GRIP: Latent Field-Guided Graph Policy for Budget-Constrained Multi-Agent Routing

Yujiao Hu[1], Zuyu Chen[2], MengJie Lee[3], Jinchao Chen[3], Meng Shen[4], Hailun Zhang[5], Wei Li[1], Yan Pan[2] *

[1]School of Data Science and Artificial Intelligence, Chang'an University, China
[2]Science and Technology on Information Systems Engineering Laboratory, National University of Defense Technology, China
[3]School of Computer Science, Northwestern Polytechnical University, China
[4]Pervasive Communication Research Center, Purple Mountain Laboratories, China
[5]School of Automobile, Chang'an University, China
huyujiao@chd.edu.cn, chenzuyu19@nudt.edu.cn, lmj1023@mail.nwpu.edu.cn, cjc@nwpu.edu.cn, shenmeng@pmlabs.com.cn, zhanghailun@chd.edu.cn, grandy@chd.edu.cn, panyan@nudt.edu.cn

Abstract

Subset selection under budget constraints is critical in applications like multi-robot patrolling, crime deterrence, and targeted marketing, where multiple agents must jointly select targets and plan feasible routes. We formalize this challenge as Multi-Subset Selection with Budget-Constrained Routing (MSS-BCR), involving complex, non-additive cost structures that defy traditional methods. We propose GRIP, a graph-based framework integrating spatial reward fields and policy learning to enable coordinated, budget-aware target selection and routing. GRIP uses attention-based embeddings and constraint-triggered pruning with utility recovery to produce high-quality, feasible solutions. Experiments based on multiple synthetic and real-world datasets show GRIP outperforms baselines in reward efficiency and scalability across varied scenarios.

Introduction

A wide range of decision-making problems in science and engineering require selecting a subset of informative or high-reward elements under resource constraints. This broad class of problems is generally known as the Subset Selection (SS) problem (Natarajan 1995; Davis, Mallat, and Avellaneda 1997) and has been extensively studied across fields such as operations research, machine learning, and data mining. Classical examples include maximum coverage (Feige 1998) and influence maximization (Kempe, Kleinberg, and Tardos 2003), where the goal is to select a subset that maximizes a utility function while satisfying additive cost constraints, that is, the total cost is defined as the sum of individual element costs and must not exceed a given limit such as subset size, total weight, or budget. Owning to the simple additive nature of these constraints, a variety of algorithms (e.g., greedy methods with approximation guarantees) have been developed to solve SS efficiently.

With the increasing complexity of real-world applications, single-agent formulations have become insufficient to capture the collaborative and distributed nature of many decision-making tasks. This has motivated the development of the Multi-Subset Selection (MSS) framework (Ohsaka and Yoshida 2015; Qian et al. 2017a; Rafiey and Yoshida 2020; Oshima 2021; Spaeh, Ene, and Nguyen 2025; Nguyen and Thai 2021), where a team of agents, each with distinct budgets, operational scopes, or deployment objectives, jointly select subsets from a shared ground set to optimize a global utility. MSS naturally arises in scenarios such as multi-robot exploration, coordinated vehicle dispatch, and distributed workforce deployment, and introduces new challenges stemming from inter-agent coordination.

However, even the MSS formulation falls short in many realistic scenarios where the cost of selecting elements is no longer additive, but is governed by the feasibility of planning routes among the selected elements (Zhang and Vorobeychik 2016; Qian et al. 2017b; Bian et al. 2020; Roostapour et al. 2022). For example, in door-to-door marketing, each salesperson must select a subset of potential customers to visit and construct a daily route that maximizes marketing impact while staying within time or travel constraints (Zhang and Vorobeychik 2016; Roostapour et al. 2022). In urban crime monitoring and deterrence, police patrol vehicles are tasked with visiting a subset of high-crime or high-risk areas to enhance public safety. However, each vehicle has operational constraints such as fuel capacity or shift time, which limits the total patrol length (Sun et al. 2022; Lin et al. 2023; Hepp, Nießner, and Hilliges 2018). Moreover, to be effective, the patrol path must form a feasible tour connecting the selected hotspots. This requires jointly optimizing both which areas to visit and how to traverse under strict budget constraints.

These scenarios share a unifying structure: each agent must select a subset of targets and construct a cost-feasible route through them, where the cost, such as travel time or path length, is determined by solving a combinatorial routing problem. Crucially, the cost of a subset depends not only on the elements selected but also on their spatial configuration, violating the additive assumptions that underlie classical SS and MSS models. We formalize this practically important yet computationally challenging class of problems as Multi-Subset Selection with Budget-Constrained Routing (MSS-BCR). Despite its relevance to domains such

*Corresponding Author

as robotics, logistics, and public safety, MSS-BCR has received limited attention, particularly with regard to scalable, learning-based approaches capable of handling real-world complexity.

To address these challenges, we propose GRIP (Graph-based Routing-aware Information Planning), a novel learning-based framework for efficiently solving the MSS-BCR problem. GRIP is fundamentally driven by Information Field Theory (Enßlin 2013; Bai et al. 2021; Said et al. 2021), which models the spatial distribution of reward as a continuous information field. Guided by this field, GRIP integrates information-driven subset assignment via graph policy learning, constraint-trigger field collapse and route refinement and field-gradient reinsertion for residual utility capture. Leveraging an S-sample batch reinforcement learning paradigm, GRIP generalizes across diverse environments and scales to large problem instances. Extensive experiments on both synthetic benchmarks and a real-world crime patrol task demonstrate GRIP's superiority in reward accumulation, constraint satisfaction, and cost efficiency.

Our main contributions are summarized as follows:

- We formulate the MSS-BCR problem, a novel extension of subset selection to multi-agent, routing-constrained settings, motivated by real-world applications such as crime deterrence patrol, robotic sensing, and logistics planning.
- We introduce GRIP, a learning-based framework explicitly driven by Information Field Theory, which serves as both a theoretical foundation and a design principle guiding graph-based subset allocation, routing optimization, and local re-insertion for reward enhancement.
- We conduct comprehensive experiments on synthetic and real-world datasets, demonstrating that GRIP consistently outperforms strong learning-based and heuristic baselines in global reward maximization, constraint compliance, and scalability.

