A Quantitative Evaluation Pipeline for Map Partitioning for Grid Pathfinding

Yan Wu[1], Chenyuan Zhang[2], Yangmengfei Xu[1], Guang Hu[1]
[1]The University of Melbourne
Melbourne, Australia
[2]Monash University
Melbourne, Australia
wy459903122@gmail.com, chenyuan.zhang@monash.edu, yangmengfeix@student.unimelb.edu.au,
guang.hu@unimelb.edu.au

Abstract

With the rapid development of smart factories and autonomous driving, grid-map pathfinding has become a core enabling technology for intelligent navigation systems. However, with the growth in map scales and environmental complexity, The computational cost of traditional global pathfinding methods such as A* increases significantly, rendering them incapable of satisfying the stringent real-time requirements of modern autonomous systems. Map partitioning decomposes the map into subregions and significantly accelerates the overall pathfinding process. Nevertheless, existing literature lacks a systematic and quantitative framework to evaluate the effectiveness of different map partitioning schemes on pathfinding performance. To address this gap, we propose a standardized evaluation pipeline consisting of three core steps: sampling representative pathfinding query points on grid maps, precomputing hierarchical pathfinding structures for partitioned maps, and quantitatively assessing partitioning quality using hierarchical pathfinding-derived metrics. Furthermore, we employ machine learning models to predict partitioning effectiveness in terms of the Expanded Node Reduction Percentage, leveraging both map partitioning-derived features and inherent map structural features. This avoids costly pathfinding computations and enables efficient performance evaluation.

1 Introduction

With the widespread deployment of mobile robots and autonomous navigation systems, grid-based maps have emerged as the dominant spatial representation for path planning across diverse application domains (Raja and Pugazhenthi 2012; Debnath et al. 2024). This representation provides a straightforward and computationally tractable approach to discretizing continuous spatial environments and encoding binary occupancy information, distinguishing between traversable free space and non-traversable obstacles. However, as map scales expand and environmental complexity increases, the computational complexity of traditional global path planning algorithms—most notably A*—grows superlinearly, leading to prohibitive performance degradation in large-scale scenarios (Harabor and Botea 2008). Specifically, the polynomial growth in the number of node expansions incurred by global search results in excessive time and memory overhead, severely limiting the real-time responsiveness of autonomous systems.

To mitigate this computational bottleneck, map partitioning has emerged as a highly effective optimization paradigm for grid-based path planning (Zhou, Lu, and Lv 2024). This approach decomposes grid maps into non-overlapping subregions for hierarchical pathfinding such as Hierarchical Path-Finding A* (HPA*) and Hierarchical Annotated A* (HAA*) , which precomputes region-level connectivity and short paths offline to drastically reduce online node expansions and improve computational efficiency (Botea, Müller, and Schaeffer 2004; Harabor and Botea 2008). Existing partitioning methods use diverse criteria to generate subregions, and varying algorithms or subregion counts can produce numerous distinct partitioning schemes for any grid map (Aurenhammer 1991; Chazelle 1984; Shi and Malik 2000). However, these schemes produce highly variable pathfinding performance gains, hierarchical algorithms like HPA* and HAA* fail to guarantee path optimality, and no standardized quantitative framework exists to systematically evaluate them and select the optimal partitioning strategy for a given map.

Existing literature on grid map partitioning predominantly focuses on the development of novel partitioning algorithms, while relatively little attention has been paid to establishing rigorous benchmarks and evaluation methodologies for assessing partitioning performance (Bormann et al. 2016; Jones, Djahel, and Welsh 2023). Furthermore, existing grid map pathfinding benchmarks primarily center on sampling strategy design but lack a unified, standardized methodology to quantitatively measure the pathfinding efficiency improvements specifically brought about by map partitioning. While they provide diverse grid maps and adopt single-type path-related metrics for evaluation, they do not establish a systematic sampling and quantitative evaluation pipeline tailored for partitioning performance assessment (Sturtevant 2012; Harabor, Hechenberger, and Jahn 2022). The core challenge addressed in this work is the development of a standardized evaluation pipeline that can quantitatively measure pathfinding efficiency improvements while rigorously preserving the optimality of the generated paths.

The primary contribution of this paper is a standardized quantitative evaluation pipeline for systematically assessing the impact of grid map partitioning on pathfinding efficiency. Central to this pipeline is a novel two-dimensional stratified sampling (2DSS) method, which jointly considers

path length and node expansion count to construct a statistically representative test set for robust partitioning performance evaluation. This study also includes Linear Regression and Random Forest models based on core structural features to directly predict the Expanded Node Reduction Percentage (ENRP), which measures the proportion of expanded nodes saved by Hierarchical A* (which designates all subregion boundaries as traversable gateways, unlike HAA* and HPA* that only use partial boundaries) compared to global A* after map partitioning, thereby reducing the computational effort required for partitioning quality assessment (Botea, Müller, and Schaeffer 2004; Harabor and Botea 2008).

2 Related Work

2.1 Hierarchical Pathfinding

Hierarchical pathfinding methods are fundamentally based on the principles of decomposition and abstraction. They partition the original grid or raster map based on spatial or semantic characteristics, first computing a coarse-grained path on the high-level abstract graph and then refining it locally, which significantly reduces the search space and computational time. Representative methods include HPA* and HAA*. HPA* is designed for single-agent pathfinding on large-scale maps, with an emphasis on spatial partitioning and gateway-to-gateway cost caching (Botea, Müller, and Schaeffer 2004). HAA* extends the hierarchical framework with capability annotations to handle heterogeneous agents with different sizes and terrain traversal capabilities (Harabor and Botea 2008). However, these methods use only partial boundaries as gateways, a design choice that does not guarantee path optimality. In contrast, the hierarchical pathfinding approach proposed in this paper uses all boundaries as gateways, which ensures that all possible entry and exit points between adjacent subregions are fully covered, thus guaranteeing the optimality of the resulting paths.

