921609632 9.356934985122818, f 41.63518262935315 -2.5469139619557266, f 49.88853460665634 9.733303201427368, f 50.92
5217480776716 9.4503] 257743.8674597577
```



Da se ne dogodi beskonačno dugo popravljanje rešenja, proširen je domen za onoliko koliko je minimalna dozvoljena distanca izmedju objekta i klijenta, i posle određenog broja iteracija prestaje pametno da menja smer gde da pomeri objekat koji krši ograničenje, tako da je izbavak iz nedozvoljenog rešenja zagarantovano.

4. Zakljucak:

Za jako male vrednosti p, može se koristiti brute-force algoritam, za malo veće vrednosti p pokazao se da je najbolji VNS jer uglavnom daje najbolja rešenja ali je sporiji u odnosu na ostale, dok za vece vrednosti p najbolji je taboo search jer je jedan od najbržih i ne daje najgore rešenje. Ako treba da odredimo gde treba da postavimo objekte umesto da od određenog broja rešenja biramo p, arbitrary positioned facilities ispunjava te uslove, podseća na gradijentni spust, mogu se dodati ograničenja ali nije lako raditi sa njima jer rešenja svakog objekta nisu međusobno nezavisna, što je razlog zašto nije imalo smisla koristiti swarm intelligence metaheuristike.

5. Literatura:

1. “The obnoxious p-median problem: theory and praxis” https://epub.jku.at/download/pdf/8507667.pdf
2. “Multi-objective Memetic Optimization for the Obnoxious p-Median Problem” https://www.uv.es/rmarti/paper/docs/multio4.pdf
3. Ulazni podaci: contains original data downloaded from https://opendata.edf.fr, updated on 29/10/2025, under the terms and conditions of the Etalab licence.
4. “Matf-RI” https://github.com/PetarP02/matf-RI