Large Scale Optimization

Emmanouil Zachariadis, Assistant Professor
Department of Management Science and Technology, Athens University of Economics and Business
E: ezach@aueb.gr
A: Evelpidon 47A and Lefkados 33 street, 9th floor, Room 906, 11362, Athens, Greece
T: +30-210-8203 674

LOCAL SEARCH

- In the context of Local Search algorithms, the procedure iteratively transitions from one solution to another via specific modifications to the current solution structure.
- This transition is named *Move*
- Therefore, Local Search algorithms DO NOT construct a solution (unlike a greedy constructive algorithm)
- Local Search algorithms modify a given solution via the Moves
- Objective: Transition to a better solution (Improvement)
- A Move is classified to a specific *Move type*.
- The term *Move Type* is also referred as *Local Search Operator* or Neighborhood Type.

LOCAL SEARCH

- In each local search algorithm iteration, the incumbent (current) solution is denoted as $s \in S$, with S being the set of all feasible solutions of the examined problem.
- The *Move Type* defines a *Set of Moves* that can be applied to the incumbent solution s and modify it to a new solution.
- A Move Type t connects a solution s (via the moves that defines) with a set of new solutions $N_t(s) \in S$
- The set of solutions $N_t(s)$ is named Neighborhood of solution s by move type t