2.2 Map Partitioning

Various map partitioning strategies have been proposed for hierarchical pathfinding. One classic method is Normalized Cut (NCut), which models the environment as an undirected weighted graph $G = (V, E)$ where nodes represent spatial positions and edge weights encode the similarity and reachability between nodes. For a partition of the graph into two subsets A and B, the optimization objective of NCut is defined as:

$$\text{NCut}(A, B) = \frac{\text{cut}(A, B)}{\text{assoc}(A, V)} + \frac{\text{cut}(A, B)}{\text{assoc}(B, V)}$$

where $\text{cut}(A, B)$ denotes the connection cost between regions A and B, and $\text{assoc}(\cdot)$ measures the degree of association between the interior of a region and the entire graph. Minimizing NCut yields partitions with strong internal connectivity and low inter-region coupling, which align well with the natural topological structure of the environment (Shi and Malik 2000). However, directly solving the NCut problem is NP-hard; thus, spectral graph theory is commonly used for approximate optimization, achieving efficient partitioning by computing the eigenvectors of the normalized Laplacian matrix (Von Luxburg 2007).

Another widely used method is the Voronoi diagram, a geometric partitioning approach based on spatial distance (Aurenhammer 1991). Given a set of seed points, the entire grid is divided into non-overlapping subregions, where any point in a subregion is closer to the seed point of that region than to any other seed point. The boundaries between regions are the equidistant trajectories of adjacent seed points.

2.3 Benchmark Sampling Strategies

Benchmark sampling strategies play a critical role in evaluating hierarchical pathfinding methods. In the Moving AI benchmark dataset (Sturtevant 2012), the sampling process first generates 100,000 random point pairs, then groups these pairs into different buckets based on $\lfloor \ell/4 \rfloor$ (where ℓ is the length of the optimal path for each pair). Only the first ten sampled points in each bucket are retained, and buckets that are not fully filled are discarded. However, using only path length as the metric tends to oversample point pairs with long paths but simple structures.

Harabor, Hechenberger, and Jahn (2022) partitioned 50,000 points pairs into n buckets (each containing 10 to 19 points) based on expansion cost, selected the buckets with the highest and lowest expansion costs, clustered the remaining buckets into 198 clusters using k-means, and chose the bucket with the highest expansion cost from each cluster, resulting in a total of 2,000 selected points. However, relying solely on expansion costs fails to distinguish between two fundamentally different scenarios that lead to high expansion costs: one where more nodes are expanded simply because the path itself is long, and another where it is due to the complex local structure of the map.

Both sampling methods rely on a single-dimensional metric, making it difficult to account for the diversity of paths at different hierarchical levels.

3 Methodology

For a given grid map, different partitioning methods and partition settings (such as the number of regions) may produce multiple non-overlapping region decompositions. Since these partitioning schemes can affect hierarchical pathfinding performance in different ways, this study establishes a standardized evaluation pipeline to assess their impact on pathfinding efficiency while preserving path optimality. The pipeline is designed as follows:

1. Import the grid map and the corresponding partitioning results (Section 3.1);
2. Perform two-dimensional stratified sampling on the map to generate a set of test points (Section 3.2);
3. Execute the global A* algorithm on the sampled test points and record path cost, pathfinding time, and the number of expanded nodes as the baseline for comparison (Section 3.2);
4. Conduct preprocessing based on the partitioning results to extract relevant features (Section 3.3);

5. Execute the Hierarchical A* algorithm and compare its results against the baseline (Section 3.4).

3.1 Import of Map and Partition Results

The first step of this pipeline is to import the grid map to be evaluated and its corresponding precomputed partition results, laying the foundation for all subsequent experiments and evaluations.

Input Content:

- The grid map consists of traversable grids and obstacle grids and is represented by an undirected graph $G = (V, E)$, where V denotes the set of traversable nodes and E denotes the adjacency relation between them. Obstacle grids are excluded from V.
- The traversable part of the map is partitioned into n non-overlapping connected sub-regions $\mathbb{X} = \{X_1, \ldots, X_n\}$, whose union covers all traversable nodes in the map.

3.2 Hierarchical Sampling of Test Point Set

This sampling framework uses two core indicators of pathfinding difficulty: path length and the number of expanded nodes. Therefore, we propose the two-dimensional stratified sampling (2DSS) scheme, where the first dimension groups samples by path length and the second groups them by expanded nodes using a globally fixed bin width. For each resulting two-dimensional cell, one of two fixed sampling sizes (1 or 3 samples) is used; if a cell contains fewer samples than required, all available samples are retained. In addition, the global weight of each path-length bucket and the local weight of each cell are recorded to support weighted evaluation. The specific steps are as follows.

Initial Random Sampling and Global A* Path Calculation We randomly generate 50,000 start-goal pairs $(\boldsymbol{s}_t, \boldsymbol{g}_t)$ from the traversable node set V of the global map, where $t \in \{1, 2, \ldots, T\}$ and $T = 50000$. For each pair, we run the standard global A* algorithm using Manhattan distance as the heuristic and record two core metrics:

1. l_t: the shortest path length for the pair, i.e., $l_t = d_G(\boldsymbol{s}_t, \boldsymbol{g}_t)$, where $d_G(\cdot, \cdot)$ denotes the shortest path distance in G;
2. n_t: the number of unique nodes expanded by the global A* algorithm during the search, used as the primary metric of computational cost and search effort.

