A Machine Learning enhanced Variable Neighborhood Search approach for the Uncapacitated Facility Location problem

International Conference on Variable Neighborhood Search 2025 (ICVNS)

Lucas Martín-García (lucas.martin@urjc.es)
Isaac Lozano-Osorio (isaac.lozano@urjc.es)
J. Manuel Colmenar (josemanuel.colmenar@urjc.es)
Belén Melián-Batista (mbmelian@ull.es)

Universidad Rey Juan Carlos

grafo RESEARCH

Universidad de La Laguna

INTERNATIONAL CONFERENCE ON VARIABLE NEIGHBORHOOD SEARCH VNS 2025

MINISTERIO DE CIENCIA E INNOVACIÓN

AGENCIA ESTATAL DE INVESTIGACIÓN

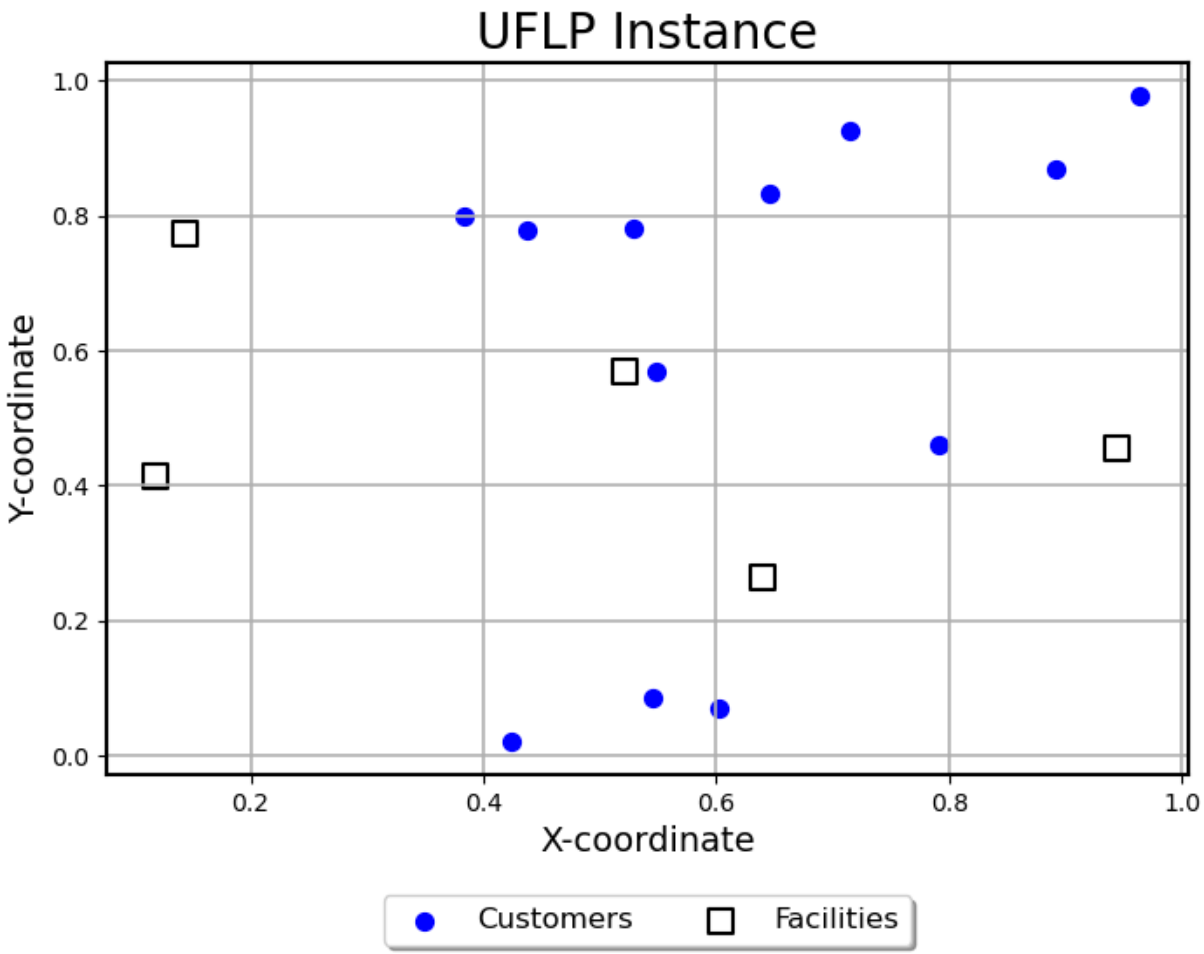
Outline

Introduction

- The **UFLP** (Uncapacitated Facility Location Problem) belongs to the family of **Facility Location Problems**.
- It aims to **minimize** the sum of the **cost of opening facilities** plus the **cost of assigning customers** to these facilities.
- Real-world applications:
  - Resource allocation and network configuration.
  - Logistics and transportation.

