Report on the synergies of the Artificial Intelligence approaches for the Set Team Orienteering Problem (STOP)

1. Introduction

In optimization, the Greedy Randomized Adaptive Search Procedure (GRASP) is a well-established technique that utilizes random initialization of starting points to mitigate the risk of local optima and explore the solution space more broadly. This randomness enhances the algorithm's ability to discover diverse solutions by avoiding premature convergence. However, the potential to refine this approach by leveraging Machine Learning (ML) to intelligently select starting points has not been extensively explored. Our research introduces a novel application of ML to the Set Team Orienteering Problem, aiming to enhance GRASP's effectiveness. By employing ML techniques to optimize the selection of initial solutions, we seek to improve the algorithm's performance and solution quality, building on GRASP's foundational principles while incorporating advanced predictive capabilities.

2. The problem

The Set Team Orienteering Problem (STOP), introduced by Nguyen et al. (2025), extends the Set Orienteering Problem (SOP) into a multi-vehicle framework. This combinatorial optimization challenge seeks to maximize the total profit collected by a fleet of vehicles operating within a constrained timeframe. The problem is defined on a graph where nodes are grouped into disjoint clusters; the consolidated profit of a cluster is realized by visiting at least one distinct node within it. This structure effectively models modern logistics scenarios, such as last-mile distribution to smart lockers or neighborhood hubs, where goods are routed to aggregated locations rather than individual doorsteps. Consequently, solving the STOP is critical for enhancing operational effectiveness and profitability in decentralized supply chains.

3. The approach

Our core algorithm is a multi-restart matheuristic based on Tabu Search. In general, restarting the search process from various points aims to explore the solution space more thoroughly. The classic GRASP approach includes random selection of starting points. However, integrating Machine Learning (ML) into this process offers a transformative leap. The ML-enhanced approach employs sophisticated techniques to refine and optimize initial solution selection, which is pivotal for improving overall outcomes.

This approach consists of three key steps:

1. **Generating Initial Solutions**: This step focuses on producing a broad and varied set of initial solutions to ensure a rich exploration of the solution space.
2. **Feature Association**: Each solution is linked with a set of meaningful features that capture its essential characteristics and semantics. These features provide critical insights into the solution's potential, helping to evaluate its promise.
3. **Machine Learning Evaluation**: A Machine Learning model is employed to analyze the extracted features and discern which solutions hold the most promise. This model leverages the feature data to predict and identify the most viable solutions from the initial set.

3.1 Generating Initial Solutions

The generation of initial solutions is a critical step in the optimization process. For this, a Minimum Insertions algorithm with a Restricted Candidate List (RCL) was employed, which facilitates the creation of diverse and high-quality initial solutions. This algorithm is designed to ensure that the initial solutions cover a wide range of possibilities, thus providing a robust foundation for the subsequent optimization process.

3.2 Feature Association

To evaluate the quality of each generated solution, 25 distinct features were calculated, categorized into two main groups: Solution Features and Model Features. This dual approach allows for a comprehensive understanding of each solution's performance and the context in which it was generated.

I) **Solution Features**: These metrics reflect the characteristics of the solution and its routing.

Metric Name	**Description**	**Calculation Details**
intersections	Total intersections within a solution	Calculates intersections between all route pairs, divided by total number of edges
longest_edge	Longest edge between two customers	Finds the maximum distance between any two customers (excluding depot), divided by t_max
average_edge_from_depot	Average distance of depot edges	Mean of depot edges (start and end of routes), divided by t_max
average_distance_from_centroids	Mean distance between route centroids	Average distance between centroids of different routes, divided by t_max
max_route_width	Maximum route width in solution	Maximum width among all routes in the solution, divided by t_max
max_rad_corner	Maximum angle between customers and depot	Largest angle formed by any two customers and the depot across all routes
route_compactness_distance	Route compactness based on distance	Mean distance from gravity line across all routes, divided by t_max
route_compactness_angle	Route compactness based on angle	Mean angle deviation from gravity line for each route
route_max_depth	Maximum distance from depot	Furthest distance any customer is from the depot, divided by t_max
var_node_pct_difference	Variance in node distribution	Variance of percentage differences from average nodes per route
min_unvisited_node_distance	Minimum distance to unvisited nodes	Closest distance from solution to any unvisited node, divided by t_max
percentage_of_collected_profits	Profit collection efficiency	Ratio of collected profit to total possible profit in the model
percentage_of_visited_clusters	Cluster visitation rate	Ratio of visited clusters to total available

		clusters
average_percentage_of_tmax_used	Time utilization efficiency	Average percentage of t_max time constraint used across all vehicles
close_unvisited_profit	Nearby missed opportunities	Unvisited profit within 5% of t_max distance from solution, as % of possible profit

II) **Model Features**: These metrics capture essential contextual variables influencing the solution's performance.

Metric Name	**Description**	**Calculation Details**
number_of_vehicles	Vehicle count	Total number of vehicles available in the model
number_of_clusters	Cluster count	Total number of clusters available in the model
zone1_profit_percentage	Zone 1 profit distribution	% of possible profit within Zone 1 (0–25% of t_max/2 from depot)
zone2_profit_percentage	Zone 2 profit distribution	% of possible profit within Zone 2 (25–50% of t_max/2 from depot)
zone3_profit_percentage	Zone 3 profit distribution	% of possible profit within Zone 3 (50–75% of t_max/2 from depot)
zone4_profit_percentage	Zone 4 profit distribution	% of possible profit within Zone 4 (75–100% of t_max/2 from depot)
quartile1_profit_percentage	Quartile 1 profit distribution	% of possible profit in quadrant where x ≥ depot.x and y ≥ depot.y
quartile2_profit_percentage	Quartile 2 profit distribution	% of possible profit in quadrant where x < depot.x and y ≥ depot.y
quartile3_profit_percentage	Quartile 3 profit distribution	% of possible profit in quadrant where x < depot.x and y < depot.y
quartile4_profit_percentage	Quartile 4 profit distribution	% of possible profit in quadrant where x ≥ depot.x and y < depot.y

3.3 Machine Learning Evaluation

The features derived from the solutions were utilized as inputs for a Machine Learning model, specifically designed to identify the most promising solutions among the generated candidates. We primarily focused on tree-based algorithms, such as Random Forest, XGBoost, and LightGBM, due to their robustness and effectiveness in handling complex feature interactions. Various labeling strategies were explored, employing both classification and regression methodologies.