Related Work

To contextualize our work, we review the literature from two complementary perspectives: (1) Problem Settings and Application Background, and (2) algorithmic Solutions.

Problem Settings and Application Background

Subset selection problems arise in many fields, where the goal is to choose a subset of elements that maximize a utility under cost constraints.

Subset Selection (SS) is the classical setting with additive costs, studied in applications such as maximum coverage (Feige 1998) and influence maximization (Kempe, Kleinberg, and Tardos 2003). Greedy algorithms with performance guarantees are common here (Nemhauser, Wolsey, and Fisher 1978).

Multi-agent Subset Selection (MSS) extends SS by involving multiple agents, each selecting a subset under individual budgets to maximize a joint objective. Typical applications include multi-topic influence maximization (Ohsaka and Yoshida 2015) and multi-sensor placement (Nguyen and Thai 2021). Most MSS work assumes additive costs (Ward and Živnỳ 2014; Spaeh, Ene, and Nguyen 2025).

Subset Selection with Budget-Constrained Routing (SS-BCR) further generalizes SS by introducing routing-based, non-additive costs. Examples include door-to-door marketing (Zhang and Vorobeychik 2016) and robotic sensing (Hepp, Nießner, and Hilliges 2018), where the selected subset must be connected via a feasible route respecting travel budgets. This coupling of subset selection and routing makes the problem NP-hard.

Algorithmic Solutions

Algorithmic approaches for these problems can be divided into two main categories: greedy algorithms and Evolutionary Learning (EL) algorithms.

Greedy algorithms iteratively add or remove elements based on marginal gains to optimize the objective. The seminal work by Nemhauser et al. (Nemhauser, Wolsey, and Fisher 1978) established performance guarantees for greedy algorithms in SS. For MSS, Ward et al. (Ward and Živnỳ 2014) proposed a multiple greedy algorithm. Recently, Spaeh et al. (Spaeh, Ene, and Nguyen 2025) developed a greedy method for online MSS, where elements arrive sequentially rather than being predetermined. Zhang et al. (Zhang and Vorobeychik 2016) proposed the first greedy algorithm tailored for SS-GCC.

Evolutionary Learning (EL) algorithms explore a larger solution space and often achieve better empirical performance than greedy methods. Qian et al. (Qian, Yu, and Zhou 2015) theoretically proved that EL can achieve the same approximation ratio as greedy algorithms and experimentally showed superior performance. However, EL methods can be computationally expensive. To alleviate this, parallel and distributed EL algorithms have been developed (Qian et al. 2016, 2018), accelerating the learning process using multiple processors. EL algorithms have also been adapted specifically for SS-GCC problems (Bian et al. 2020; Qian et al. 2017b; Roostapour et al. 2022).

Limitations of Existing Work

Despite significant progress, current research faces several limitations when applied to MSS-GCC problems:

- Most MSS algorithms (Ohsaka and Yoshida 2015; Rafiey and Yoshida 2020; Spaeh, Ene, and Nguyen 2025) assume the additive costs, which are not suitable for applications with routing-based constraints.
- While EL algorithms (Bian et al. 2020; Qian et al. 2017b; Roostapour et al. 2022) are more expressive, they are computationally expensive and scale poorly in high-dimensional or multi-agent settings.
- To the best of our knowledge, no existing work has systematically addressed the MSS-BCR problem using learning-based frameworks that jointly consider subset selection and routing.

These limitations motivate our work, which proposes a novel reinforcement learning (RL)-based approach, GRIP, to tackle the MSS-BCR problem by integrating spatial reward modeling and routing feasibility within a unified framework.

MSS-BCR Problem Formulation

We study the Multi-Subset Selection with Budget-Constrained Routing (MSS-BCR) problem, which models scenarios where multiple agents must each select and visit a subset of nodes by traversing a cost-feasible route, with the goal of maximizing the total collected reward.

Notations

Let $\mathcal{V} = \{v_1, v_2, \ldots, v_N\}$ denote a set of N nodes, each associated with a non-negative reward $r(v_i) \in \mathbb{R}_{\geq 0}$. Let $\mathcal{A} = \{a_1, a_2, \ldots, a_M\}$ be a set of M agents. Each agent a_j has a start location s_j and a budget B_j representing the maximum allowed route cost. Let $d(u, v)$ denote the travel cost (e.g., Euclidean distance) between nodes u and v. Each agent a_j selects a subset of nodes $\mathcal{V}_j \subseteq \mathcal{V}$, such that the cost of the planned tour T_j over $\mathcal{V}_j$ satisfies $\text{cost}(T_j) \leq B_j$. The subsets are mutually disjoint, i.e., $\mathcal{V}_j \cap \mathcal{V}_{j'} = \emptyset$ for any $j \neq j'$. The objective is to maximize the total collected reward.

Problem Definition

Let $\mathbf{X} = [x_{ij}] \in \{0, 1\}^{N \times M}$ be the assignment matrix, where $x_{ij} = 1$ if node v_i is assigned to agent a_j, and 0 otherwise. Let $\mathcal{P}_j(\mathbf{X})$ denote the planned route (tour) for agent a_j visiting nodes assigned by $\mathbf{X}$. The MSS-BCR problem is formulated as:

$$\max_{\mathbf{X}} \quad \sum_{j=1}^{M} \sum_{v_i \in \mathcal{P}_j(\mathbf{X})} r(v_i) \tag{1}$$

$$\text{s.t.} \quad \text{cost}(\mathcal{P}_j(\mathbf{X})) \leq B_j, \quad \forall j = 1, \ldots, M \tag{2}$$

$$\sum_{j=1}^{M} x_{ij} \leq 1, \quad \forall i = 1, \ldots, N \tag{3}$$

$$x_{ij} \in \{0, 1\}, \quad \forall i = 1, \ldots, N, j = 1, \ldots, M \tag{4}$$

This problem is combinatorially hard due to the coupling between node assignment and budget-constrained routing.