Introduction

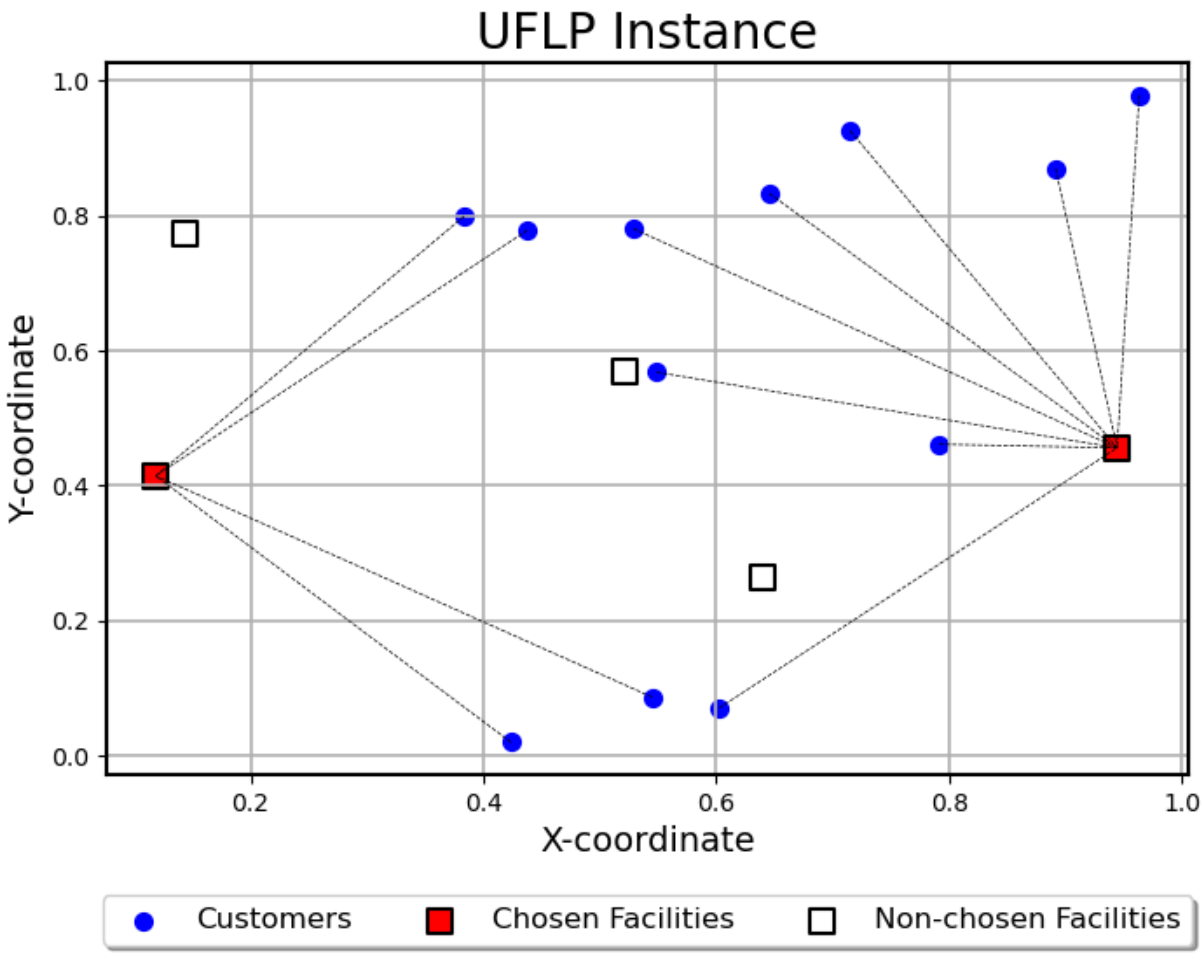
Solution representation



- Set of **customers**
- Set of **facilities**

Introduction

Solution representation

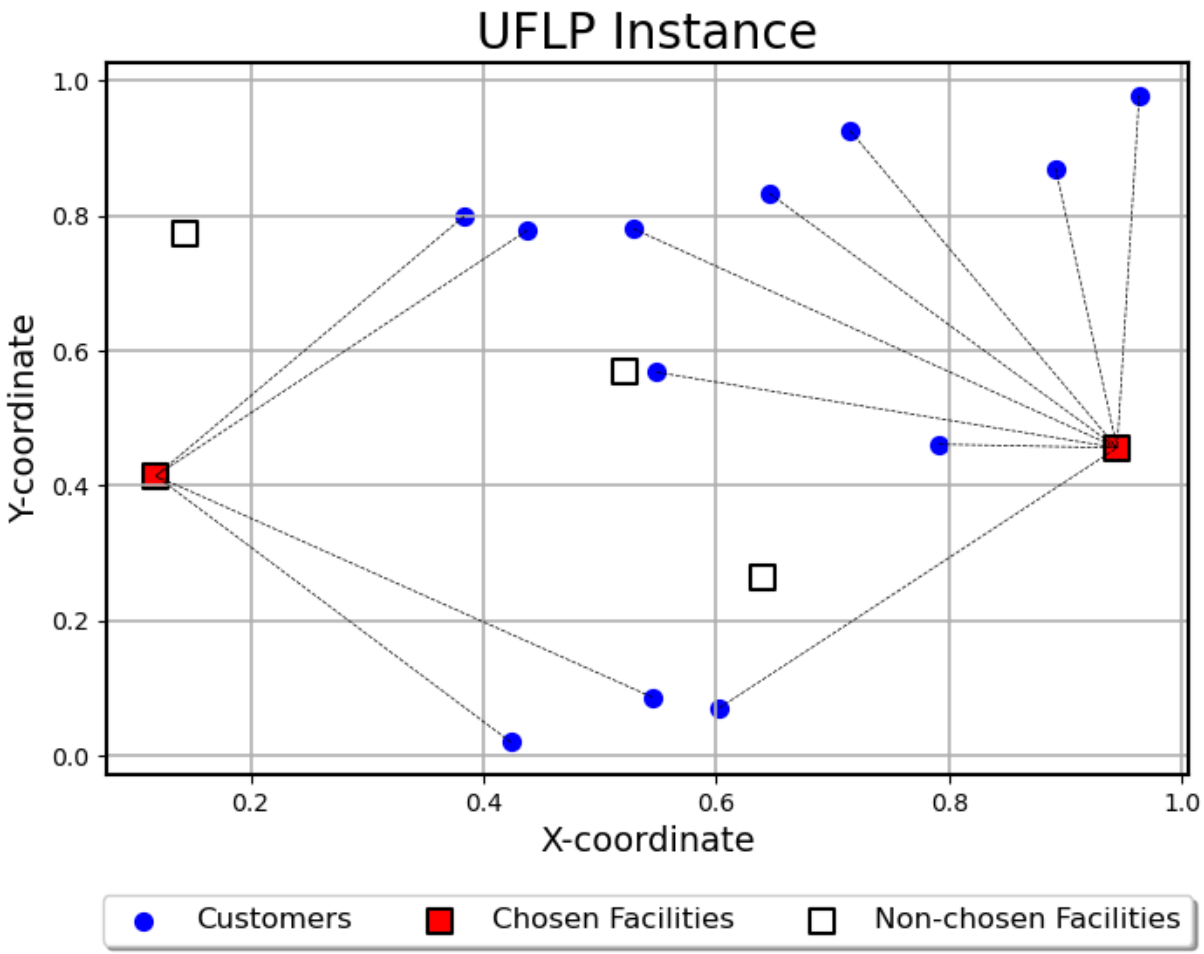


- Set of **customers**
- Set of **facilities**

Solution: open an unbounded number of facilities and assign customers.

