Politecnico di Torino

Master’s degree in Engineering and Management

Analysis and algorithms to solve the bilevel assignment problem

Candidate:
Francisco Brogiolo

Supervisors:
Federico Della Croce di Dojola
Rosario Scatamacchia

Outline

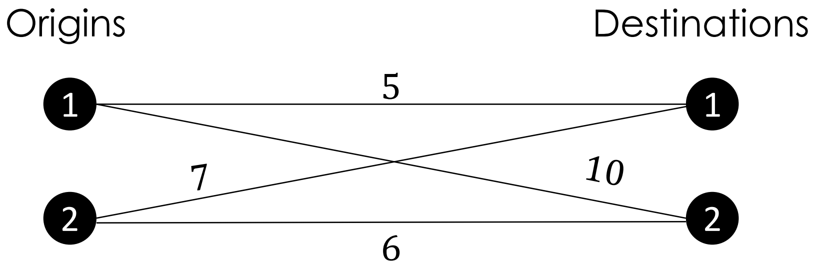
1. Introduction
2. Exact and relaxed models
3. Heuristics
4. Matheuristics
5. Results and conclusions

1. Introduction

1.1Assignment problem

Given n origins and n destinations, with their corresponding distances or costs, assign each origin to one destination **minimizing the total distance**.

n=2



$$\begin{bmatrix} \mathbf{5} & 10 \\ 7 & \mathbf{6} \end{bmatrix} \qquad Total\ cost = 5 + 6 = 11$$

Example of applications:

- Assign taxi drivers to passangers
- …

1.2 Bilevel assignment problem

Given *n* origins and *n* destinations

1. The **leader** selects *k* origins and *k* destinations (*k*<*n*)
2. The **follower** solves the assignment problem

The follower's objective function is $$g_{follower} = min \sum_{j=1}^{n} \sum_{i=1}^{n} c_{ij} x_{ij}$$

While the leader tries to **maximize** the solution of the follower, then:

$$g_{leader} = max\,(g_{follower})$$

→ The considered bilevel assignment problem has not been solved by the scientific community of operational research yet.

1.2 Bilevel assignment problem

Feasible solution Example for n=4 and k=2

Given *4* origins and *4* destinations

$$\begin{bmatrix} 5 & 10 & 8 & 3 \\ 7 & 6 & 2 & 13 \\ 15 & 4 & 1 & 9 \\ 12 & 7 & 8 & 10 \end{bmatrix}$$

The leader selects 2 origins and 2 destinations

$$\begin{bmatrix} \mathbf{5} & 10 \\ 7 & \mathbf{6} \end{bmatrix}$$

$$Leader's\ selection = Origins\ 1\ and\ 2; Destinatinons\ 1\ and\ 2$$

The follower solves the assignment problem

$$Follower's\ selection = [5, 6]$$

$$Total\ cost = 5 + 6 = 11$$

1.2 Bilevel assignment problem

Optimal solution Example for n=4 and k=2

$$\begin{bmatrix} 5 & 10 & 8 & 3 \\ 7 & 6 & 2 & 13 \\ 15 & 4 & 1 & 9 \\ 12 & 7 & 8 & 10 \end{bmatrix}$$

Given *4* origins and *4* destinations

The leader selects 2 origins and 2 destinations

$$\begin{bmatrix} 15 & \mathbf{9} \\ \mathbf{12} & 10 \end{bmatrix}$$

$$Leader's\ selection = Origins\ 3\ and\ 4; Destinatinons\ 1\ and\ 4$$

The follower solves the assignment problem

$$Follower's\ selection = [12, 9]$$

$$Total\ cost = 12 + 9 = 21$$

2. Exact and relaxed models

2. Exact and relaxed models

Primal formulation	Dual formulation	Relaxation of the dual formulation
$g_{leader} = max(min \sum_{j=1}^{n} \sum_{i=1}^{n} c_{ij} x_{ij})$	$g_{leader} = max \sum_{i=1}^{n} u_i + \sum_{j=1}^{n} v_j$	$u_i, v_j \geq 0 \ \forall \text{ i, j}$

Relaxed model advantages and disadvantage:

Advantages	Disadvantage
Faster for small instances	Does not always find the optimal solution
Better solutions for big instances	

→ **Relaxed model is preferred**

The solver IBM ILOG CPLEX Optimization Studio 20.1 was used.
The code was written in Python 3, and the API to connect Python with CPLEX is *docplex*.

