CIVIL-557

Decision Aid Methodologies In Transportation

Lectures 7 & 8: Network Design

Nikola Obrenović

Transport and Mobility Laboratory TRANSP-OR
École Polytechnique Fédérale de Lausanne EPFL

TRANSP-OR

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Case Study

- **Network Design for CFF Cargo**

TRANSP-OR

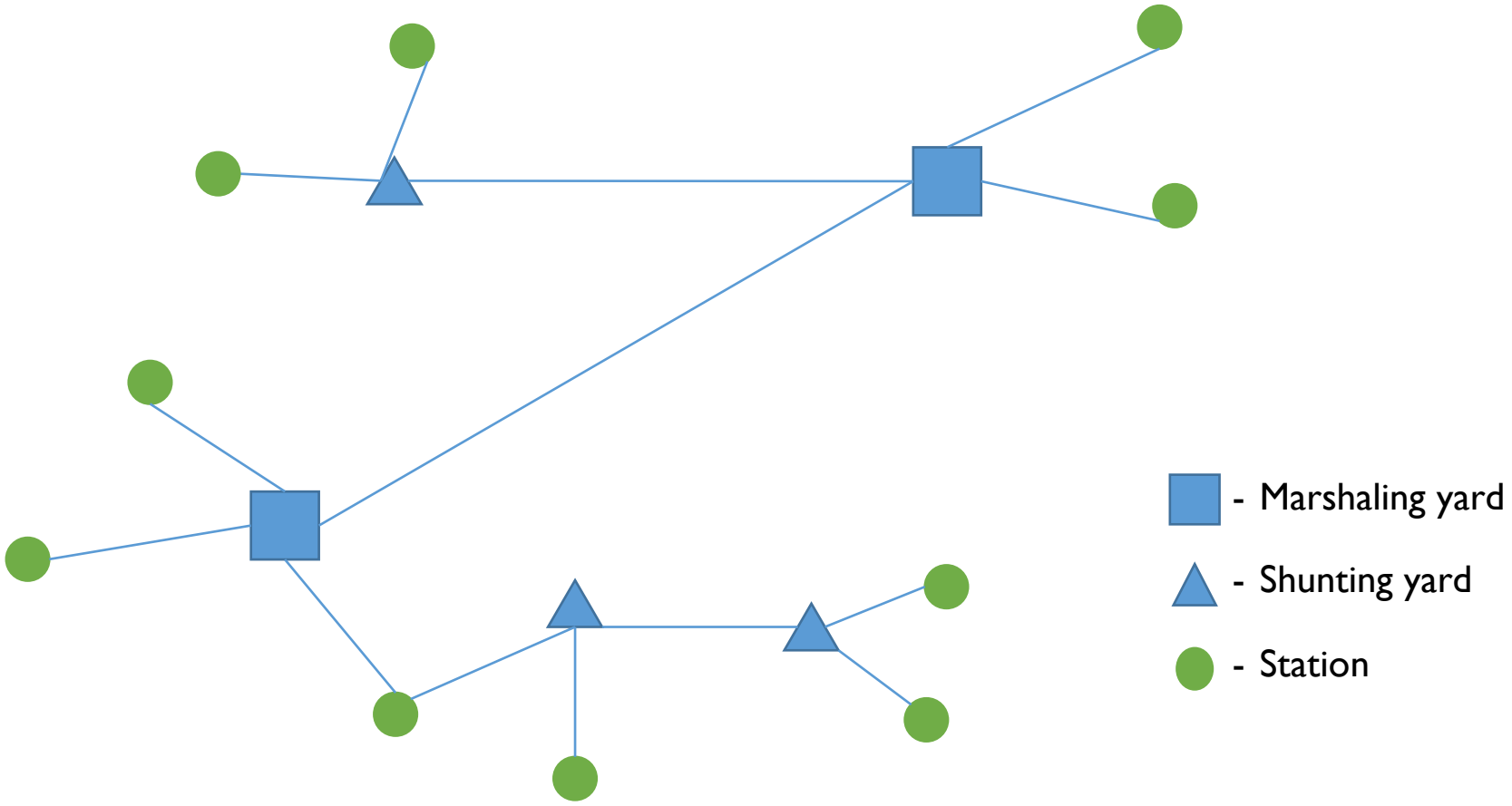
ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

SBB Cargo Network

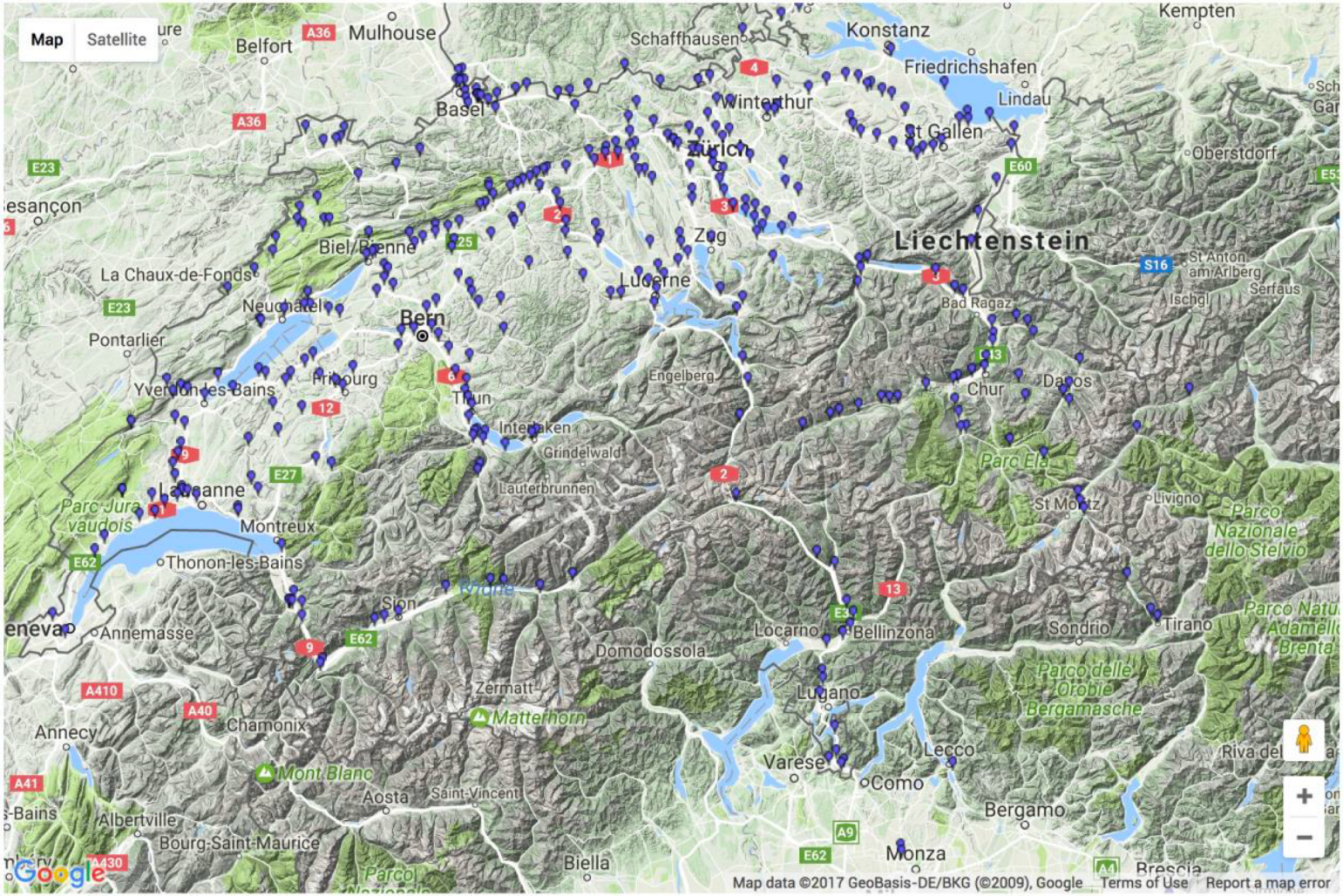
- Stations
- Tracks
- Demand
- **Bundling points**
  - **Marshaling yards**
  - **Shunting yards**
  - **Small bundling point**

Marshaling and shunting yards

- Bundling different commodities with close origins and close destinations



SBB Cargo Network



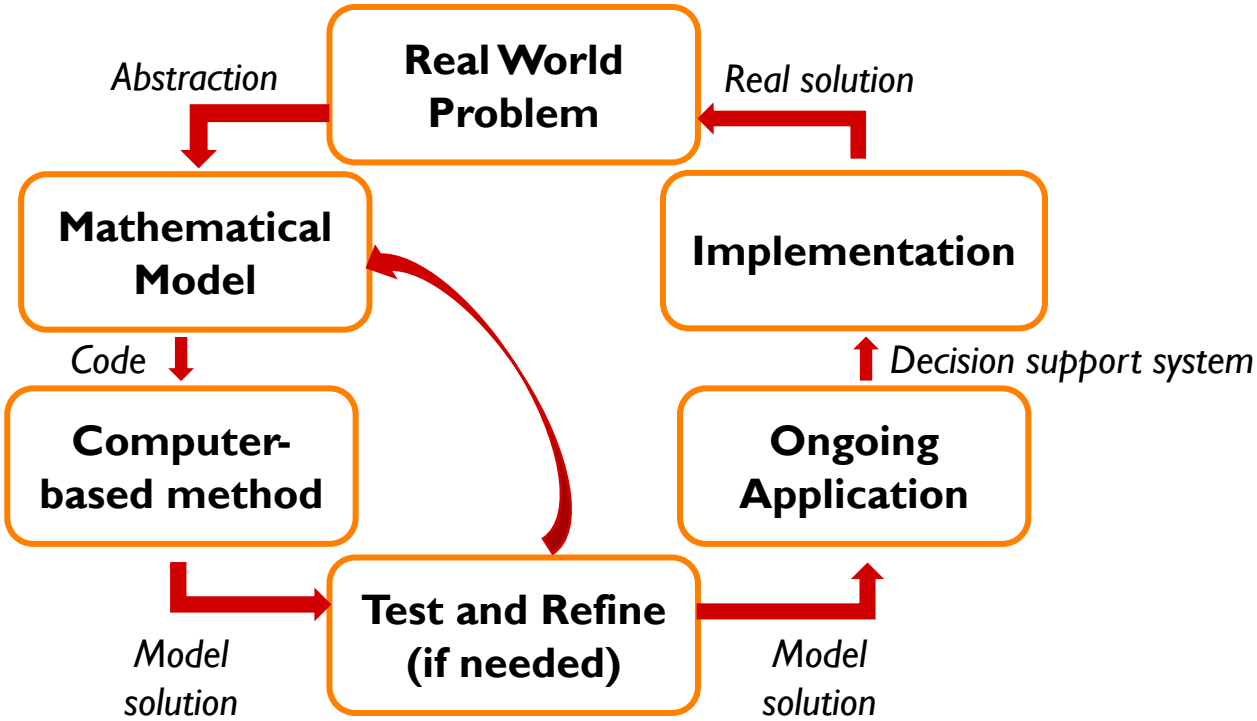
TRANSP-OR

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

SBB Cargo Network

- Goal: determine the optimal number and location of bundling points
- Consequently: lower the price of transport and yard operation

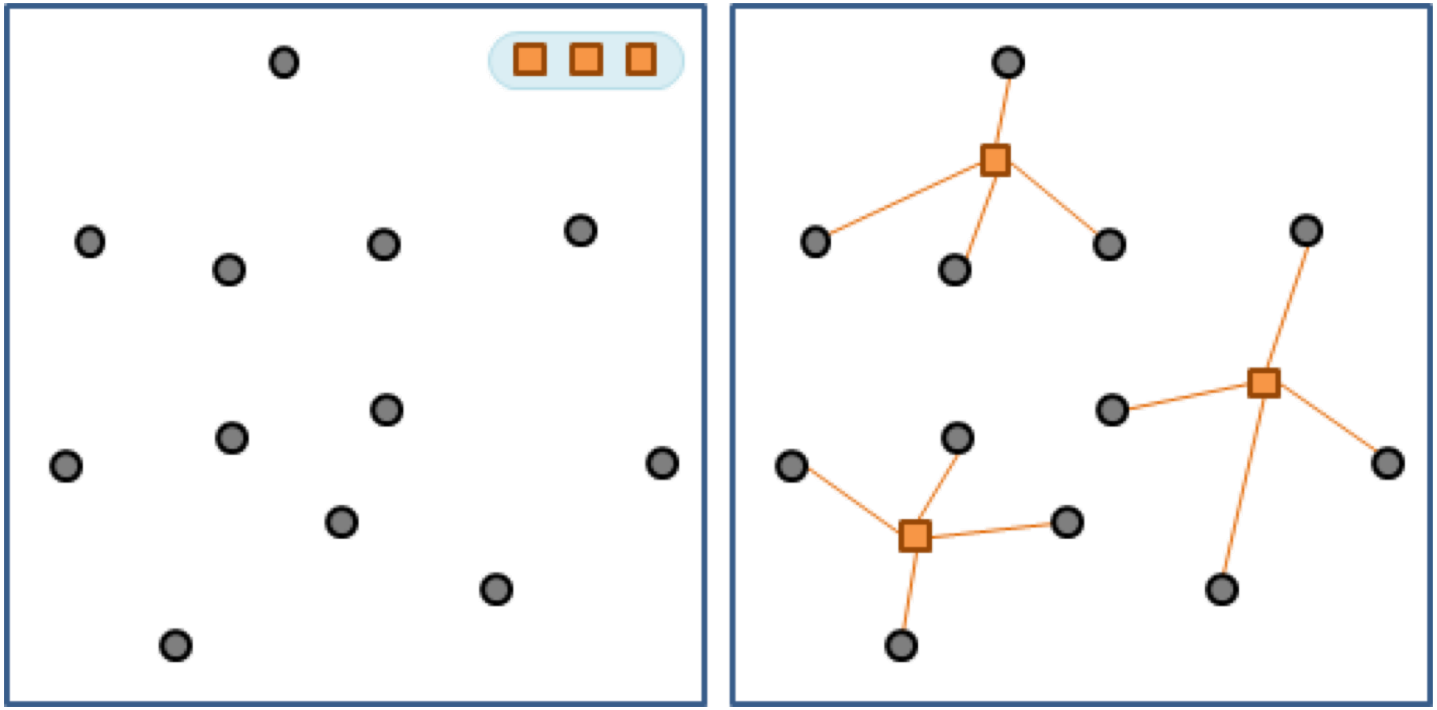
General Framework



Related problems

- Facility location problem (FLP)
- Hub location problem (HLP)
- Multicommodity flow problem (MFP)
- Multicommodity network design problem (MNDP)