GRIP Framework

This section introduces the proposed GRIP (Graph-based Routing-aware Information Planning) framework, designed to efficiently solve the MSS-BCR problem. GRIP is driven by Information Field Theory, treating agents as dynamic entities that perceive and act within a latent spatial-reward field defined over a graph. To operationalize this perspective, GRIP proceeds in four stages.

Stage I: Modeling Node Environments as a Latent Information Field

GRIP begins by formulating the environment as a discrete latent information field defined over the graph $G = (\mathcal{V}, E)$ to enable information-theoretic reasoning under spatial constraints. Kernel operations include:

Information Potential Encoding In G, Each node $v_i \in \mathcal{V}$ serves as a spatial stimulus, contributing to the overall field through an intrinsic information potential. We explicitly define the node-wise information potential as

$$\phi_i = \beta \cdot r_i \tag{5}$$

where r_i denotes the intrinsic reward of node v_i, and β is a tunable scaling coefficient. This scalar field $\{\phi_i\}_{i=1}^{N}$ represents a reward landscape over the spatial graph and serves as the foundation for downstream agent perception and interaction.

Graph-Aware Embedding Initialization While (5) encodes explicit node utility, topological context is captured implicitly via learnable embeddings. Each agent a_j is initialized with feature vector $f_j = [c_j; d_j; B_j] \in \mathbb{R}^d$, where c_j is the agent's current position, d_j is the depot location, and B_j denotes its path-length budget. Similarly, each node v_i is described by $h_i = [x_i; \phi_i] \in \mathbb{R}^d$, where x_i is the node's coordinate.

These inputs are projected into initial embeddings through agent- and node-specific multilayer perceptrons (MLPs):

$$f_j^{(0)} = W_j f_j + b_j \tag{6}$$

$$h_i^{(0)} = W_i h_i + b_i \tag{7}$$

with learnable parameters W_j, W_i, b_j, b_i.

Multi-Directional Attention Propagation To encode higher-order field interactions, GRIP applies L layers of message-passing attention across three relational dimensions:

- Node $\rightarrow$ Node (Spatial Context Propagation): Each node v_i communicates with all other nodes to capture long-range spatial and structural dependencies via a multi-head attention (MHA) module with K heads as (8)-(12).

$$q_{i,k}^{(l)} = W_{q,k}^{(l)} h_i^{(l-1)} \tag{8}$$

$$y_{n,k}^{(l)} = W_{y,k}^{(l)} h_i^{(l-1)}, \quad s_{n,k}^{(l)} = W_{v,k}^{(l)} h_i^{(l-1)} \tag{9}$$

$$u_{ni,k}^{(l)} = \frac{(q_{i,k}^{(l)})^T y_{n,k}^{(l)}}{\sqrt{d_y}}, \quad t_{ni,k}^{(l)} = \frac{u_{ni,k}^{(l)}}{\sum_j u_{ji,k}^{(l)}} \tag{10}$$

$$\hat{q}_{i,k}^{(l)} = \sum_n t_{ni,k}^{(l)} s_{n,k}^{(l)} \tag{11}$$

$$h_i^{(l)} = \sum_k W_{a,k}^{(l)} \hat{q}_{i,k}^{(l)} \tag{12}$$

- Node $\rightarrow$ Agent (Perception Assimilation): Each agent receives field-level messages from all nodes through a node-to-agent attention module, allowing the agent to integrate environmental potential. This module follows the structure of (8)–(12), replacing queries with agent features $f_j^{(l-1)}$.
- Agent $\rightarrow$ Agent (Mutual Awareness): Agents exchange messages via an agent-to-agent MHA module, enabling them to infer relative positions, loads, and overlapping coverage patterns.

Contextual Feature Fusion Since each agent receives signals from both nodes and other agents, we fuse these streams via a weighted context aggregation:

$$\hat{f}_j^{(l)} = W_{na}^{(l)} \cdot f_j^{s(l)} + W_{aa}^{(l)} \cdot f_j^{a(l)} \tag{13}$$

where $f_j^{s(l)}$ and $f_j^{a(l)}$ are the node- and agent-sourced embeddings, respectively. A residual connection, batch normalization, and linear projection are applied to finalize the layer update:

$$f_j^{(l)} = \mathrm{FC}_a^{(l)} \left(\mathrm{BN}_a^{(l)} \left(\hat{f}_j^{(l)} + f_j^{(l-1)} \right) \right) \tag{14}$$

Latent Field Embedding Through multi-level attention and contextual fusion, each agent and node acquires a personalized latent representation of the environment, that is, an information field shaped by both spatial reward distribution and cooperative interactions. This learned embedding serves as a compact perceptual prior, guiding the downstream graph policy learning and subset assignment in Stage II.

Stage II: Information-Driven Assignment via Graph Policy Learning

In Stage II, GRIP operationalizes the latent information field constructed in Stage I by learning a routing-aware node-to-agent assignment policy. This is achieved through a graph-based stochastic policy network $\pi(\theta)$, which outputs a soft probabilistic mapping from nodes to agents. Specifically, the policy is represented as a matrix $\mathbf{P} \in \mathbb{R}^{|V| \times M}$, where each entry $P_{i,j}$ denotes the likelihood that node v_i is assigned to agent a_j.

Graph-Based Policy Generation Stage I produces the latent field embeddings, specifically agent and node embeddings,denoted by $\{h_i^{(L)}\}_{i=1}^N$ and $\{f_j^{(L)}\}_{j=1}^M$. These embeddings encapsulate each entity's spatial configuration and topological context as shaped by the latent information field. To derive actionable assignment decisions from this representation, GRIP projects the embeddings into a shared latent interaction space using learnable linear transformations:

$$f_j = W_{pa} f_j^{(L)}, \qquad h_i = W_{pn} h_i^{(L)} \tag{15}$$

where W_{pa} and W_{pn} are parameter matrices specific to agents and nodes. This shared space allows agents and nodes to interact in a field-aligned manner.

