CIVIL-557

Decision aid methodologies in transportation

Lab 1: Using a mathematical solver (GUROBI)

Tom Haering

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

TRANSP-OR

20.02.2024

EPFL

Overview

- Using a mathematical solver (GUROBI)
  - What is a mathematical solver
  - What is GUROBI?
  - Installing GUROBI
  - Using GUROBI
  - Example
- Container storage problem
- Transportation problem

Using a mathematical solver

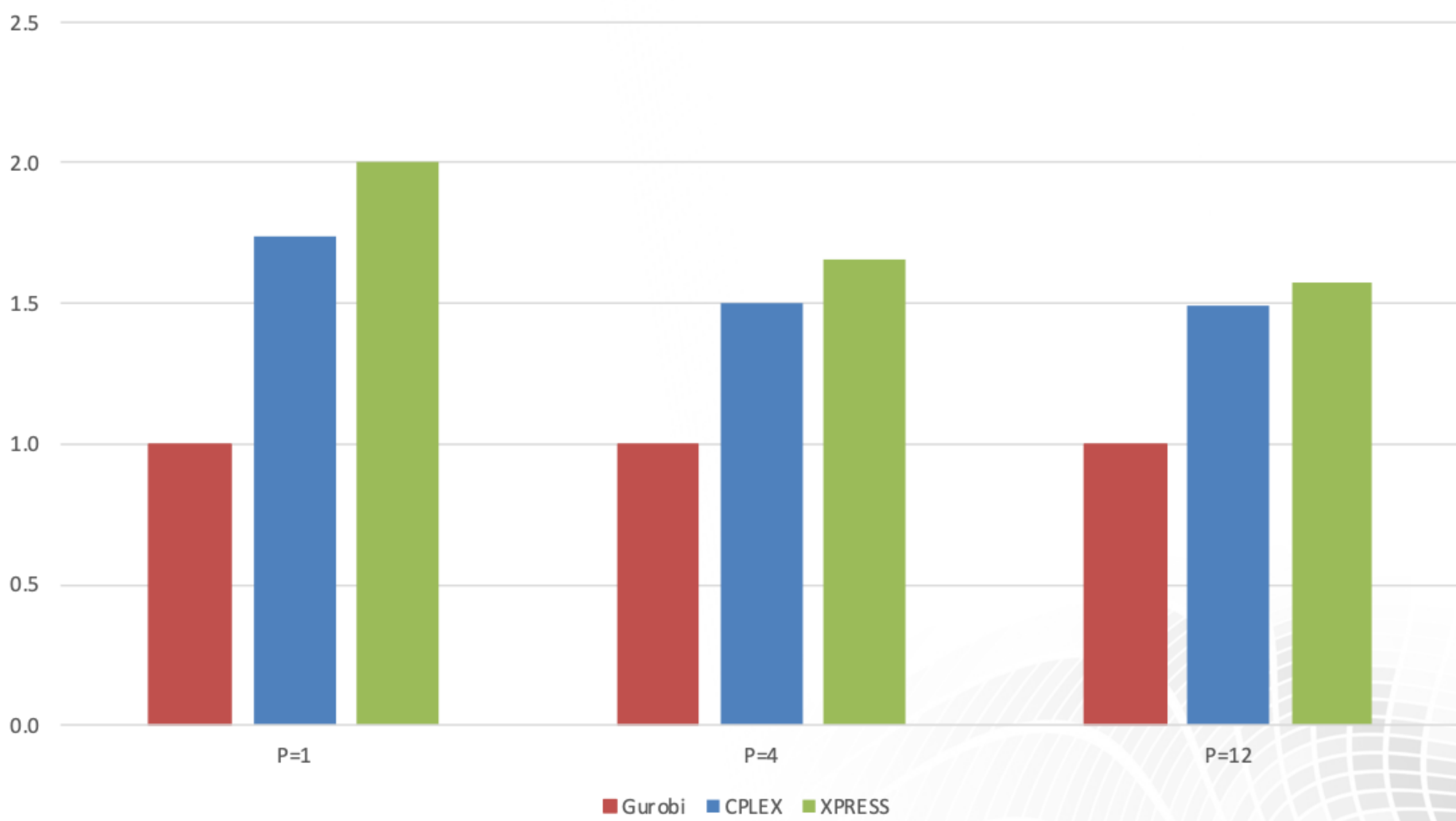
What is a mathematical solver?