Bucket Division Based on Path Length Following the Moving AI Benchmark (Sturtevant 2012), we first filter out invalid start-goal pairs (those without a valid shortest path or with coincident start and goal points, i.e., $l_t = 0$). We then divide all remaining valid pairs into buckets using a fixed path-length interval of 4. This ensures the number of buckets is automatically determined by the path length distribution of the map, while preserving full coverage of all observed path length ranges.

The bucket ID b_t for each valid pair $(\boldsymbol{s}_t, \boldsymbol{g}_t)$ is calculated as:

$$b_t = \left\lfloor \frac{l_t}{4} \right\rfloor,$$

where path lengths in the interval $[4k, 4(k+1))$ are assigned to bucket k.

Let $B = \{b_1, b_2, \ldots, b_M\}$ denote the set of non-empty valid buckets (i.e., buckets containing at least one valid sample), where M is the total number of valid buckets. For each bucket $b \in B$, we define two parameters:

1. Sample count of bucket b: $N_b = |\{(\boldsymbol{s}_t, \boldsymbol{g}_t) \mid b_t = b\}|$;
2. Global weight of bucket b: $w_b = \frac{N_b}{N_B}$, where $N_B = \sum_{b \in B} N_b$ is the total number of valid samples.

Global Stratification of Expanded Nodes To ensure the consistency of stratification across different path-length buckets, we calculate a global unified stratification standard for expanded nodes using all valid samples:

1. Extract expanded node values of all valid samples to form the global set $E_{\text{global}} = \{n_t \mid l_t > 0 \wedge n_t \geq 1\}$, and calculate the global minimum and maximum values:

$$n_{\text{global,min}} = \min(E_{\text{global}}), \quad n_{\text{global,max}} = \max(E_{\text{global}})$$

2. Calculate the global unified bin step size Δn_{global} based on the total number of valid buckets M:

$$\Delta n_{\text{global}} = \frac{n_{\text{global,max}} - n_{\text{global,min}}}{M}$$

3. Divide the global value range of expanded nodes into M continuous, non-overlapping, equal-width bins. Let the set of all bins be $K = \{1, 2, \ldots, M\}$. The k-th bin interval $[n_k^{\text{low}}, n_k^{\text{high}})$ $(k \in K)$ is defined as:

$$n_k^{\text{low}} = n_{\text{global,min}} + (k-1) \cdot \Delta n_{\text{global}}$$

$$n_k^{\text{high}} = n_{\text{global,min}} + k \cdot \Delta n_{\text{global}}$$

Two-Dimensional Cell Sampling and Weight Recording After determining the path-length buckets and global unified expanded node bins, we perform fixed-size sampling in each two-dimensional cross-interval formed by bucket $b \in B$ and bin $k \in K$ as the core step of the 2DSS method. Let $C_{b,k}$ denote the set of samples falling within the cross-interval of bucket b and bin k, i.e., $C_{b,k} = \{(\boldsymbol{s}_t, \boldsymbol{g}_t) \mid b_t = b \wedge n_t \in [n_k^{\text{low}}, n_k^{\text{high}})\}$. Two fixed sampling sizes are adopted: 1 sample or 3 samples per cross-interval, and all samples in $C_{b,k}$ are selected in full if the size of $C_{b,k}$ is insufficient for the target sampling number.

We also record the weight corresponding to each sampled test point. Let $S_{b,k}$ denote the number of samples selected from the cross-interval (b, k) (i.e., the number of sample points in this single cross-interval) and $S_b = \sum_{k \in K} S_{b,k}$ denote the total number of samples selected from the bucket b. We first calculate the local weight of the cross-interval (b, k) within its corresponding bucket:

$$w_{b,k} = \frac{S_{b,k}}{S_b}$$

where $\sum_{k \in K} w_{b,k} = 1$.

For each test point t selected from the cross-interval (b, k), its associated weight w_t is defined by first determining the local weight $w_{b,k}$ of that cross-interval within its corresponding path-length bucket, and then uniformly allocating $w_{b,k}$ among all test points selected from the interval.

This sampling framework ensures that the stratification standard of expanded nodes is globally unified, avoiding the bias caused by per-bucket independent stratification. The recorded global bucket weights and local cross-interval weights enable rigorous weighted performance evaluation, ensuring the test set is representative of the underlying distribution of the original dataset.

3.3 Partition Feature Preprocessing

This section describes the precomputation procedures performed after map partitioning, followed by the intra-region pathfinding algorithm for queries where both the start and goal states reside within the same subregion. Once the hierarchical partition is established, each region X_i is characterized by a set of boundary nodes that enable cross-region navigation. Formally, a boundary node is defined as any node in X_i that has at least one neighbor belonging to a different adjacent region. Pairs of adjacent boundary nodes from neighboring regions then constitute the undirected inter-region edges in the high-level abstract graph. In contrast to the common design choice in hierarchical pathfinding methods such as HPA* and HAA*to select only a subset of boundaries as gateways for inter-region traversal (Botea, Müller, and Schaeffer 2004; Harabor and Botea 2008), Hierarchical A* designates all nodes adjacent to any other region as traversable boundary nodes.