The affinity between node v_i and agent a_j is then computed via a bounded bilinear interaction:

$$u_{ij} = C_1 \cdot \tanh(h_i f_j^\top) \tag{16}$$

where C_1 is a temperature constant controlling the sharpness of interaction. Intuitively, this operation measures the directional alignment of node v_i's potential vector with agent a_j's trajectory within the information field, which is analogous to computing local field strength in classical potential theory.

These affinity scores are normalized across all agents using a softmax function to yield a distributional assignment policy:

$$p_{ij} = \frac{e^{u_{ij}}}{\sum_{k=1}^M e^{u_{ik}}} \tag{17}$$

Algorithm 1: Bi-Criteria Pruning via Field Collapse

Input : Tour $\mathcal{T}$, length L, coordinates $\mathbf{x}$, rewards $\mathbf{r}$, budget B, trade-off α
Output: Pruned tour $\mathcal{T}^*$ with $L \le B$
Initialize open list with $(\mathcal{T}, L, R = \sum_{v_i \in \mathcal{T}} r_i)$;
while $L > B$ **do**
  foreach *internal node* $v_i \in \mathcal{T}$ **do**
    Estimate ΔL_i by shortcutting v_i:
    $\Delta L_i = d(v_{i-1}, v_i) + d(v_i, v_{i+1}) - d(v_{i-1}, v_{i+1})$;
    Estimate $\Delta R_i = r_i$;
    Compute score
    $\delta_i = \alpha \cdot (1 - \Delta L_i / L) + (1 - \alpha) \cdot \Delta R_i / R$;
  Remove node $v_k = \arg\max_i \delta_i$ from $\mathcal{T}$;
  Update L and R accordingly;
return $\mathcal{T}^*$

where $p_{ij} \in (0, 1)$ quantifies the likelihood of assigning node v_i to agent a_j. In this view, the policy network acts as a differentiable, field-sensitive inference mechanism that dynamically interprets spatial task utility and agent capabilities under a constrained assignment regime.

Stage III: Constraint-Triggered Field Collapse and Route Refinement

Given the initial assignment matrix $\mathbf{P} \in \mathbb{R}^{|V| \times M}$, each agent a_j is assigned a candidate node set S_j through sampling on $\mathbf{P} \in \mathbb{R}^{|V| \times M}$. With S_j, a tentative tour $\mathcal{T}_j$ can be constructed using a standard TSP solver (e.g., OR-Tools). However, when the tour cost $L(\mathcal{T}_j)$ exceeds the agent's budget B_j, the agent must prune its path. The step is modeled as a collapse of the agent's effective information field, i.e. contracting its region of influence to satisfy operational constraints.

To guide the pruning, we propose a bi-criteria heuristic that jointly considers: (i) the relative path length reduction, and (ii) the relative reward loss. For each internal node v_i in the tour, we compute:

$$\delta_i = \alpha \cdot \left(1 - \frac{\Delta L_i}{L}\right) + (1 - \alpha) \cdot \frac{\Delta R_i}{R}$$

where ΔL_i is the length saved by removing v_i, and $\Delta R_i = r_i$ is the reward lost. Nodes with the highest δ_i are iteratively removed until the tour becomes budget-feasible. The process is detailed in Algorithm 1.

This mechanism promotes removal of nodes that contribute least in terms of information per unit path cost, leading to a dynamic contraction of the agent's effective field. Moreover, the pruning results are fed back into Stage II during training, enabling the policy to learn more efficient assignment strategies

Stage IV: Field-Gradient Reinsertion for Residual Utility Capture

While pruning ensures feasibility, it may prematurely discard nodes with high latent potential due to local subopti-

Algorithm 2: Residual Field Reinsertion for Budget-Feasible Tour Enhancement

Input: Node coords $\mathbf{X}$, rewards $\mathbf{r}$; current tours $\{\mathcal{T}_j\}$ with lengths $\{\ell_j\}$; budgets $\{B_j\}$
Output: Enhanced tours $\{\hat{\mathcal{T}}_j\}$
repeat
  Let $\mathcal{U} \leftarrow$ unassigned nodes ;
  Initialize best gain $\gamma^* \leftarrow 0$;
  foreach *agent* a_j **do**
    foreach $v_i \in \mathcal{U}$ **do**
      foreach *insertion point* $(v_j, v_{j+1}) \in \mathcal{T}_j$ **do**
        Compute ΔL_i via (18) ;
        if $\ell_j + \Delta L_i \leq B_j$ **then**
          Compute $\gamma_i = \phi_i / \Delta L_i$ via (19) ;
          if $\gamma_i > \gamma^*$ **then**
            Record best (a_j, v_i, j) and update γ^* ;
          end
        end
      end
    end
  end
  if *valid insertion* (a^*, v^*, j^*) *found* **then**
    Insert v^* between v_{j^*} and v_{j^*+1} in $\mathcal{T}_{a^*}$;
    Update ℓ_{a^*} and remove v^* from $\mathcal{U}$;
  end
until *no feasible reinsertion found*;
return $\{\hat{\mathcal{T}}_j\}$

mality. To mitigate this, GRIP performs a final field-gradient reinsertion phase that greedily recovers residual value by leveraging the information field's gradient.

Let $\mathcal{U}$ denote the set of unvisited nodes after pruning. For each unassigned node $v_i \in \mathcal{U}$ and agent a_j, we evaluate the marginal insertion cost ΔL_i between two consecutive nodes (v_j, v_{j+1}) in tour $\mathcal{T}_j$:

$$\Delta L_i = d(v_j, v_i) + d(v_i, v_{j+1}) - d(v_j, v_{j+1}), \tag{18}$$

and define a reinsertion priority based on the field-efficiency ratio:

$$\gamma_i = \frac{\phi_i}{\Delta L_i}, \tag{19}$$

where ϕ_i is the information potential from the latent field, shown in (5) in Stage I. A reinsertion is allowed only if it respects the agent's budget constraint: $\ell_j + \Delta L_i \leq B_j$.