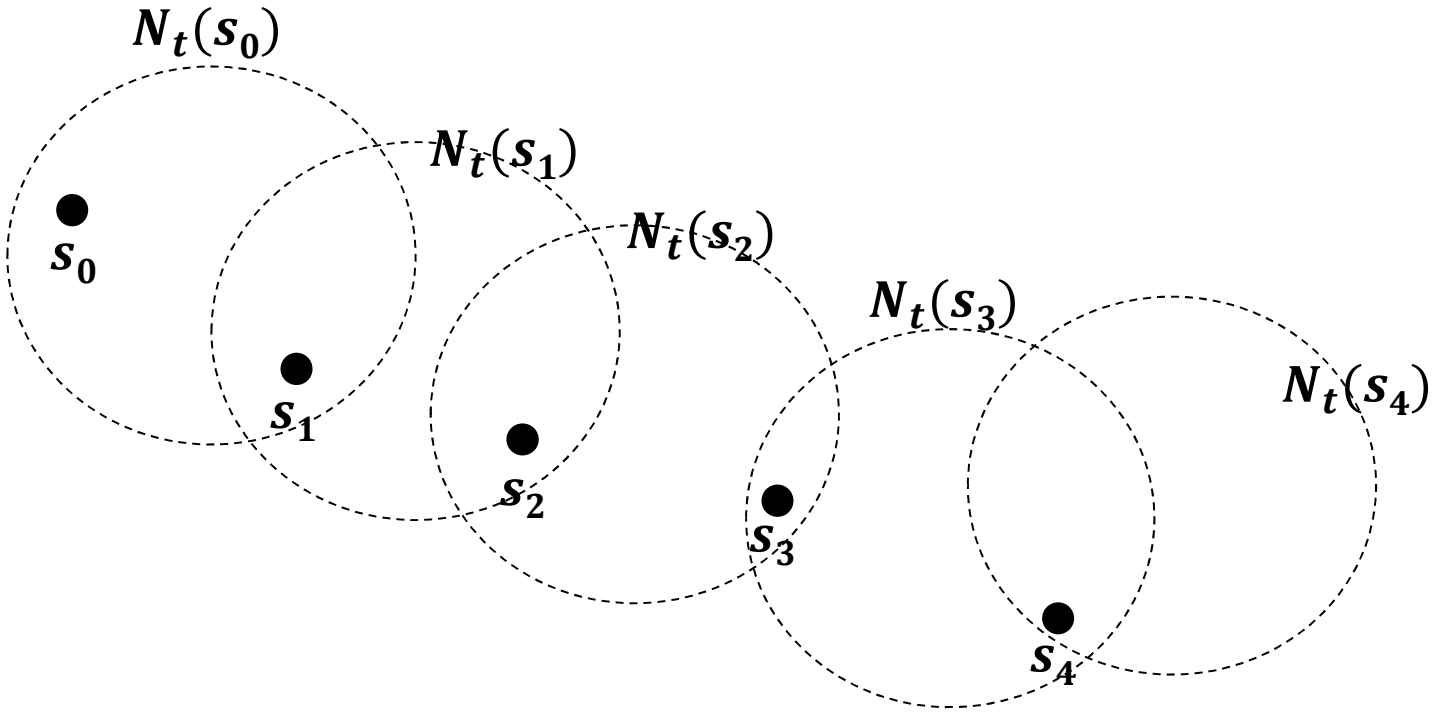
LOCAL SEARCH

- Every solution $s' \in N_t(s)$ is called neighbor solution of the incumbent solution s, with respect to a specific Move type t.
- Obviously, the set $N_t(s)$ is a subset of the set of all feasible solutions S of the examined problem.

LOCAL SEARCH

A solution s comprises the local minimum with respect to move type t, iff

$$z(s) \leq z(s'), \forall s' \in N_t(s)$$



The basic scheme of a local search algorithm with move type t is terminated in the first local minimum that is visited: $z(s_4) \leq z(s'), \forall s' \in N_t(s_4)$

Basic Local Search Scheme

Input: *Initial solution* $s_{initial}$

Set $s_{initial}$ as incumbent solution:

$i \leftarrow 0$

$s_i \leftarrow s_{initial}$

Repeat

Select one the neighboring solutions of the current solution: $s_i' \in N_t(s_i)$

Implement the move towards this solution: $\{ i \leftarrow i + 1, s_i \leftarrow s_i' \}$

Until *the termination criterion is satisfied/activated*

Basic Local Search Scheme

- The local search is an iterative algorithm that transitions from one solution to another within the problem solution space (polytope).
- In every iteration i, we transition from one solution s_i to another solution s_i'
- Solution s_i' is one the neighbor solutions of s_i, i.e., $s_i' \in N_t(s_i)$
- The neighbor solutions of a solution are defined by the move type N_t that is applied

  Question: Which neighbor solution of solution s_i should be selected as incumbent s_i' for the next iteration?

  Answer: It depends on the selection criterion that is used.

Basic Local Search Scheme

- Neighbor solution selection criterion example:

  Selection of best quality neighbor solution

- In every algorithm iteration i:
  - All neighbor solutions of s_i
  - s_i' is the neighbor solution with the selection criterion minimum value

  If $z(s_i') < z(s_i)$ (i.e., solution s_i' is better than s_i)

  Then the algorithm transitions to solution s_i'

  Else (if solution s_i' is of the same or lowest quality than $s_i \Leftrightarrow s_i$ is a local minimum) the algorithm terminates and s_i is returned as final solution

- This structure of the Local Search algorithm is known as Basic Local Search Scheme

Basic Local Search Scheme

Input: *Initial solution* $s_{initial}$

Set $s_{initial}$ as incumbent solution:

$i \leftarrow 0$

$s_i \leftarrow s_{initial}$

Repeat

Select the best of the neighbor solutions of incumbent solution: $s_i' \in N_t(s_i)$

if $z(s_i') < z(s_i)$

Implement the move towards this solution $s_i' : \{ i \leftarrow i + 1, s_i \leftarrow s_i' \}$

else

return s_i

Move Type Design

- Two different Move Types may produce the same or different number of neighbor solutions, when applied to the same initial solution
- The neighbor solutions produced by a Move type are normally different compared to the neighbor solution produced by another move type, even if both move types are applied to the same solution
- Additionally, in order to effectively search through the solution space, i.e., to select effective subsets of solutions (Neighborhoods), the used move types aim to optimize the objective function of the problem and not just modify the solution structure.