Boundary Definition For any partition region, we define its boundary node set as the collection of all nodes that lie within the region itself and are adjacent to at least one node belonging to a different adjacent traversable region. This boundary set makes no distinction between entry and exit directions for traversal; every node within the set acts as an undirected connection point between the region and its neighboring regions in the undirected grid graph.

For any two distinct and adjacent partitioned regions, the boundary relationship between them is symmetric. This means that for every boundary node in one region that is adjacent to the other region, there exists a matching boundary node in the adjacent region that is directly adjacent to it, and these two nodes together form an undirected cross-region edge with a traversal cost of 1. This symmetric boundary definition ensures undirected connectivity in the high-level abstract region graph, and guarantees that the Hierarchical A* algorithm can accurately model the traversal capabilities between adjacent regions in the original undirected grid map.

Intra-Region Cost Computation Within each region X_i, the shortest path costs between all pairs of boundary nodes and internal nodes must be precomputed to accelerate the hierarchical search. Let $d^*(p, q)$ denote the exact shortest path cost between nodes p and q within region X_i. We first introduce the calculation method for Boundary-to-All-Nodes Propagation, which forms the local transition costs:

For each boundary node $pos \in \text{Boundary}(X_i)$, compute the cost from pos to all nodes within the same region (including other boundary nodes):

$$\text{Cost}(pos, pos') = d^*(pos, pos'), \quad \forall pos' \in X_i$$

Through these computations, a local boundary-to-all-nodes cost table is constructed for each region, providing a basis for higher-level boundary-to-boundary transitions. By combining intra-region symmetric boundary-to-boundary costs with cross-region undirected adjacency edges, a global abstract boundary graph is established, which serves as the input for the Hierarchical A*'s high-level planning phase.

Intra-Region Boundary-to-Boundary Cross-Cost Computation After defining the boundary nodes of each region, we precompute the shortest path costs between all pairs of boundary nodes within the same region. These costs form the core edge weights of the high-level abstract graph used by the Hierarchical A* algorithm, and are symmetric due to the undirected nature of the grid map.

For each region X_i, we first obtain its complete boundary node set $\text{Boundary}(X_i)$. For every boundary node $v \in \text{Boundary}(X_i)$, we perform a Breadth-First Search (BFS) restricted to the interior of X_i, computing the exact shortest path cost from v to all other nodes in X_i, including all other boundary nodes.

The results are stored in two lookup tables: - $\text{B2SCost}(X_i, v, u)$: The shortest path cost from boundary node v to any node u within region X_i - $\text{B2BCost}(X_i, v, v')$: The shortest path cost between two boundary nodes v and v' within the same region X_i

Following the boundary node identification process that defines all valid cross-region connection points, these precomputed intra-region transition cost tables enable constant-time lookup during high-level planning, eliminate the need for repeated low-level searches within regions, and collectively constitute the complete cross-boundary precomputation infrastructure for the Hierarchical A* algorithm.

3.4 Hierarchical A* Based on start-goal pairs

When the start node and goal node lie in the same region, the global A* algorithm can be used to ensure an optimal path. When they lie in different regions, the Hierarchical A* algorithm proceeds as follows to find an abstract high-level path and then construct the concrete low-level path:

In this stage, given a start node s and a goal node t, the system does not explicitly enumerate all possible boundary-to-boundary combinations. Instead, it performs an implicit A* search on an abstract boundary graph, which is dynamically generated on-the-fly from precomputed data. The vertices of this abstract graph consist of the start node s, the goal node t, and boundary nodes encountered during the search. By searching over this compact abstract graph rather than the full grid, the algorithm dramatically reduces the search space while still guaranteeing correctness—this is enabled by the precomputed intra-region cost structure.

Edges and Costs. All edge costs are symmetric due to the undirected nature of the grid map, and are derived entirely from precomputed data:

- Edges from the start node to all boundary nodes in the start region R_s: (s, u) for $u \in \text{Boundary}(R_s)$, with cost $c(s, u) = d^*(s, u)$, where $d^*(s, u)$ denotes the exact shortest path cost from s to u within R_s;

- Intra-region boundary edges between any two boundary nodes within the same region: (u, v) for $u, v \in \text{Boundary}(R_i)$ and $u \neq v$, with cost $c(u, v) = d^*(u, v)$, taken from the precomputed boundary-to-boundary cost table;
- Cross-region edges between adjacent boundary nodes from neighboring regions: (u, v) where $u \in \text{Boundary}(R_i)$, $v \in \text{Boundary}(R_j)$, and u and v are directly adjacent in the grid, with a fixed unit traversal cost $c(u, v) = 1$;
- Edges from all boundary nodes in the target region R_t to the goal node: (v, t) for $v \in \text{Boundary}(R_t)$, with cost $c(v, t) = d^*(v, t)$.

Heuristic Function. To guarantee admissibility for the undirected abstract graph, we use the standard Manhattan distance as the heuristic function, which is consistent with the heuristic used in low-level A* search:

$$\widehat{h}(x) = |r_x - r_t| + |c_x - c_t|$$

where (r_x, c_x) denotes the grid coordinates of node x, and (r_t, c_t) denotes the coordinates of the goal node t.

Same-Region Case. If $R_s = R_t$:

- A full global A* with the Manhattan distance as the heuristic function is executed to ensure an optimal path.

Different-Region Case. When $R_s \neq R_t$, the total cost of a candidate abstract path can be decomposed into the sum of intra-region shortest path costs and cross-region unit traversal costs. A* on the dynamically generated abstract graph finds the path that minimizes this total cost without explicitly enumerating all possible boundary node combinations.