Introduction

Solution representation



- Set of **customers**
- Set of **facilities**

Solution: open an unbounded number of facilities and assign customers.

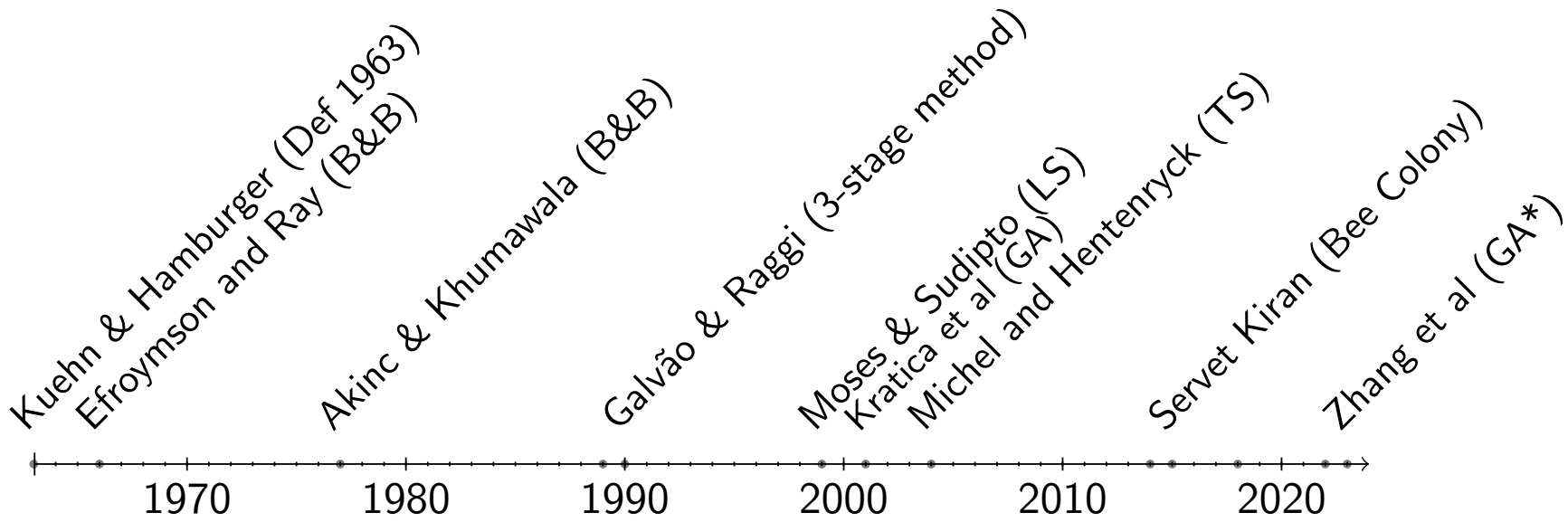
Objective: minimize opening cost plus service cost.

$$\sum_{i=1}^{m}\sum_{j=1}^{n} d_{ij}y_{ij} + \sum_{j=1}^{n} g_j x_j$$

Introduction

Literature review

It has been proven to be $\mathcal{NP}$-Hard (Karp 1972).



Introduction

Mathematical formulation

$$\text{Minimize} \quad \sum_{i=1}^{m}\sum_{j=1}^{n} d_{ij}y_{ij} + \sum_{j=1}^{n} g_j x_j \tag{1}$$

Subject to

$$\sum_{j=1}^{n} y_{ij} = 1, \quad i = 1, 2, \ldots, m \tag{2}$$

$$y_{ij} \leq x_j, \quad i = 1, 2, \ldots, m, \quad j = 1, 2, \ldots, n \tag{3}$$

$$y_{ij}, x_j \in \{0, 1\}, \quad i = 1, 2, \ldots, m, \quad j = 1, 2, \ldots, n \tag{4}$$

Implemented in Gurobi.

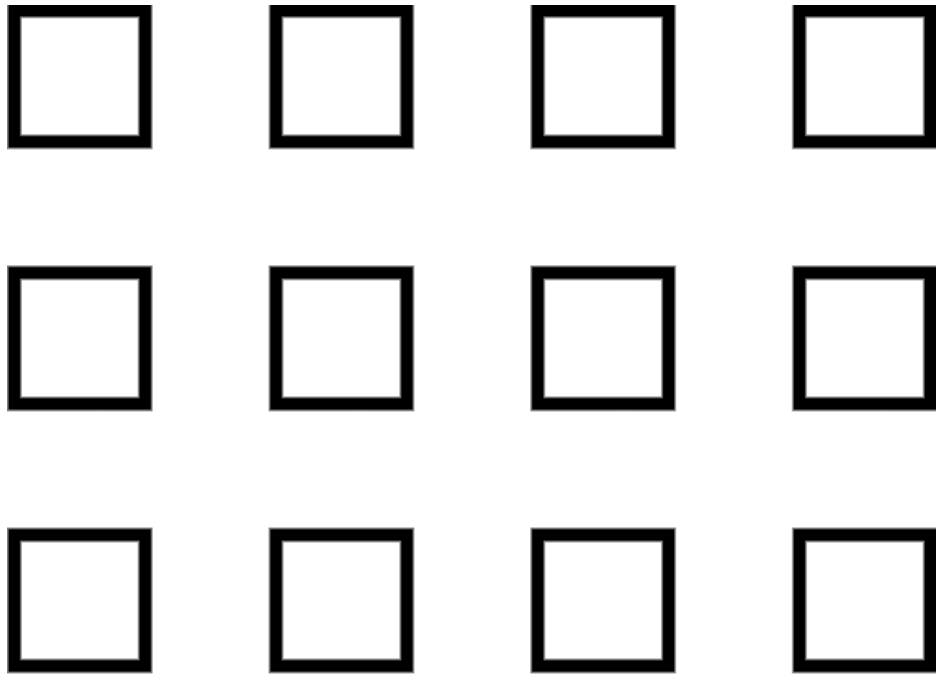
BVNS

Algorithm 1 BVNS(k_{max})

```
1: S ← Constructive()
2: S* ← S
3: k ← 0
4: while k < k_max do
5:     S ← Shake(S,k)
6:     S ← Improvement(S)
7:     if F(S) < F(S*) then
8:         S* ← S
9:         k ← 1
10:    else
11:        k ← k + 1
12:    end if
13: end while
14: return S*
```