Facility Location Problem



Facility Location Problem

- Minimizes facility opening and transportation costs.
- Does not assume each site has the same fixed costs
- Does not assume that there is a set number of facilities p that should be opened
- Determines optimal number and locations of facilities, as well as assignments of demand to a facility
- Transport occurs only between the demand point and the facility
- Transportation cost is proportional to the demand and distance

Facility Location Problem

- Example:
  - Design of the production/distribution network
    - Locations of plants, distribution centers and/or warehouses w.r.t. the customers
  - Design of computer networks
    - Locations of servers and switching equipment w.r.t. terminals

FLP Formulation

- Notations
  - I – set of demand nodes
  - J – set of candidate sites for facilities
  - f_j - fixed cost of locating a facility at candidate site j
  - C_j - capacity of a facility at candidate site j
  - α - cost per unit demand per unit distance
  - d_{ij} - distance between demand node i and candidate site j
  - h_i - demand at node i
  - x_j - 1 if node j is a facility
  - y_{ij} - 1 if demand node i is assigned to facility at node j, 0 otherwise

FLP Formulation

$$Min \sum_{j \in J} f_j x_j + \alpha \sum_{iinI} \sum_{j \in J} h_i d_{ij} y_{ij} \quad (1)$$

$$s.t. \sum_{j \in J} y_{ij} = 1 \qquad \forall i \in I \quad (2)$$

$$y_{ij} - x_j \leq 0 \qquad \forall i \in I, \forall j \in J \quad (3)$$

$$\sum_{j \in J} h_i y_{ij} - C_j x_j \leq 0 \qquad \forall i \in I \quad (4)$$

$$x_j \in \{0,1\} \qquad \forall j \in J \quad (5)$$

$$y_{ij} \in \{0,1\} \qquad \forall i \in I, \forall j \in J \quad (6)$$

- (1) minimizes sum of fixed facility location costs and total travel costs for demand to be served.
- (2) requires each demand node be assigned to exactly one facility
- (3) restricts demand node assignments only to open facilities
- (4) prohibits total demand assigned to a facility from exceeding capacity of the facility C_j
- (5) establishes the siting decision variable as binary
- (6) a binary constraint requiring all demand points be single sourced

Neighborhood search

- Basic idea:
  - Start from a feasible solution
  - In each solution, create one or more neighbors
    - A neighbor is created by a "significant" modification of the current solution
  - Choose the neighbor with the improved objective function
- The neighborhood is defined based on the structure of the problem.

Local search

Algorithm 29.3: Local search

Objective

Find a good feasible solution of the optimization problem $\min_x f(x)$ subject to $x \in \mathcal{F}$.

Input

The objective function $f : \mathbb{R}^n \to \mathbb{R}$

The feasible set $\mathcal{F}$

An initial feasible solution $x_0 \in \mathcal{F}$

A neighborhood structure N

Output

A feasible solution x^*

Repeat

$x_c := \operatorname{argmin}_{x \in N(x_k) \cap \mathcal{F}} f(x)$

if $f(x_c) < f(x_k)$ **then**

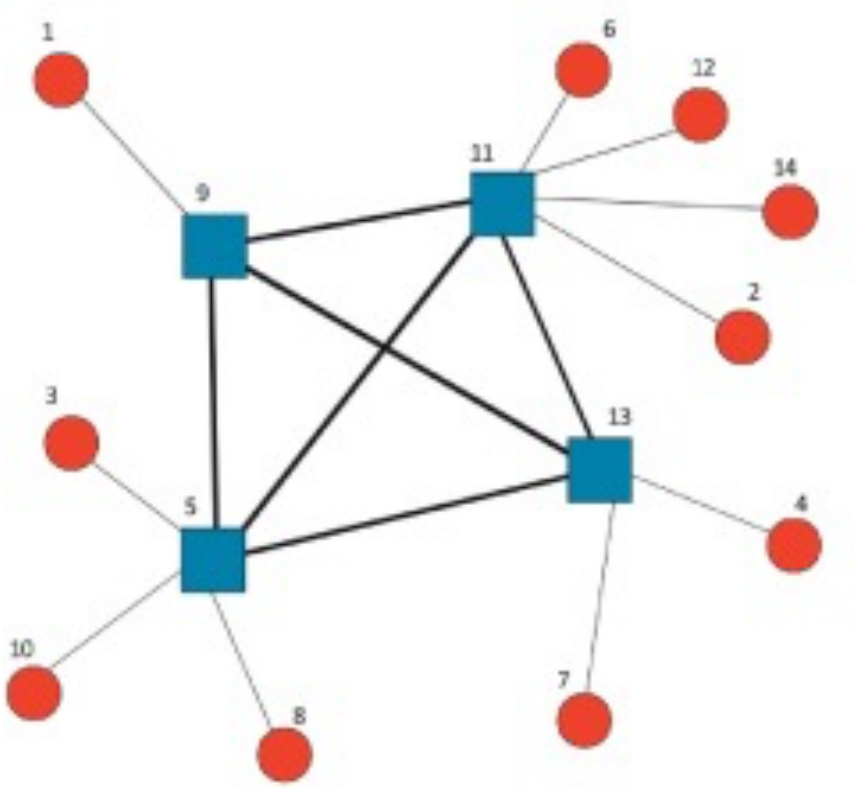
$x_{k+1} := x_c$

$k := k + 1$

Until $f(x_c) \geq f(x_k)$ *or* $N(x_k) \cap \mathcal{F} = \emptyset$

$x^* := x_k$

Hub location problem (HLP)



TRANSP-OR

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

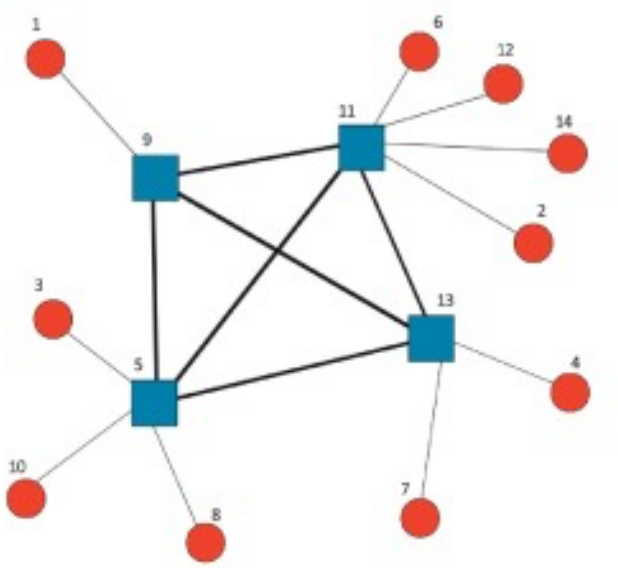
Hub location problem (HLP)

- Involves locating facilities and designing hub networks.
- Goal to minimize total cost (as a function of distance) of transportation between hubs, facilities and demands.
- Rather than serving every origin-destination demand with a direct link, a hub network provides service via smaller set of links between origin/destinations and hubs, and between pairs of hubs.
    - Every origin-destination path includes at least one hub node, and cost per unit flow is discounted between all hub pairs
- Use of fewer links in the network concentrates flows and allows economies of scale to be exploited.

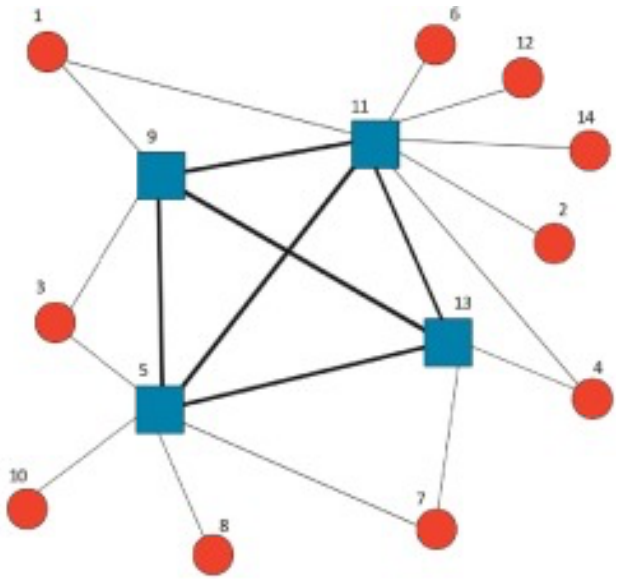
HLP Applications

- Transportation
  - Air passenger travel, air freight travel, express shipments, large trucking systems, postal operations and rapid transit systems.
  - Demand usually specified as flows of passengers or goods between city pairs; transported in vehicles of some type.
- Telecommunication
  - Distributed data networks in computer communication, telephone networks, video teleconferences, distributed computer communication, telephone networks, etc.
  - Demand is transmission of information and occurs over telephone lines, fiber optic cables, co-axial cables, or satellite channels and microwave links.

HLP Model Variants



(a) single allocation



(b) multiple allocation

HLP Model Variants

- Multiple and single allocation versions exist for hub location problems
- Single allocation:
  - Each demand point must be allocated to communicate with one hub
  - All flows to and from each demand point travel via the same hub node
- Multiple allocation:
  - Each demand point may be allocated to communicate with more than one hub
  - Greater flexibility allows lower cost solutions, and simplifies solution, since for a given set of hub nodes, each origin-destination flow can be routed separately from all others via the least cost path.

Basic p-Hub Location Model

- Minimize total cost (as a function of distance) of transportation between hubs, facilities and demands
- Notations
  - h_{ij} = number of units of flow between nodes *i* and *j*
  - c_{ij} = unit cost of transportation between nodes *i* and *j*
  - α = discount factor for transport between hubs
  - Sets: *N* = all nodes
  - x_j = 1 if a hub is located at node *j*, 0 otherwise
  - y_{ij} = 1 if demands from node *i* are assigned to a hub located at node *j*, 0 otherwise

Basic p-Hub Location Model

$$\min \sum_{i \in N} \sum_{j \in N} h_{ij} \left(\sum_{k \in N} c_{ik} y_{ik} + \sum_{m \in N} c_{jm} y_{jm} + \alpha \sum_{k \in N} \sum_{m \in N} c_{km} y_{ik} y_{jm} \right) \quad (1)$$

$$s.t. \sum_{j \in N} x_j = p \quad (2)$$

$$\sum_{j \in N} y_{ij} = 1 \qquad \forall i \in N \quad (3)$$

$$y_{ij} - x_j \leq 0 \qquad \forall i, j \in N \quad (4)$$

$$x_j \in \{0,1\} \qquad \forall j \in N \quad (5)$$

$$y_{ij} \in \{0,1\} \qquad \forall i, j \in N \quad (6)$$

Basic p-Hub Location Model

- (1) Minimizes sum of cost of moving items between a non-hub node and the hub to which the node is assigned, the cost of moving from the final hub to the destination of the flow, and the interhub movement cost which is discounted by a factor of α.
- (2) limits the number of hubs
- (3) each node should be assigned to exactly one hub
- (4) a non-hub node can only be assigned to a hub node
- (5) and (6) integrality constraints

Multiple allocation p-hub

- Notations
  - h_{ij} = number of units of flow between nodes *i* and *j*
  - C_{ij} = unit cost of transportation between nodes *i* and *j*
  - α = discount factor for transport between hubs
  - Set *N* = all nodes
  - X_k = 1 if a hub is located at node *j*, 0 otherwise
  - Z_{ij}^{km} = the fraction of flow h_{ij} which is transferred via hubs *k* and *m*

Multiple allocation p-hub

- A commodity transport cost is defined as the unit transportation cost between start node *i* and end node *j*, via hub nodes *k* and *m*

$$C_{ij}^{km} = C_{ik} + \alpha C_{km} + C_{mj}$$

Multiple allocation p-hub

$$\min \quad \sum_i \sum_j \sum_k \sum_m C_{ij}^{km} h_{ij} Z_{ij}^{km}$$

Subject to

$$\sum_k X_k = P$$

$$\sum_k \sum_m Z_{ij}^{km} = 1 \quad \forall\, i,j$$

$$Z_{ij}^{km} \leqslant X_m \quad \forall\, i,j,k,m$$

$$Z_{ij}^{km} \leqslant X_k \quad \forall\, i,j,k,m$$

$$Z_{ij}^{km} \geqslant 0 \quad \forall\, i,j,k,m$$

$$X_k \in \{0,1\} \quad \forall\, k$$

The Genetic Algorithm

- Directed search algorithms based on the mechanics of biological evolution
- Developed by John Holland, University of Michigan (1970's)
  - To understand the adaptive processes of natural systems
  - To design artificial systems software that retains the robustness of natural systems
- Provide efficient, effective techniques for optimization and machine learning applications
- Widely-used today in business, scientific and engineering circles