3. Heuristic approach

3.1.Algorithms

Greedy algorithm → Local search → Iterated local search

Greedy algorithm

- ✓ Very fast solution
- ✓ Better than a random solution

Local search

1. Greedy algorithm
2. Neighbor generation
3. Acceptance criterion
4. Search rule
5. Stopping criterion

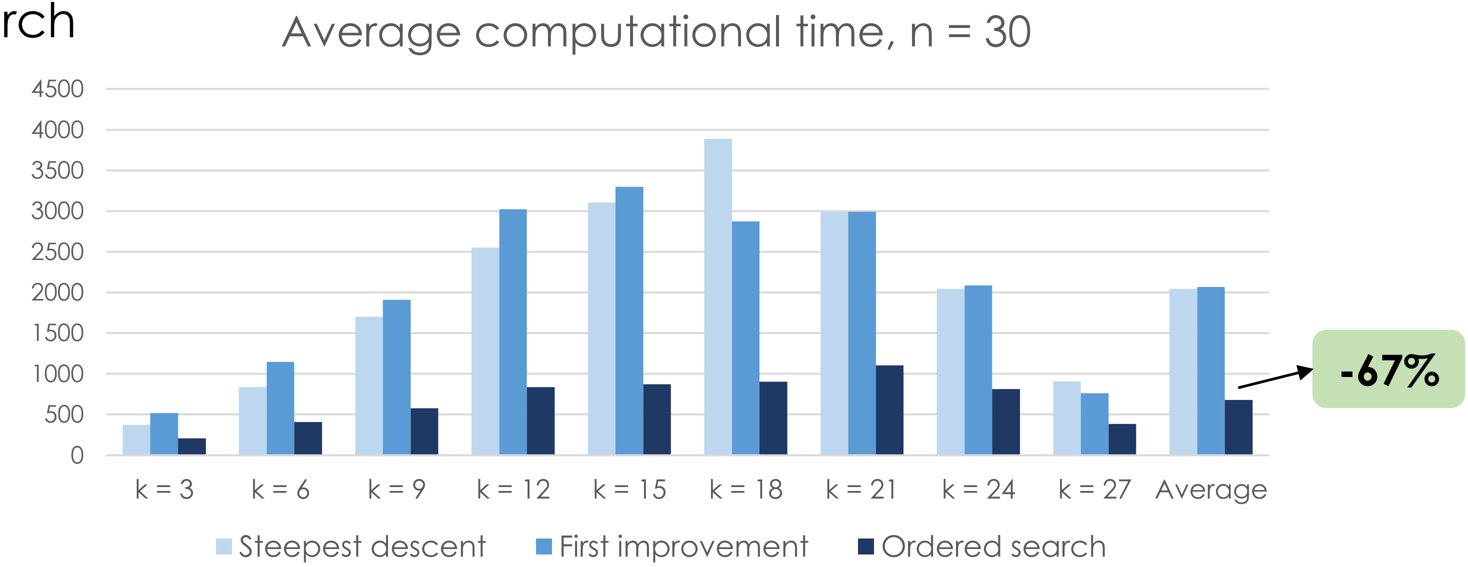
Iterated local search

1. Local search
2. Perturbation
3. Intensification (Local Search)
4. Acceptance criterion
5. Stopping criterion

3.1.Algorithms

Search rules of local search:

- ✓ Steepest descent
- ✓ First improvement
- ✓ Ordered search



Time metric considered: number of times in which the AP is solved on average of 10 instances

3.2 Acceptance criterion

The acceptance criterion of local search algorithm takes $O(k^3)$

But…

We can generate upper bounds in $O(k^2)$, and:

If: $Upper\ bound\ \leq\ Current\ best\ solution$

Then we **reject** the neighbor

$O(k^3) \Rightarrow O(k^2)$

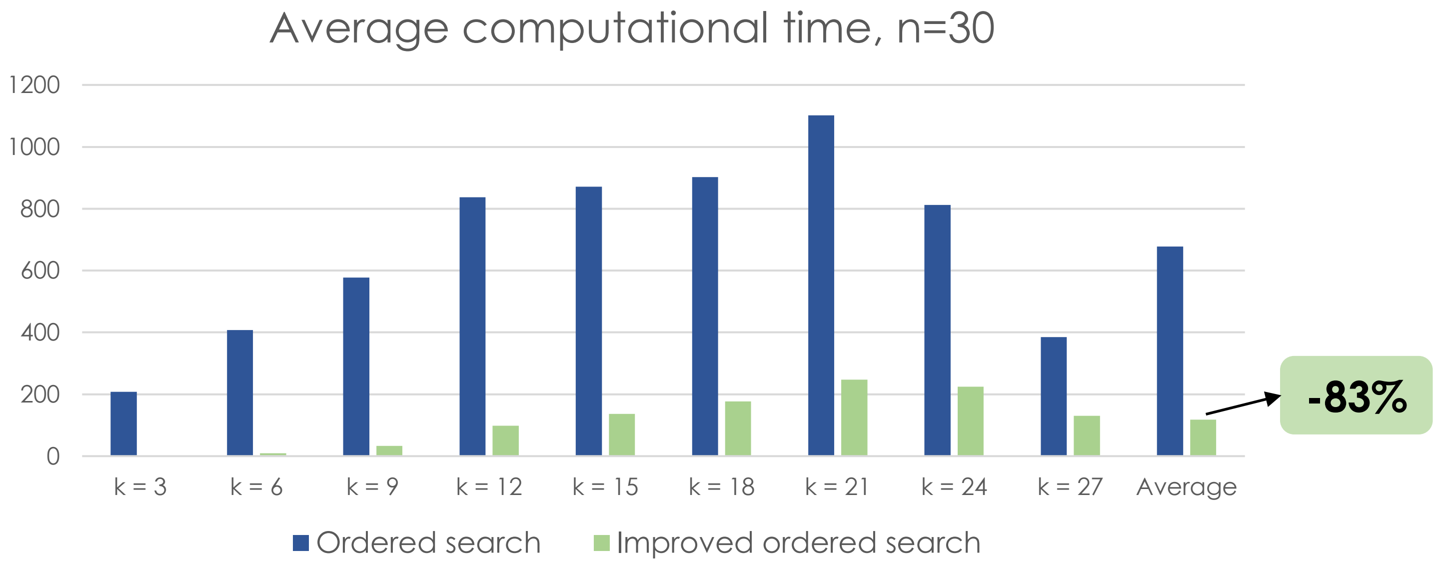
The computational time to evaluate that neighbor decreases

3.2 Acceptance criterion

¿How to construct k^2 upper bounds?

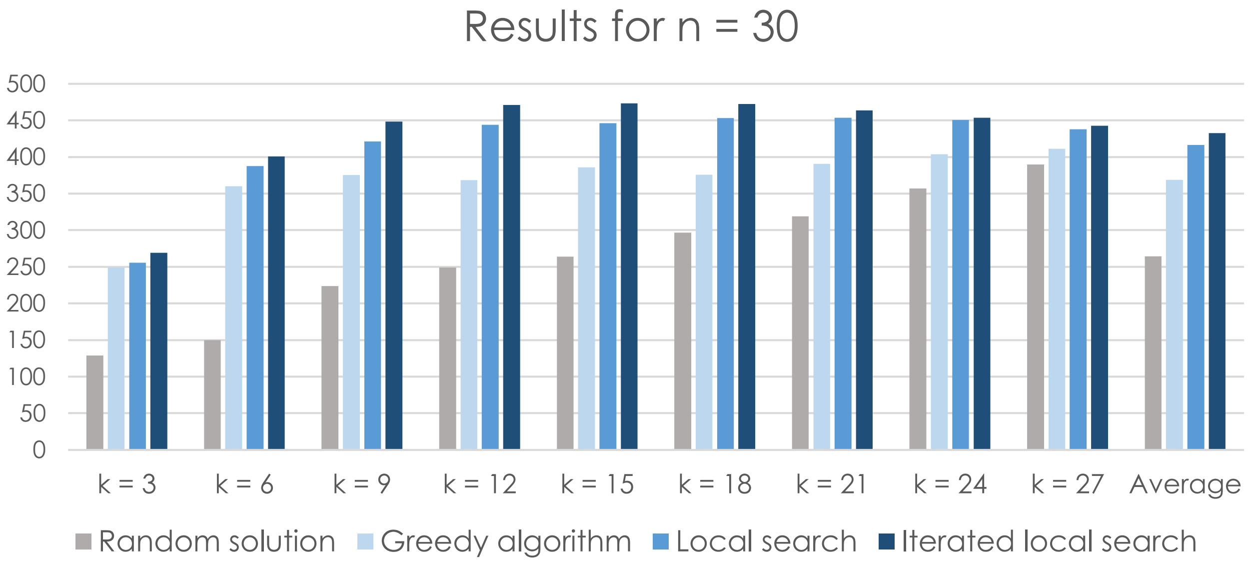
✓ Two nested **for loops**, each one from 1 to k

```
For i in [1,2…k]:
    for j in [1,2..k]:
        generate upper bound
        check if the neighbor can be rejected
```



