Evaluation Report: Spatio-Temporal Graph Coarsening for Vehicle Routing Problem with Time Windows (VRPTW)

1 Introduction

This report analyzes a spatio-temporal graph coarsening approach for Vehicle Routing Problems with Time Windows (VRPTW) using Solomon datasets. It evaluates two heuristic solvers, Greedy and Clarke and Wright Savings, across four scenarios:

1. **Uncoarsened Greedy:** Solving directly on the original graph with the Greedy heuristic.
2. **Uncoarsened Savings:** Solving directly on the original graph with the Clarke and Wright Savings heuristic.
3. **Inflated Greedy:** Solving on the coarsened graph using the Greedy heuristic, followed by route inflation.
4. **Inflated Savings:** Solving on the coarsened graph using the Clarke and Wright Savings heuristic, followed by route inflation.

The objective is to assess coarsening's impact on solution quality (distance, time, vehicles) and feasibility (time window, capacity violations).

2 Key Findings and Overall Performance

2.1 Pervasive Infeasibility

A critical observation across almost all problem instances and solution methods is the `Is Feasible: False` status. This indicates that the generated routes frequently violate time windows or vehicle capacity constraints. While the coarsening process and subsequent solving aim to find efficient routes, the heuristics employed (Greedy and Savings) are not robust enough to consistently produce fully feasible solutions for these complex VRPTW instances under strict time window constraints. The high number of `Time Window Violations` and `Capacity Violations` confirms this.

2.2 Demand Coverage - A Crucial Distinction

A significant finding is the inconsistent `Total Demand Served` by the **Uncoarsened Greedy** solver. For many instances (e.g., C101.csv, C102.csv, C105.csv), the `Total Demand Served` by Uncoarsened Greedy is notably lower than the total demand of the problem (which is consistently served by the coarsened approaches). This implies that the Uncoarsened Greedy solver often fails to visit all customers, making its other metrics (Total Distance, Total Route Duration) incomparable and misleading.

In contrast, both **Inflated Greedy** and **Inflated Savings** consistently report `Total Demand Served` values that match the expected total demand for the respective datasets (e.g., 1680.00 for C-type, 1366.00 for R-type). This is a crucial success for the coarsening approach, as it enables the solvers to cover all customer demands, even if feasibility remains a challenge.

3 Detailed Comparison by Scenario

3.1 Uncoarsened Solvers

- **Uncoarsened Greedy:**

- **Vehicles:** Produces a moderate to high number of vehicles (e.g., 30-42 for C1xx, 15-40 for R1xx).
- **Distance/Time:** Shows relatively lower total distances and route durations compared to Uncoarsened Savings.
- **Feasibility:** Consistently `False` due to time window violations, and critically, often fails to serve all customer demand.
- **Conclusion:** While seemingly "better" in some metrics than Uncoarsened Savings, its failure to serve all demand makes it an incomplete solution.

- **Uncoarsened Savings:**
  - **Vehicles:** Exhibits an exceptionally high number of vehicles (consistently 92 for C-type and R-type problems), which is often close to the total number of customers. This indicates that very few merges are successfully performed, or that the algorithm is creating individual routes for most customers.
  - **Distance/Time:** Consequently, its `Total Distance` and `Total Route Duration` are extremely high.
  - **Feasibility:** Suffers from a very high number of `Time Window Violations` and `Capacity Violations` (often 2 capacity violations), leading to consistent `False` feasibility.
  - **Conclusion:** The Uncoarsened Savings solver performs very poorly on these datasets, suggesting a fundamental issue with its implementation, its suitability for these specific VRPTW instances, or overly strict feasibility checks preventing merges.

3.2 Coarsened and Inflated Solvers

- **Inflated Greedy (Coarsening + Greedy):**
  - **Vehicles:** This combination consistently yields the **lowest number of vehicles** across all methods (e.g., 7-14 for C1xx, 2-4 for C2xx, 4-14 for R1xx). This is a major success of the coarsening strategy.
  - **Distance/Time:** Achieves the **lowest total distances** and `Total Route Duration` among all scenarios, indicating superior route efficiency.
  - **Demand Coverage:** Successfully serves all customer demand.
  - **Feasibility:** While still predominantly `False` for feasibility, the number of `Time Window Violations` is significantly reduced compared to uncoarsened methods, and `Capacity Violations` are consistently zero. Notably, for `C205.csv` and `C206.csv`, it even achieves `True` feasibility.
  - **Conclusion:** This is the **most effective approach** observed, demonstrating that coarsening can significantly improve the efficiency and vehicle utilization of the Greedy heuristic, and enable full demand coverage.