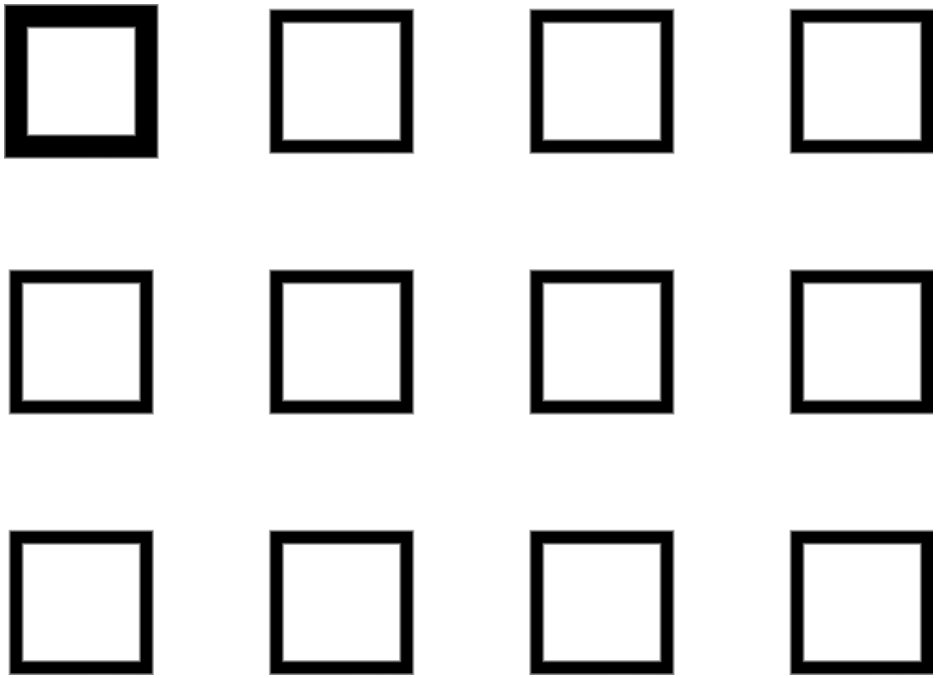
Constructive

Random constructive



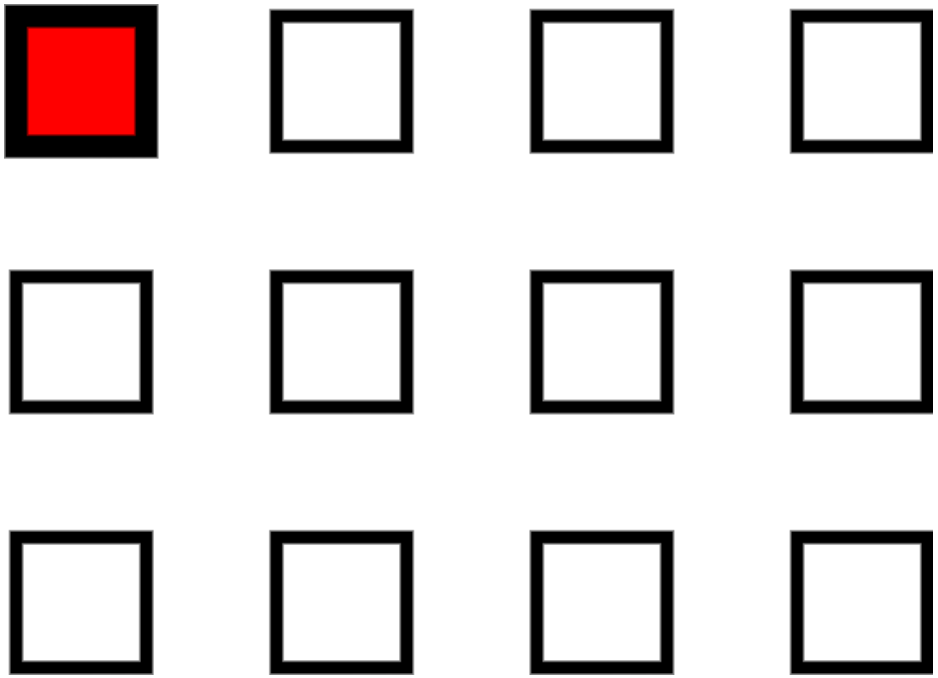
Constructive

Random constructive



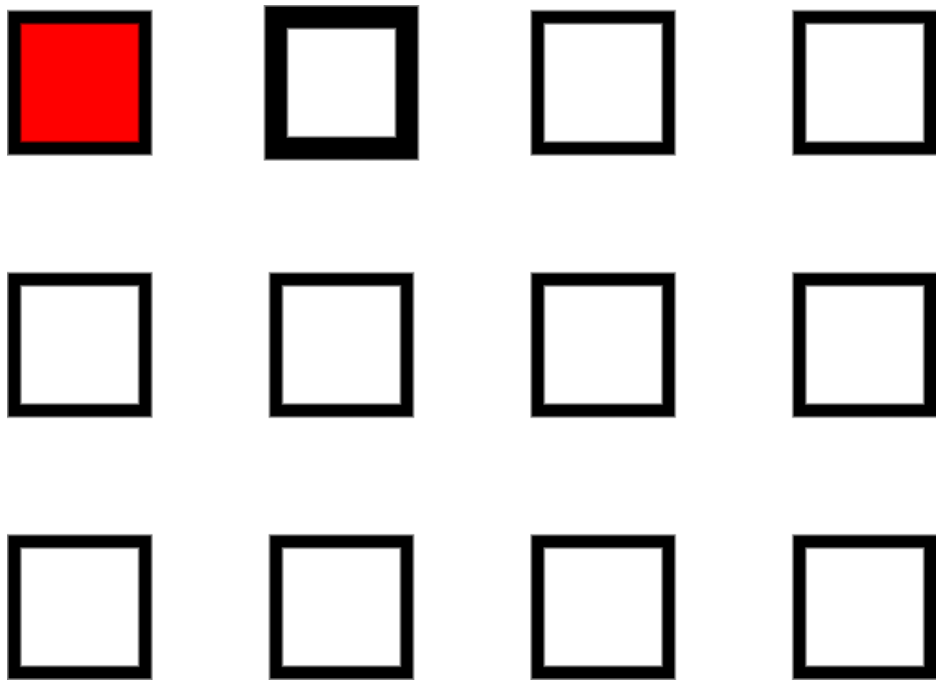
Constructive

Random constructive



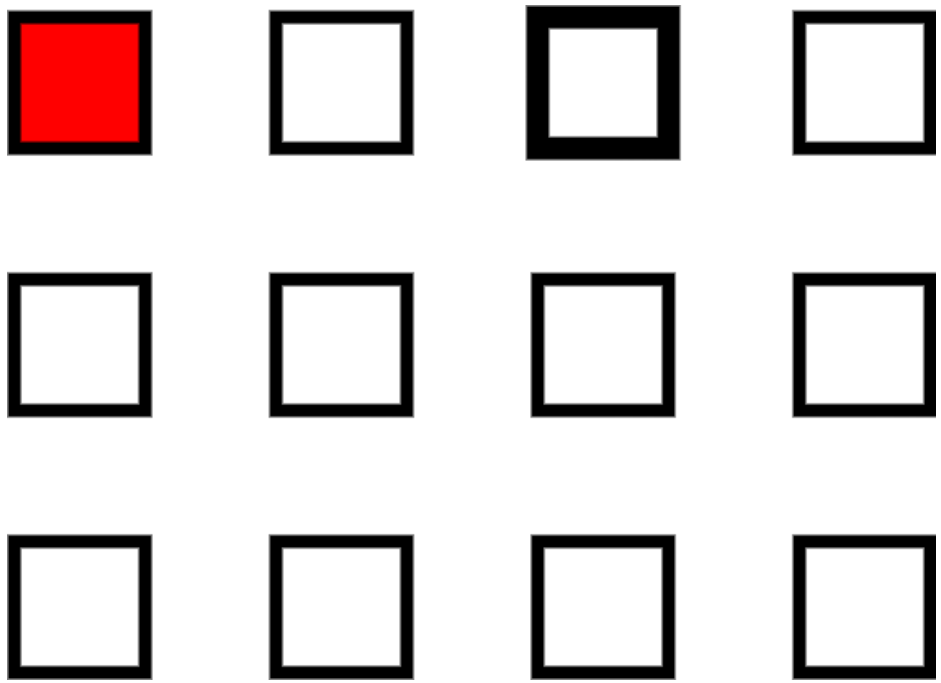
Constructive

Random constructive



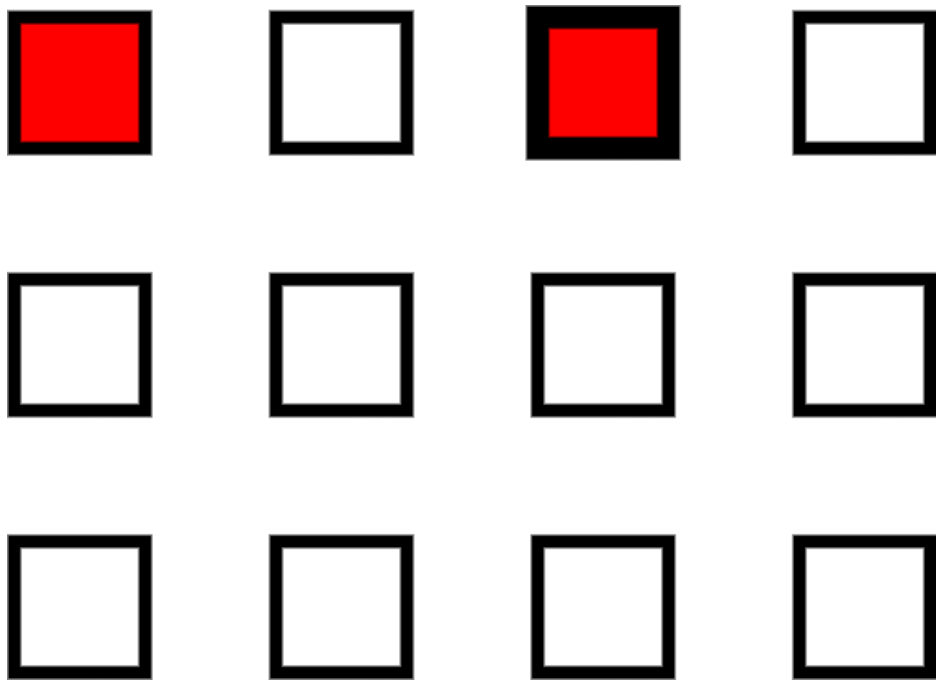
Constructive

Random constructive



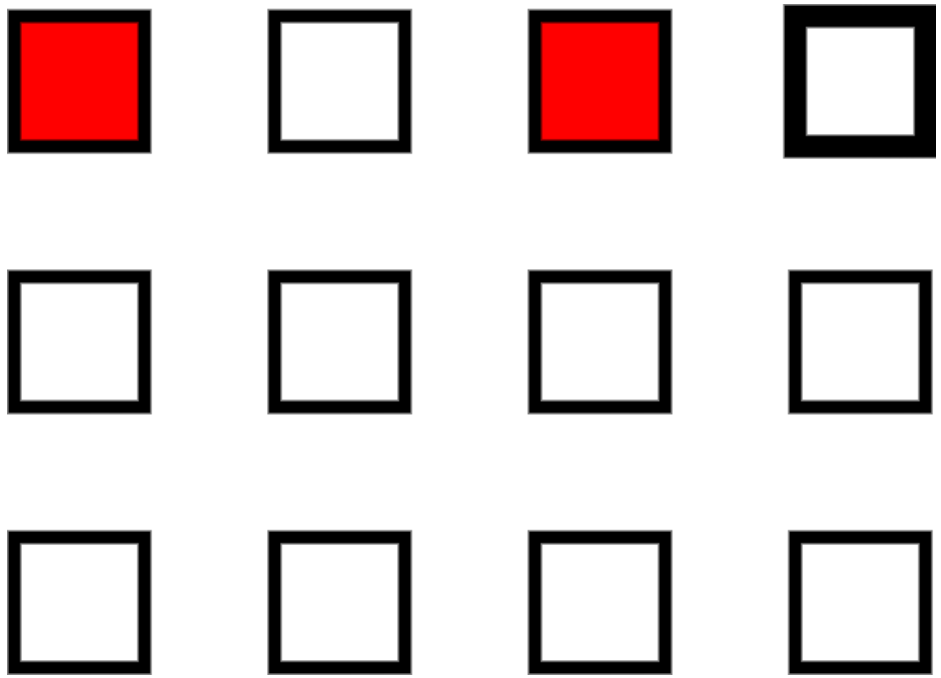
Constructive

Random constructive



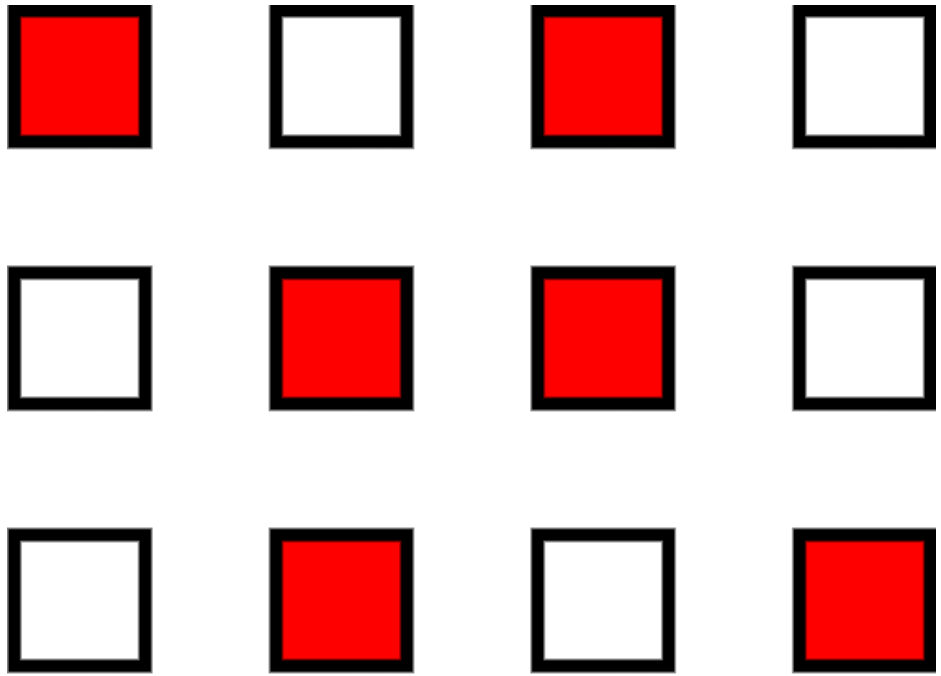
Constructive

Random constructive

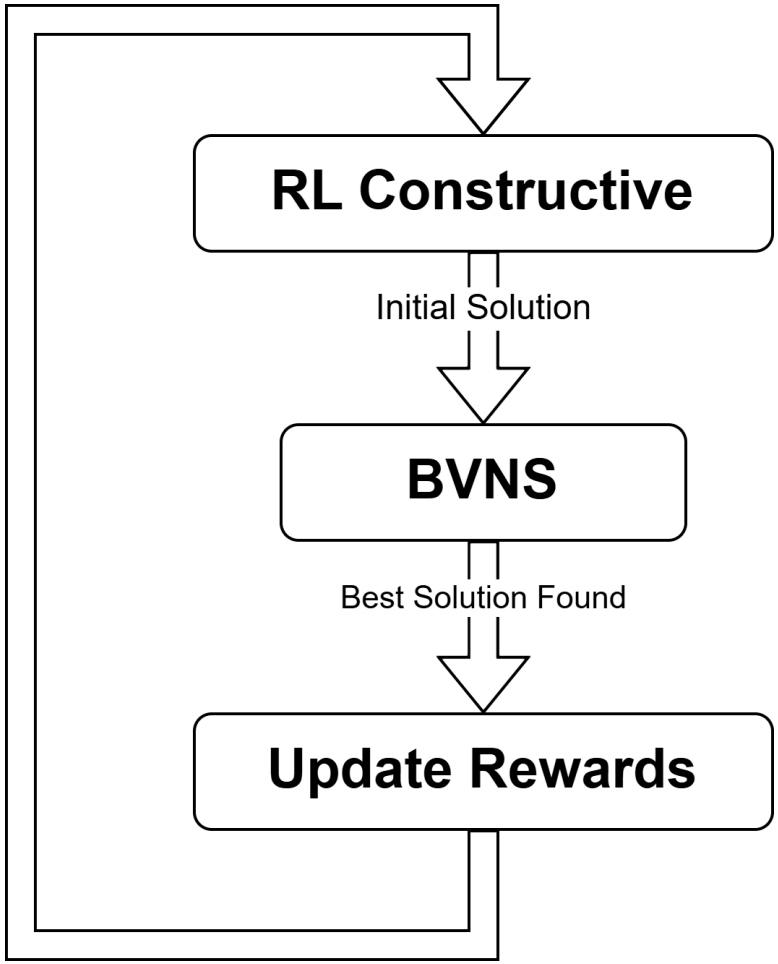


Constructive

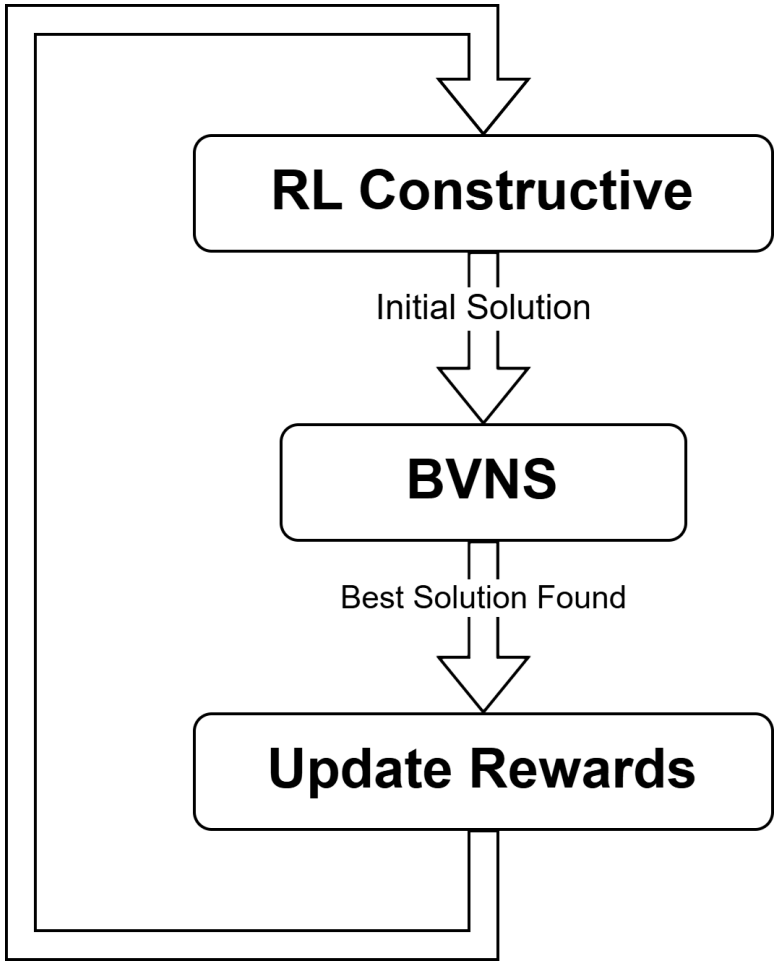
Random constructive



Reinforcement Learning Constructive



Reinforcement Learning Constructive



Reward Update

$$R_{k+1}(i) = R_k(i) + \Delta R(i)$$

where for $i = 1, \ldots, N$:

$$\Delta R(i) = \begin{cases} +1/N & \text{if } i \in S^* \\ -1/N & \text{if } i \notin S^* \end{cases}$$

Reinforcement Learning Constructive

Algorithm 2 `RLConstructive(N)`

1: $S' \leftarrow \emptyset$
2: **for** $i = 1$ **to** N **do**
3: **if** `nextRandom(0,1)` $<$ `Reward(i)` **then**
4: $S \leftarrow S \cup \{i\}$
5: **end if**
6: **end for**
7: **return** S'

Improvement method

Algorithm 3 `Improvement(S)`

1: $improvement \leftarrow$ `True`
2: **while** *improvement* **do**
3: $improvement \leftarrow$ `False`
4: $x \leftarrow \arg\min_{i \in S} \mathcal{F}(S \setminus \{i\})$
5: $S' \leftarrow S \setminus \{x\}$
6: **if** $\mathcal{F}(S') < \mathcal{F}(S)$ **then**
7: $improvement \leftarrow$ `True`
8: $S \leftarrow S'$
9: **end if**
10: **end while**
11: **return** S

Shake

Algorithm 4 `Shake(`S`,`k`)`

1: $C \leftarrow \{ a_i : a_i \in A \wedge a_i \notin S\}$
2: $C' \leftarrow$ `RandomSelection(C,k)`
