Proceedings of the 4th International Conference on Signal Processing and Machine Learning
DOI: 10.54254/2755-2721/79/20241082

Optimization and comparative analysis of maze generation algorithm hybrid

Kaicheng Yang[1,5], Sutong Lin[2,*], Yu Dai[3,6], Wentai Li[4,7]

[1]School of Mechanical And Electrical Engineering, Soochow University, Suzhou, 215137, China
[2]Maynooth International School of Engineering, Fuzhou University, Fuzhou, 350108, China
[3]International School of Information and Software, Dalian University of Technology, Linyi, 116024, China
[4]Maple Leaf International School, Maple Leaf International School, Dalian, 116699, China

[5]kcyang1848@stu.suda.edu.cn
[*]832103215@fzu.edu.cn
[6]1786933922@qq.com
[7]3048422460@qq.com

Abstract. The complexity of generating intricate and random mazes is a captivating challenge that finds applications in various fields, including computer science, mathematics, gaming, and simulations. This study presents an innovative approach by integrating two prominent perfect maze generation algorithms, Aldous-border (AB) and Wilson. Both are celebrated for their strong randomness and efficiency, yet their combination offers a novel way to optimize maze generation. Our research commenced with a detailed analysis of the relationship between the coverage rate, uniquely characterized by the AB algorithm, and map size. We then formulated a mechanism that transitions seamlessly into the Wilson algorithm, aiming to minimize time consumption. Through a series of carefully designed experimental trials, we hope to use a model to find the most suitable algorithm for switching to minimize the time it takes to generate a maze. These were subsequently evaluated and compared to identify the most fitting solution. Under the framework of our synthesized algorithm, an average time saving of 34.124% was achieved, demonstrating a promising enhancement in efficiency. Although still in the exploratory phase, the outcomes of this research provide foundational insights into maze generation's underlying principles and techniques. The outcomes of this research offer insights into maze generation and its applications and may serve as a useful reference for future studies and potential technological advancements.

Keywords: Algorithm, Maze, Aldous-border, Wilson, Machine Learning.

1. Introduction

In game development, the efficient generation of random mazes is pivotal for enhancing the gaming experience. From the earliest iteration of "Pac Eating Man" to the contemporary game "Dead Cells,"

DOI: 10.54254/2755-2721/79/20241082

developers frequently need to create a lot of mazes in different sizes. Existing maze generation algorithms, such as recursive segmentation, Prim's, and Kruskal's algorithms, often suffer from low generation efficiency or excessive regularity, or unequal possibility of generating all possible mazes. In light of the unique advantages of Aldous-Broder and Wilson's algorithms in generating uniform random spanning trees, this paper proposes to employ them in the design of new maze generation algorithms. Recent research has revealed that by defining and analyzing the branch generation process of Aldous-Broder and Wilson's algorithms, an equivalence between the two can be found at the branch completion point, thereby enabling the design of a hybrid algorithm. This hybrid algorithm exhibits enhanced spanning tree efficiency for complete graphs. It proposes and establishes the initial hybrid algorithm when the f i^{th} branch is generated as the turning point of the hybrid algorithm. However, it does not give the optimal value of i or the calculation formula of i, and we argue that the existing branch-based equivalence analysis is too constrained to the internal mechanisms of the algorithms. As i is hard to use in practical application, a more macroscopic evaluation perspective, considering factors such as coverage, can enable a systematic study of both algorithms, facilitating the discovery of their optimal application range and switching points. Consequently, this article intends to develop a new hybrid algorithm with coverage as its turning point and find out a calculation formula of coverage so that it can be used for maze generation in different sizes. This essay will employ machine learning methods to automate the search for the optimal switching strategy for Aldous-Broder and Wilson's algorithms for mazes of varying sizes. Use a variety of regression analysis and cross-validation methods to build regression models and try to pick out more generalized models. then, this essay will assess and contrast the algorithm optimization effects under different coverage, offering the accuracy of the coverage calculation formula and how much efficiency has been improved compared to Aldous-Broder Algorithm and Wilson Algorithm. What sets this research apart from existing work is the adoption of a new perspective on evaluation indicators and the utilization of automated search methods to pinpoint the optimal switching strategy. Our work focus transcends the mere analysis of the internal mechanisms of algorithms and emphasizes the comprehensive optimization effect of the algorithms. The findings of this study can be directly applied to the efficient generation of mazes of specific sizes in practical games, thereby contributing original ideas and solutions for game maze generation through the innovative fusion of classic algorithms. Wishing it can not only be limited to the generation of mazes, but also hope to further solve the problem of generating random trees in graphs and other scenarios.

2. Related Works

With the development of the gaming industry, random maze generation has become increasingly prevalent, especially in various two-dimensional platform games, where each level is a randomly generated maze. The website [1] describes in detail the features and advantages of different random maze generation algorithms. In particular, according to the website, the Aldous-Broder Algorithm and the Wilson Algorithm are able to generate all possible mazes equally well compared to other algorithms. Compared to the Recursive Division Algorithm, the Aldous-Broder Algorithm and the Wilson Algorithm do not have the inability to generate a certain randomized maze, so they have a wider range of generating results. In contrast to Prim's Algorithm, Aldous-Broder Algorithm and Wilson Algorithm guarantee that each result is equally likely to be generated, so although they were created earlier, they still have their own advantages today. It should be noted, however, that even though the Wilson Algorithm is an improvement on the Aldous-Broder Algorithm and reduces the runtime significantly, the Wilson Algorithm still takes some time. Therefore, it is still necessary to optimize Aldous-Broder Algorithm and Wilson Algorithm or propose a new algorithm with shorter running time while maintaining their superiority. One of the optimization ideas is to mix the two algorithms. Geo Brown et al. further explored the Aldous-Border Algorithm and provided proof for general Markov chains [1]. The shortcomings of the Aldous-Border Algorithm and Wilson Algorithm became apparent, and IGOR NUNES et al. proposed a hybrid generation algorithm based on these two classic methods: Using the notion of branches – paths generated by the two algorithms on stopping times, we show that the trees built by the two algorithms when running on a complete graph are statistically equivalent on these