Components of a GA

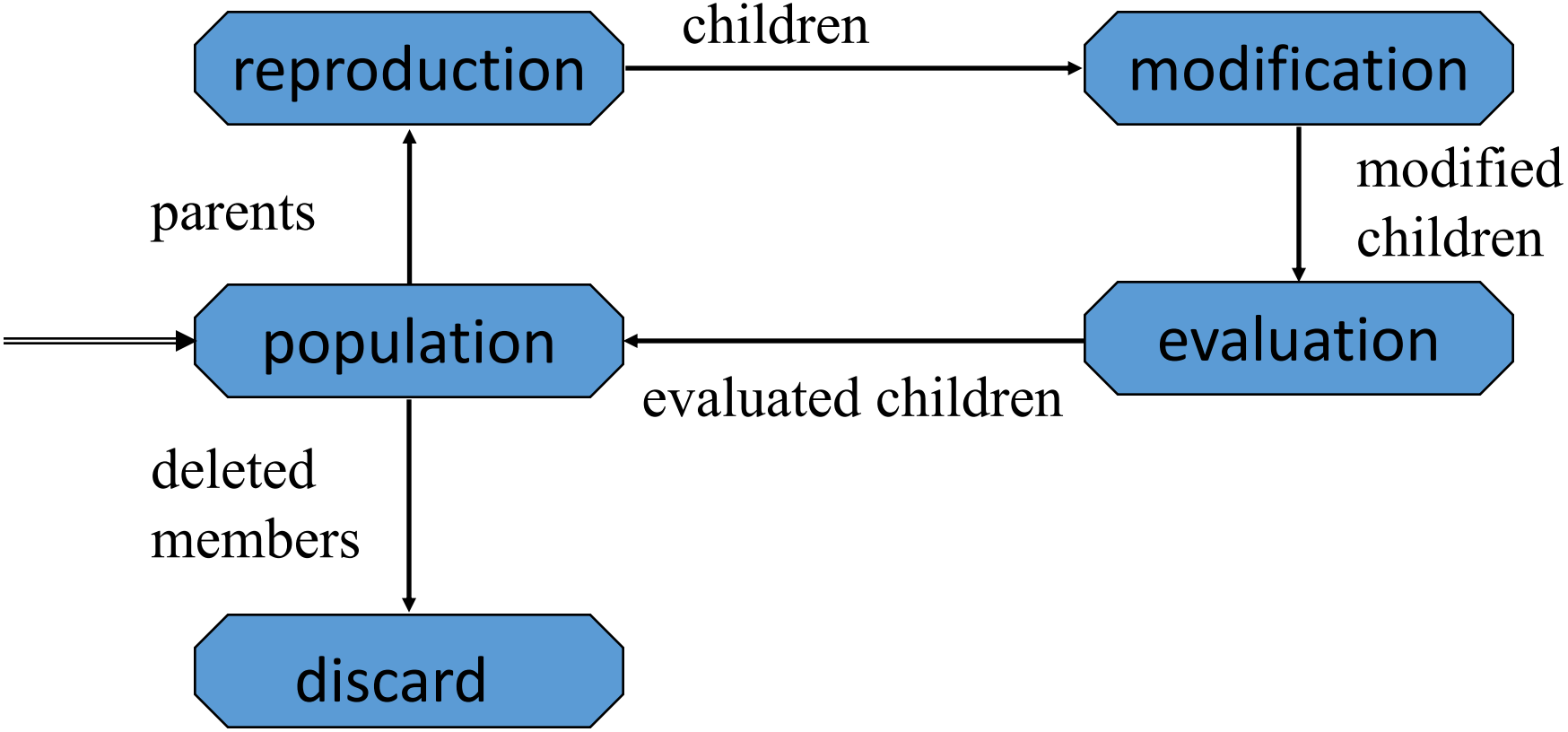
A problem to solve, and ...

- Encoding technique (*gene, chromosome*)
- Initialization procedure *(creation)*
- Selection of parents *(reproduction)*
- Genetic operators *(recombination, mutation)*
- Evaluation function *(environment)*
- Termination criteria *(optimality)*
- Parameter settings *(practice and art)*

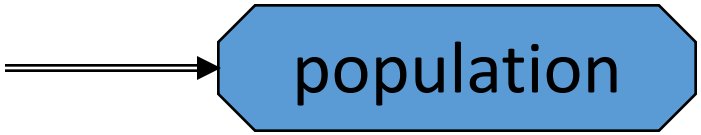
Basic Genetic Algorithm

```
{
    initialize population;
    evaluate population;
    while TerminationCriteriaNotSatisfied
    {
        select parents for reproduction;
        perform recombination and mutation;
        evaluate population;
    }
}
```

The GA Cycle of Reproduction



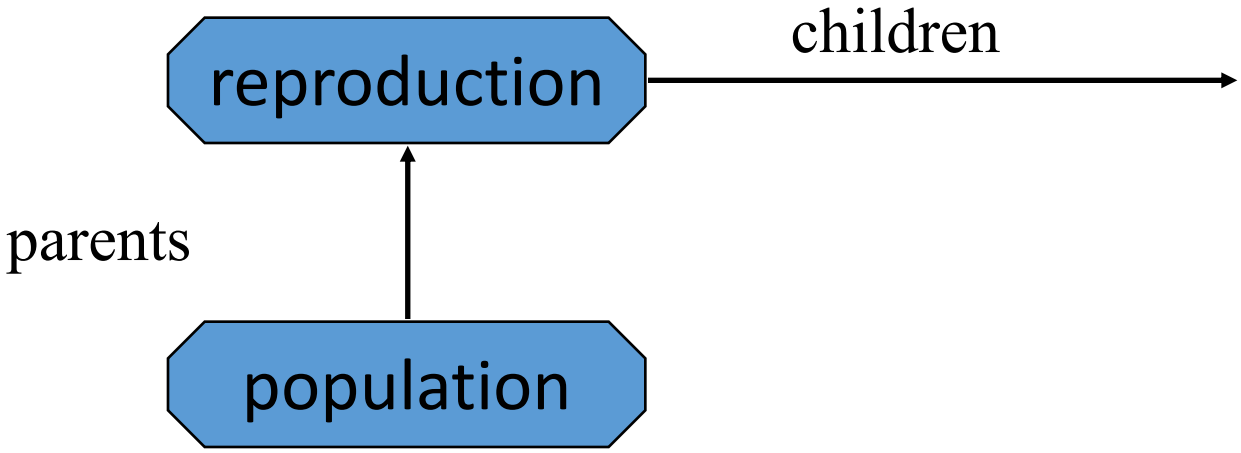
Population



Chromosomes could be:

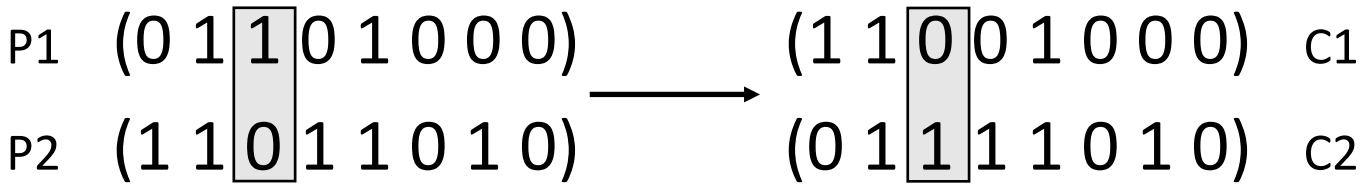
- Bit strings (0101 ... 1100)
- Real numbers (43.2 -33.1 ... 0.0 89.2)
- Permutations of element (E11 E3 E7 ... E1 E15)
- Lists of rules (R1 R2 R3 ... R22 R23)
- Program elements (genetic programming)
- ... any data structure ...

Reproduction



Parents are selected at random with selection chances biased in relation to chromosome evaluations.

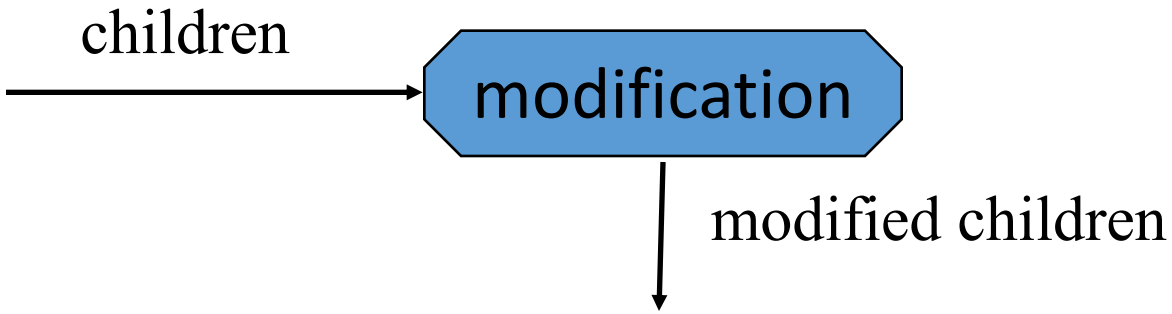
Crossover (Recombination)



Crossover is a critical feature of genetic algorithms:

- It greatly accelerates search early in evolution of a population

Chromosome Modification



- Modifications are stochastically triggered
- Operator types are:
  - Mutation
  - Crossover (recombination)

Mutation: Local Modification

Before: (1 0 1 1 0 1 1 0)

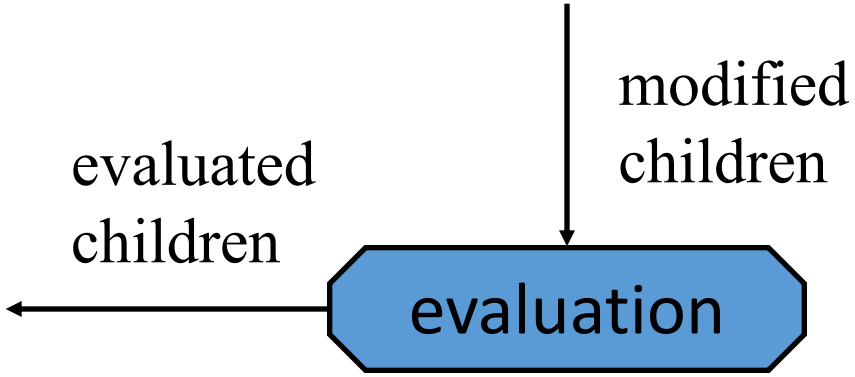
After: (0 1 1 0 0 1 1 0)

Before: (1.38 -69.4 326.44 0.1)

After: (1.38 -67.5 326.44 0.1)

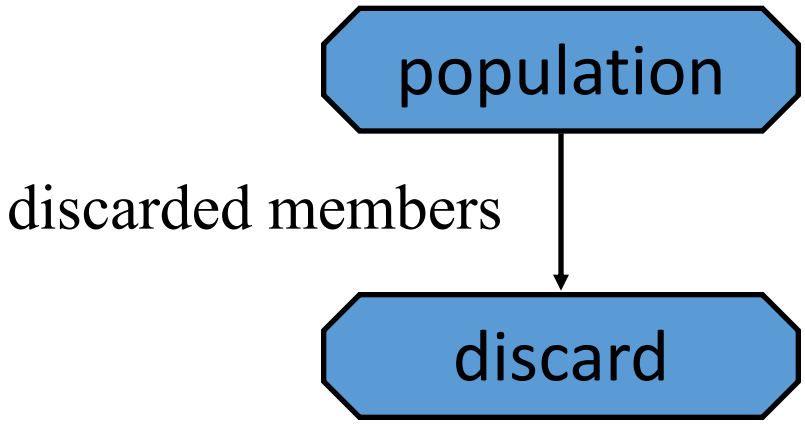
- Causes movement in the search space (of random size)
- Restores lost information to the population

Evaluation



- The evaluator decodes a chromosome and assigns it a fitness measure
- The fitness measure is e.g. the value of the objective function or the error between the expected and obtained results

Deletion



- *Generational* GA:
  entire populations replaced with each iteration
- *Steady-state* GA:
  a few members replaced each generation

Issues for GA Practitioners

- Choosing basic implementation issues:
  - representation
  - population size, mutation rate, ...
  - selection, deletion policies
  - crossover, mutation operators
- Termination Criteria
- Performance, scalability
- Solution is only as good as the evaluation function

Benefits of Genetic Algorithms

- Concept is easy to understand
- Modular, separate from application
- Supports multi-objective optimization
- Inherently avoids local optima
- Provides always an answer; answer gets better with time
- Parallel by construction and easily distributed

Downsides of Genetic Algorithms

- Not interpretable
- Weakly driven by the domain knowledge of the solved problem

When to Use a GA

- Exact solutions and domain-driven heuristics are too slow or overly complicated
- Need an exploratory tool to examine new approaches
- Problem is similar to one that has already been successfully solved by using a GA

Some GA Application Types

Domain	Application Types
Control	gas pipeline, pole balancing, missile evasion, pursuit
Design	semiconductor layout, aircraft design, keyboard configuration, communication networks
Scheduling	manufacturing, facility scheduling, resource allocation
Robotics	trajectory planning
Machine Learning	designing neural networks, improving classification algorithms, classifier systems
Signal Processing	filter design
Game Playing	poker, checkers, prisoner's dilemma
Combinatorial Optimization	set covering, travelling salesman, routing, bin packing, graph colouring and partitioning

Multicommodity flow problem

Problem elements – a network with traffic demand:

- a set of nodes,
- a set of links connecting them, and
- a set of connection requests, between pairs of nodes of the network

Goals:

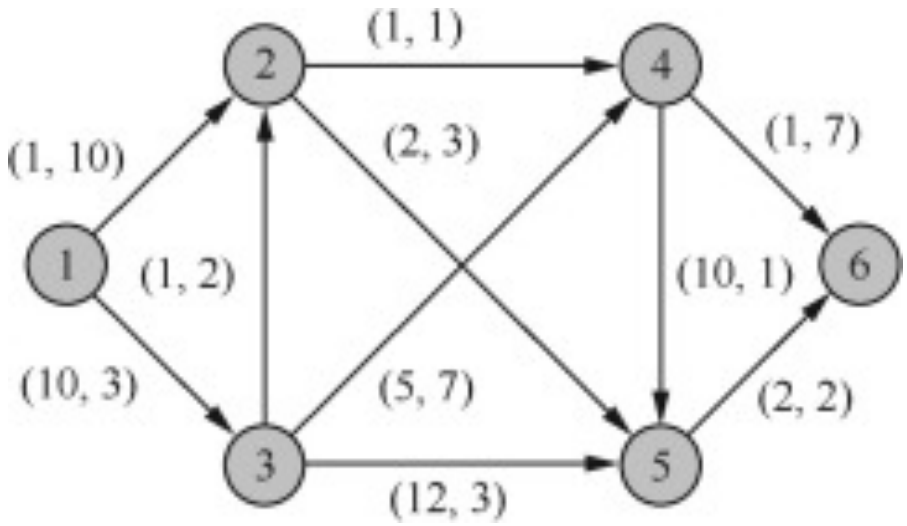
- determine a route for each commodity, in order to
- minimize delay time or transportation cost

Multicommodity flow problem

Some assumptions:

- Each link has a unit transport cost,
  - which can depend on the transferred commodity,
- the capacity of links may not be enough for all connection requests,
- different connections can be routed on the same links ...
- ... provided the capacity of each link is enough.