Algorithm 2 details this residual completion procedure. At each iteration, feasible insertion points across all tours are evaluated, and the node with the highest γ_i is inserted. This continues greedily until no further improvement is possible.

This final stage can be interpreted field-theoretically as extending each agent's local decision boundary along the residual gradient of its collapsed field. Rather than re-optimizing full tours, the agent incrementally advances into nearby high-potential regions, recovering previously discarded utility in a budget-aware manner. By aligning local reinsertion with the latent field structure, GRIP completes a fully differentiable, field-aware information acquisition loop that integrates feasibility, reward efficiency, and spatial reasoning.

GRIP Training

To effectively train GRIP in the context of the MSS-BCR problem, we adopt an S-sampled batch reinforcement learning (RL) framework (Hu, Yao, and Lee 2020; Hu et al. 2024). Unlike traditional single-sample policy gradient methods, this approach utilizes multiple sampled assignments per graph instance to estimate the expected performance of downstream routing, thereby reducing gradient variance and improving convergence speed.

Given a training dataset D and a graph instance $g \sim D$, we sample S complete node-to-agent assignments $\{\pi_s\}_{s=1}^{S}$ from the learned assignment policy $\pi(\theta)$ where θ represents all learnable parameters. The policy is defined by the assignment probabilities $\{P_{i,j}\}_{i=1,j=1}^{N,M}$, and the reward $\Psi(\pi_s|g)$ for each assignment is computed by executing the downstream routing process in Stage III.

To guide training, we compute a normalized advantage function by (20), which quantifies the relative merit of each sample against the batch average.

$$A(\pi_s|g) = \Psi(\pi_s|g) - \frac{1}{S}\sum_{k=1}^{S}\Psi(\pi_k|g) \tag{20}$$

The overall objective is to maximize the expected advantage-weighted log-likelihood of sampled assignments under the current policy:

$$\mathcal{L}(\theta|D) = \mathbb{E}_{g\sim D}\left[\sum_{s=1}^{S}\log P_\theta(\pi_s|g)\cdot A(\pi_s|g)\right] \tag{21}$$

where $P_\theta(\pi_s|g)$ denotes the joint likelihood of constructing the assignment π_s via independent sampling over all nodes:

$$P_\theta(\pi_s|g) = \prod_{i=1}^{N} p_{i,j_i^*}, \quad \text{where } j_i^* = \pi_s(v_i) \tag{22}$$

GRIP Inference

At inference time, GRIP operates in a deterministic and efficient manner by deploying its learned assignment policy to generate agent routes. Specifically, we apply greedy decoding by assigning each node v_i to the agent a_j with the highest assignment probability: $j^* = \arg\max_j p_{ij}$. Each agent's assigned node set then undergoes budget-constrained route refinement via the field-collapsing heuristic described in Stage III (Algorithm 1). After initial pruning, the algorithm performs field-gradient-based reinsertion (Algorithm 2) to augment agent tours by capturing high-value nodes within residual budget margins.

Evaluation

This section presents the experimental setup, including synthetic evaluation under in-domain and out-of-domain settings, cross-domain generalization to real-world data, and an ablation study on GRIP's design components.

Type	Method	R ↑	RPCB ↑	VLP ↑	R ↑	RPCB ↑	VLP ↑	R ↑	RPCB ↑	VLP ↑	R ↑	RPCB ↑	VLP ↑
		<5,50>			<5,100>			<7,70>			<10,100>		
NT	EL	15.229	3.773	0.545	24.192	5.201	0.405	25.000	4.080	0.637	39.542	4.380	0.710
	Greedy	15.666	3.534	0.524	25.609	5.457	0.414	25.777	3.951	0.624	41.545	4.438	0.717
	GRIP	**17.329**	**3.930**	**0.661**	**28.696**	**6.048**	**0.520**	**28.680**	**4.533**	**0.791**	**45.442**	**5.255**	**0.889**
		<5,200>			<5,300>			<10, 500>			<20, 1000>		
GT	EL	31.667	6.648	0.252	35.996	7.370	0.188	74.587	7.629	0.235	159.283	8.103	0.251
	Greedy	39.567	8.217	0.310	50.968	10.307	0.263	125.953	12.829	0.395	338.740	17.223	0.540
	GRIP	**43.272**	**8.877**	**0.369**	**54.062**	**10.805**	**0.299**	**138.061**	**13.976**	**0.478**	**367.544**	**18.704**	**0.669**

Table 1. Performance on Synthetic Dataset under Normal and Generalization Testing. $< M, N >$ indicates the number of agents (M) and candidate nodes (N).

Experimental Setup

Training and Testing Scales GRIP is trained on synthetic instances with the number of agents $M \in [5, 10]$ and candidate nodes $N \in [5M, 100]$. Evaluation includes two settings: *Normal Testing (NT)* matches the training scale, while *Generalization Testing (GT)* extends M and N beyond the training range to assess scalability.

Synthetic Dataset Generation Each synthetic instance is constructed by uniformly sampling $M + N$ points in $[0, 1]^2$. Node rewards are drawn from $[0, 1]$; M depots are randomly chosen from these points to serve as both start and end points. Agent path-length budgets are sampled from $[0, 2]$. This setup follows prior work (Gao et al. 2023; Hu et al. 2025; Park, Kwon, and Park 2023; Kim, Park, and Park 2023). For each pair $\langle M, N \rangle$, we generate 1000 instances.

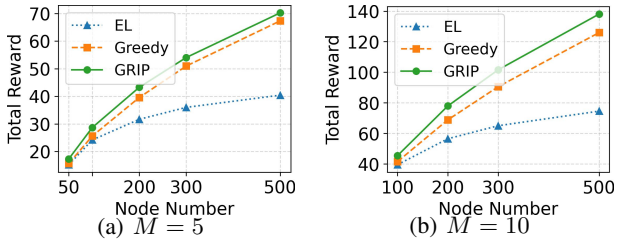
Real-World Dataset We use public incident data from Seattle (Department 2024b), Los Angeles (Department 2024a), and Denver (Mooney 2014), containing coordinates of accidents and crimes. Each city is normalized to $[0, 1]^2$ while preserving spatial layout. Incidents are clustered into $M + N$ groups via K-Means; cluster sizes define rewards. M centers are randomly selected as depots, and agent budgets are sampled as in the synthetic case. We generate 100 test instances per scale.