- **Input** = mathematical formulation of an optimization problem
- **Output** = optimal solution to the problem
- **Options** for optimality gap, time limits, numerical precision etc.
- Applies several **optimization techniques** in order to find the optimal solution(s).
- Includes **simplex** + many other methods, such as **branch and bound**, cutting planes, probing, **pre-processing**, and heuristics.

What is GUROBI?

GUROBI
OPTIMIZATION

What is GUROBI?



For example, on time to optimality benchmark (87 models) using 4 threads (P=4), CPLEX was 50% slower (1.50) and XPRESS was 66% slower (1.66) than Gurobi.

TRANSP-OR

EPFL

What is GUROBI?

- **Free** for **academic use**
  https://www.gurobi.com/academia/academic-program-and-licenses/

- **No local installation** of optimization suite necessary (compared to for ex. CPLEX)
  - https://support.gurobi.com/hc/en-us/articles/360044290292-How-do-I-install-Gurobi-for-Python

- Latest version (11.0) **compatible** with **Python 3.8+**

- Easy to use with **Pandas** etc.

Installing GUROBI

1. Create **Gurobi account** using epfl email
2. Download **academic license**
3. pip install gurobipy

- Use the python environment you're **most comfortable** with
- **Jupyter notebooks**: great to play around with, no automatic code completion
- To use jupyter notebooks we recommend using **anaconda**, convenient to set up environments etc. https://docs.anaconda.com/anaconda/install/

Installing GUROBI

- Let's take some minutes until everyone can do this in some way:

```
import gurobipy as gp
```

```
m = gp.Model()
m.optimize()
```

```
Set parameter Username
Academic license - for non-commercial use only - expires 2025-01-24
Gurobi Optimizer version 11.0.0 build v11.0.0rc2 (mac64[x86] - Darwin 21.5.0 21F79)

CPU model: Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz
Thread count: 6 physical cores, 12 logical processors, using up to 12 threads

Optimize a model with 0 rows, 0 columns and 0 nonzeros
Model fingerprint: 0xf9715da1
Coefficient statistics:
  Matrix range     [0e+00, 0e+00]
  Objective range  [0e+00, 0e+00]
  Bounds range     [0e+00, 0e+00]
  RHS range        [0e+00, 0e+00]
Presolve time: 0.00s
Presolve: All rows and columns removed
Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    0.0000000e+00   0.000000e+00   0.000000e+00      0s

Solved in 0 iterations and 0.01 seconds (0.00 work units)
Optimal objective  0.000000000e+00
```

Using GUROBI (example)

$$
\begin{aligned}
\text{max} \quad & x + y + 2z \\
\text{s.t.} \quad & x + 2y + 3z \leq 5 \\
& x + y \geq 1 \\
& x,, z, y \in \{0,1\}
\end{aligned}
$$

```python
from gurobipy import GRB

# Create a new model
m = gp.Model("mip1")

# Create variables
x = m.addVar(vtype=GRB.BINARY, name="x")
y = m.addVar(vtype=GRB.BINARY, name="y")
z = m.addVar(vtype=GRB.BINARY, name="z")

# Set objective
m.setObjective(x + y + 2 * z, GRB.MAXIMIZE)

# Add constraint: x + 2 y + 3 z <= 4
m.addConstr(x + 2 * y + 3 * z <= 4, "c0")

# Add constraint: x + y >= 1
m.addConstr(x + y >= 1, "c1")

# Optimize model
m.optimize()
```

First example to solve

$$\begin{aligned} \min \quad & 3x + 2y \\ \text{subject to} \quad & x - y \geq 5 \\ & 3x + 2y \geq 10 \end{aligned}$$

?