Move Type Design

- The size of the neighborhood of the incumbent solution is essential.
- The computational time required for finding the best neighbor solution (like in the basic local search scheme) increases with the number of neighboring solution produced
- The design of large neighborhoods with many solutions (Rich Neighborhoods) increases the computational time required for evaluating all neighbor solutions.
- However, Rich Neighborhoods increase the probability of finding a neighboring solution that improves the incumbent
- Increased ability to find good quality solutions

Steepest Descent Local Search – "1-1 EXCHANGE" Move Type

- Let a TSP with $n = 6$ clients.
- The distance matrix is given below (Salesman starts at node 0 and distances are symmetrical) :

	0	1	2	3	4	5	6
0	0	22.52	74.35	49.82	46.73	31.17	36.84
1		0	70.86	44.88	79.58	66.99	65.95
2			0	32.48	38.32	42.61	22.73
3				0	73.68	58.20	70.21
4					0	54.39	52.49
5						0	24.67
6							0

Steepest Descent Local Search – "1-1 EXCHANG" Move Type

- Let a greedy constructive algorithms that constructs an initial solution $s_{init} = \{0,3,6,2,5,1,4,0\}$ με κόστος $z(s_{init}) = 378.67$
- We apply the basic local search scheme to solve the problem
- 1-1 exchange Move type: this move type swaps the positions of two nodes i and j of the solutions, e.g. from solution

$$s = (0, \dots, a, i, b, \dots, m, j, n, .., 0)$$

We transitions to solution

$$s' = (0, \dots, a, j, b, \dots, m, i, n, .., 0)$$

- Execute two iterations of the algorithms and report the objective of the best solution found.

Steepest Descent Local Search – "1-1 EXCHANGE" Move Type

- Neighborhood $N(s_0)$:
  - {0,6,3,2,5,1,4,0}, z=375.44
  - {0,2,6,3,5,1,4,0}, z=418.79
  - {0,5,6,2,3,1,4,0}, z=282.24
  - {0,1,6,2,5,3,4,0}, z=332.42
  - {0,4,6,2,5,1,3,0}, z=326.25
  - {0,3,2,6,5,1,4,0}, z=323
  - {0,3,5,2,6,1,4,0}, z=365.62
  - {0,3,1,2,5,6,4,0}, z=332.06
  - {0,3,4,2,5,1,6,0}, z=374.21
  - {0,3,6,5,2,1,4,0}, z=384.48
  - {0,3,6,1,5,2,4,0}, z=380.63
  - {0,3,6,4,5,1,2,0}, z=439.11
  - {0,3,6,2,1,5,4,0}, z=381.73
  - {0,3,6,2,4,1,5,0}, z=358.82
  - {0,3,6,2,5,4,1,0}, z=341.86

$$s_0 = \{0,3,6,2,5,1,4,0\}$$

The best solution is $s_0' = \{0,5,6,2,3,1,4,0\}$ that is generated by swapping nodes 3 and 5

Therefore, the new incumbent solution is $(s_1 \leftarrow s_0')$

$s_1 = \{0,5,6,2,3,1,4,0\}$ with $z(s_1) = 282.24$

Steepest Descent Local Search – "1-1 EXCHANGE" Move Type

2nd Iteration: Neighborhood $N(s_1)$:

- {0,6,5,2,3,1,4,0}, z=307.79
- {0,2,6,5,3,1,4,0}, z=351.14
- {0,3,6,2,5,1,4,0} z=378.67
- {0,1,6,2,3,5,4,0}, z=303
- {0,4,6,2,3,1,5,0}, z=297.47
- {0,5,2,6,3,1,4,0}, z=337.91
- {0,5,3,2,6,1,4,0}, z=336.84
- {0,5,1,2,3,6,4,0}, z=370.93
- {0,5,4,2,3,1,6,0}, z=304.03
- {0,5,6,3,2,1,4,0}, z=355.7
- {0,5,6,1,3,2,4,0}, z=284.2
- {0,5,6,4,3,1,2,0}, z=372.1
- {0,5,6,2,1,3,4,0}, z=314.72
- {0,5,6,2,4,1,3,0}, z=291.17
- {0,5,6,2,3,4,1,0}, z=286.83

$s_1 = \{0,5,6,2,3,1,4,0\}$ with $z(s_1) = 282.24$

The best solution is solution $s_1' = \{0,5,6,1,3,2,4,0\}$ that is generated by swapping nodes 2 and 1, with objective $z(s_1') = 284.20$

Since $z(s_1') > z(s_1)$, solution s_1 is a local minimum and the algorithm terminates

THE QUADRATIC ASSIGNMENT PROBLEM - QAP

- How can the construction of clinics to hospital be decided?
- Which is the best location of warehouses in a shopping mall in order to minimize the cost of moving products?
- Which is the optimal way of connecting the numerous materials on a computer motherboard?
- All these problems seems different by the share the feature that are modelled as Quadratic Assignment Problems.