Implementation Details Experiments are run on an NVIDIA V100S GPU with Ubuntu 22.04, Python 3.10, and PyTorch 2.5.1. The learning rate is set to 5×10^{-5}, with $\alpha = 0.7$ in the pruning heuristic and $S = 12$ in (20).

Evaluation Metrics The three metrics together reflect each method's effectiveness, efficiency, and coverage.

- Total reward (R) : The accumulated reward across all agents, reflecting the total task value achieved.
- Reward per Consumed Budget (RPCB): Defined as the total reward divided by the actual consumed budget. It evaluates cost-efficiency.
- Visited Location Proportion (VLP): The ratio between the number of visited nodes and the total available candidate nodes, capturing spatial task coverage.

Baselines As no prior work directly solves MSS-BCR, we extend two representative single-agent budget-constrained subset selection methods to the multi-agent setting:



(a) $M = 5$ (b) $M = 10$

Figure 1: Scalability evaluation on synthetic dataset. Total reward vs. number of candidate nodes N under fixed agent numbers (a) M=5 and (b) M=10.

- **Evolutionary Learning (EL)** (Bian et al. 2020): Each agent independently applies evolutionary search to select and route among unvisited nodes, with selected nodes broadcast to avoid duplication.
- **Greedy Algorithm (Greedy)** (Zhang and Vorobeychik 2016): Iteratively assigns nodes to agents based on marginal reward, stopping when all nodes are selected or budgets are exceeded.

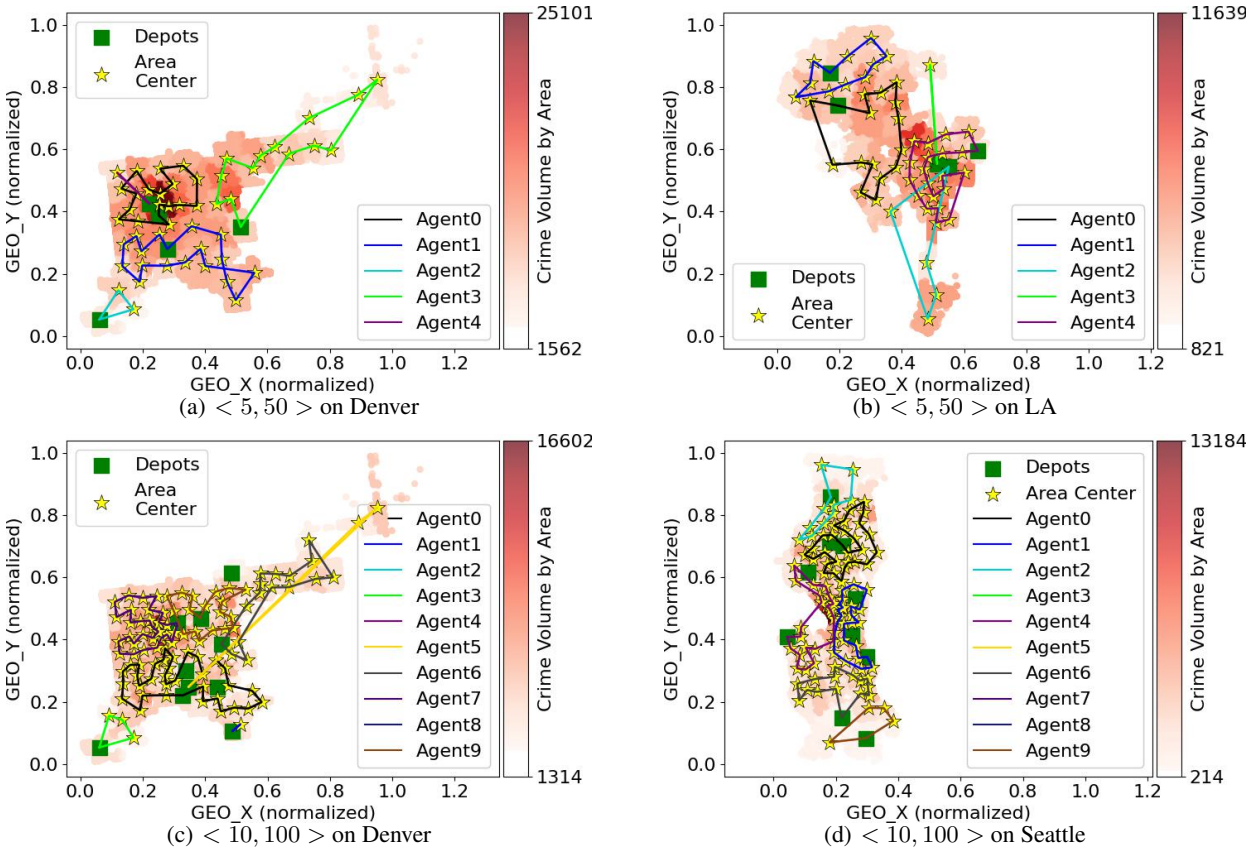
Performance on Synthetic Dataset

As shown in Table 1, GRIP consistently achieves superior performance across all problem scales. In NT scenarios, GRIP yields the highest total reward and RPCB across all scales. For example, at $< 10, 100 >$, GRIP surpasses EL and Greedy by 15% and 9.4% in reward, and by 20% and 18.4% in RPCB, respectively. Notably, GRIP also maintains the highest VLP, indicating that more target locations are effectively covered under budget constraints. In GT scenarios where both the number of agents and candidate nodes exceed training conditions, GRIP demonstrates strong generalization. At $< 20, 1000 >$, GRIP achieves the highest reward (367.5), significantly outperforming EL (159.3) and Greedy (338.7). Moreover, GRIP's RPCB and VLP continue to dominate, indicating not only its reward-maximizing capability but also its ability to scale effectively while maintaining spatial exploration efficiency.