DOI: 10.54254/2755-2721/79/20241082

stopping times. This leads to a hybrid algorithm that can generate uniform spanning trees of complete graphs faster than either of the two algorithms [2]. Meanwhile, there are many commonly used maze-generation algorithms today, such as assembling mazes to generate large mazes [3], generating mazes using topological constraints [4], creating ring-shaped mazes using L-systems [5], etc.

The hybrid algorithm in this paper also relies on machine learning models, including the linear regression model and the advanced multivariate linear regression model [6], with evaluations based on eigenvalue assessment [7]. Additionally, the paper explores ridge regression models, with R. Uemukai et al. providing proof and deriving the mean squared error (MSE) [8]. Yaojie Zhang et al. further investigates the source of predictability from a variable selection perspective. In the experiments, the lasso model was also tested for fitting [9]; finally, the Elastic-Net Regression model was utilized, selecting two variables for fitting, in an attempt to derive a more optimal model. During the evaluation process, Hong, CS (Hong, Chong Sun) et al. propose an alternative graphical method called the SSR plot for use with a multiple regression model [10], assisting in the calculation of R^2 for the experimental model.

3. Method

1. Code implementation of algorithms

We designed a small model generator using pygame and based on the descriptions of Aldous-Broder and Wilson, we wrote the corresponding code to demonstrate. As figure 1 shows, Wilson's generation method is divided into the following four steps:

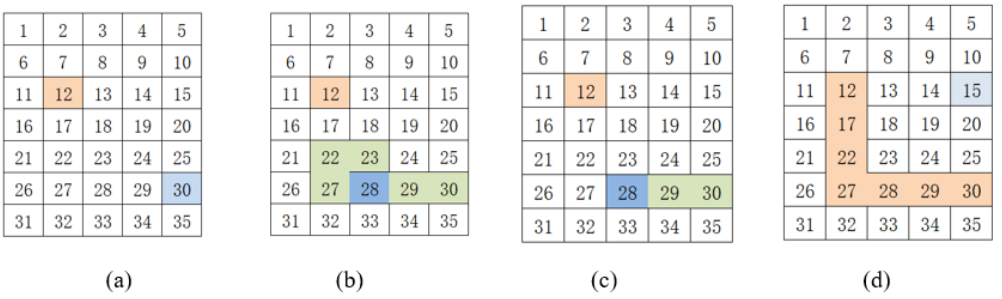

Figure 1. Step Display

The maze maps are shown in Figure 2. The Aldous-Broder generation method involves randomly selecting a point on the map for walking, and if it has not been visited, it will be connected until the entire map has been visited.

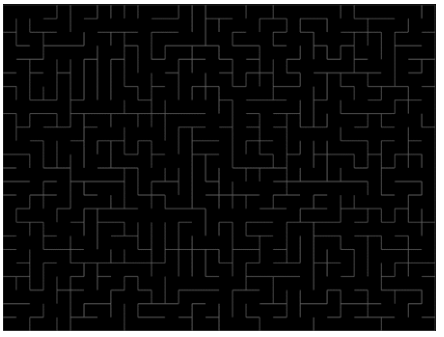

Figure 2. Wilson Maze Generation Display

The generated maze is shown in Figure 3.

DOI: 10.54254/2755-2721/79/20241082

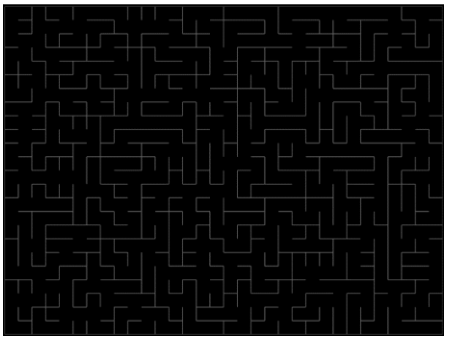

Figure 3. AB Maze Generation Display

We also compared the efficiency characteristics of the two algorithms, as shown in Figure 4 and Figure 5. The horizontal axis represents the generation time, and the vertical axis represents the number of maze cells that have already been generated. It can be seen that the AB algorithm grows rapidly to around 650 cells within the first 6ms but grows by less than 200 cells in the subsequent 8ms. In contrast, the Wilson algorithm grows by 300 cells in the last 12 to 15ms. The AB algorithm generates quickly in the early stage and slowly in the later stage, while Wilson is just the opposite. Furthermore, according to the description in the paper [2], these two algorithms can generate the same tree, meaning that merging the two algorithms will not affect the result. Therefore, we want to merge them.