THE QUADRATIC ASSIGNMENT PROBLEM - QAP

- The QAP is one of the most interesting and challenging computational problems in the field of combinatorial optimization
- Initially, it was presented in 1957 by Tjalling C. Koopmans and Martin Beckman, in an attempt to model the facilities location problem.
- Many scientist, mathematicians, computer scientist, operational research analysts and economists have been using QAP to model several optimization problems and support decision making.

THE QUADRATIC ASSIGNMENT PROBLEM - QAP

- This category of problems refer to the way of assigning employees or machines to specific jobs or locations, while taking into consideration the flows between them.
- If the size of the problem is n, all possible solutions (assignments) are $n!$.
- For example, if we have 10 warehouses than need to be built to 10 locations of a region, there are $10! = 3{,}6 \cdot 10^6$ ways do so. In other words, there are $10!$ feasible problem solutions.
- Due to the complexity of the problem, usually efficient approximation algorithms are applied in order to ensure good quality solution in limited computational time.

THE QUADRATIC ASSIGNMENT PROBLEM - QAP

- Therefore, we observe that QAP is an assignment problem where the elements of set A need to be assigned to the elements of set B.
- In the basic for, these to sets have the exact same number and each element of A is assigned to exactly one element of B.
- Bear in mind that each element of set A (or B) is related/affected by each other element of set A (or B).
- This relationships is usually the flow between the elements of set A and element of set B.
- This flow can be either a physical flow (e.g. product flow from a warehouse to another) or an information flow (e.g. weight and frequency of cooperation between two employees).

THE QUADRATIC ASSIGNMENT PROBLEM - QAP

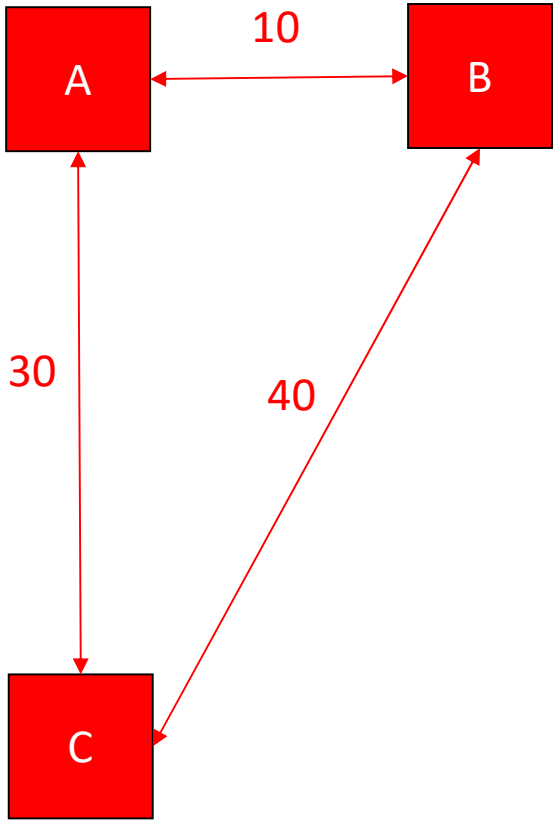
- Facility location example
  - The elements of set A are warehouses of distribution centers (facilities)
  - The elements of set B are the locations that these warehouse can be constructed on
  - Flows among elements of A (Facilities): Frequency of trips
  - Flows among elements of B (Locations): Distances
- The objective function of a QAP solutions is the sum of the produce of the flows of set A and set B, given the assignment of this solution, e.g.,

  Cost = Trip frequency (Flow A) * Distance (Flow B)

