

# A Generalized Vehicle Routing Problem Optimization based on Gavish and Graves Formulation (1978)

Christos - Chrysovalantis Paraschakis    Dimitrios Charistes

University of Western Macedonia -  
Department of Electrical and Computer Engineering

January 30, 2026

# Outline

- 1 Problem Formulation
- 2 Proposed Methodology
- 3 Results and Analysis
- 4 Key Observations
- 5 Conclusion

# Introduction

- The GVR problem that was modeled is the Ganish and Graves (1976) formulation.
- The goal of the GVRP is to minimize total traversal costs across the set of arcs  $A$  by servicing all cluster demands using exactly one node within it.

$$\min \sum_{(i,j) \in A} c_{ij} x_{ij}$$

where:

- $A$  is the set of all arcs in the directed Graph  $G(V, A)$ .
- $c_{ij}$  is the cost of the arc  $(i, j)$
- $x_{ij} \in \{0, 1\}$  is the arc  $(i, j)$ . If the arc is used it is 1.

# Problem Formulation

- Modeled as an MILP with the arc variable being a integer and the flow variable by implication is integer although it is modeled as continuous.
- Constraints:
  - Cluster Visit Rule
  - Vehicle Usage Rule
  - Flow Conservation Rule
  - Commodity Flow Balance
  - Capacity Constraint
- Objective: **Minimize travel cost.**

# GVRP Mathematical Formulation

## Objective Function:

$$\text{Minimize} \quad \sum_{(i,j) \in A} c_{ij} x_{ij}$$

## Subject to:

- Cluster Visit Rule:

$$x(\delta^+(C_k)) = 1, \quad x(\delta^-(C_k)) = 1 \quad \forall k \in M \setminus \{0\}$$

- Vehicle & Flow Conservation:

$$\begin{aligned} x(\delta^+(C_0)) &= K, & x(\delta^-(C_0)) &= K \\ x(\delta^+(i)) &= x(\delta^-(i)) & \forall i \in V \end{aligned}$$

- Commodity Flow Balance:

$$f(\delta^+(i)) - f(\delta^-(i)) = \frac{1}{2} q_{\alpha(i)} (x(\delta^-(i)) + x(\delta^+(i)))$$

- Capacity & Domain Constraints:

$$\begin{aligned} q_{\alpha(i)} \leq f_{ij} &\leq (Q - q_{\alpha(i)}) x_{ij} & \forall (i,j) \in A \\ x_{ij} &\in \{0, 1\} & \forall (i,j) \in A. \end{aligned}$$

# Using Gurobi as a Solver

- 1. **Instance Generation (gvrppgenerator)**
  - We create randomized problem instances using our custom generator.
  - **Inputs:** Grid size ( $N \times N$ ), Total number of Vertices ( $V$ ), Number of Clusters, Vehicles ( $K$ ), Capacity ( $Q$ ), and the number of problems.
  - **Output:** A .txt file containing all the information needed about the custom problem.
- 2. **Model Construction & Execution**
  - The generated data is loaded into **Gurobi**, and the SCF constraints (Equations 1-9) are initialized.
  - The standard `model.optimize()` function is called, triggering Gurobi's internal **Branch-and-Cut** algorithm to find the best solution.
- 3. **Solution Extraction**
  - We extract the active arcs ( $x_{ij} = 1$ ) to reconstruct the optimal route.

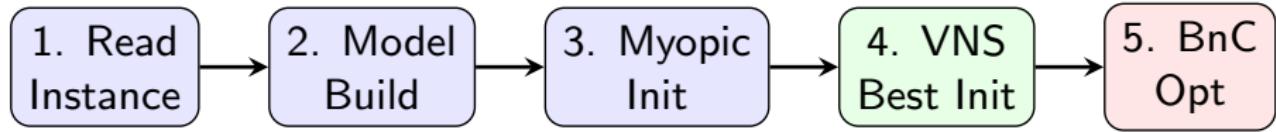
# Branch-and-Bound Method

The Proposed methodology is based on a Branch and Cut algorithm. The problem is remodeled as LP in GurobiPy API. It will be compared to the PyOmo model.

- Utilizes **Best-First Search** technique.
- The nodes are inserted into a min-heap based on the relaxed objective solution.
- The most fractional variable of the node is explored first  
(Most-Fractional technique) →  $\min_{i \in O_V} (value_i - 0.5)$
- A Cutting Plane method is implemented inside this algorithm.