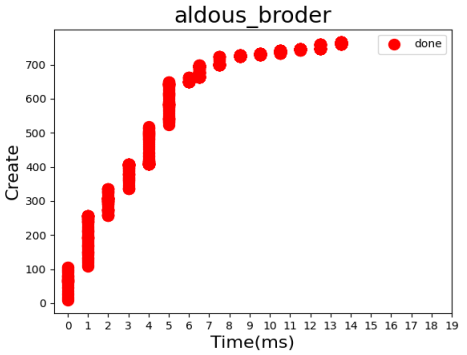

Figure 4. AB1

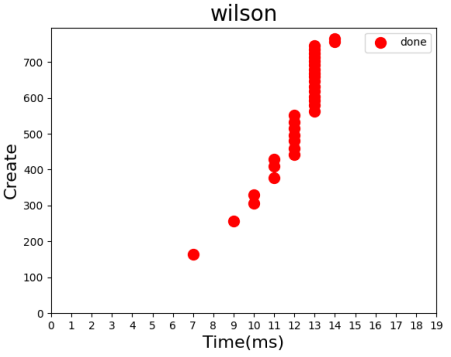

Figure 5. Wilson1

The research gives a detailed demonstration proving the feasibility of the implementation of the hybrid algorithm and gives the hybrid algorithm with the generation of the i^{th} branch as the turning point. It can be implemented like this:

Algorithm 1 Wilson Maze Generation Algorithm

```
Initialize grid with walls
Choose a random starting point s
Mark s as visited
while There are unvisited cells do
    Choose a random unvisited cell c
    Create a random walk starting from c until it hits a visited cell
    Connect the visited cells in the walk
    for each cell in the walk do
        Mark the cell as visited
    end for
end while
```

However, we choose coverage, rather than the generation of the i^{th} branch, as an indicator of turning points. Branching was proposed in the study, but it is not intuitive and easy to understand. And the total number of branches that ultimately generate a maze is not only related to the size of the maze, i.e., the number of nodes in the tree, but also to the algorithm used to generate it, so using the generation of the i^{th} branch as a turning point will not facilitate the tuning of the algorithm.

DOI: 10.54254/2755-2721/79/20241082

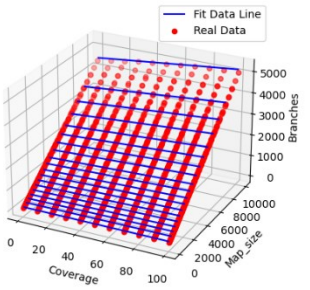

Figure 6. Relationship Fit Result of Coverage, Map_size and Branches

By recording the average total number of branches after every 10 mazes generation for map sizes ranging from 100 to 10000 in a real-machine operation and fitting to visualize relationships as Figure 6, we find that the Wilson algorithm usually generates fewer branches compared to the Aldous-Broder algorithm. And as the coverage rises, the total number of branches generated by the hybrid algorithm will also increment from the total number of branches generated by the Wilson algorithm to the total number of branches generated by the Aldous-Broder algorithm.

Therefore, the number of generated branches should be highly or perfectly correlated with the map size with respect to the coverage and considering that the number of generated branches is not always adjusted as an independent variable, we decided to use the coverage as an indicator of the turning point. The new hybrid algorithm will be realized like this:

Algorithm 2 Hybrid Algorithm(I, Column, Row)

```
Input: I, Column, Row
Unvisited Neighbor[Column][Row] ← every Neighbor
Branch Count ← 0
while Branch Count < I do
    Neighbor Grid ← Randomly Choose Neighbor(Unvisited Neighbor)
    if Is VisitedNeighbor Grid then
        if Is First VisitNeighbor Grid then
            Branch Count ← Branch Count + 1
        end if
    else
        if Is First VisitNeighbor Grid then
            Set Neighbor As New BranchNeighbor Grid
        end if
        Mark Neighbor As Visited And LinkNeighborGrid
        Remove Neighbor From Unvisited ListNeighborGrid
    end if
end while
Wilson Algorithm (Unvisited-Neighbor)
```

The process of the merged hybrid algorithm is to first use the AB algorithm, and if the best value of the characteristic quantity we set is reached, it switches to the Wilson algorithm. The algorithms within this paper test the maze generation speed in different map sizes by passing different coverage rates as feature values.

2. Coverage based regression model

A. Data processing

Step 1: Data Acquisition and Cleaning.

We first generated 100 sets of data as the learning set, with each set containing data on coverage rates ranging from 0% to 100% for map sizes ranging from 100 to 10,000. Within this, we obtained the average.

Step 2: Feature scaling

We use map size and coverage as feature vectors. However, since the variation range of map size from 100 to 10000 is too large compared to the variation range of coverage from 0 to 1, it will make the whole convergence trajectory complicated and make the convergence time longer. In practice, we also found that the complex vectors without feature scaling will make the convergence effect become very poor, so it is necessary to perform feature scaling on the features.

DOI: 10.54254/2755-2721/79/20241082

We use Max-Min normalization for feature scaling. The reason for not using standard deviation normalization is that after standard deviation normalization, the whole data set will be shifted to the position centered at 0, and at the same time, it will be scaled to the interval where the standard deviation is 1. Therefore, the predicted time in the interval where the size of the map is small will be concentrated in the negative interval after standard deviation normalization, and a large number of predicted times after inverse standard deviation normalization will be still smaller than or even significantly smaller than 0, which is obviously not in line with the reality. significantly less than 0, which is clearly not realistic. This will have a negative impact on the final results, especially for intervals with small map sizes. Moreover, we found that after using standard deviation normalization, the learning models ridge regression, Lasso regression and elastic net do not work to adjust coefficients or filter features to improve the impact of feature multicollinearity on fitting. In the later section, we will show that there is multicollinearity between map size and coverage, and thus feature filtering is necessary.