Time metric considered: number of times in which the AP is solved on average of 10 instances

3.3 Results



Iterated local search improves **82%** of the solutions of local search

(Time limit: 1 minute)

Results represent the average of 10 instances

The algorithms were implemented in Python 3

4. Matheuristic approach

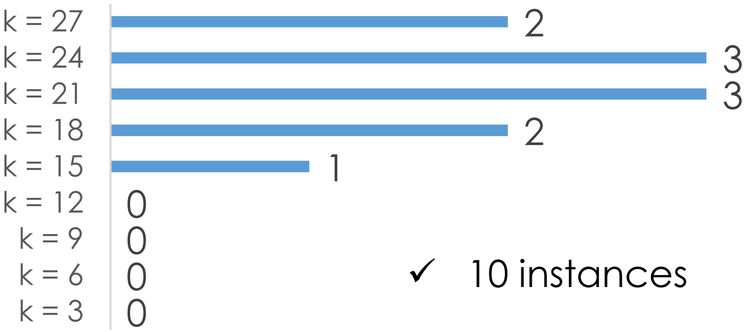
4. Matheuristic approach

Iterated local search → Matheuristic

Matheuristic: hybrid approach, it uses exact and heuristics

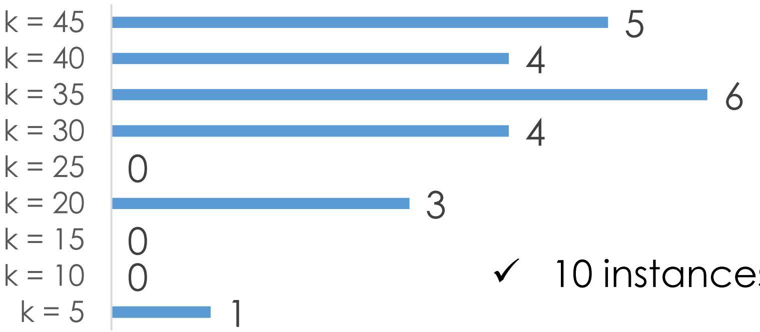
Same time limit

N° of improved solutions, n = 30



✓ 10 instances

N° of improved solutions, n = 50



✓ 10 instances

$k \leq 0.5n$ ⇨ Small/null number of improvements

$k > 0.5n$ ⇨ Medium/small number of improvements

5. Results and conclusion

5.1 Results

MIP: Relaxed dual formulation
ILS: Improved ordered search
Math.: Matheuristic algorithm

- ✓ 10 instances for each n
- ✓ Values uniformly distributed between 10 and 99
- ✓ ks = 0.1n, 0.2n…0.9n
- ✓ 90 tests per each n

	n	~~MIP~~	ILS	Math.
# Best	30	~~33~~	**78**	76
# Best	50	~~13~~	**61**	55

The solver IBM ILOG CPLEX Optimization Studio 20.1 was used.
The code was written in Python 3, and the API to connect Python with CPLEX is *docplex*.

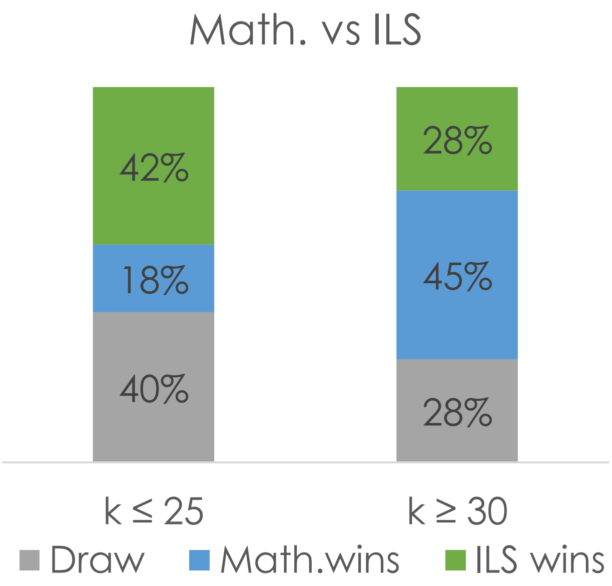
5.2 Conclusion

- ✓ MIP is outperformed by ILS and Math.
- ✓ For smaller ks ILS is preferred.
- ✓ For bigger ks Math is slightly preferred.

Future development:

→ Study the complexity status of the bilevel assignment problem.

Pairwise comparison, n=50



(Similar results for n=30)

Thank you for your attention!