The admissibility of the Manhattan distance heuristic ensures the optimality of the search result. The performance improvement of the Hierarchical A* algorithm is evaluated by comparing its runtime and node expansion metrics with those of the standard global A* algorithm.

3.5 Performance Prediction of Partitioning Schemes

To further quantify the correlation between partitioning features and pathfinding efficiency improvement, and to enable the automatic selection of optimal partitioning schemes, this study constructs a prediction model that maps partitioning features to pathfinding performance. The core framework includes feature engineering, model evaluation, and optimal scheme selection, with detailed steps as follows.

Feature Engineering The input features of the model focus on core indicators that capture the structural characteristics of partitioning schemes, covering the following two core categories:

- **Subregion Structural Features**
  These features are exclusively determined by the map partitioning result, capture the structural properties of the resulting subregions, and serve as the core inputs for distinguishing the performance of different partitioning schemes.
  - **Number of subregions**: Total number of subregions obtained from the global map partitioning, reflecting the partitioning granularity.
  - **Total boundary node count**: Total number of boundary nodes across all subregions, measuring the overall inter-region connection overhead.
  - **Subregion size balance**: Standard deviation of the areas of all subregions, characterizing the size uniformity of subregions.
  - **Subregion morphology**: Average eccentricity of all subregions, reflecting the geometric regularity of subregion shapes.
  - **Structural efficiency**: Mean and standard deviation of the ratio of subregion area to the number of boundary nodes in that subregion, comprehensively measuring the trade-off between subregion compactness and inter-region connection cost.
- **Map Inherent Global Features**
  These features are determined by the original grid map itself, independent of the partitioning scheme. They depict the inherent complexity of the map and provide consistent global contextual information for the model.
  - **Number of traversable nodes**: Total number of passable nodes in the global map, reflecting the scale of the pathfinding search space.
  - **Average node degree**: Average degree of all traversable nodes in the grid graph, characterizing the connectivity density of the underlying map.
  - **Topological complexity**: Ratio of the number of dead-end nodes to the number of bottleneck nodes in the map, measuring the inherent path search difficulty of the original grid.

The output feature of the model is defined as the ENRP. This indicator quantifies the pathfinding efficiency improvement of a given partitioning scheme compared to the standard global A* algorithm, with a value range of $[0, 100\%]$. A higher ENRP value indicates a better optimization effect of the partitioning scheme.

Model Evaluation Metrics Four metrics are used to comprehensively assess the prediction performance of the model, each selected to evaluate a distinct aspect of predictive quality:

- Mean Squared Error (MSE): Computed as the average squared difference between predicted values and ground-truth values, reflecting the overall regression error of the model for ENRP value prediction (Chicco, Warrens, and Jurman 2021). It is selected as a fundamental metric to directly quantify the magnitude of prediction deviation in absolute terms.
- Coefficient of Determination (R^2): Represents the proportion of variance in the target variable that can be explained by the input map features, indicating the goodness of fit of the regression model (Chicco, Warrens, and Jurman 2021). It is chosen to measure how effectively map structural features account for variations in ENRP.

- Kendall Tau (K-Tau): Measures the ordinal association between predicted and true rankings of partitioning schemes by counting the proportion of concordant and discordant pairs, evaluating the consistency of ranking predictions (van den Heuvel and Zhan 2022). It is adopted to assess the reliability of ranking different partition strategies independently of absolute prediction values.
- Spearman Rho (S-Rho): Calculates the Pearson correlation coefficient between the predicted ranks and the true ranks of partitioning schemes, quantifying the accuracy of ordinal ranking predictions (van den Heuvel and Zhan 2022). It is selected to complement K-Tau and robustly evaluate the correctness of the relative ordering of partition schemes.

4 Experiment

4.1 Test Environment

All experiments are conducted on Google Colab, equipped with 12GB of RAM and 100GB of available storage.

4.2 Experimental Setup

Research Objectives All experiments are conducted based on 4-directional grid pathfinding as the unified evaluation foundation. The core objectives of the experiments are as follows:

- Verify that the 2DSS method can cover pathfinding scenarios with different path lengths and expanded nodes, ensuring the representativeness and generalization ability of the test set.
- Evaluate the efficiency improvement of the Hierarchical A* algorithm compared with the global A* algorithm under various partitioning schemes, including the reduction in expanded nodes, generated nodes, and the weighted reduction value.
- Validate the prediction accuracy of the Random Forest (RF) and Linear Regression (LR) models for the ENRP values of partitioning schemes, as well as the effectiveness of model-based optimal partitioning scheme selection.

Datasets and Partitioning Schemes The base experimental data are sourced from the public Moving AI dataset, which is established and maintained by Sturtevant (2012). We select 30 grid maps with dimensions smaller than 100×100 from the dataset as the test scenarios for all experiments. This selection is driven by computational efficiency considerations: when evaluating different sampling methods across multiple map partition schemes, each sampling point requires hundreds of independent runs of either global A* or hierarchical A*. For maps larger than 100×100, the cumulative computational cost of such repeated executions becomes prohibitively high. We therefore restrict our test set to maps below this size threshold to ensure the feasibility of the extensive experimental evaluations and accelerate the overall experimental pipeline.