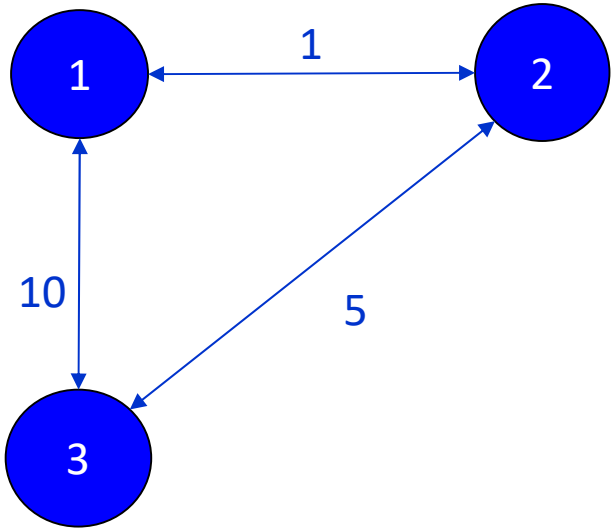
  Distance/Day= (Trips number/day) * (Distance/Trip)
- The goal is to assign the elements of set A (facilities) to the elements of set B (locations) in way such that the cost of the flow is minimized.

THE QUADRATIC ASSIGNMENT PROBLEM - QAP

Let a QAP with symmetrical data as below:

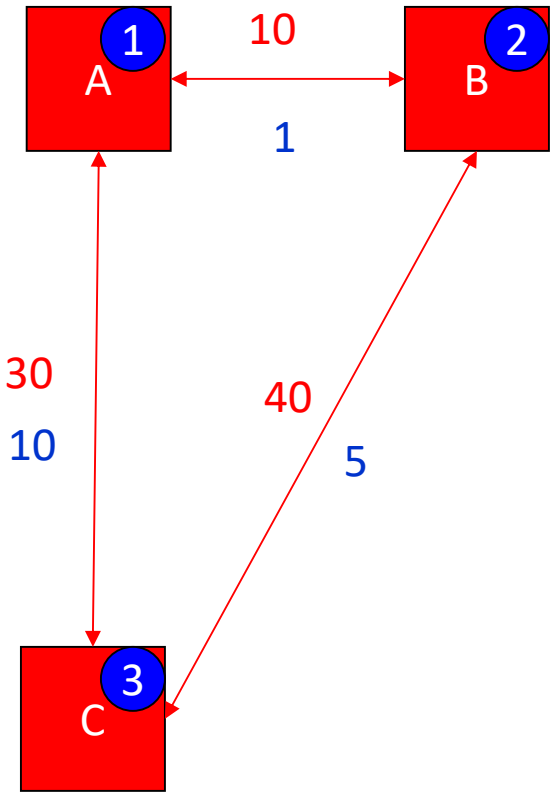


Locations and distances

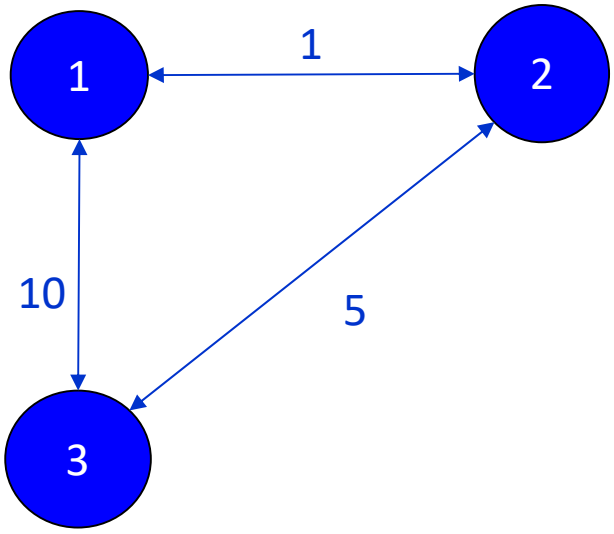


Facilities and flows

THE QUADRATIC ASSIGNMENT PROBLEM - QAP



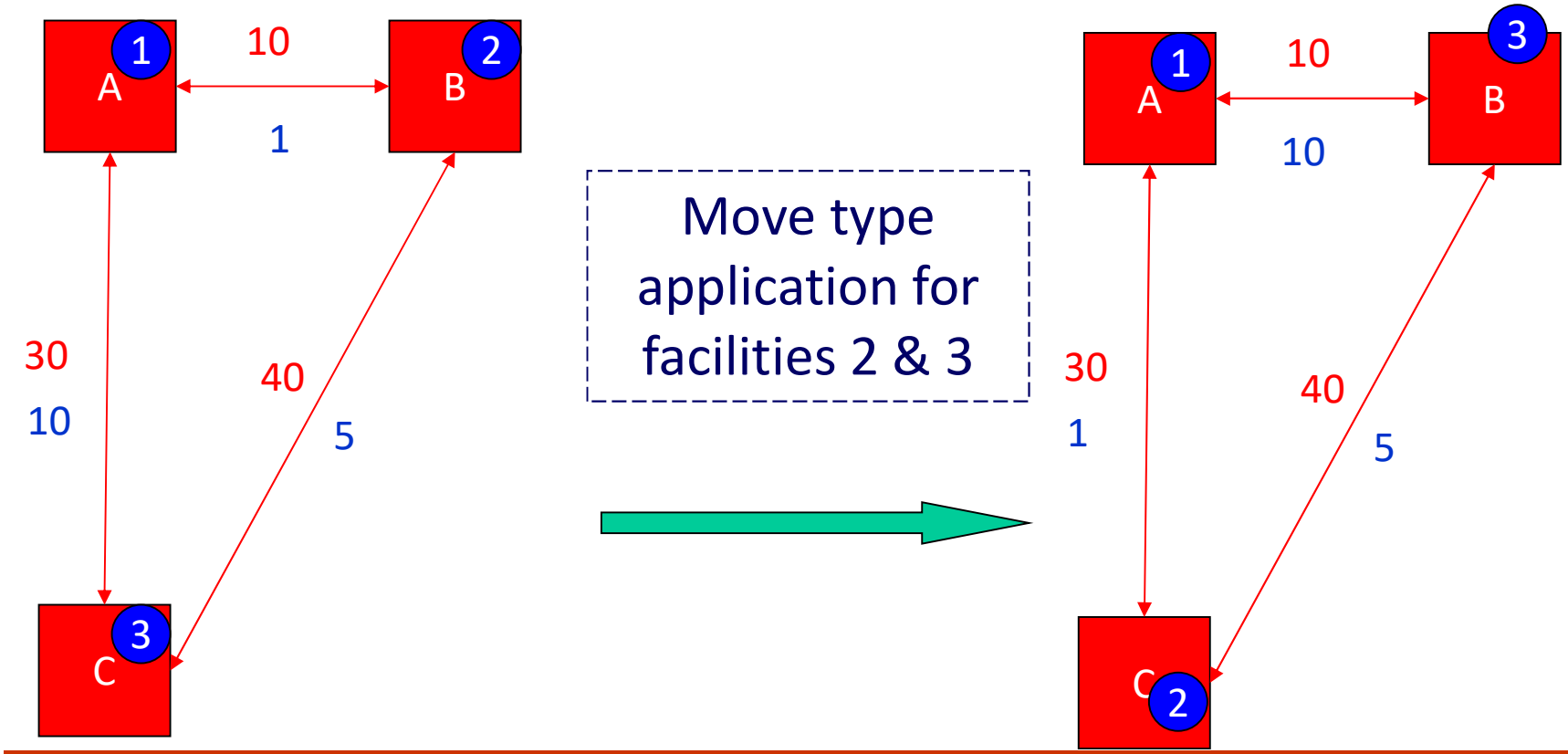
Let a solution (1A,2B,3C) with cost
c(s)= [(dAB*f12)+(dAC*f13)+(dBA*f21)+(dBC*f23) +
(dCA*f31) + (dCB*f32)] = [(10 × 1)+ (30 × 10) +
(10 × 1)+ (40 × 5)+ (30 × 10) + (40 × 5)] = 1020



Facilities and flows

QAP – Move implementation

A move type for (1A,2B,3C) : Swap two facilities i and j, for each i and j.



Application: Storage facilities allocation

- A company needs to install large logistic centers in Greece. The first logistic center that will be constructed will contain 4 warehouse units (e.g. warehouses 1, 2, 3, 4).
- Due to the size of the region, each of the warehouses needs to be installed in one of the locations A, B, C, D.
- Every warehouse is related to the others in terms of the frequency of transactions with the other warehouses.
- Similarly, each location is related to the other location in terms of distance.