# Heuristics

- Heuristics are implemented to speed up the branching algorithm
- First, a Myopic Heuristic to extract an initial cost
- Then a metaheuristic VNS to improve the initial
- VNS's cost becomes the upper bound of the BnC



# Variable Neighborhood Heuristic

- Parameters  $k_{max}$  and max\_iterations are set.
- Finds routes through the process:
  - Shaking then local search
  - Increases the intensity of shaking (if no improvement)
  - The heuristic moves to the improved solution  $S''$  only if feasibility criteria are checked.

$$(S, k) \leftarrow \begin{cases} (S'', 1) & \text{if } C(S'') < C(S) - \epsilon \\ (S, k + 1) & \text{otherwise} \end{cases} \quad (1)$$

# Capacity Cut

- The cut implemented in this algorithm is the *Rounded Capacity Inequality*
- The cut enforces that at least  $2r(S)$  arcs must cross the subset's boundary (accounting for both entry and exit).

$$r(S) = \left\lceil \frac{\sum_{k \in S} q_k}{Q} \right\rceil \leq \frac{1}{2} \sum_{(i,j) \in \delta(S)} x_{ij} \quad (2)$$

where:

- $S$  is a subset
- $D(S)$  its total demand
- and  $r(S)$  min vehicles required

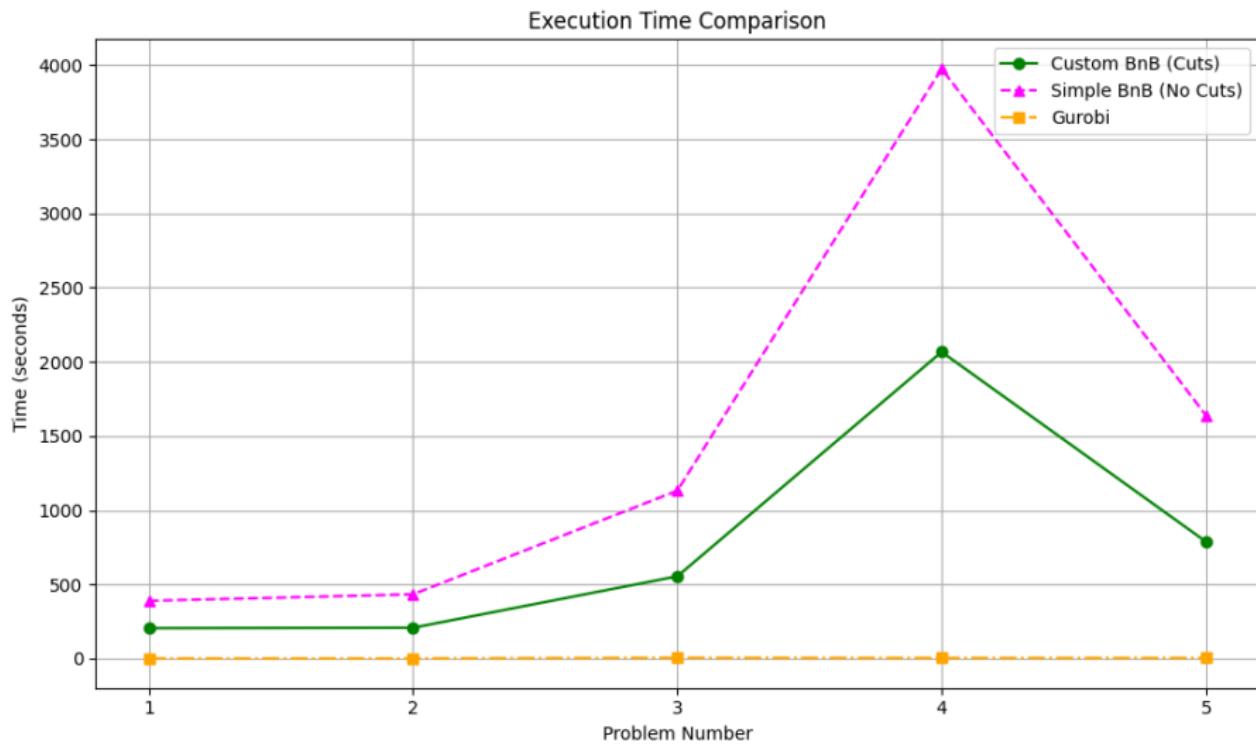
# Experimental Setup

- VNS parameters:  $k_{max} = 4$  and  $max_{it} = 100$ .
- 5 problems with grid dimensions:  $\{10, 12, 14, 15, 35, 45\}$  generated.
- All Problems are solved by the BnC (final), BnB (simple) and Gurobi.