To carry out comparative experiments on different partitioning strategies, we generate multiple partitioning schemes for each base map using a three-stage partitioning and optimization framework:

- Graph-based partitioning: The Normalized Cut (NCut) algorithm and Voronoi-based partitioning method are adopted, which divide the map into 2 to 30 subregions.
- Coordinate-based uniform partitioning: A set of uniform grid partitioning schemes is constructed, including fixed divisions of 1×30 and 2×15, as well as a series of divisions with gradually increasing computational workload, ranging from 2×2, 2×3, 3×2, 3×3 up to 5×5.
- Connectivity post-processing: Disconnected or fragmented subregions generated by the above two strategies are merged into adjacent complete regions to ensure that each final subregion is fully connected internally.

For all subsequent comparative experiments on different sampling methods, we exclusively adopt the partitioning schemes generated by the NCut algorithm to maintain a unified experimental baseline. NCut is chosen because it tends to produce relatively balanced subregions, whereas Voronoi-based partitioning and coordinate-based uniform partitioning often exhibit significant size imbalance among subregions due to the distribution of obstacles (Shi and Malik 2000; Aurenhammer 1991). This balanced partitioning is particularly important given that our sampling baseline is based on random point sampling, ensuring that randomly selected start and end points are evenly distributed across the entire map.

Model Overview To investigate both linear and non-linear relationships between map structural features and ENRP, and to quantify the impact of map features on pathfinding performance, we employ two fundamental machine learning models to construct the prediction model for partitioning features and pathfinding performance:

- Linear Regression (LR): A basic regression model that assumes a linear relationship between map structural features and the ENRP. It serves as a baseline model for performance comparison due to its simplicity and interpretability. In our implementation, the model is configured with `fit_intercept=True` to incorporate an intercept term into the linear formulation.
- Random Forest (RF): Taking decision trees as base learners, an ensemble model is built via bootstrap resampling, and the final ENRP prediction is obtained by averaging outputs from all base learners (Rigatti 2017). In this work, the random forest regressor is implemented with three key hyperparameters: `n_estimators=200` to ensure stable and accurate prediction, `random_state=RANDOM_STATE` for experimental reproducibility, and `n_jobs=-1` to enable full parallelism across all available CPU cores for efficient training.

4.3 Experimental Design

Path Sampling Methods To comprehensively compare the performance of different sampling strategies, we set up 6 groups of path sampling schemes in the experiments. The proposed two-dimensional stratified sampling method

serves as the core experimental group, and the remaining 5 groups are set as baseline control methods. The specific implementation rules of each method are described below:

- **Proposed Two-Dimensional Stratified Sampling Method** We adopt a two-dimensional stratified sampling strategy based on path length and expanded nodes. An initial sample pool is constructed with 50,000 random start-goal pairs, which are divided into path-length buckets and global equal-width expanded node bins, respectively. Two sampling variants are set for test set construction: extracting 1 sample (`2DSS`) or 3 samples (`2DSS_3`) from each cross-interval.
- **Random Sampling Method** This method generates start-goal pairs in a completely random manner(`random`), with a significantly larger number of samples than the final sample size of 2DSS, serving as the unbiased baseline for comparison.
- **Default Sampling Method from Moving AI Benchmarks** This method adopts the native path sampling scheme matched with the Moving AI dataset(`movingai_old`), which is a widely used benchmark sampling protocol for grid map pathfinding algorithm experiments (Sturtevant 2012).
- **One-Dimensional Full-Bucket Retention Sampling Method** This method is also built on 50,000 initial random start-goal pairs, and performs bucketing on the path length dimension with a step size of 4(`movingai`). The core difference from `movingai_old` is that this method fully retains all buckets (even if the bucket is empty or not filled with samples) before sampling based on the bucketing results.
- **K-means Clustering-Based Sampling Method** This method is implemented based on the K-means clustering sampling framework(`iron`) proposed by Harabor, Hechenberger, and Jahn (2022). The final number of sampled points is close to 2DSS; if the number of samples to be extracted from a single cluster is a non-integer value, it is rounded up to the nearest integer to ensure the rationality of sample allocation.
- **Iron Sampling Method** This method strictly follows the sampling protocol proposed by Harabor, Hechenberger, and Jahn (2022), which extracts exactly 2,000 samples from the initial pool of 50,000 random start-goal pairs to construct the test set(`iron_old`).

Hierarchical A* Execution Hierarchical A* is performed on the stratified test set under two scenarios:

- Same-region case: If the start and goal nodes lie in the same subregion, the global A* algorithm is executed to ensure path optimality.
- Different-region case:
  1. Preprocess: Extract boundary nodes and precompute intra-region and cross-region costs.
  2. Construct the abstract boundary graph G_{abs} and perform Hierarchical A* search.
  3. Record the number of expanded nodes.

Machine Learning Model Training & Validation **Feature and Label Construction** The model inputs are divided into two core categories, as defined in Section 3.1: Subregion Structural Features and Map Inherent Global Features.

Two performance labels are defined:

- Expanded Node Reduction Percentage (ENRP):

$$\text{ENRP} = 100\% \times \frac{\text{ExpandedNodes}_{\text{GlobalA}^*} - \text{ExpandedNodes}_{\text{HierarchicalA}^*}}{\text{ExpandedNodes}_{\text{GlobalA}^*}}$$

- Weighted ENRP: The weighted average of ENRP over all test points according to their sampling weights:

$$\text{Weighted ENRP} = \sum_{t=1}^{T} w_t \cdot \text{ENRP}_t$$

where w_t is the weight of the t-th test point and ENRP_t is its corresponding ENRP value.