Hint:

```
inf = GRB.INFINITY
x = m.addVar(lb = -inf, vtype=GRB.CONTINUOUS)
```

Container storage problem

Exercise

- Solve the **container storage problem** (even distribution)

Storage blocks $i \in \{1, \ldots, B\}$

Parameters

- a_i: Initial number of stored containers in block i
- N: Number of new containers expected to arrive for storage in this period
- B: Total number of blocks in the storage yard
- A: Number of storage positions in each block
- F: The fill-ratio in the whole yard at the end of this period ($(N + \sum_i a_i) / AB$)

Decision variables

- x_i: Number of arriving containers in this period to be stored to block i

$$min \sum_{i=1}^{B} |a_i + x_i - AF|$$

$$s.t. \sum_{i=1}^{B} x_i = N$$

$$x_i \in \mathbb{N} \quad \forall i$$

Exercise

- How can we model an absolute value in a linear program?

$$\min \sum_{i=1}^{B} |a_i + x_i - AF|$$

$$s.t. \sum_{i=1}^{B} x_i = N$$

$$x_i \in \mathbb{N} \qquad \forall i$$

?

Exercise

- **Set** N=15166, B=100, A=600, F = $\frac{N + \sum_i a_i}{AB}$
- a can be read from the numpy file on the moodle (Lab1_a.npy)

```
import numpy as np
```

```
a = np.load("Lab1_a.npy")
```

- **Hint 1:** use **dictionaries** for variables

```
x = {i: m.addVar(vtype=GRB.INTEGER) for i in range(B)}
```

- **Hint 2:** for sums use **gp.quicksum()**

$$\sum_{i=1}^{B} x_i =$$

```
gp.quicksum(x[i] for i in range(B))
```

TRANSP-OR

EPFL

Surprise exercise!

Surprise exercise

- How can we model logical constraints linearly?

$$c_i = \begin{cases} d_i & \text{if } d_i \geq 5 \\ e_i & \text{else} \end{cases}$$

?

Surprise exercise solution

- Introduce auxiliary **binary variables** ω_i and a **large constant** M (can be optimized, i.e. minimized)

$$c_i = \begin{cases} d_i & \text{if } d_i \geq 5 \\ e_i & \text{else} \end{cases}$$