The reason for not using robust scaling is that while robust scaling maximizes the retention of anomalies, i.e., outliers, in the dataset and does not affect the ability of the learned model to perform its role of filtering out features, it can lead to segmental outliers in the predicted values of the learned model in predicting the best coverage, which greatly weakens the accuracy of the prediction.

Using Max-Min normalization for feature scaling, i.e.

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

x represents the feature vector, x' represents the normalized feature vector. After Min-Max normalization, the dataset as a whole will be flattened to the interval [0,1], with the data distribution unchanged.

Figure 7 shows Time min-max-scaled-heights standard deviation = 0.0970320034046752

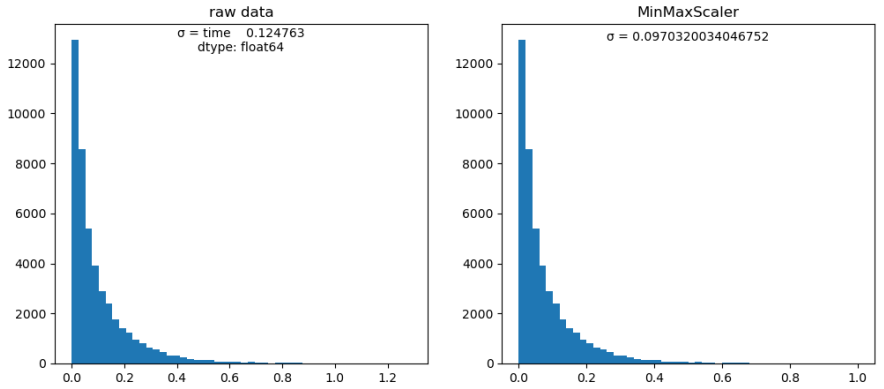

Figure 7. Time Min-Max-Scaled Display

Figure 8 shows Cover min-max-scaled-heights standard deviation = 0.31622776601683794

DOI: 10.54254/2755-2721/79/20241082

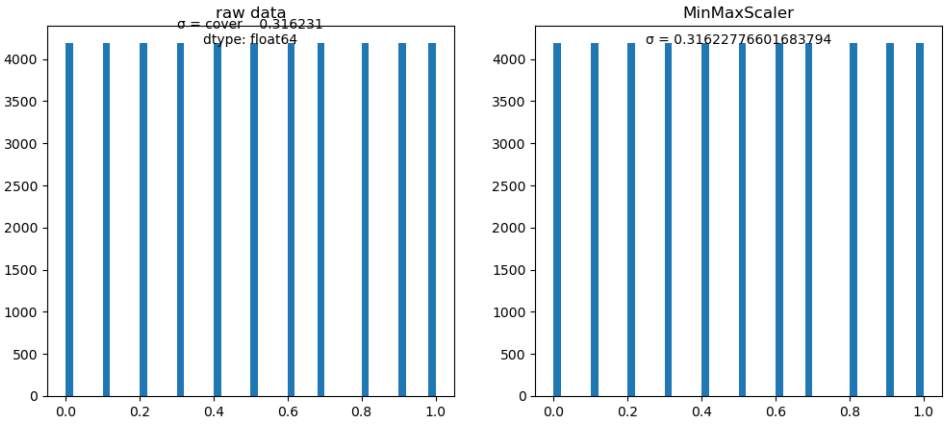

Figure 8. Cover Min-Max-Scaled Display

Figure 9 shows Map min-max-scaled-heights standard deviation = 0.21898601439955426

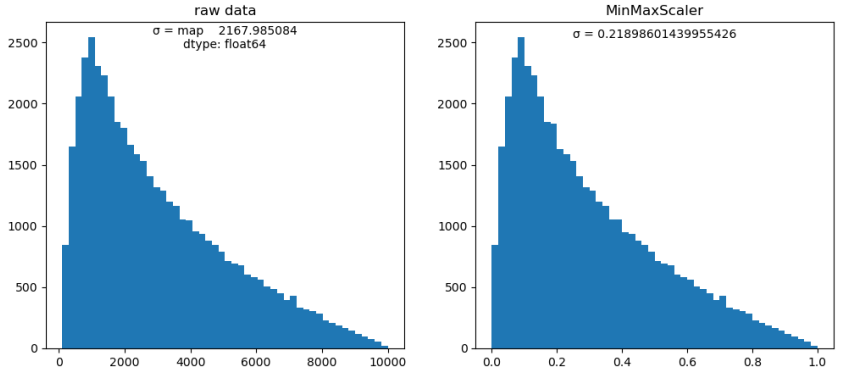

Figure 9. Map Min-Max-Scaled Display

B. Establishing a Regression Model

We initially thought of using linear regression to establish the model. The loss function formula for the least squares linear regression model is as follows:

$$Cost(\vec{w}) = \frac{1}{2}\sum_{i=1}^{N}\left(h_{\vec{w}}(x^i) - y^i\right)^2 \quad [2]$$

To minimize the loss function, the value of $\boldsymbol{w}$ can be obtained as:

$$\boldsymbol{w} = (X^T X)^{-1} X^T y \quad [3]$$

X represents the feature matrix.

From the above equation [3], it can be observed that for w to have a solution, $\mathrm{X^TX}$ must have an inverse matrix. The necessary and sufficient condition for the existence of an inverse matrix is that the determinant is $\left|\mathrm{X^TX}\right| \neq 0$. The necessary and sufficient condition for the determinant $\left|\mathrm{X^TX}\right| \neq 0$ is that matrix $\mathrm{X^TX}$ must be a full-rank matrix. A necessary and sufficient condition for matrix $\mathrm{X^TX}$ to be a full-rank matrix is that there should be no multicollinearity among the features.