Training Protocol

- 5-fold cross-validation under the 2DSS_3 sampling method is adopted to ensure the reliability of model evaluation.
- Both Linear Regression and Random Forest models are trained, with key parameters of RF (number of decision trees, maximum depth, minimum sample split) optimized.
- Four metrics are used to comprehensively assess model performance: Mean Squared Error (MSE), Coefficient of Determination (R^2), Kendall Tau (K-Tau), and Spearman Rho (S-Rho).

Optimal Partitioning Scheme Screening The trained model predicts the ENRP and Weighted ENRP for unseen partitioning schemes. The scheme with the highest predicted score is selected as the model-recommended optimal partition. Its performance is compared with the true optimal scheme obtained from full traversal tests to validate selection accuracy.

5 Results

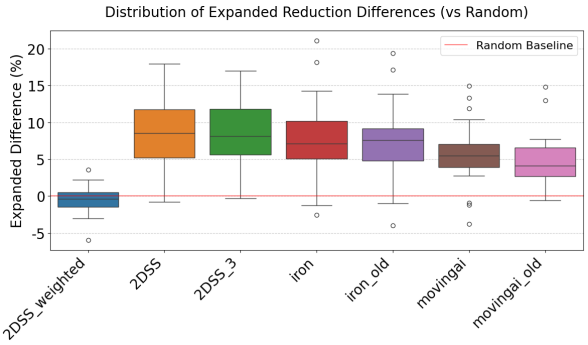

Figure 1: Performance comparison of six path sampling methods on all 30 maps, while the red line represents the random sampling baseline.

5.1 Performance Comparison of Six Sampling Methods

This section compares the test results of the six path sampling methods on all 30 maps using visualization plots, where the results of the random sampling method are used as the baseline (marked as the red line in Figure 1). Additionally, a number of NCut partitioning examples on the same map are provided in Figure 2 to demonstrate the performance of different sampling methods as the number of subregions varies.

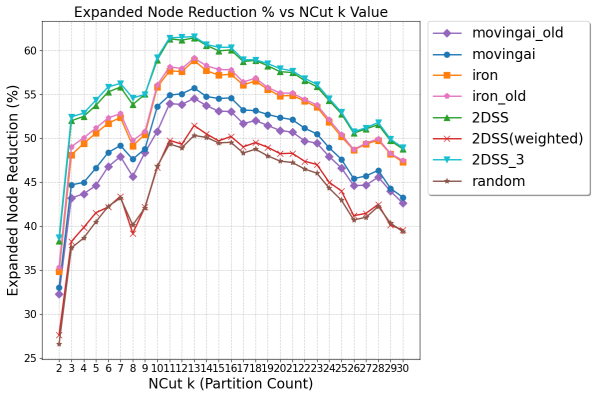

Figure 2: An example of NCut partitioning on a single map.

The results show that all unweighted non-random sampling methods consistently outperform random sampling, with mean expanded-node reduction improvements ranging from 4.81% (`movingai_old`) to 8.64% (`2DSS_3`) across all maps. The main reason is that random sampling is strongly concentrated on short-path cases. As path length increases, the number of valid start-goal pairs decreases sharply, leading to a highly skewed distribution. In contrast, the non-random methods explicitly increase the proportion of long-path pairs and therefore cover rarer and more diverse pathfinding scenarios. Since long-path pairs are typically more sensitive to the node-reduction effect of Hierarchical A* after partitioning, these methods tend to produce higher reduction values than random sampling.

In addition, the path-length-only sampling methods (`movingai` and `movingai_old`) achieve the smallest improvements over the random baseline, with mean gains of 5.68% and 4.81% respectively. The reason is that these methods consider only path length and ignore the number of expanded nodes. As a result, even when paths have the same length, they do not consistently include cases with different search costs, and therefore fail to stably cover more difficult scenarios. Compared with the other targeted sampling methods, they contain fewer samples that are highly sensitive to expanded-node reduction.

The 2DSS variants consistently outperform the Iron-series methods. 2DSS achieves a mean improvement of 8.43% compared to 7.47% for Iron and 7.23% for Iron_old, representing a 0.96% to 1.20% performance advantage. This is because Hierarchical A* is more effective at reducing cross-region expansions than intra-region expansions. However, under the same number of expanded nodes, the Iron-series methods do not ensure sufficient diversity in path length. Consequently, they may include short paths with many expanded nodes, where much of the search effort occurs within regions and is therefore less reducible by Hierarchical A* after partitioning. This further suggests that 2DSS can adapt the sample size to the map scale and provide more informative performance evaluation without fixing the sample size to 2,000 points as in the original Iron methods.

The 3-sample variant, `2DSS_3`, shows statistically negligible differences from the basic `2DSS` method. The mean difference is only 0.21% (8.64% vs 8.43%), the median difference is 0.38% (8.14% vs 8.52%), and the interquartile ranges are nearly identical (6.17 vs 6.52). This indicates that `2DSS` is highly stable with respect to the number of samples selected per cell.

Among the non-random sampling methods, weighted 2DSS (`2DSS_weighted`) is the closest to the random-sampling baseline, with a mean deviation of only -0.46%. All observed deviations fall within the range of -4.52% to 3.56%, which is well within the ±5% threshold for practical equivalence. Given that random sampling uses five times as many samples as 2DSS, this suggests that the proposed weighting strategy can recover the overall distributional effect of global random sampling using a much smaller but more representative test set.