$$\Rightarrow$$

$$c_i \geq d_i - (1 - \omega_i)M$$
$$c_i \leq d_i + (1 - \omega_i)M$$

$$c_i \geq e_i - \omega_i M$$
$$c_i \leq e_i + \omega_i M$$

$$d_i \leq 4 + \omega_i M$$
$$d_i \geq 5 - (1 - \omega_i)M$$

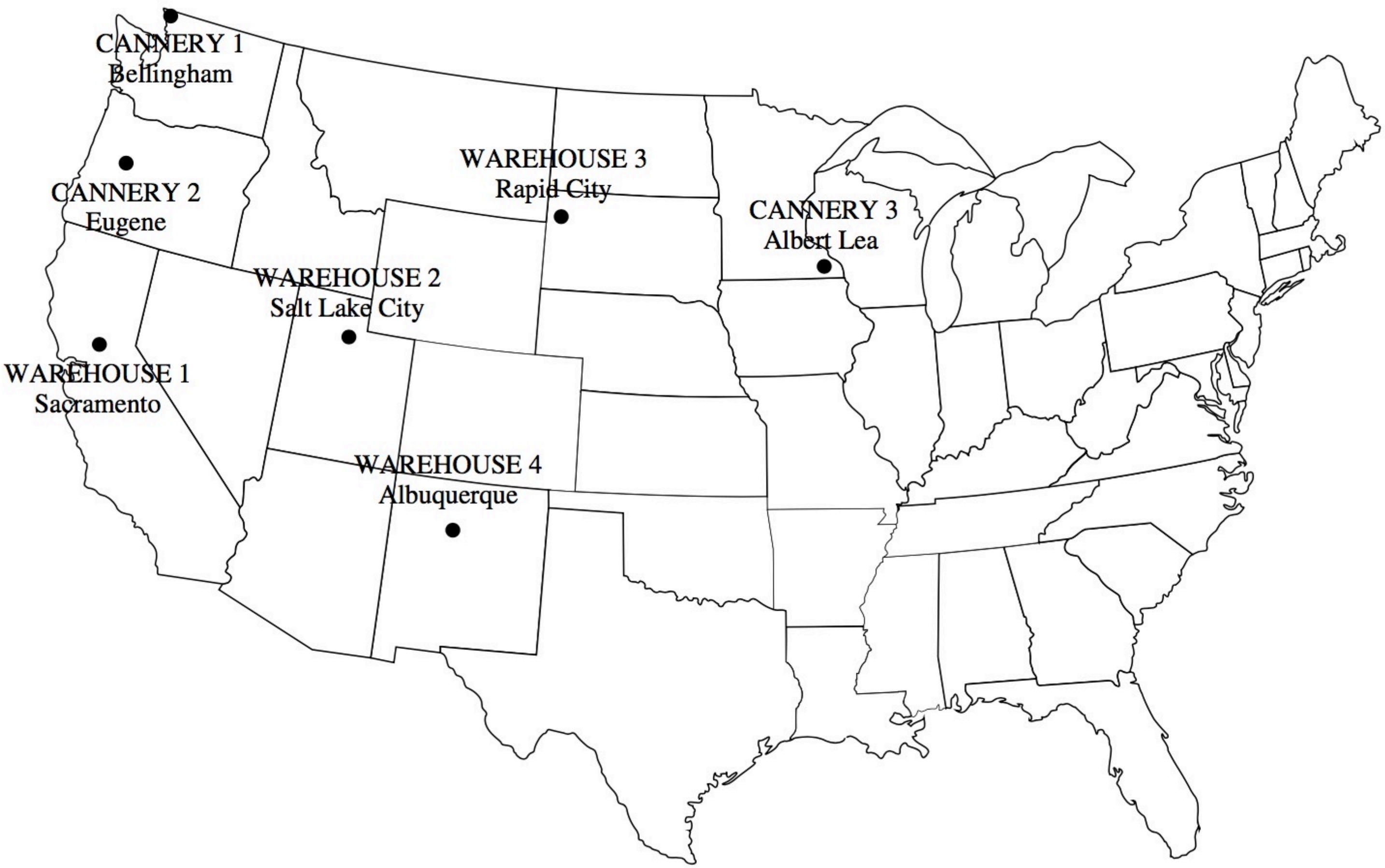
- Use with caution. Large **computational burden, weak relaxations**

Transportation Problem (Bonus exercise)

Transportation Problem

One of the main products of the P&T COMPANY is canned peas. The **peas are prepared at three canneries** (near Bellingham, Washington; Eugene, Oregon; and Albert Lea, Minnesota) and then **shipped by truck to four distributing warehouses** in the western United States (Sacramento, California; Salt Lake City, Utah; Rapid City, South Dakota; and Albuquerque, New Mexico). Because the **shipping costs are a major expense**, management is initiating a study to **reduce them as much as possible**. For the upcoming season, **an estimate has been made of the output from each cannery**, and **each warehouse has been allocated a certain amount from the total supply of peas**. This information (in units of truckloads), along with the shipping cost per truckload for each cannery-warehouse combination, is given in Table 8.2 (and is provided as data). Thus, there are a **total of 300 truckloads to be shipped**. The problem now is to determine which plan for **assigning these shipments to the various cannery-warehouse combinations** would **minimize the total shipping cost.**

Transportation Problem – Layout



Transportation Problem – Table 8.2

TABLE 8.2 Shipping data for P & T Co.

		Shipping Cost ($) per Truckload				
		Warehouse				
		1	**2**	**3**	**4**	**Output**
Cannery	1	464	513	654	867	75
	2	352	416	690	791	125
	3	995	682	388	685	100
Allocation		80	65	70	85	

Transportation Problem

- Formulate the problem mathematically
- Implement the problem and report the optimal solution
- Solve the problem for a large instance (use N= 20, M = 30, and files Lab1_cost.npy, Lab1_ output.npy and Lab1_ allocation.npy).