3: $S \leftarrow S \cup C'$
4: **return** S

Results

- Programming languages: **Java 21**, **Python 3.8** and **Gurobi 11**.
- Server characteristics: AMD EPYC 7643 32-core with 32 GB of RAM.
- Instances: 15 CAP (66-1100 nodes), 87 K90 (502-914 nodes) and 700 G700 (3006-3906 nodes).
- Metrics:
  - $\mathcal{F}$: objective function value.
  - **Time (s)**: execution time in seconds.
  - **GAP (%)**: deviation from the exact value.
  - **# Opt**: total number of instances where the algorithm achieves the optimal solution.

Preliminary results

Random constructive

Local Search	K	$\mathcal{F}$	Time (s)	GAP (%)	# Opt
First Improvement	0.1	3 174 785.33	792.34	0.274205	58
	0.2	3 170 779.44	1007.23	0.233152	66
	0.3	3 171 017.58	1240.25	0.243865	65
	0.4	3 171 210.25	1502.95	0.244462	66
	0.5	3 170 757.61	1728.05	0.233776	65
Best Improvement	0.1	3 167 364.00	**93.92**	0.006122	133
	0.2	3 167 281.28	227.00	0.001724	148
	0.3	3 167 279.87	423.14	0.001707	**151**
	0.4	3 167 278.90	684.37	0.001595	**151**
	0.5	**3 167 276.27**	1017.10	**0.001489**	150

Table 1: Results of the BVNS algorithm on 20% of the dataset.

Preliminary results

Reinforcement learning constructive

Local Search	K	$\mathcal{F}$	Time (s)	GAP (%)	# Opt
First Improvement	0.1	3 177 592.01	939.49	0.497938	58
	0.2	3 172 158.29	1093.90	0.245668	65
	0.3	3 171 408.92	1316.49	0.239123	66
	0.4	3 174 379.58	1598.42	0.262672	63
	0.5	3 170 757.10	1853.15	0.232076	63
Best Improvement	0.1	3 169 410.98	**88.01**	0.210642	139
	0.2	3 167 271.68	224.93	0.001082	**151**
	0.3	**3 167 269.01**	424.91	**0.000836**	**151**
	0.4	3 167 276.50	683.34	0.001309	147
	0.5	3 167 271.70	1020.71	0.001015	149

Table 2: Results of the BVNS-RL algorithm on 20% of the dataset.