To build a comprehensive training dataset, we conducted a 20-restart run for each dataset, utilizing random initializations. To expand the dataset size, solutions were not only taken from restart 0 but also from those appearing up to *iterations_before_best* iterations before the best solution. This approach, with a *step* parameter to avoid very similar solutions, ensured a diverse set of initial solutions.

Two primary approaches were utilized for labeling solutions: Regression and Classification. Each approach was tailored to evaluate and categorize solutions based on their predicted quality.

1. **Regression**:
   - *General Logic*: In regression, the goal was to quantify the quality of a solution by normalizing its performance relative to other solutions. This approach provides a continuous measure of how well a solution performed.
   - *Specific Implementation (**Regression 0-1**)*: a min-max normalization formula was applied:

$$Label = \frac{Restart\ Best - Dataset\ Worst}{Dataset\ Best - Dataset\ Worst}$$

   This approach assigns values closer to 1 to solutions that have evolved to be near the best solution found for the dataset, while values closer to 0 are given to solutions that remain further from this best solution.

2. **Classification**:
   - *General Logic:* Binary Classification aimed to categorize solutions into discrete categories, and specifically promising or not promising. This method provides a clear, binary distinction, making it easier to identify which solutions meet the criteria for further consideration and which do not.
   - *Specific Implementation (**Classification 0.90**)*: The classification method used the same normalization formula as Regression. However, it applied a threshold to this normalized score:
     - If the normalized score exceeded 0.90, the solution was labeled as promising (1).
     - If the score was 0.90 or below, it was labeled as not promising (0).

A key aspect of the approach is that a separate model is trained for each dataset category, ensuring that the category's own rows are excluded from the training set. This means that when identifying the most promising solutions at the start of a run, the model used has not been trained on that specific dataset. This approach demonstrates that the features indicating a solution's promise are transferable across different dataset instances, enhancing the generalizability and robustness of the proposed method.

3. Results

The Local Search algorithm was configured to operate with 20 restarts, and the Machine Learning component selected from a pool of 1,000 initial solutions to ensure diverse and high-quality candidate solutions. To evaluate the efficacy of our approach, we established two baseline methodologies:

- **Baseline 1**: An algorithm using 20 random initializations.
- **Baseline 2**: An algorithm that selects the 20 highest profit solutions from the pool of 1,000 initial solutions.

The evaluation dataset was derived from the single-vehicle SOP, by assigning a vehicle number $v \in \{2, 3\}$.

The evaluation focused on the 10 most challenging dataset categories, encompassing 180 instances, as performance on easier instances was consistently favorable across all approaches.

3.1 Baseline 1 and Baseline 2 Comparison

The results of our evaluation demonstrate that Baseline 2 outperforms Baseline 1. Here are the findings summarized:

Baseline 1 Best	40
Baseline 2 Best	78
Same	62
Baseline 2 Best / Baseline 1 Best	1.95

This stark contrast indicates that the method of selecting the highest profit solutions yields a considerable advantage over random initialization, underscoring the importance of informed solution selection in optimization tasks.

3.2 Regression 0-1 Results

Evaluating the Regression 0-1 labeling approach revealed further insights into the effectiveness of the Machine Learning-enhanced method. The results are as follows:

	Baseline 1	**Baseline 2**
Regression 0-1 Best	82	61
Baseline Best	31	37
Same	67	82
Regression 0-1 Best / Baseline Best	2.65	1.65

The results indicate that the Regression 0-1 method consistently yields better performance, which demonstrates a significant improvement in identifying high-quality solutions through the application of Machine Learning techniques.

3.3 Classification 0.90

The Classification 0.90 approach yielded similarly promising results, as shown below:

	Baseline 1	**Baseline 2**
Classification 0.90 Best	88	63
Baseline Best	32	39
Same	60	78
Regression 0-1 Best / Baseline Best	2.75	1.62

These findings highlight the enhanced capability of our Machine Learning model to classify and predict the most promising solutions effectively, leading to substantial improvements over both baseline methods.

4. Conclusion

In conclusion, our Machine Learning-enhanced approach to the Set Team Orienteering Problem demonstrates promising potential, particularly in its ability to expand the solution space via a multi-restart matheuristic that combines Tabu Search and the GRASP methodology. By using Machine Learning to inform the selection and evaluation of initial solutions through carefully engineered features, we have introduced a novel integration of predictive modeling within a combinatorial optimization framework.

While the results were better than those obtained from a pure local-search scheme, they were not competitive enough when compared to results obtained on benchmark instances from state-of-the-art solvers. This indicates that, while promising, the integration of AI in OR problems has not reached the necessary maturity to outperform most traditional practices. Nonetheless, this work lays important groundwork for future efforts to more deeply integrate Machine Learning into traditional optimization pipelines. As the complexity of real-world routing and planning problems continues to grow, approaches that bridge heuristic methods with data-driven insights remain a compelling area for continued research and refinement.