Multicollinearity encompasses both exact linear relationships and high degrees of correlation. Under exact linear relationships, matrix $\mathrm{X^TX}$ is not a full-rank matrix, and its determinant is $\left|\mathrm{X^TX}\right| = 0$. When

DOI: 10.54254/2755-2721/79/20241082

solving for $(X^TX)^{-1} = \frac{1}{|X^TX|}(X^TX)^*$, it can be observed that the coefficient $\frac{1}{|X^TX|}$ becomes zero, making it unsolvable. In the case of high degrees of correlation, although matrix X^TX is a full-rank matrix, its determinant $|X^TX| \approx 0$ is near zero. When solving for $(X^TX)^{-1} = \frac{1}{|X^TX|}(X^TX)^*$, it is noticeable that the coefficient $\frac{1}{|X^TX|} \to \infty$ leads to $(X^TX)^{-1}$ being extremely large, which consequently affects the solution for w and ultimately impacts the results of the model.

We aim to identify the optimal relationship between map size and coverage that minimizes time. Hence, we believe that coverage can be calculated based on map size, which leads us to hypothesize that multicollinearity may exist between map size and coverage. To investigate this, we initially employ both a linear regression and a polynomial regression to calculate the variance and mean deviation.

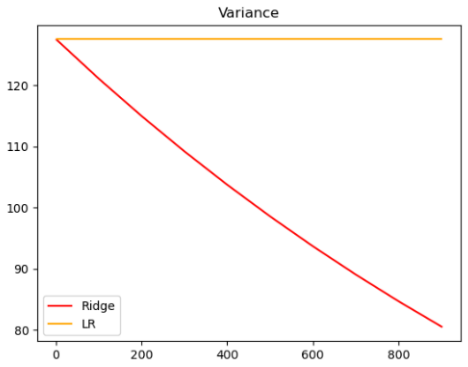

Figure 10. Real Data Variance

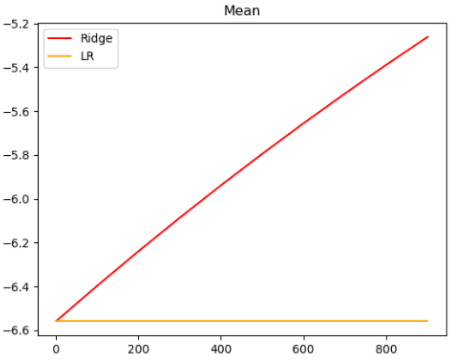

Figure 11. Real Data Mean

By observing Figure 10 and Figure 11, it has been noted that as the regularization coefficient in ridge regression increases, both the variance and bias gradually decrease. This suggests that multicollinearity exists between map size and coverage. As a result, we need to address the issue of multicollinearity.

To tackle the multicollinearity problem, we can transform the solution process into a constrained optimization problem and then solve it using the least squares method. Specifically, when solving for ***w***, we shift from minimizing the loss function of multiple linear regression to minimizing the sum of the loss function and a regularization term. In the case of ridge regression, the regularization term is expressed as the L2 norm of the coefficients multiplied by the regularization coefficient, denoted as $\lambda \sum_{j=1}^{N} w_j^2$. For Lasso regression, the regularization term is expressed as the L1 norm of the coefficients multiplied by the regularization coefficient, denoted as $\lambda \sum_{j=1}^{N} w_j^2$. Different regularization terms will impact the final solution for ***w*** differently. Therefore, we initially consider using either ridge regression or Lasso regression to establish the model, and subsequently evaluate their results to choose the better learning model.

The loss function for the ridge regression model (L2 norm) is as follows:

$$Cost(\vec{w}) = \frac{1}{2}\sum_{i=1}^{N}\left(h_{\vec{w}}(x^i) - y^i\right)^2 + \lambda\sum_{j=1}^{N} w_j^2 \quad [4]$$

λ represents the regularization coefficient, and $h_{\vec{w}}$ represents the undetermined feature coefficient matrix.

To minimize the loss function, the value of ***w*** can be obtained as:

$$\boldsymbol{w} = (X^TX + \lambda I)^{-1}X^Ty \quad [5]$$

Ridge regression can effectively mitigate the impact of exact linear relationships. For matrices with high degrees of correlation, adjusting to a larger λ can reduce the determinant of the $X^TX + \lambda I$ matrix, thereby controlling the deviation of $\boldsymbol{w}$. The larger the value of λ, the less the model is affected by multicollinearity.

DOI: 10.54254/2755-2721/79/20241082

However, λ effectively weakens the contribution of the original feature matrix when estimating $\boldsymbol{w}$. An excessively large λ could lead to significant deviations in $\boldsymbol{w}$, rendering the model incapable of accurately capturing the true nature of the data. Therefore, to identify the optimal value of λ, we employ cross-validation to select the best λ.

For cross-validation, we employ Leave-One-Out Cross-Validation and use R^2 as the evaluation metric for the model. The range for the regularization coefficient λ is set between 0.001 and 100. The optimal regularization coefficients obtained is: for octave regression: $\lambda = 0.001$,

The L2 norm tends to have lower sensitivity to the coefficients, so ridge regression may suffer from insufficient penalization. Therefore, we also employed Lasso regression to build the model.