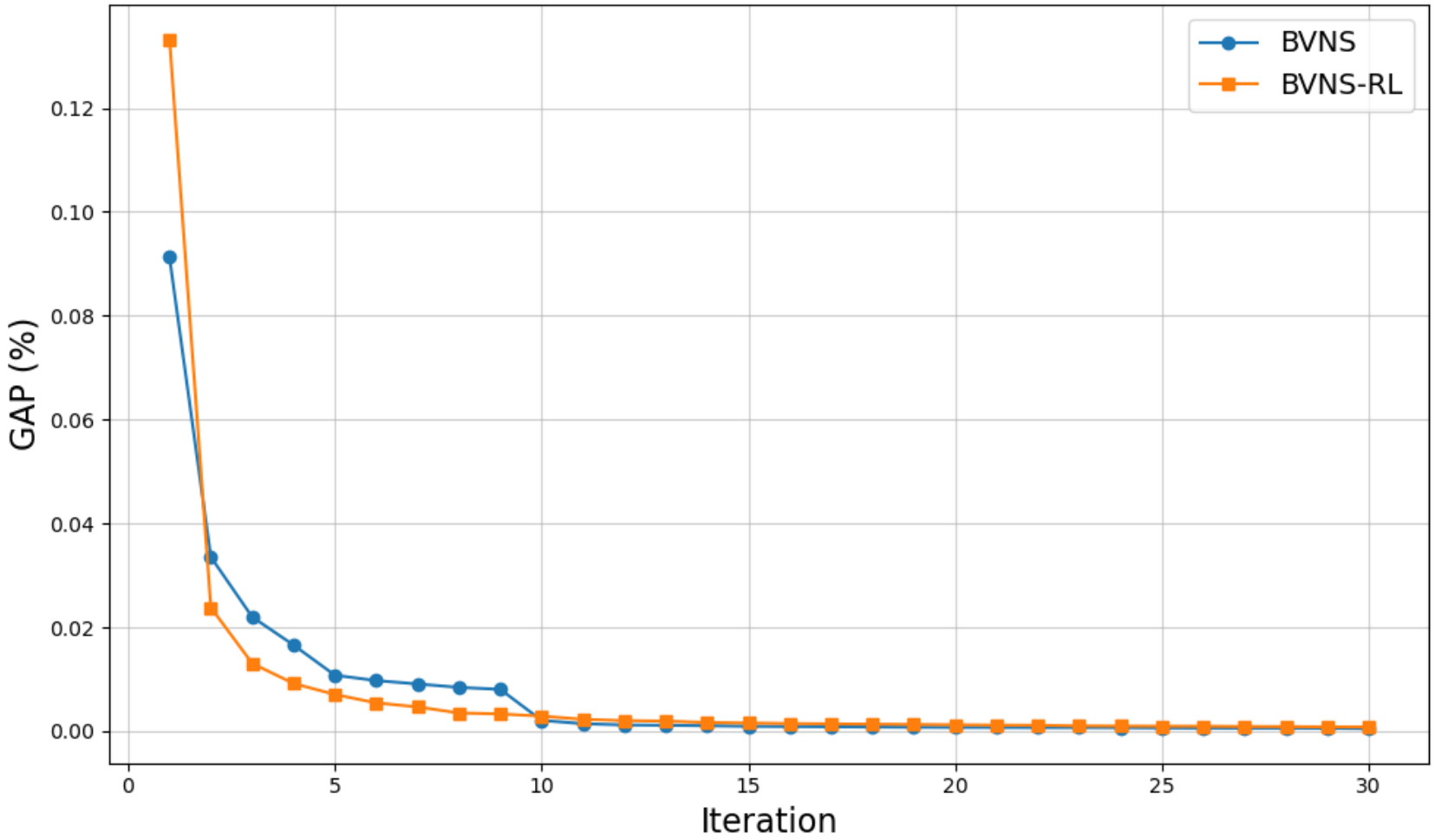
Final results

Instances	Algorithm	$\mathcal{F}$	Time (s)	GAP (%)	# Opt
Cap	EGTOA (Zhang)	3 502 545.59	258.931	0.00000	15
	BVNS	3 502 796.74	0.254	0.00218	14
	BVNS-RL	3 502 545.59	0.208	0.00000	15
	Exact Model	3 502 545.59	0.802	0.00000	15
K90	EGTOA (Zhang)	3 069 045.61	1 405.238	12.12858	0
	BVNS	2 765 428.69	3.825	0.00568	71
	BVNS-RL	2 765 390.11	3.508	0.00374	73
	Exact Model	2 765 298.57	2.642	0.00000	87
G700	EGTOA (Zhang)	27 202 096.73	3 597.980	174.81629	23
	BVNS	2 943 809.46	248.588	0.00018	666
	BVNS-RL	2 943 809.24	239.329	0.00010	678
	Exact Model	2 943 809.00	62.861	0.00000	700
Aggregated	EGTOA (Zhang)	24 140 913.79	1 236.663	153.89849	38
	BVNS	2 934 913.80	217.392	0.00081	751
	BVNS-RL	2 934 904.73	209.275	0.00050	766
	Exact Model	**2 934 894.58**	**55.168**	**0.000000**	**802**

Table 3: Final results of the EGTOA, BVNS, and Exact algorithms.

Results

GAP Convergence



Conclusions

Conclusions:

- Two constructive heuristics were explored with the algorithm.
- **BVNS** consistently achieves competitive solutions, comparable to state-of-the-art.
- **Reinforcement Learning (RL)** showed an interesting fast progression towards better solutions, improving the performance of the random constructive.

Future work:

- Speed up the objective function calculation
- Test different machine learning methods to guide the solution search.

A Machine Learning enhanced Variable Neighborhood Search approach for the Uncapacitated Facility Location problem

International Conference on Variable Neighborhood Search 2025 (ICVNS)

Lucas Martín-García (lucas.martin@urjc.es)
Isaac Lozano-Osorio (isaac.lozano@urjc.es)
J. Manuel Colmenar (josemanuel.colmenar@urjc.es)
Belén Melián-Batista (mbmelian@ull.es)

Universidad Rey Juan Carlos

grafo RESEARCH

Universidad de La Laguna

INTERNATIONAL CONFERENCE ON VARIABLE NEIGHBORHOOD SEARCH VNS 2025

MINISTERIO DE CIENCIA E INNOVACIÓN

AGENCIA ESTATAL DE INVESTIGACIÓN