To further examine scalability under varying candidate node densities, we plot the total reward as a function of node number N under fixed agent counts M=5 and M=10 in Fig. 1.

Dataset	Method	<5,50> R ↑	<5,50> RPCB ↑	<5,50> VLP ↑	<10,100> R ↑	<10,100> RPCB ↑	<10,100> VLP ↑	<5,300> R ↑	<5,300> RPCB ↑	<5,300> VLP ↑	<10,500> R ↑	<10,500> RPCB ↑	<10,500> VLP ↑
Denver	EL	390923	3.611	0.765	420756	2.839	0.884	171156	4.222	0.227	214622	3.577	0.297
	Greedy	405577	3.818	0.797	443648	3.286	0.940	303083	7.428	0.471	413431	6.919	0.719
	GRIP	**422117**	**4.516**	**0.869**	**447098**	**4.507**	**0.975**	**324510**	**7.977**	**0.548**	**437612**	**7.438**	**0.825**
LA	EL	170043	3.359	0.797	186643	3.256	0.902	72098	4.883	0.242	85551	4.882	0.299
	Greedy	177488	3.576	0.832	194233	3.771	0.948	138281	9.335	0.520	179301	10.364	0.733
	GRIP	**185772**	**4.168**	**0.911**	**196206**	**5.268**	**0.987**	**141191**	**9.492**	**0.565**	**185950**	**10.910**	**0.805**
Seattle	EL	261534	2.638	0.876	273613	2.339	0.940	138642	3.885	0.265	162582	3.834	0.331
	Greedy	267033	2.793	0.900	277924	2.680	0.965	222641	6.235	0.546	274048	6.501	0.784
	GRIP	**272572**	**3.646**	**0.960**	**278549**	**4.038**	**0.995**	**233787**	**6.567**	**0.634**	**284201**	**6.910**	**0.883**

Table 2. Performance Comparison on Real-world Datasets (Denver, LA, Seattle)



(a) $< 5, 50 >$ on Denver (b) $< 5, 50 >$ on LA

(c) $< 10, 100 >$ on Denver (d) $< 10, 100 >$ on Seattle

Figure 2: Visualized Planned Paths of GRIP on Real-World Datasets.

All methods benefit from increased node availability, but GRIP consistently achieves the highest reward across all settings. This further validates GRIP's advantage in efficiently navigating large, combinatorial decision spaces via its hierarchical policy.

Performance on Real-world Data

Table 2 presents the performance on urban incident datasets from Denver, LA, and Seattle. Across all cities and problem scales, GRIP consistently achieves the best performance on all three metrics. In Denver at $< 10, 500 >$, GRIP achieves a 5.8% reward gain over Greedy and a significant 104% gain over EL. GRIP also shows better RPCB and VLP, indicating both improved efficiency and broader incident site coverage.

The improvements are particularly pronounced in larger-scale settings. In LA at $< 10, 500 >$, GRIP yields a VLP of 0.805, compared to 0.733 and 0.299 for Greedy and EL, respectively. This reflects GRIP's superior ability to plan diverse and effective paths under budget constraints. Similar trends are observed in Seattle, where GRIP outperforms in all metrics and scales, especially in budget efficiency (RPCB: 6.91 vs. 6.50/3.83) and spatial coverage.

Method	<5,50>			<5,100>		
	R ↑	RPCB ↑	VLP ↑	R ↑	RPCB ↑	VLP ↑
GRIP-S3	6.319	1.833	0.216	9.157	2.436	0.148
GRIP-S3S4	14.067	3.201	0.527	22.842	4.886	0.415
GRIP-S1S2S3	16.352	3.920	0.608	26.084	5.803	0.450
GRIP-Full	**17.329**	**3.930**	**0.661**	**28.696**	**6.048**	**0.520**

Table 3. Ablation Study on GRIP Variants.

To qualitatively assess the behavior of GRIP in realistic environments, we visualize four representative planning instances in Fig. 2. Across all scenarios, GRIP demonstrates a consistent ability to construct well-structured and spatially diverse routes that effectively cover high-reward regions. These cases further validate GRIP's capacity for coordinated multi-agent planning under complex real-world constraints.

Ablation Study

We perform an ablation study to evaluate the contribution of each GRIP component, as summarized in Table 3. GRIP-S3, which uses only the pruning stage (Stage III), performs poorly across all metrics, showing that local greedy pruning alone is ineffective without prior assignment guidance. Introducing policy feedback in GRIP-S3S4 significantly boosts reward and coverage, highlighting the value of constraint-aware adaptation. Adding node assignment stages in GRIP-S1S2S3 brings further gains, confirming that directing agents toward high-potential regions provides a strong initialization. Finally, the full GRIP model achieves the best results, demonstrating the effectiveness of integrating global assignment, local pruning, and feedback in a unified framework.

These results demonstrate the complementary roles of each stage in GRIP, especially the necessity of combining learning-based node assignment with field-based constraint-aware pruning. The progressive improvement across variants validates the effectiveness of our hierarchical design.

Conclusion

We propose GRIP to solve the MSS-BCR problem, integrating multi-agent subset selection with budget-constrained routing. GRIP achieves superior performance in reward efficiency and coverage across synthetic and real-world datasets. In future work, we plan to extend GRIP to dynamic environments and incorporate temporal or priority-aware constraints.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China (Grant Number: 62372376).

References

Bai, S.; Shan, T.; Chen, F.; Liu, L.; and Englot, B. 2021. Information-driven path planning. *Current Robotics Reports*, 2(2): 177–188.

Bian, C.; Feng, C.; Qian, C.; and Yu, Y. 2020. An efficient evolutionary algorithm for subset selection with general cost constraints. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 3267–3274.

Davis, G.; Mallat, S.; and Avellaneda, M. 1997. Adaptive greedy approximations. *Constructive approximation*, 13: 57–98.