The loss function for Lasso regression is as follows (L1 norm):

$$Cost(\vec{w}) = \frac{1}{2}\sum_{i=1}^{N}\left(h_{\vec{w}}(x^i) - y^i\right)^2 + \lambda\sum_{j=1}^{N}|w_j| \quad [6]$$

To minimize the loss function, the value of $\boldsymbol{w}$ can be obtained as:

$$\boldsymbol{w} = (X^T X + \lambda I)^{-1} X^T y \quad [7]$$

For Lasso regression, we also employed cross-validation to select the best value for λ. We used Leave-One-Out Cross-Validation and set R^2 as the evaluation metric for the model. The regularization coefficient λ was set to range between 0.001 and 100. Ultimately, the optimal regularization coefficient was obtained as follows: for octave regression: $\lambda = 0.001$ It is worth noting that the L1 norm used in Lasso regression imposes a more severe penalty on the coefficients compared to ridge regression. Generally speaking, Lasso regression is more often used for feature selection. In the case of map size and coverage, although the coefficients for the higher-order terms do not significantly approach zero, the penalty for the higher-order terms may be too severe, leading to the lowest regularization coefficient value of 0.001. Therefore, we employed Elastic Net regression to combine the advantages of both methods.

The formula for Elastic Net is as follows:

$$Cost(\vec{w}) = \frac{1}{2}\sum_{i=1}^{N}\left(h_{\vec{w}}(x^i) - y^i\right)^2 + \lambda\left(\rho\sum_{j=1}^{N}|w_j| + (1-\rho)\sum_{j=1}^{N}w_j^2\right) \quad [8]$$

$$\boldsymbol{w} = (X^T X + \lambda I)^{-1} X^T y \quad [9]$$

For Elastic Net, we also utilized cross-validation to select the optimal value for λ. We employed Leave-One-Out Cross-Validation and used R^2 as the model evaluation metric. The range for the regularization coefficient λ was set between 0.001 and 100. The optimal regularization coefficient was found to be: for octave regression, $\lambda = 0.001$, $l1\_ratio = 0.1$;

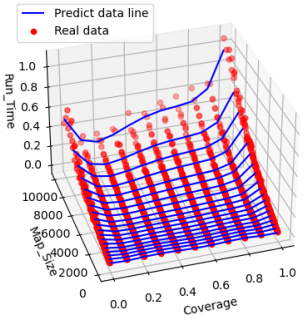

Figure 12. Elastic Net Display

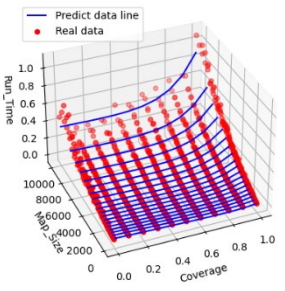

Figure 13. Lasso Regression Display

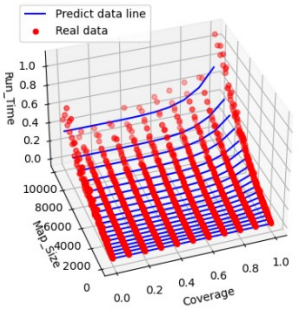

Figure 14. Ridge Regression Display

DOI: 10.54254/2755-2721/79/20241082

Comparing the learning results of the three regression models (Figure 12-14), we finally decided to choose the learning results of the ridge regression model. Although compared to ridge regression and Lasso regression, the learning results of elastic net are more stable when part of the features are screened, the self-evaluation R2 score is higher, and the model image is smoother, but in fact, after our actual testing, such as After calculating the accuracy of the final learning results, we found that the learning effect of elastic net is not as good as ridge regression. Since the maze generation algorithm is essentially a random tree generation algorithm, when the interval is small, the generation time fluctuates greatly and there is strong randomness. In order to reduce this randomness, we repeated the operation 100 times, and took the average every 10 times to reduce the randomness. This may cause the feature screening of Lasso regression and elastic net to have a negative impact on the fitted regression.

In order to obtain better learning results as much as possible, we use higher-order models for fitting. To avoid overfitting, we need to plot the fitted model and analyze the predicted results.

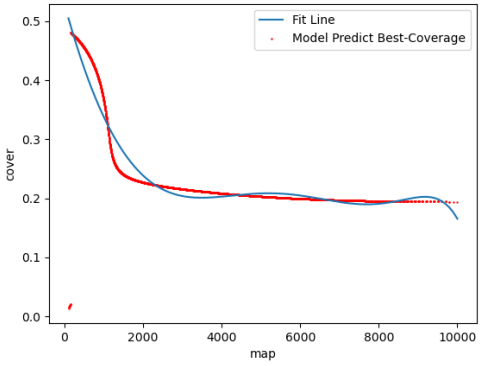

Figure 15. Original Predicted Results Display

Predicted Results are shown in Figure 15. It should be noted that when the order increases, the cross-sectional image of the model tends to be stable in the interval with higher map size, while in the area with lower map size, such as the interval of 100-2500 nodes, the cross-sectional image of the model changes. The fluctuations are huge. When the learning order is too high, some prediction results in the low range of the map size will be abnormal. Since the fitting order of the learning model for fitting the prediction model is too high, we cannot directly solve the prediction model. Therefore, we plan to directly calculate the optimal coverage under each map size and fit the data again to obtain new forecasting model. This will mean that the accuracy of the results of the original prediction model will greatly affect the accuracy of the new prediction model. However, after our calculations, most of the inaccurate predictions in the results predicted by the prediction model are concentrated in the intervals with lower map size. This shows that it is necessary for us to separate the intervals with lower map size separately. The prediction results of the prediction model obtained by combining more data with denser coverage intervals are merged with the original prediction results in the interval with higher map size to improve the accuracy of the new prediction model. In this regard, we use the map size = 1296 where the prediction result coverage of the prediction model mutates as the dividing point, and take 100-1296 as the new map size interval for separate fitting.