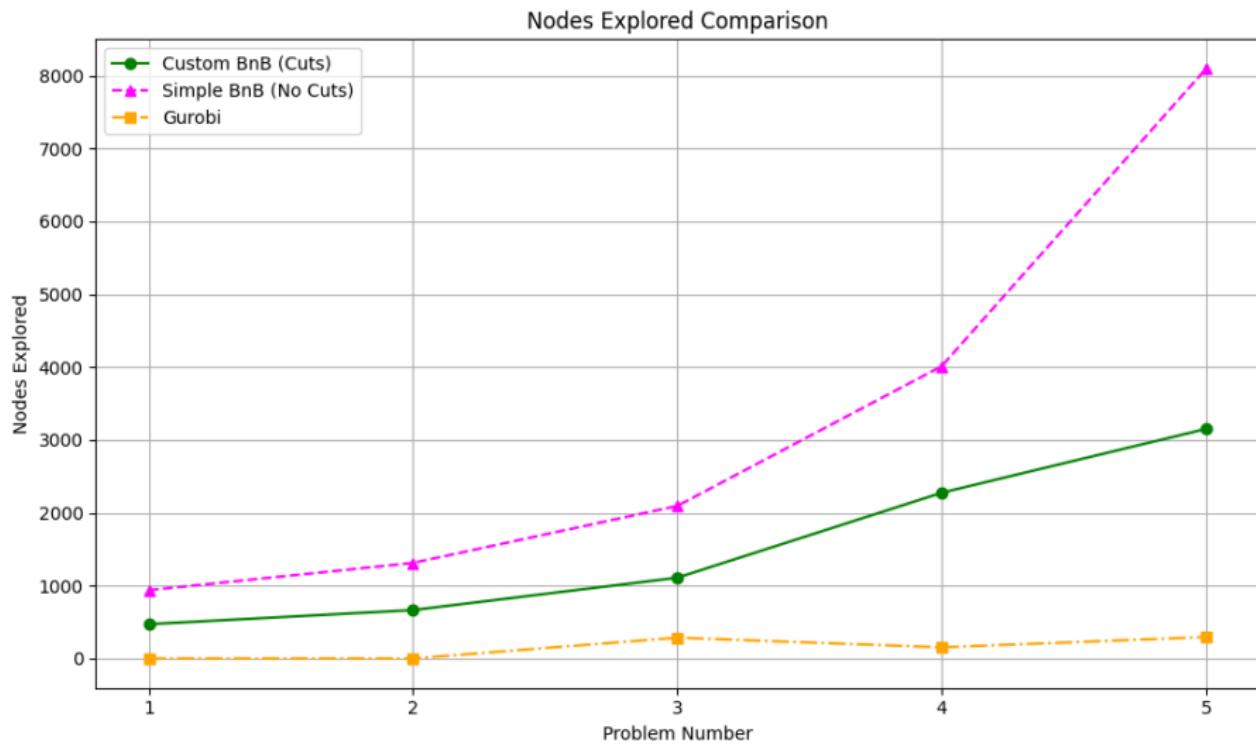
# Performance Metrics

- Key Metric 1: **Computation Time.**
- Key Metric 2: **Nodes Explored**
- Comparison between:
  - ILP Model (Gurobi)
  - Custom Branch and Cut Method
  - Simple Branch and Bound

# Execution Time



# Nodes Explored



# Key Observations

- The execution time increases with the size of the problem (exponential growth) but no distinct correlation with problem's order.
- The state-of-the-art has the best performance.
- Simple BnB explores twice the number of nodes of BnC. This is attributed to the Cut method
- BnB provides competitive performance for small-to-medium instances.

# Conclusion

- The Problem's order is not the only difficulty metric. A computational analysis that accounts more factors for problem's difficulty growth.
- The proposed Branch and Cut can be further optimized from the heuristic aspect.
- More Cutting Plane techniques can be added to further improve the nodes explored number.

# Thank You!