Multicommodity flow problem



MFP Formulation

Given a network, build a *directed* graph $G = (V, A)$ having:

- one vertex $i \in V$ for each node of the network,
- one arc $a \in A \subseteq V \times V$ for each link of the network,
- capacities $u(i, j)$ on each arc $(i, j) \in A$.

Then, consider the set K of connection requests (commodities), and enrich the graph with:

- costs $c^k_{(i,j)}$ on each arc $(i, j) \in A$ for each commodity $k \in K$,
- flow excess b^k_i for each node $i \in V$ and for each commodity $k \in K$.

MFP Assumptions

We assume that

- **Homogeneous commodities w.r.t. capacity**: each unit of flow of uses one unit of capacity on each arc, independently of *k*,
- **No congestion**: cost is linear in the amount of flow on each arc (until capacity limit is reached),
- **Fractional flows**: no integrality condition is imposed on flows.

WLOG we assume also that

- $b_i^k > 0$ for a unique $i \in V$ (origin of commodity $k \rightarrow s_k$),
- $b_i^k < 0$ for a unique $i \in V$ (destination of commodity $k \rightarrow t_k$).

We search for *k* min cost flows on the network, one for each commodity.

MFP Definition

Let $x^k_{(i,j)}$ be decision variables representing the amount of flow for commodity k sent on arc (i,j). Let v represent the total cost of routing packets in the network.

$$\min v = \sum_{(i,j)\in A} \sum_{k\in K} c^k_{(i,j)} x^k_{(i,j)}$$

$$s.t. \sum_{j\in V} x^k_{(i,j)} - \sum_{l\in V} x^k_{(l,i)} = b^k_i, \qquad \forall i \in V, i \neq s,t; \forall k \in K$$

$$\sum_{k\in k} x^k_{(i,j)} \leq u_{(i,j)}, \qquad \forall (i,j) \in A$$

$$x^k_{(i,j)} \geq 0, \qquad \forall (i,j) \in A; \forall k \in K$$

MFP Path-based formulation

Idea: represent overall flow as sum of partial flows, each following a single path, and combine them in a feasible way.

Notations:

- P_k is the set of all paths from s_k to t_k
- c_k^p is the unit cost of transferring commodity k on the path p
- f_k^p is the amount of flow of commodity k sent on path p
- $a_{ij}^p = 1$ if path p includes arc (i, j), and = 0 otherwise
- u_{ij} - the capacity of the arcs (i, j)
- b^k - total amount of commodity k to be transferred

MFP Path-based formulation

LP model:

$$\min v = \sum_{k \in K} \sum_{p \in P^k} c_k^p f_k^p$$

$$s.t. \sum_{k \in K} \sum_{p \in P^k} a_{ij}^p f_k^p \leq u_{ij}\,, \qquad \forall (i,j) \in A$$

$$\sum_{p \in P^k} f_k^p = b^k, \qquad \forall k \ \in K$$

$$f_k^p \geq 0, \qquad \forall p \in P^k; \forall k \in K$$

Is it possible to straightly optimize it?

$\left|P^k\right|$ grows combinatorially with problem dimension => we need an iterative approach.

Column generation

- **Idea**: in a good MFP solution, only very few good paths are chosen.
- General steps:
  - We replace each P^k with a "well chosen" subset $S^k \subset P^k$
  - $\bigcup_{k \in K} S^k$ contains at least one feasible solution
  - Optimize the restricted problem
  - Further, we search for a path in $P^k \backslash S^k, \forall k \in K$ that would improve the solution
  - If such path does not exist, we have found the solution

Pricing problem – the most challenging step

Pricing

- In the LP model:
  - let $\lambda_{ij} \geq 0$ and μ_k be the dual variables
  - the reduced cost of each path variable f_k^p is

$$\bar{c}_k^p = c_k^p - \sum_{(i,j)\in A} \left(-\lambda_{ij} \cdot a_{ij}^p\right) - \mu_k$$

  - or: $\bar{c}_k^p = \sum_{(i,j)\in A}(c_{ij}^k + \lambda_{ij}) \cdot a_{ij}^p - \mu_k$
  - a solution v is optimal if **all variables** have **non-negative reduced cost**
- Luckily: searching for the variable with most negative reduced cost is like searching for a **minimum cost path** in the network where **arc costs** are **increased** with λ_{ij}

Column generation

The algorithm:

BEGIN
Initialize $\mathcal{S}^k$
do
    solve the **restricted** LP model, considering $\mathcal{S}^k$
    get the values of dual variables $\lambda_{ij} \geq 0$ and μ_k
    for each $k \in K$
        find a **shortest path** on G using (red.) costs
            $\bar{c}_{ij} = c^k_{ij} + \lambda_{ij}$
        obtain a path P of (reduced) cost $\bar{c}^P$
        if $\bar{c}^P - \mu_k < 0$, add P to $\mathcal{S}^k$
while (*any new path has been added to* $\mathcal{S}^k$)
END

Column generation

Question:

- How should we initialize S^k?

Multicommodity Network Design Problem (MNDP)

- Setting: it is required to send flows (which may be fractional) to satisfy demands given arcs with existing capacities, or to install, in discrete amounts, additional facilities with fixed capacities.
- Price: not only for routing flows, but also for using an arc or installing additional facilities.
- Goal: to determine the optimal routes of the commodities and the number and location of facilities to be installed.
- Appears in transportation and telecommunication systems
- NP-hard problem
- Arc-based vs. path-based formulation

MNDP Formulation

- Arc-based formulation
- Inputs:
  - a directed graph $G = (N, A)$
  - a set of commodities K to be routed
  - a set of facility types L to be installed on each arc
- Objective: minimize the sum of flow and facility installation costs
- Constraints:
  - Flow conservation
  - Arc capacities
  - Max number of facilities

MNDP Formulation

- Constants:
  - c_{ij}^k – flow cost per unit of commodity k on arc (i, j)
  - f_{ij}^l – the design cost for each facility of type l installed on arc (i, j)
  - $O(k)$ – origin nodes for commodity k
  - $D(k)$ – destination nodes for commodity k
  - $T(k)$ – transshipment nodes
  - b_{ij}^k – an upper bound on the amount of flow of commodity k that may pass through arc (i, j)
  - h_{ij}^l – an upper bound on the number of facilities of type l installed on arc (i, j)

MNDP Formulation

- Constants:
  - e_{ij}^{k} - commodity k weight
  - v_{ij} – initial (existing) arc capacity
  - u_{ij}^{l} - added capacity by installing one facility
- Decision variables:
  - x_{ij}^{k} – the amount of flow of commodity k that will pass through arc (i, j)
  - y_{ij}^{l} – the number of facilities of type l to be installed on arc (i, j)

MNDP Formulation

$$\min \sum_{k \in K} \sum_{(i,j) \in A} c_{ij}^k x_{ij}^k + \sum_{l \in L} \sum_{(i,j) \in A} f_{ij}^l y_{ij}^l$$

$$\sum_{j \in N^+(i)} x_{ij}^k - \sum_{j \in N^-(i)} x_{ji}^k = \begin{cases} o_i^k & i \in O(k) \\ -d_i^k & i \in D(k) \\ 0 & i \in T(k) \end{cases} \qquad i \in N,\ k \in K$$

$$0 \leq x_{ij}^k \leq b_{ij}^k \quad (i,j) \in A,\ k \in K$$

$$\sum_{k \in K} e_{ij}^k x_{ij}^k \leq v_{ij} + \sum_{l \in L} u_{ij}^l y_{ij}^l \quad (i,j) \in A$$

$$Ay \leq g$$

$$0 \leq y_{ij}^l \leq h_{ij}^l \quad (i,j) \in A,\ l \in L$$

$$y_{ij}^l \ integer \quad (i,j) \in A,\ l \in L.$$

MNDP Solution Methods

- Exact method:
  - B. Gendron, M. Larose : Branch-and-price-and-cut for large-scale Multicommodity capacitated fixed-charge network design
- Heuristics:
  - I. Gamvros, B. Golden, S. Raghavan, and D. Stanojevic : Heuristic search for network design

Case Study

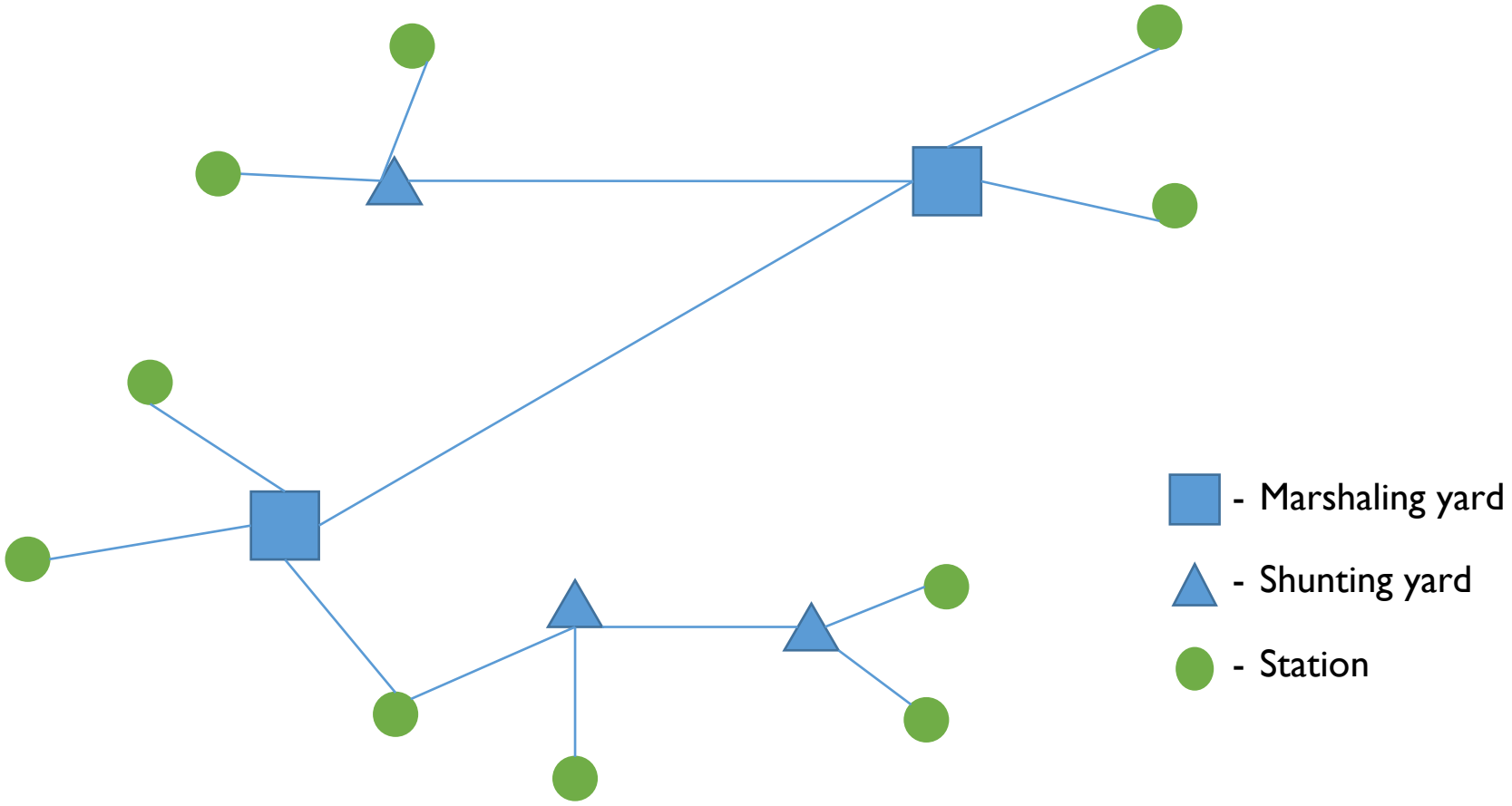
- **Network Design for CFF Cargo**

TRANSP-OR

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

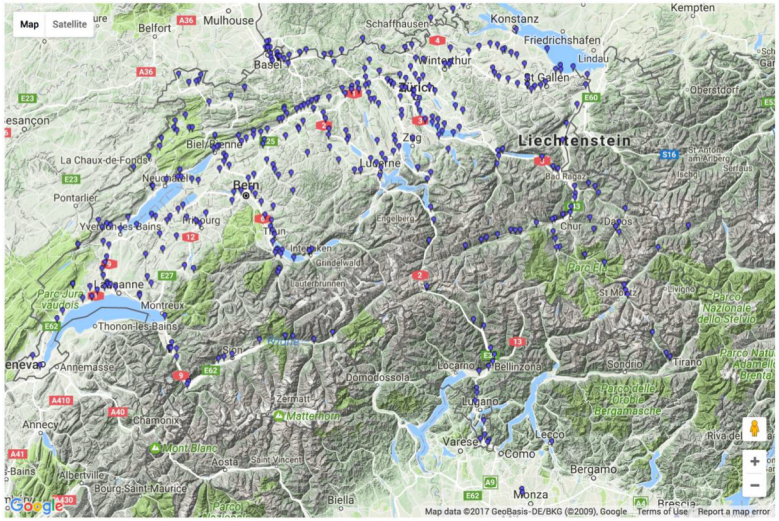
Marshaling and shunting yards

- Bundling different commodities with close origins and close destinations



Problem setting

- Existing SBB Cargo network
  - 2 inner marshaling yards
  - 3 border marshaling yards
  - Approx. 70 shunting yards – 50 can be changed