Department, L. A. P. 2024a. Crime in Los Angeles. https://www.kaggle.com/datasets/cityofLA/crime-in-los-angeles. Accessed on 2024-10-01.

Department, S. P. 2024b. Seattle Police Department 911 Incident Response. https://www.kaggle.com/datasets/sohier/seattle-police-department-911-incident-response. Accessed on 2024-10-02.

Enßlin, T. 2013. Information field theory. In *AIP Conference Proceedings*, volume 1553, 184–191. American Institute of Physics.

Feige, U. 1998. A threshold of ln n for approximating set cover. *Journal of the ACM (JACM)*, 45(4): 634–652.

Gao, H.; Zhou, X.; Xu, X.; Lan, Y.; and Xiao, Y. 2023. AMARL: An Attention-Based Multiagent Reinforcement Learning Approach to the Min-Max Multiple Traveling Salesmen Problem. *IEEE Transactions on Neural Networks and Learning Systems*.

Hepp, B.; Nießner, M.; and Hilliges, O. 2018. Plan3d: Viewpoint and trajectory optimization for aerial multi-view stereo reconstruction. *ACM Transactions on Graphics (TOG)*, 38(1): 1–17.

Hu, Y.; Jia, Q.; Chen, J.; Yao, Y.; Pan, Y.; Xie, R.; and Yu, F. R. 2024. CoRaiS: Lightweight Real-Time Scheduler for Multiedge Cooperative Computing. *IEEE Internet of Things Journal*, 11(17): 28649–28666.

Hu, Y.; Yao, Y.; Chen, J.; Wang, Z.; Jia, Q.; and Pan, Y. 2025. Solving Scalable Multiagent Routing Problems With Reinforcement Learning. *IEEE Transactions on Neural Networks and Learning Systems*, 1–15.

Hu, Y.; Yao, Y.; and Lee, W. S. 2020. A reinforcement learning approach for optimizing multiple traveling salesman problems over graphs. *Knowledge-Based Systems*, 204: 106244.

Kempe, D.; Kleinberg, J.; and Tardos, É. 2003. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, 137–146.

Kim, M.; Park, J.; and Park, J. 2023. Learning to CROSS exchange to solve min-max vehicle routing problems. In *The Eleventh International Conference on Learning Representations*.

Lin, C.; Yang, W.; Dai, H.; Li, T.; Wang, Y.; Wang, L.; Wu, G.; and Zhang, Q. 2023. Near Optimal Charging Schedule for 3-D Wireless Rechargeable Sensor Networks. *IEEE Transactions on Mobile Computing*, 22(6): 3525–3540.

Mooney, P. 2014. Denver Crime Data. https://www.kaggle.com/datasets/paultimothymooney/denver-crime-data. Accessed on 2024-10-01.

Natarajan, B. K. 1995. Sparse approximate solutions to linear systems. *SIAM journal on computing*, 24(2): 227–234.

Nemhauser, G. L.; Wolsey, L. A.; and Fisher, M. L. 1978. An analysis of approximations for maximizing submodular set functions—I. *Mathematical programming*, 14: 265–294.

Nguyen, L. N.; and Thai, M. T. 2021. Minimum Robust Multi-Submodular Cover for Fairness. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 9109–9116.

Ohsaka, N.; and Yoshida, Y. 2015. Monotone k-submodular function maximization with size constraints. *Advances in Neural Information Processing Systems*, 28.

Oshima, H. 2021. Improved randomized algorithm for k-submodular function maximization. *SIAM Journal on Discrete Mathematics*, 35(1): 1–22.

Park, J.; Kwon, C.; and Park, J. 2023. Learn to Solve the Min-max Multiple Traveling Salesmen Problem with Reinforcement Learning. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*, 878–886.

Qian, C.; Li, G.; Feng, C.; and Tang, K. 2018. Distributed Pareto Optimization for Subset Selection. In *IJCAI*, 1492–1498.

Qian, C.; Shi, J.-C.; Tang, K.; and Zhou, Z.-H. 2017a. Constrained monotone k-submodular function maximization using multiobjective evolutionary algorithms with theoretical guarantee. *IEEE Transactions on Evolutionary Computation*, 22(4): 595–608.

Qian, C.; Shi, J.-C.; Yu, Y.; Tang, K.; and Zhou, Z.-H. 2016. Parallel Pareto Optimization for Subset Selection. In *IJCAI*, 1939–1945.

Qian, C.; Shi, J.-C.; Yu, Y.; Tang, K.; and Zhou, Z.-H. 2017b. Subset selection under noise. *Advances in neural information processing systems*, 30.

Qian, C.; Yu, Y.; and Zhou, Z.-H. 2015. Subset selection by Pareto optimization. *Advances in neural information processing systems*, 28.

Rafiey, A.; and Yoshida, Y. 2020. Fast and private submodular and k-submodular functions maximization with matroid constraints. In *International conference on machine learning*, 7887–7897. PMLR.

Roostapour, V.; Neumann, A.; Neumann, F.; and Friedrich, T. 2022. Pareto optimization for subset selection with dynamic cost constraints. *Artificial Intelligence*, 302: 103597.

Said, T.; Wolbert, J.; Khodadadeh, S.; Dutta, A.; Kreidl, O. P.; Bölöni, L.; and Roy, S. 2021. Multi-robot information sampling using deep mean field reinforcement learning. In *2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 1215–1220. IEEE.

Spaeh, F. C.; Ene, A.; and Nguyen, H. 2025. Online and streaming algorithms for constrained k-submodular maximization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 20567–20574.

Sun, Y.; Lin, C.; Dai, H.; Wang, P.; Wang, L.; Wu, G.; and Zhang, Q. 2022. Trading off Charging and Sensing for Stochastic Events Monitoring in WRSNs. *IEEE/ACM Transactions on Networking*, 30(2): 557–571.

Ward, J.; and Živnỳ, S. 2014. Maximizing bisubmodular and k-submodular functions. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, 1468–1481. SIAM.

Zhang, H.; and Vorobeychik, Y. 2016. Submodular optimization with routing constraints. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30.