DOI: 10.54254/2755-2721/79/20241082

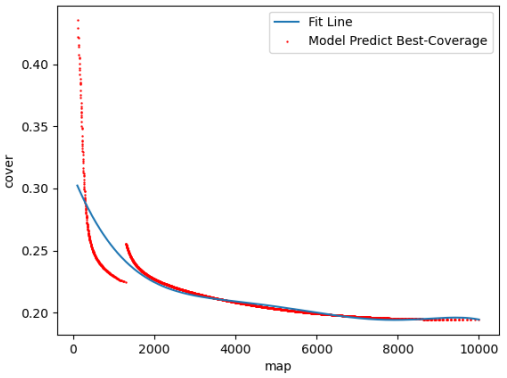

Figure 16. Predicted Results after First Time Separation Display

Predicted Results are shown in Figure 16. After merging, we found that the merged data showed significant breaks, which substantially affected the fit of the new predictive model. However, even so, when comparing the unseparated intervals and fitting them to the merged, the accuracy still showed a more significant improvement, which proves that our thinking is correct. We further increase the cutoff point to 2025 by fitting the 100-2025 interval in the map size separately, which improves the prediction accuracy of the new prediction model at lower map sizes while minimizing the degree of breakage that would occur if the data were merged.

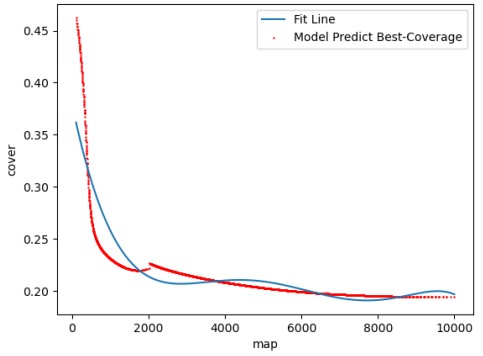

Figure 17. Predicted Results after Second Time Separation Display

Predicted Results are shown in Figure 17. After comparison, the new predictive model fitted by the new merged data has higher accuracy, but the enhancement is more limited.

ii. Final relationship

$$coverage = -10.522289687741 * map^6 + 17.0755712129219 * map^5 + 8.7305043885282 * map^4 - 31.9438842640165 * map^3 + 22.0572275319729 * map^2 - 6.01193589227465 * map + 0.625123916505794$$

C. Model Evaluation

We built an algorithm to test the accuracy of the predictive model, i.e., to ensure that the coverage predicted for each map size should be the best coverage in the interval 100-10000 of the map size. if there exists a coverage that makes the predicted coverage run longer, it is recorded that the predicted coverage is not the optimal coverage, i.e., the predictive model is inaccurate. We will count and finally calculate the inaccuracy rate of the prediction model.

DOI: 10.54254/2755-2721/79/20241082

Algorithm 4 Calculation of Inaccuracy Rate of the Prediction Model

```
T ← 0
CT ← []
InaccurateCount ← 0
for every MapSize do
    BestCoverage ← BestCoverageCalculation(MapSize)
    c1, c2, c3 ← RandomlyChooseThreeCoverages()          ▷ Not 0, 1, and BestCoverage
    for I ← 1 to AverageTimes step 1 do
        BestTime ← HybridAlgorithm(BestCoverage)
        T ← T + BestTime
        for CompareCoverage ← 0, c1, c2, c3, 1 do
            CompareTime ← HybridAlgorithm(CompareCoverage)
            CT[CompareCoverage] ← CT[CompareCoverage] + CompareTime
        end for
    end for
    for time ← CT[every coverage] do
        if T/AverageTimes > time/AverageTimes then
            InaccurateCount ← InaccurateCount + 1
        end if
    end for
end for
InaccurateRate ← InaccurateCount/TotalMapSize
```

We also carried out the actual code testing and obtained Optimization through equation [12].

$$\text{Optimization} = \frac{before - after}{before} \quad [12]$$

In equation [11], we selected the value of 'before' as the average time taken to generate using only the AB or Wilson algorithm, and the value of 'after' as the time taken using the optimal mixed algorithm obtained from the previous step. The calculated result shows that the optimization efficiency of this paper is 74.834% for AB algorithm and 52.326% for Wilson algorithm.

4. Evaluation of the Result

4.1. Experimental Environment

Processor: AMD Ryzen 7 4800H with Radeon Graphics, 2.90 GHz
Installed RAM: 16.0 GB (15.4 GB available)
System Type: 64-bit operating system, x64-based processor
Version: Windows 11 Home Chinese Edition
Build: 22H2