- Solution should provide:
  - Optimal number and locations of marshaling and shunting yards
  - Set of used trains
  - Assignment of commodities to trains

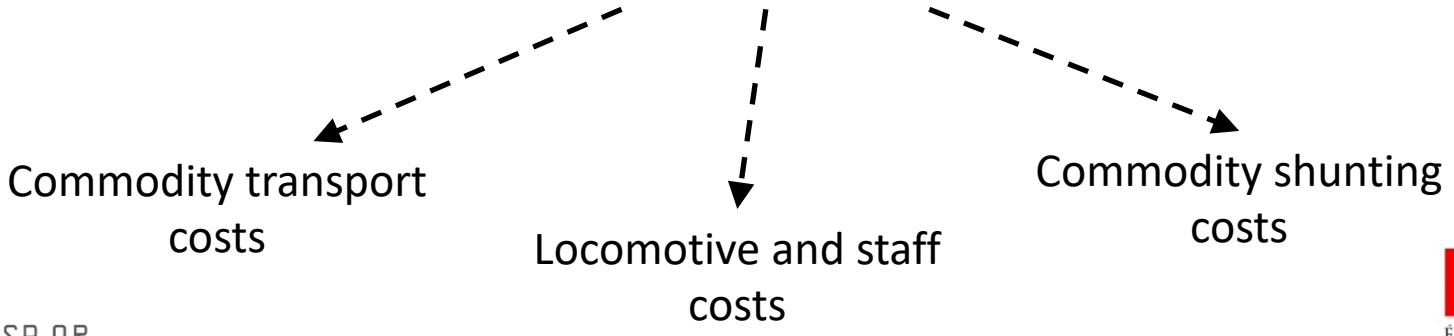
Input Data

- Input data set contained:
  - Infrastructure data
    - Stations
    - Bundling points (with capacities)
    - Tracks (with capacities)
  - Demand data
  - Cost data

Problem definition

- Combination and extension of the HLP and MNDP
  - Model of trains
- Network elements:
  - N – Set of stations, including potential marshaling and shunting yards
  - A – Set of direct links between the stations
  - K – Set of transported commodities each described with the origin, destination, weight and number of wagons
- Objective function:

$$\min C_{ct} + C_{ls} + C_{cs}$$



Problem definition *(cont.)*

- Commodity transport costs:

$$C_{ct} = \sum_{k \in K} \sum_{p \in N} \sum_{q \in N} \sum_{(i,j) \in A} d_{ij} w^k P_w x_{ij}^{pq} f_{pq}^k$$

  - Constants:
    - d_{ij} – Distance between nodes i and j
    - w^k – Weight of commodity k
    - P_W – Transport price per weight and distance unit
  - Variables:
    - x_{ij}^{pq} - Determines if arc (i, j) is used by the train between p and q
    - f_{pq}^k - Determines if commodity k is transported on the train between p and q

Problem definition *(cont.)*

- Locomotive and staff costs:

$$C_{ls} = \sum_{p \in N} \sum_{q \in N} \sum_{(i,j) \in A} n_{pq} x_{ij}^{pq} d_{ij} P_L$$

  - Constants:
    - d_{ij} – Distance between nodes i and j
    - P_L – Locomotive and staff cost per distance
  - Variables:
    - x_{ij}^{pq} - Determines if arc (i, j) is used by the train between p and q
    - n_{pq} - Number of trains between p and q

Problem definition *(cont.)*

- Commodity shunting costs:

$$C_{cs} = \sum_{k \in K} \sum_{i \in N} S v^k s_i^k + \sum_{k \in K} \sum_{i \in N} M v^k m_i^k$$

  - Constants:
    - v^k – Number of wagons of commodity k
    - S – Shunting price per wagon, in a shunting yard
    - M – Shunting price per wagon, in a marshaling yard
  - Variables:
    - s_i^k - Determines if commodity k is shunted in the shunting yard i
    - m_i^k - Determines if commodity k is shunted in the marshaling yard i

Problem definition *(cont.)*

- Constraints from MNDP:
  - Flow conservation constraints for trains
  - Arc capacity constrains
- Constraints from HLP:
  - Hub capacity constraints
  - Maximal number of hubs
- Node type constraints:

$$r_i + s_i + m_i = 1, \qquad \forall i \in N$$

$$\sum_{k \in K} s_i^k \leq s_i \mathcal{M}_1, \qquad \forall i \in N$$

$$\sum_{k \in K} m_i^k \leq m_i \mathcal{M}_2, \qquad \forall i \in N$$

- Variables:
  - r_i - If node i is a regular station
  - s_i - If node i is a shunting yard
  - m_i - If node i is a marshaling yard

Problem definition *(cont.)*

- Commodity assignment constraints:

$$f_{pq}^k \leq s_p^k + m_p^k + o_{kp}, \qquad \forall p, q \in N, \forall k \in K$$

$$f_{pq}^k \leq s_q^k + m_q^k + d_{kq}, \qquad \forall p, q \in N, \forall k \in K$$

- Flow conservation constraints for commodities:

$$\sum_{q \in N} f_{pq}^k - \sum_{q \in N} f_{qp}^k = o_{kp} - d_{kp}, \qquad \forall p \in N, \forall k \in K$$

- Constants:
  - o_{kp} – Determines if node p is the origin of commodity k
  - d_{kp} – Determines if node p is the destination of commodity k

Problem definition *(cont.)*

- Train capacity constraints:

$$\sum_{k \in K} f_{pq}^{k} v^{k} l^{k} \leq L_t n_{pq}, \qquad \forall p, q \in N$$

- Constants:
  - l^k – Length of commodity k
  - L_t – Max. allowed train length

Input data

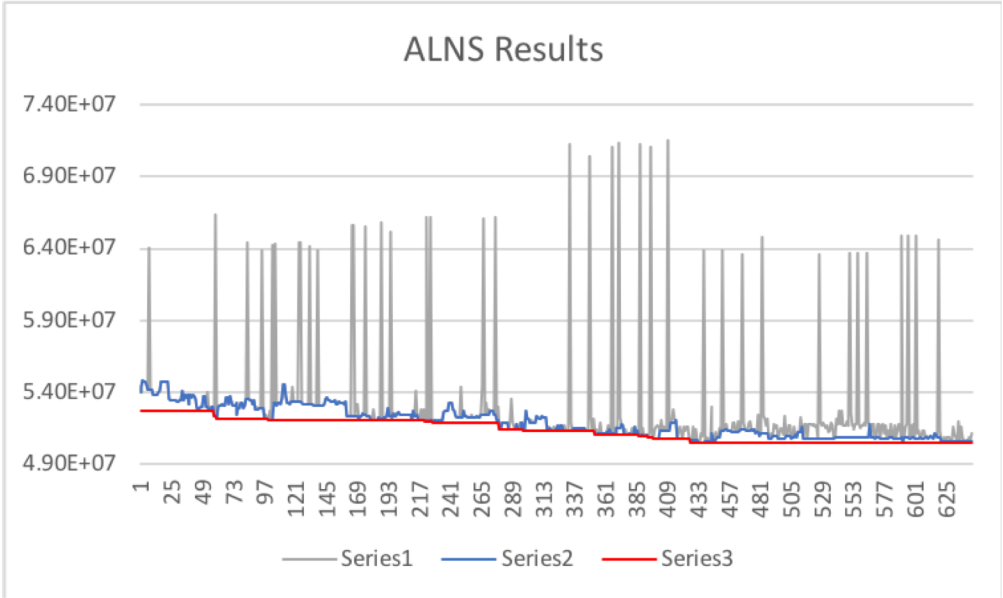
- Size of the SBB Cargo network:
  - Approx. 2100 stations
  - Approx. 2500 direct links
- Over 65000 commodities
  - Yearly demand, scaled to daily average

Heuristic algorithm

- Heuristic algorithm composed of 4 stages:
  - Yard location and sizing
  - Initial train generation
  - Commodity assignment (routing)
  - Train number reduction

Heuristic algorithm – Yard location and sizing

- Yard location:
  - Adaptive large neighborhood search
  - Variable neighborhood search



Simulated annealing

- Local search can both decrease and increase the objective function.
- Simulated annealing:
  - Choose initial and final temperatures, and the speed of the temperature decrease
  - At higher temperatures, we allow the algorithm to increase the objective function quite often (e.g. with $p = 0.95$). (Min problem is assumed.)
  - As the temperature is decreased, the increase of the objective function happens more rarely.
  - Cooling speed – speed of reduction of the probability to increase the objective function
- Analogy with metallurgy
  - Heating a metal and then cooling it down slowly improves its properties
  - The molecule structures become more ordered

Simulated Annealing

Algorithm 29.7: Simulated annealing

Objective

Find a good feasible solution of the optimization problem $\min_x f(x)$ subject to $x \in \mathcal{F}$.

Input

The objective function $f : \mathbb{R}^n \to \mathbb{R}$

The feasible set $\mathcal{F}$

An initial feasible solution $x_0 \in \mathcal{F}$

A neighborhood structure N

Initial temperature $T_0 > 0$, final temperature $0 < T_f < T_0$

A strictly decreasing function of T, $r(T)$

Number of iterations per level of temperature K

Output

A feasible solution x^*

Initialization

$x_c := x_0$, $x^* := x_0$, $T := T_0$

Simulated Annealing

```
Repeat
    for k := 1 to K do
        Select randomly y ∈ N(x_c) ∩ F
        δ := f(y) − f(x_c)
        if δ < 0 then
            x_c := y
        else
            Select randomly r ∈ ℝ between 0 and 1
            if r < exp(−δ/T) then
                x_c := y
        if f(x_c) < f(x*) then
            x* := x_c
    Reduce the temperature: T := r(T)
Until T ≤ T_f
```

SA – Temperature reduction

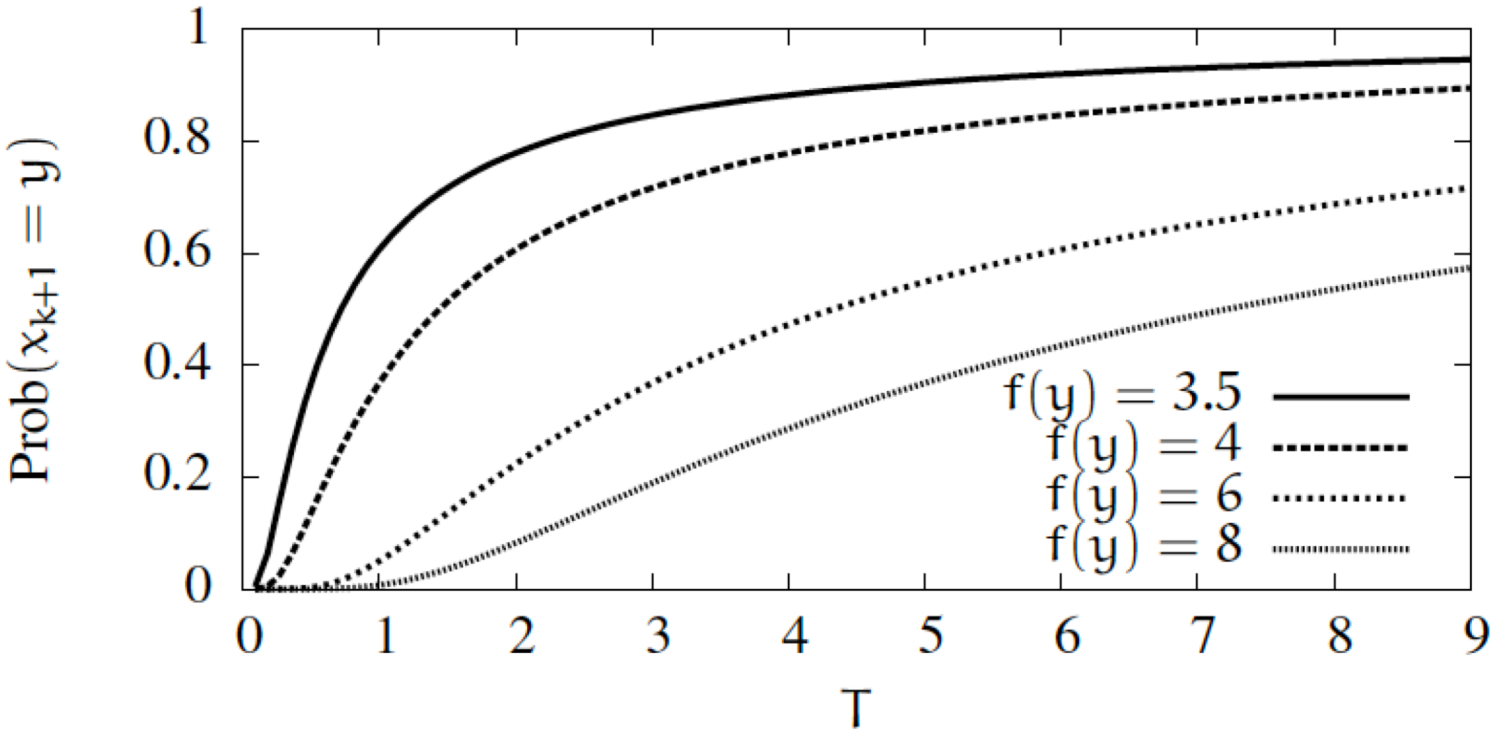
- In practice, start with high T for flexibility.
- Then, decrease T progressively