Application: Storage facilities allocation

- Flows between locations (Distances)
- Given that the distances are symmetrical, i.e., that $d_{AB} = d_{BA}$ holds, where d is the distance between two locations, the cost (distance) matrix among the location is the following:

	A	B	C	D
A	0	20.62	56.57	11.18
B	20.62	0	40.31	25.50
C	56.57	40.31	0	54.08
D	11.18	25.50	54.08	0

Distances among locations A,B,C,D

Application: Storage facilities allocation

- Flow between facilities (Cost/ distance unit)
- Below the cost matrix per distance unit between the warehouses is presented. These costs are formed according to the flows of products between each warehouse and the other ones.

	1	2	3	4
1	0	8.5	1.5	5
2	8.5	0	0.5	0.5
3	1.5	0.5	0	2
4	5	0.5	2	0

Application: Storage facilities allocation

- The goal of the company is to find the optimal locations for the warehouses in the specific region, such that the total cost of the product flows between the warehouses built in the specific locations are minimized.
- Apply the basic local search scheme, with the following initial solution:

$$(A1, B2, C4, D3)$$

- The move type that will be applied swaps two locations, where two warehouses are assigned.

Application: Storage facilities allocation

- Let us denote a solution as follows:
- Solutions (i, j, k, l) describes that warehouse i will be placed in location A, warehouse j will be placed in location B, warehouse k will be placed in location B, and finally warehouse l will be placed in location D
- Therefore, the initial solution is rewritten:

$$z(s_{init}) = z(\{1, 2, 4, 3\}) =$$

$$= [(d_{AB} \cdot f_{12}) + (d_{AC} \cdot f_{14}) + (d_{AD} \cdot f_{13}) + (d_{BC} \cdot f_{24}) + (d_{BD} \cdot f_{23}) + (d_{CD} \cdot f_{43})] \cdot 2 =$$

$$= 1231.91$$

Application: Storage facilities allocation - Solution

1st Local Search Iteration

$s_0 = \{1,2,4,3\}$ with $z(s_0) = 1231.91$

$N(s_0) = \{(2,1,4,3), (4,2,1,3), (3,2,4,1), (1,4,2,3), (1,3,4,2), (1,2,3,4)\}$

After the cost evaluation of all neighbor solutions produced by the given move type, we find that the best neighbor solution of solution s_0 is solution $s_0' = \{1,2,3,4\}$ with $z(s_0') = 914.18$

Therefore, $s_1 \leftarrow \{1,2,3,4\}$

Application: Storage facilities allocation - Solution

2nd Local Search Iteration

$s_1 = \{1,2,3,4\}$ with $z(s_1) = 914.18$

$N(s_1) = \{(2,1,3,4), (3,2,1,4), (4,2,3,1), (1,3,2,4), (1,4,3,2), (1,2,4,3)\}$

After the cost evaluation of all neighbor solutions produced by the given move type, we find that the best neighbor solution of s_1 is solution $s_1' = \{1,4,3,2\}$ with $z(s_1') = 806.79$

Therefore, $s_2 \leftarrow \{1,4,3,2\}$

Application: Storage facilities allocation - Solution

3rd Local Search Iteration

$s_2 = \{1,4,3,2\}, z(s_2) = 806.79$

$N(s_2) = \{(4,1,3,2), (3,4,1,2), (2,4,3,1), (1,3,4,2), (1,2,3,4), (1,4,2,3)\}$

After the cost evaluation of all neighbor solutions produced by the given move type, we find that the best neighbor solution of s_2 is solution $s_2' = \{2,4,3,1\}$ with $\mathrm{z}(s_2') = 845.73$

Therefore, $\mathrm{z}(s_2') = 845.73 > \mathrm{z}(s_2') = 806.79$

The algorithm is terminated with final solution:

$$\{1,4,3,2\}$$

Corresponding to the assignment:

$$\{\mathrm{A1}, \mathrm{B4}, \mathrm{C3}, \mathrm{D2}\}$$