Another advantage of the weighting mechanism is that it can be adjusted to reflect application-specific path distributions rather than a fixed sampling distribution. For example, on an NCut-partitioned 14-region map, reweighting the path-length groups toward a more realistic distribution increases the overall weighted expanded-node reduction rate from 34% to 43%. This shows that weighting can make the evaluation better reflect the practical benefits of a partitioning scheme.

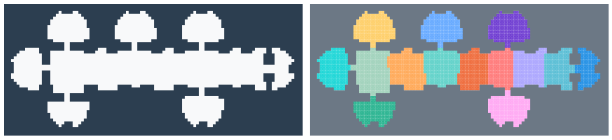

Figure 3: Map partitioning results of a sample map using Normalized Cut (Sturtevant 2012).

5.2 Machine Learning-Based Prediction of Reduction

We attempt to use machine learning methods to predict the node expansion reduction in pathfinding. The model inputs cover the two categories of features defined above: subregion structural features and map inherent global features.

The results show that there is a strong nonlinear correlation between the above map features and the expansion reduction (Table 1). Compared with linear regression, random forest achieves higher accuracy in both reduction value prediction and ordinal ranking prediction, where the Spearman Rho for ranking prediction reaches 0.95, enabling accurate prediction of the ranking of partition performance.

Model & Metric	R2	RMSE	K-Tau	S-Rho
LR (total%)	0.7156	7.4166	0.7041	0.8821
LR (weighted%)	0.7640	6.5644	0.7419	0.9118
RF (total%)	0.8924	4.5597	0.8176	0.9467
RF (weighted%)	0.9232	3.7400	0.8519	0.9657

Table 1: Comparison of Multi-Model Prediction Results

In addition, the prediction accuracy of weighted reduction is 2 percentage points higher than that of the unweighted one, indicating that weighted reduction can better reflect the inherent characteristics of a map. However, the distribution of weighted reduction is closer to random sampling and cannot represent a more comprehensive set of samples.

6 Conclusion and Future Work

This paper presents a standardized quantitative evaluation pipeline for map partitioning in grid pathfinding. The proposed framework covers map and partition-result import, hierarchical sampling of test point pairs, global A* baseline computation, partition feature preprocessing, and Hierarchical A* performance evaluation. In addition, we introduce a two-dimensional stratified sampling method based on path length and expanded nodes to improve the representativeness of the test set, especially for cases with different pathfinding difficulties. We further explore machine learning models for predicting partitioning performance from core structural features, and the experimental results show that Random Forest provides useful predictive performance for expanded-node reduction.

Future work will primarily focus on improving the generalizability of the prediction component. More expressive learning models and richer feature representations, such as graph-based or vision-based approaches, may help improve predictive performance on out-of-distribution map layouts. Once sufficiently robust, the prediction model could be further integrated into an automatic screening mechanism for selecting promising partitioning schemes prior to expensive hierarchical pathfinding evaluation.

References

Aurenhammer, F. 1991. Voronoi diagrams—a survey of a fundamental geometric data structure. *ACM computing surveys (CSUR)*, 23(3): 345–405.

Bormann, R.; Jordan, F.; Li, W.; Hampp, J.; and Hägele, M. 2016. Room segmentation: Survey, implementation, and analysis. In *2016 IEEE international conference on robotics and automation (ICRA)*, 1019–1026. IEEE.

Botea, A.; Müller, M.; and Schaeffer, J. 2004. Near optimal hierarchical path-finding. *J. Game Dev.*, 1(1): 1–30.

Chazelle, B. 1984. Convex partitions of polyhedra: a lower bound and worst-case optimal algorithm. *SIAM Journal on Computing*, 13(3): 488–507.

Chicco, D.; Warrens, M. J.; and Jurman, G. 2021. The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation. *Peerj computer science*, 7: e623.

Debnath, D.; Vanegas, F.; Sandino, J.; Hawary, A. F.; and Gonzalez, F. 2024. A review of UAV path-planning algorithms and obstacle avoidance methods for remote sensing applications. *Remote Sensing*, 16(21): 4019.

Harabor, D.; and Botea, A. 2008. Hierarchical path planning for multi-size agents in heterogeneous environments. In *2008 IEEE Symposium On Computational Intelligence and Games*, 258–265. IEEE.

Harabor, D.; Hechenberger, R.; and Jahn, T. 2022. Benchmarks for pathfinding search: Iron harvest. In *Proceedings of the international symposium on combinatorial search*, volume 15, 218–222.

Jones, M.; Djahel, S.; and Welsh, K. 2023. Path-planning for unmanned aerial vehicles with environment complexity considerations: A survey. *ACM Computing Surveys*, 55(11): 1–39.

Raja, P.; and Pugazhenthi, S. 2012. Optimal path planning of mobile robots: A review. *International journal of physical sciences*, 7(9): 1314–1320.

Rigatti, S. J. 2017. Random forest. *Journal of insurance medicine*, 47(1): 31–39.

Shi, J.; and Malik, J. 2000. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8): 888–905.

Sturtevant, N. R. 2012. Benchmarks for grid-based pathfinding. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(2): 144–148.

van den Heuvel, E.; and Zhan, Z. 2022. Myths about linear and monotonic associations: Pearson'sr, Spearman's ρ, and Kendall's τ. *The American Statistician*, 76(1): 44–52.

Von Luxburg, U. 2007. A tutorial on spectral clustering. *Statistics and computing*, 17(4): 395–416.

Zhou, Y.; Lu, Y.; and Lv, L. 2024. Grid-based non-uniform probabilistic roadmap-based agv path planning in narrow passages and complex environments. *Electronics*, 13(1): 225.