- **Inflated Savings (Coarsening + Savings):**
  - **Vehicles:** Shows a reduced number of vehicles compared to Uncoarsened Savings, but generally higher than Inflated Greedy.
  - **Distance/Time:** Improved total distance and route duration compared to Uncoarsened Savings, but still worse than Inflated Greedy.
  - **Demand Coverage:** Successfully serves all customer demand.
  - **Feasibility:** Continues to exhibit a high number of `Time Window Violations` and `Capacity Violations`, leading to consistent `False` feasibility.
  - **Conclusion:** While coarsening helps the Savings solver to serve all demand and reduces its vehicle count compared to its uncoarsened counterpart, it still struggles with feasibility and overall efficiency compared to the Inflated Greedy approach.

4 Impact of Spatio-Temporal Coarsening

The spatio-temporal graph coarsening process has a profound and generally positive impact on the solution quality, especially when combined with the Greedy solver:

- **Significant Vehicle Reduction:** The most striking benefit is the dramatic reduction in the number of vehicles required. By merging nearby and temporally compatible nodes into super-nodes, the problem size is reduced, allowing the solver to find solutions with fewer routes on the coarser graph, which then translate to fewer routes on the original graph after inflation.
- **Improved Efficiency:** Coarsening leads to substantially lower total distances and total route durations, indicating more efficient vehicle utilization and shorter travel times.
- **Full Demand Coverage:** Crucially, the coarsening and inflation process enables both Greedy and Savings solvers to serve all customer demands, a task the uncoarsened Greedy solver often failed to achieve.
- **Reduced Violations (for Greedy):** While not eliminating all violations, coarsening significantly reduces the number of time window violations when paired with the Greedy solver.

5 Limitations and Recommendations

5.1 Limitations of Current Approach

- **Heuristic Feasibility:** The primary limitation is the inability of the current Greedy and Savings heuristics to consistently produce fully feasible solutions (i.e., zero time window and capacity violations) for all VRPTW instances, even with coarsening. This suggests that the problem's constraints are very tight, or the heuristics are not sophisticated enough to navigate them perfectly.
- **Savings Solver Performance:** The `Uncoarsened Savings` solver's performance is highly problematic. Its extremely high vehicle counts and violations suggest a fundamental flaw in its current implementation, particularly in how it handles merges or respects time windows and capacities during the merging process. The `_check_merge_feasibility` function might be overly strict, or the saving calculation itself might not be adequately capturing the true benefits for VRPTW.
- **Inflation Precision:** While inflation restores the original nodes, the quality of the final route is inherently limited by the quality of the solution found on the coarsened graph.

5.2 What to do Next

1. **Debug and Refine Savings Solver:** Thoroughly review and debug the `SavingsSolver` implementation, focusing on:
   - The `_calculate_savings` function to ensure it correctly prioritizes merges that are beneficial for VRPTW (considering time windows).
   - The `_check_merge_feasibility` function to ensure it accurately assesses feasibility without being overly restrictive.
   - The merging logic to ensure it correctly updates routes and customer-to-route mappings. The high number of vehicles for the uncoarsened Savings solver is a strong indicator that merges are not happening as expected.
2. **Implement Post-Optimization:** Introduce a local search or metaheuristic (e.g., 2-opt, Tabu Search, Simulated Annealing) as a post-optimization step on the *inflated* routes. This can significantly improve solution quality and feasibility by making small, iterative improvements to the routes.
3. **Explore Advanced Heuristics/Metaheuristics:** Integrate more advanced VRPTW heuristics or metaheuristics (e.g., Adaptive Large Neighborhood Search, Genetic Algorithms) to operate on the coarsened graph. These methods are generally more capable of handling complex constraints and finding near-optimal solutions.

4. **Parameter Tuning:** Systematically tune the coarsening parameters (α, β, P, `radiusCoeff`) to observe their impact on solution quality and feasibility.

5. **Visualize Solutions:** Implementing a visualization tool for the routes on the original and coarsened graphs would provide invaluable insights into the behavior of the algorithms and the nature of the generated solutions.