$$T_m = -\frac{\delta_t}{\ln(p_0 + \frac{p_f - p_0}{M} m)}$$

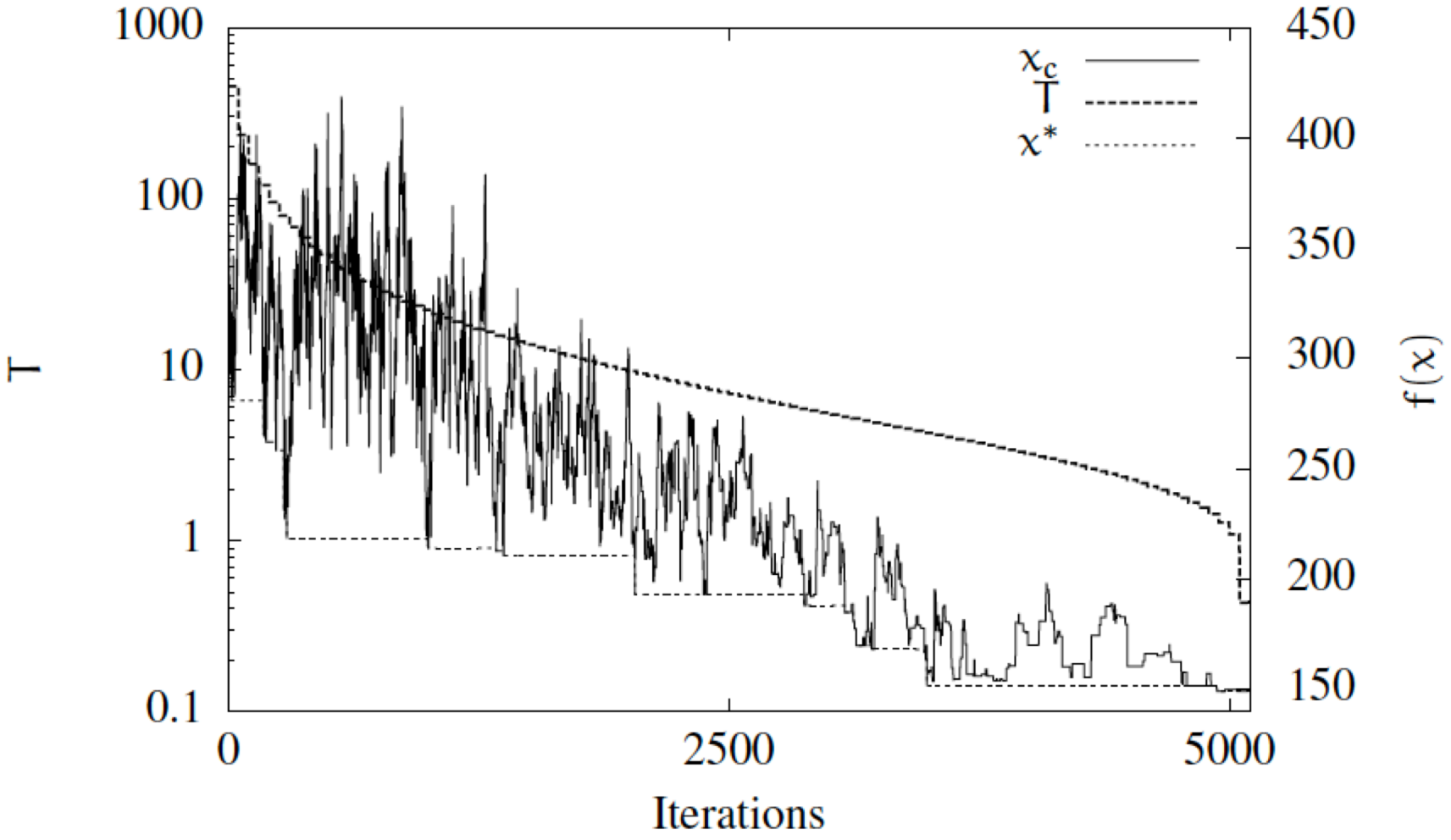
- Parameters:
  - δ_t - the typical increase of the objective function in the neighborhood structure
  - p_0 - the initial probability of acceptance
  - p_f - the final probability of acceptance
  - M- the number of times the temperature will decrease
  - $m = 0, \ldots, M$

Simulated Annealing

- $f(x_k) = 3$



Simulated Annealing Example

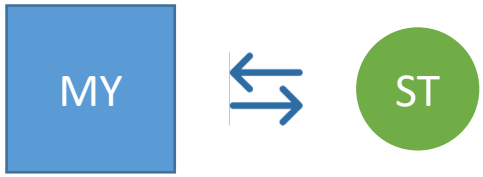


Adaptive large neighborhood search

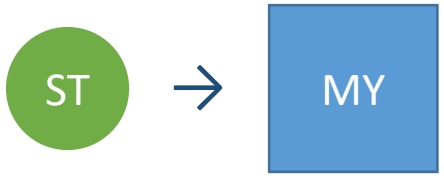
Algorithm 2 Adaptive large neighborhood search

1: input: a feasible solution x
2: $x^b = x$; $\rho^- = (1, \ldots, 1)$; $\rho^+ = (1, \ldots, 1)$;
3: **repeat**
4: select destroy and repair methods $d \in \Omega^-$ and $r \in \Omega^+$ using ρ^- and ρ^+;
5: $x^t = r(d(x))$;
6: **if** accept(x^t, x) **then**
7: $x = x^t$;
8: **end if**
9: **if** $c(x^t) < c(x^b)$ **then**
10: $x^b = x^t$;
11: **end if**
12: update ρ^- and ρ^+;
13: **until** stop criterion is met
14: **return** x^b

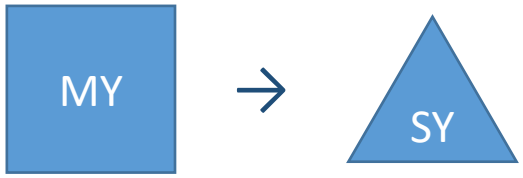
Yard location - Neighborhoods



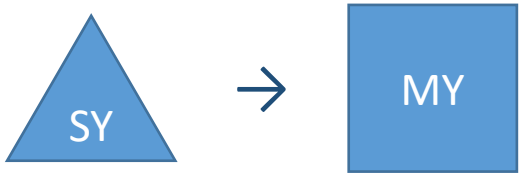
- Select the busiest station close to the MY



- Distance-dependent probability of station selection



- Select the least used MY



- Select fully utilized SY, with maximum capacity

Yard location - Neighborhoods

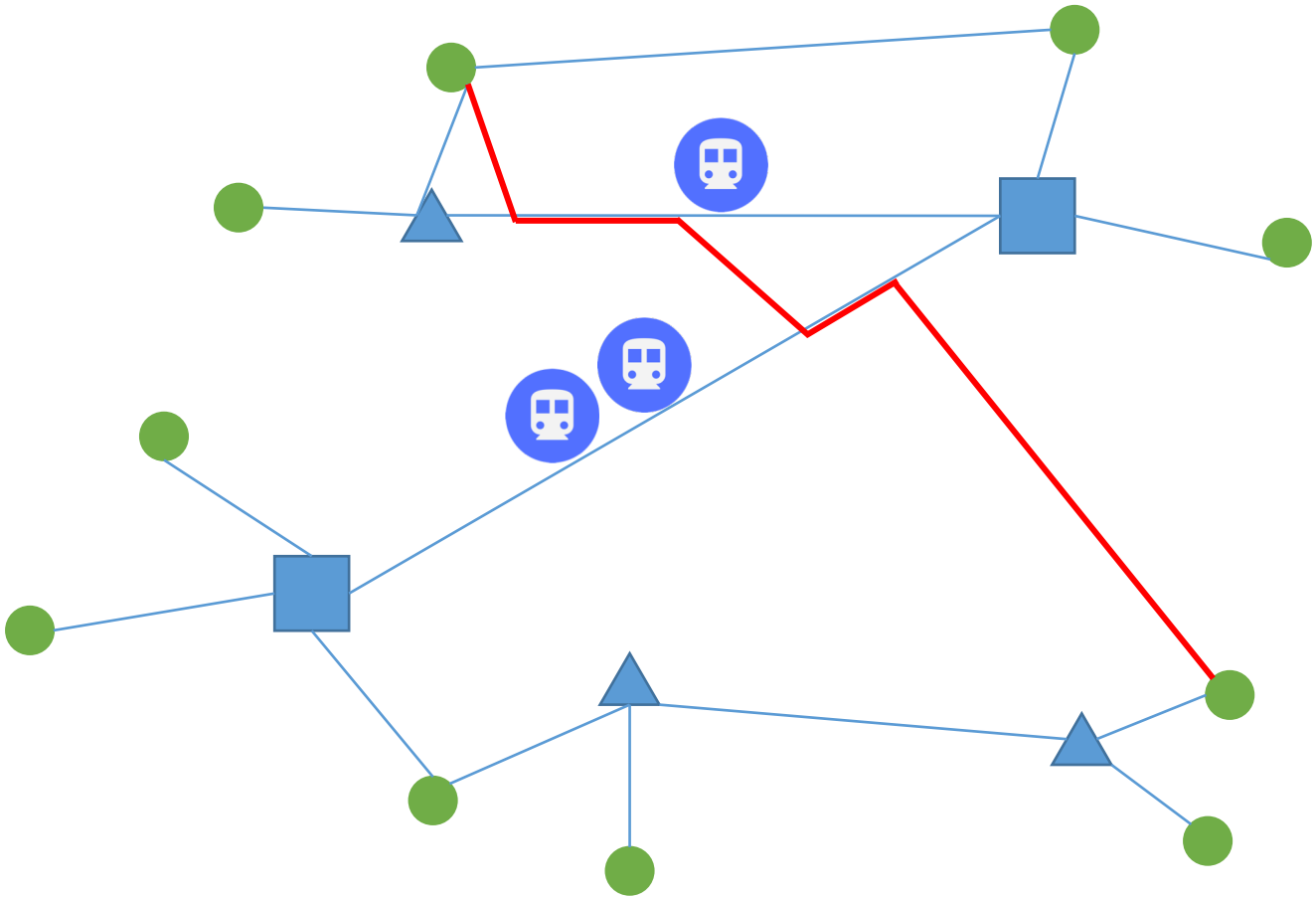
- Select SY with most unused capacity

- Select fully utilized SY with below maximum capacity

- Select underused SY with minimum capacity

- Select frequently used regular station

Heuristic algorithm – Initial trains generation



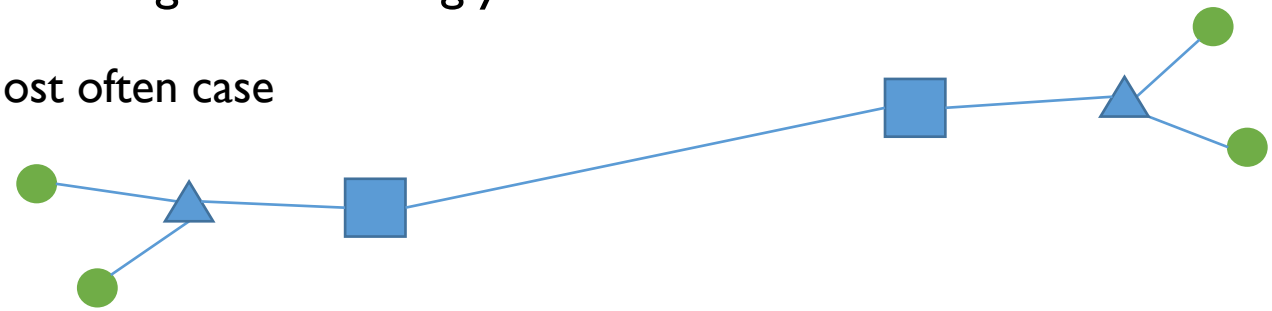
- Marshaling yard
- Shunting yard
- Station

TRANSP-OR

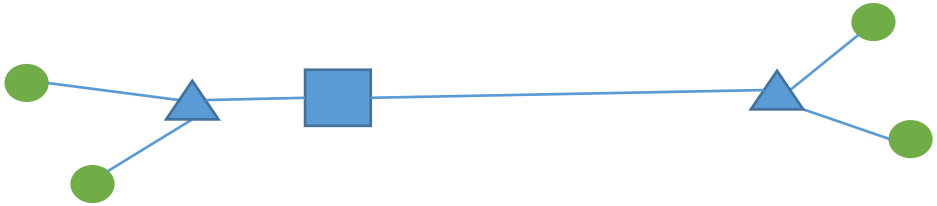
ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Heuristic algorithm - Path alternatives

- Via marshaling and shunting yards
  - Most often case



  - If the same marshaling yards is closest to both shunting yards

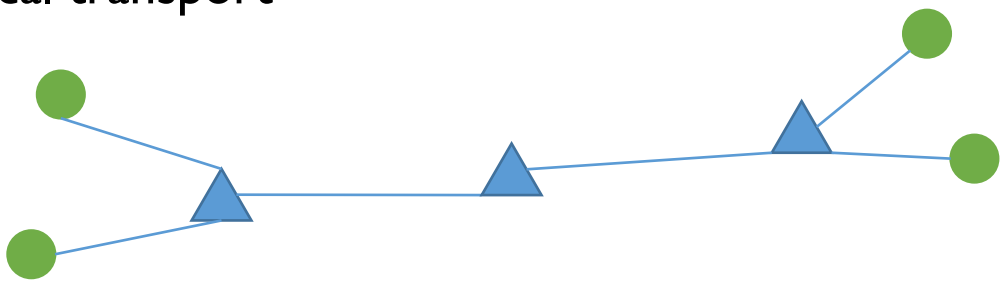


  - Skipped shunting yard

Heuristic algorithm - Path alternatives

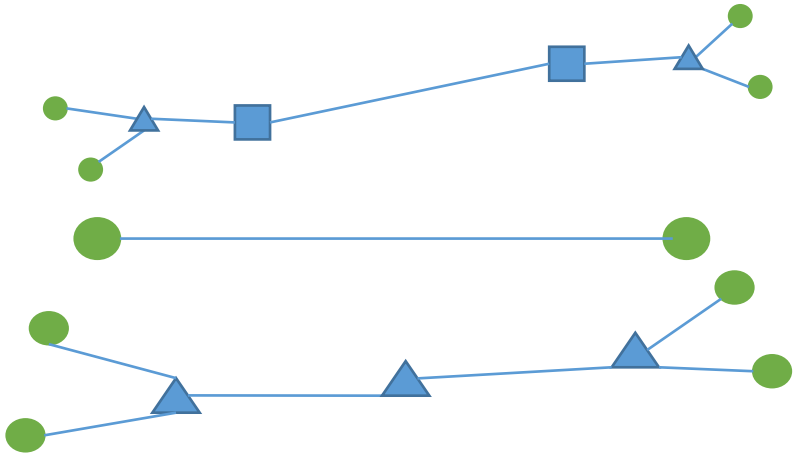
- Direct (shortest) path
  - For large commodities

- Via shunting yards
  - For local transport



Heuristic algorithm – Commodity assignment

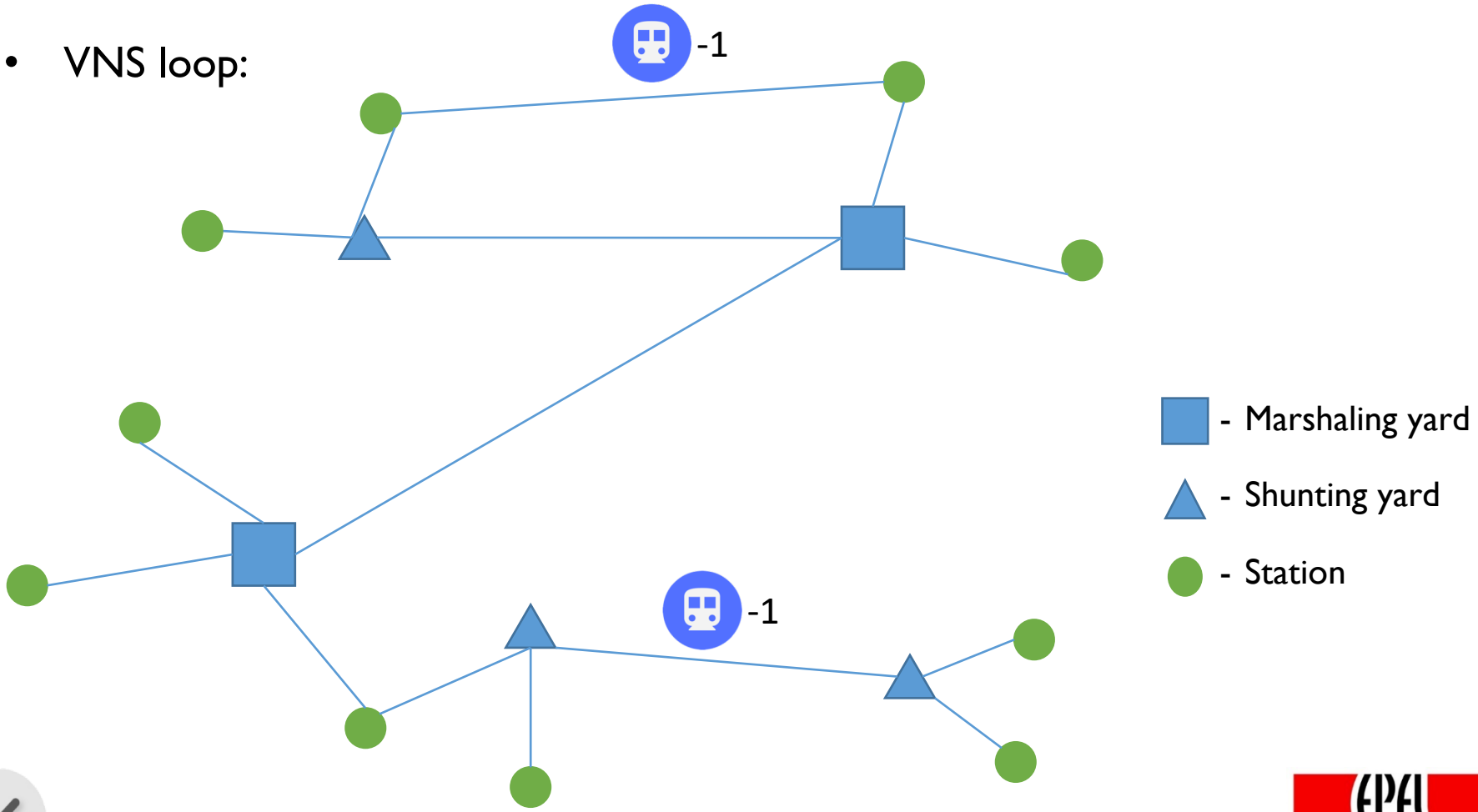
- Commodity routing:
  - Prioritized assignment algorithm



Select commodity by priority	
Path alternatives test	Via marshaling and shunting yards, shunting yards only, or direct train.
Path selection	Cheapest estimated, available path
Add necessary trains	If none of the paths is feasible

Heuristic algorithm – Reduction of train number

- Remove all unused trains
- VNS loop:

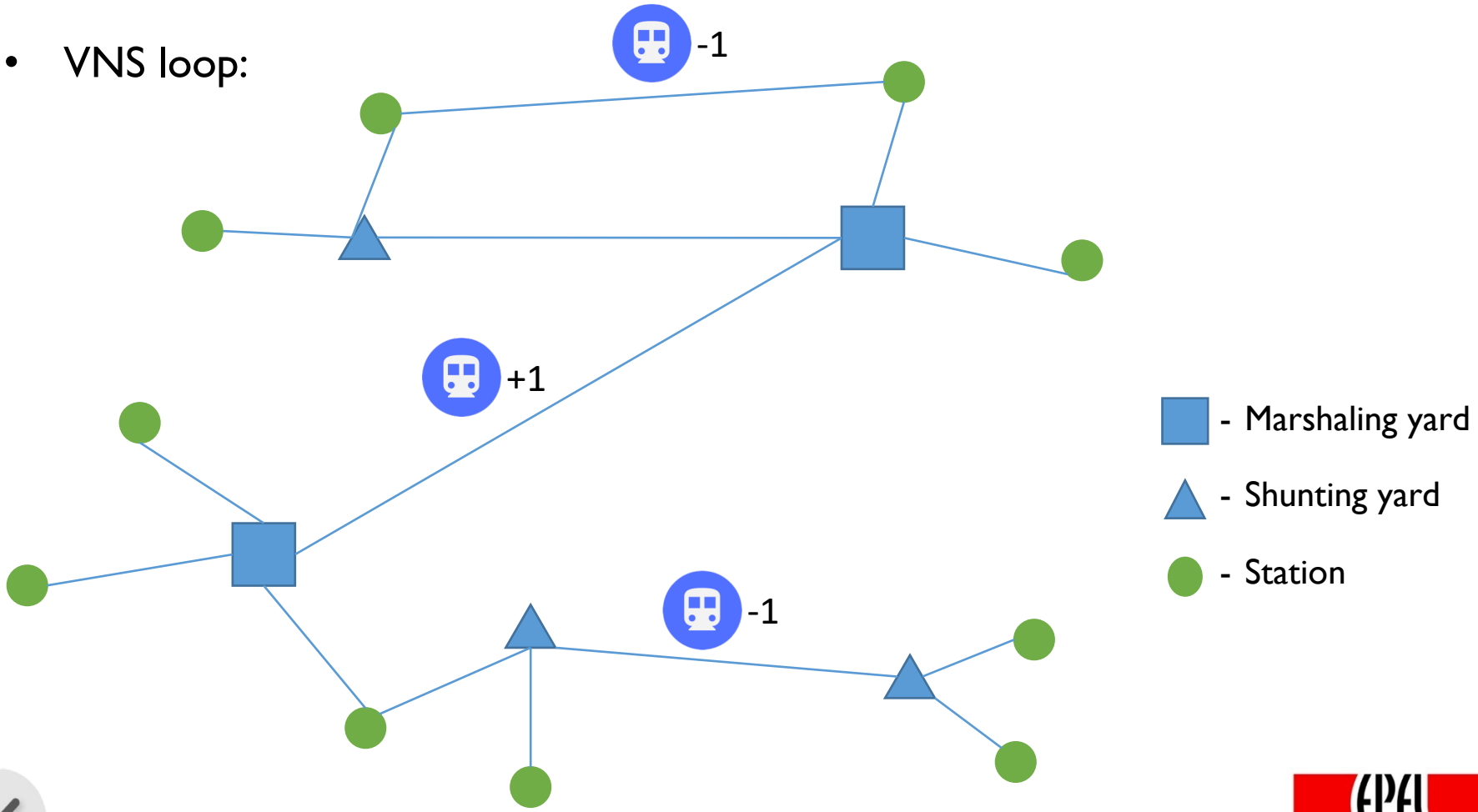


TRANSP-OR

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Heuristic algorithm – Reduction of train number

- Remove all unused trains
- VNS loop:

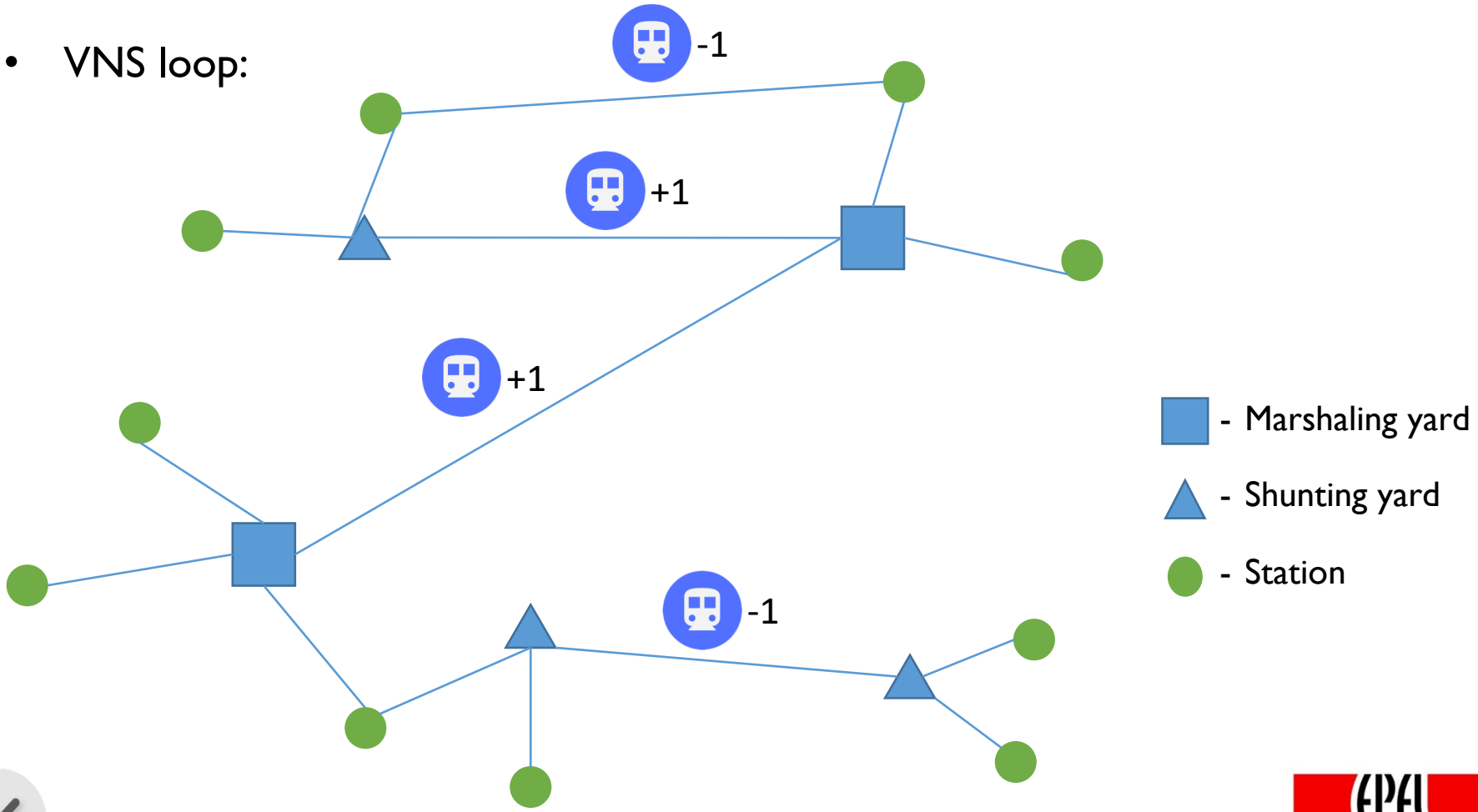


TRANSP-OR

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Heuristic algorithm – Reduction of train number

- Remove all unused trains
- VNS loop:

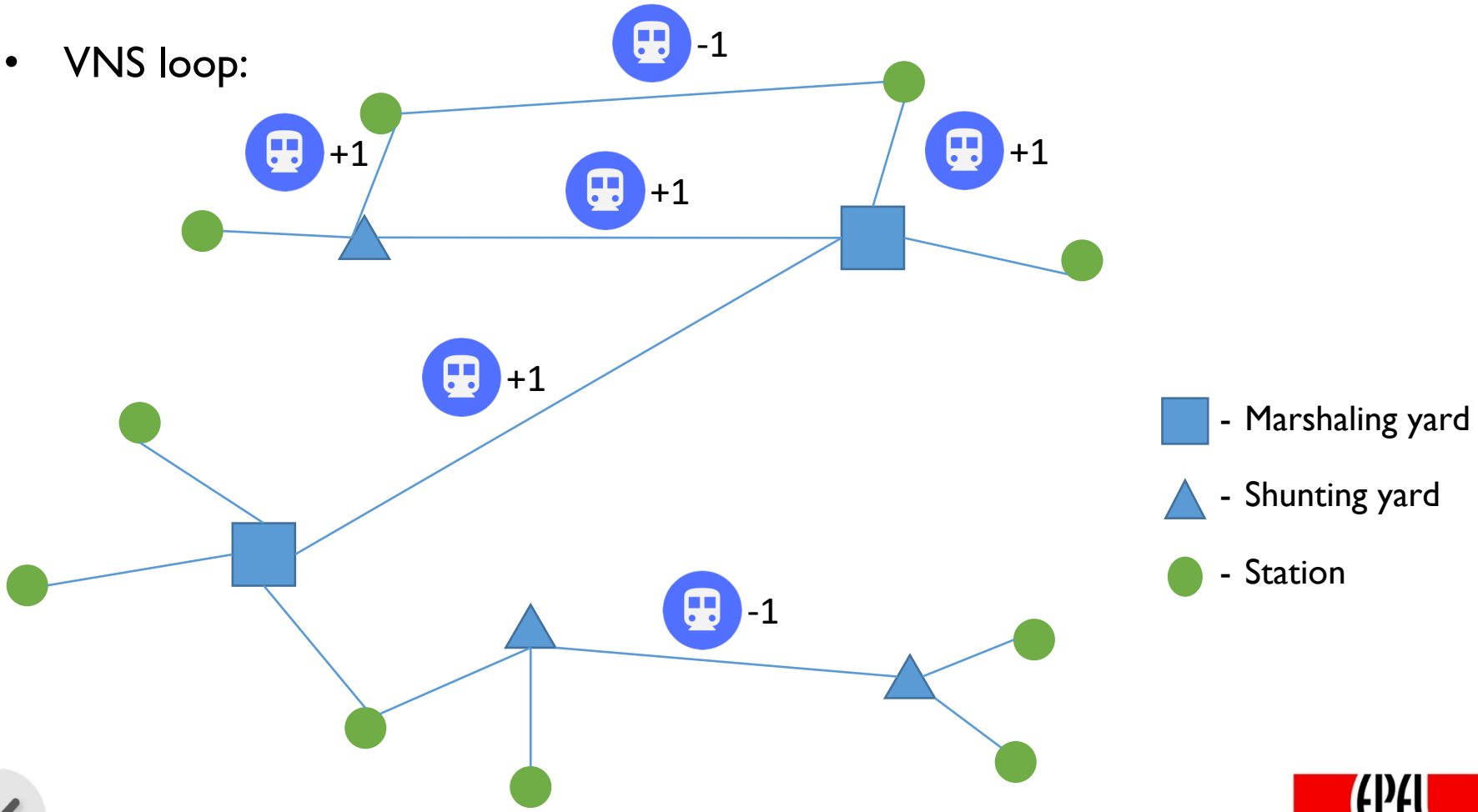


TRANSP-OR

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Heuristic algorithm – Reduction of train number

- Remove all unused trains
- VNS loop:



TRANSP-OR

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Heuristic algorithm – development details

- Developed algorithm is very flexible:
  - Easily extendable with additional neighborhood operators, i.e. network transformations
  - Easy definition of specific initial network states, e.g. all marshaling yards closed, several additional marshaling yards open, etc.
- Algorithm modes:
  - Daily average demand
  - Peak demand

Algorithm results

- Two usage strategies:
  - S1: allowing increase in the number of marshaling yards
  - S2: limiting the number of marshaling yards to the current one
- Initial network state
  - The current network state
  - Changed number and locations of the marshaling and shunting yard

Algorithm results

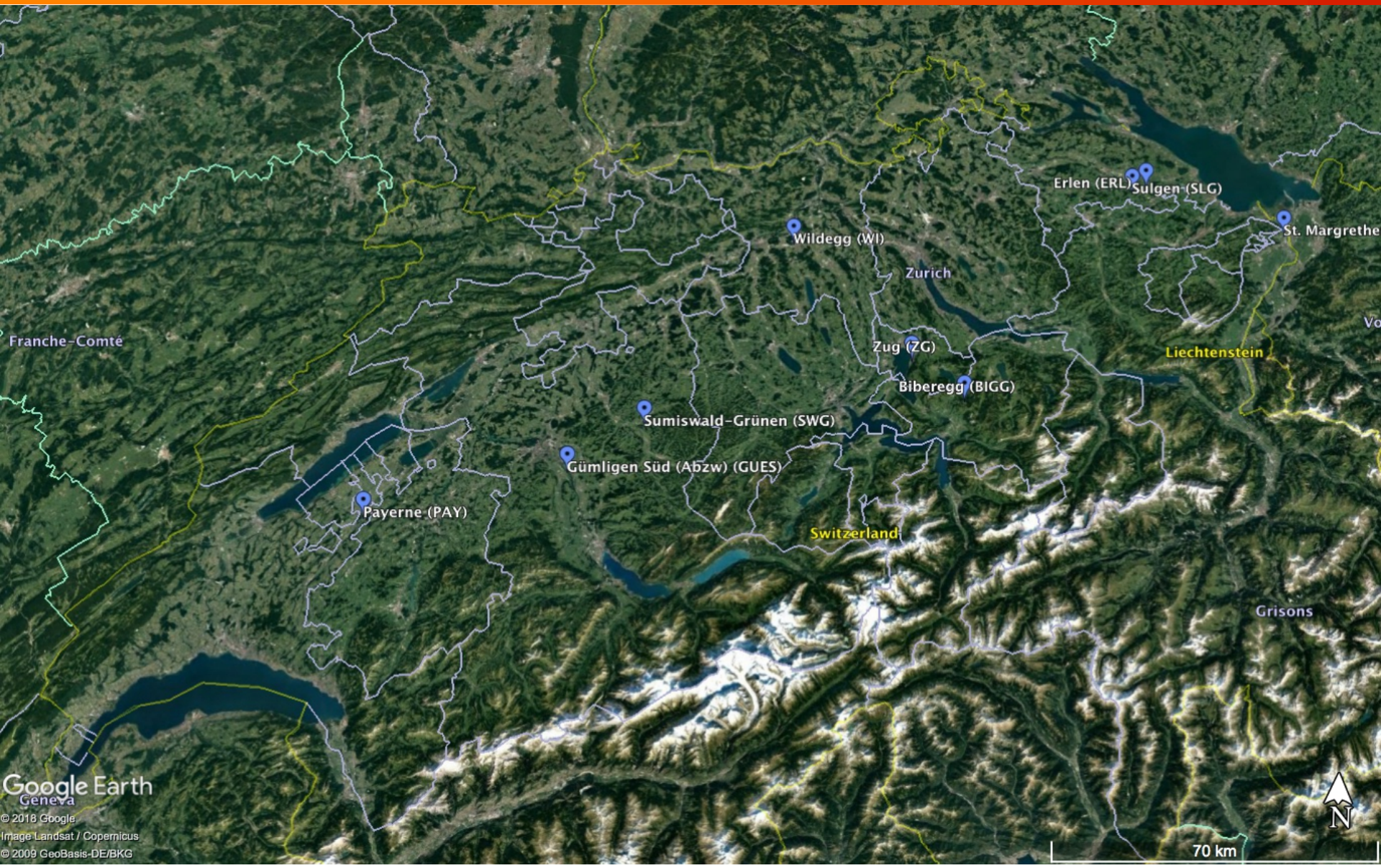
- Best resulting networks obtained from the current network state

Strat.	New MY	Rem. MY	New SY	Rem. SY	Algor.	Run. time	Cost reduct.
S1	6	1	2	2	VNS	2h	**10.01%**
S2	1	2	5	49	ALNS	17h	**16%**

- S1 should in theory yield a better result, but the investigation in this direction was shorter due to a business decision

Algorithm results – Marshaling yards

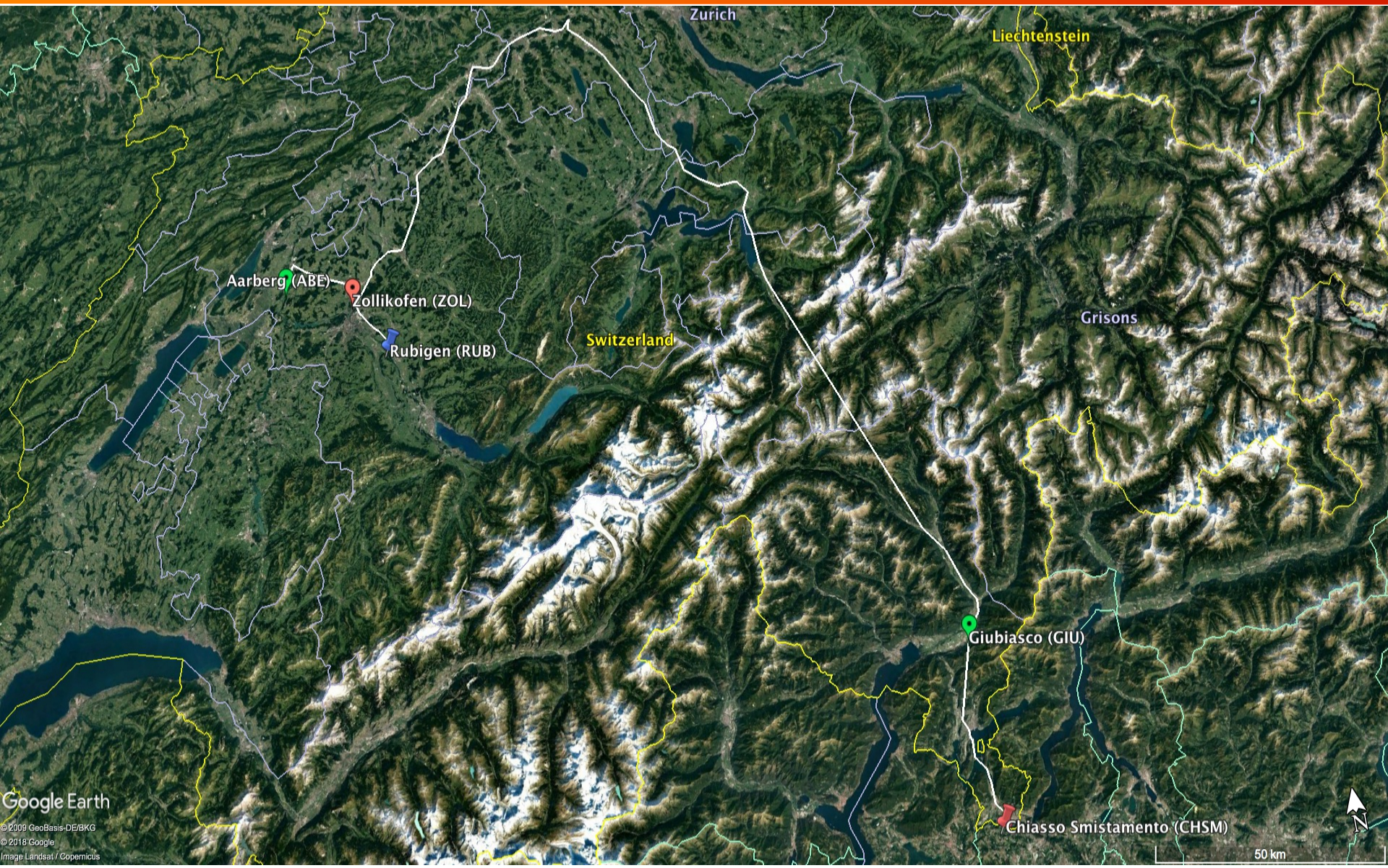
Algorithm results – Shunting yards



Results analysis

- Locomotive and personnel (distance-dependent) costs are dominant over weight-dependent commodity transportation costs
- Costs of yard opening and maintenance are not taken into account
  - Potentially would further reduce the number of yards and their size
  - Could be included in another case study
- New yards can be near the existing ones
  - The objective function has been extended to penalize this situation

Results analysis - Routing



Conclusions

- Developed algorithm explores various network changes, their combinations and their influence to the transportation costs
  - Flexible, easily extendable algorithm
- The algorithm identified network changes resulting in **transportation cost reduction**
- The objective function should be extended with the real **costs of maintenance** of the marshaling and shunting yards
- Algorithm parallelization – performance improvement

Main references

- Bierlaire, M. (2015). *Optimization: principles and algorithms*. EPFL Press, Lausanne, Switzerland.
- R. Z. Farahani, M. Hekmatfar, A. B. Arabani, E. Nikbakhsh: *Hub location problems: A review of models, classification, solution techniques, and applications*. Computers & Industrial Engineering, Vol. 64, 2013, Elsevier, pp. 1096–1109
- F. Ashwash: *Hub Location Problems* (*lecture slides*)
- W. Williams: *Genetic Algorithms: A Tutorial* (*lecture slides*)
- Barnhart, C., Krishnan, N., & Vance, P. H.: *Multicommodity Flow Problems*. In: Floudas, C. A., & Pardalos, P. M. (eds): *Encyclopedia of Optimization*, vol. 14. Springer, 2009.
- B. Gendron, T. G. Crainic, and A. Frangioni: *Multicommodity capacitated network design*. In B. Sanso and P. Soriano (eds.): *Telecommunications Network Planning*, pp. 1-19. Centre for Research on Transportation, Springer, Boston, MA, 1999.
- A. Ceselli: *Network Design and Optimization course* (*lecture slides*)