4.2. Evaluation Results

After our calculation and evaluation, we finally chose ridge regression as our learning model and learned the relationship between the final optimal coverage and map-size. In particular, it should be noted that in order to obtain the final relationship, we actually used two ridge regression models. For the first time, ridge regression was used to fit the relationship between running time and coverage and map-size. The reason is that lasso's punishment for the coverage and map-size coefficients is too serious, which affects the accuracy rate. Although in theory, elastic net combines the advantages of ridge and lasso, due to the strong randomness of the learning data, in order to weaken this strong randomness, we used the average running time of multiple runs as the learning data, which may instead lead to elastic net's excessive punishment of coverage and map-size coefficients. In the subsequent calculation of the relationship between the best coverage and map-size, we found that the high-order relationship obtained by the high-order learning model that had to be used in order to improve the accuracy rate could not be directly solved to obtain the relationship, so we settled for the next best thing by calculating the best coverage under each map-size separately to combine into a new learning. The data is learned and fitted again. At this time, the reason we use ridge regression is that since the original high-order relationship is obtained by learning and fitting from strong randomness learning data, and in the lower map-size interval, this situation is more serious, so the relationship between the best coverage and map-size obtained by the second learning fit inherits the relationship between map-size and map-size. The nature of lower size and lower accuracy. To solve this problem, we tried to separate the low map-size interval separately and improve the accuracy rate by increasing the amount of data. To bridge the data gap between the

DOI: 10.54254/2755-2721/79/20241082

combined data, but also to improve the accuracy of the relationship in the low map-size interval, we finally chose the ridge regression model again for learning fitting.

4.3. Result

The relationship between the best coverage and map-size that we finally fitted has an inaccuracy rate of about 19.6% when map-size is not optimized separately. We use map-size = 1296 as the dividing line to divide and measure it separately. The inaccuracy rate was measured. As a result, when map-size is in the range of 100-1296, the inaccuracy rate is about 53.0%. When map-size is in the range of 1298 to 10000, the inaccuracy rate is only about 13.7%. Therefore, it is necessary to map -size low range optimization. After optimizing with 1296 as the dividing line, the inaccuracy rate is about 16.2%. However, the combined learning data has large data gaps and is not continuous. This may affect the result, so we use 2025 as the dividing line. Optimized for smoother learning data. After re-optimization, the non-accuracy rate is about 15.8%, and the improvement effect is not significant.

5. Conclusion

This paper employs the Elastic Net regression model to predict the time required for the hybrid AB and Wilson algorithms under different AB coverage rates and map sizes. By establishing a model and derivative equation, we fit the best coverage. This method is highly likely to reduce the time required to generate random mazes in actual application. Overall, the mazes generated through this approach have increased randomness, making the gaming experience more enjoyable. Moreover, the use of the hybrid algorithm can significantly reduce the maze generation time, thereby enhancing the overall experience.

The model we generated is influenced by two factors. Firstly, the structure of the data possesses randomness, so further tests with more datasets should be performed to minimize the randomness of the results. Secondly, the range of computer floating-point numbers falls between 3.4E-38 and 3.4E+38, leading to potential discrepancies between the actual data and anticipated results when the dataset is small, as the generation speed might be too rapid to measure. Based on these considerations, we outline some prospects for future research:

1). Compared to regression models, there may exist more suitable and efficient methods for model construction, and there might still be room for optimization since our regression model only included terms up to the third order.

2). In the selection and training of data, the precision for smaller map sizes cannot be guaranteed, and there is a certain margin of error. Future experiments should seek to obtain more accurate results for training.

3). In practical application, game designers' mazes may not always be rectangular or presented in a fully graphical form. Future research should consider more extreme environments to make the model more universally applicable.

4). In the separation fit to the data, we performed only one separation fit. Although it was considered that the effect of separation fitting may lead to less optimization as the number of separations increases, due to time as well as equipment problems, we did not actually do it and conclude that.

Acknowledgement

KaiCheng Yang and Sutong Lin contributed equally to this work and should be considered co-first authors. Wentai Li and Yu Dai contributed equally to this work and should be considered co-second authors.

References

[1] Fredes, L., & Marckert, J. F. (2023). A combinatorial proof of Aldous–Broder theorem for general Markov chains. *Random Structures & Algorithms*, *62*(2), 430-449.

[2] Nunes, I., Iacobelli, G., & Figueiredo, D. R. (2022). A transient equivalence between Aldous-Broder and Wilson's algorithms and a two-stage framework for generating uniform spanning trees. *arXiv preprint arXiv:2206.12378*.

DOI: 10.54254/2755-2721/79/20241082

[3] C. -m. Bae, E. K. Kim, J. Lee, K. -J. Kim and J. -C. Na, "Generation of an arbitrary shaped large maze by assembling mazes," 2015 IEEE Conference on Computational Intelligence and Games (CIG), Tainan, Taiwan, 2015, pp. 538-539, doi: 10.1109/CIG.2015.7317901.

[4] Dormann, C. F., Elith, J., Bacher, S., Buchmann, C., Carl, G., Carré, G., ... & Lautenbach, S. (2013). Collinearity: a review of methods to deal with it and a simulation study evaluating their performance. *Ecography*, *36*(1), 27-46.

[5] J. Chelladurai, "Exploring Complex Toroidal Mazes with L-systems," 2023 4th International Conference on Computing and Communication Systems (I3CS), Shillong, India, 2023, pp. 1-6, doi: 10.1109/I3CS58314.2023.10127262.

[6] Leng Jianfei, Gao Xu & Zhu Jiaping. (2016). The Application of Multiple Linear Regression Statistical Forecast Models. Statistics and Decision (07), 82-85. doi:10.13546/j.cnki.tjyjc.2016.07.021.

[7] P. H. Kim and R. Crawfis, "Intelligent Maze Generation Based on Topological Constraints," 2018 7th International Congress on Advanced Applied Informatics (IIAI-AAI), Yonago, Japan, 2018, pp. 867-872, doi: 10.1109/IIAI-AAI.2018.00176.

[8] Uemukai, R. (2011). Small sample properties of a ridge regression estimator when there exist omitted variables. *Statistical Papers*, *52*, 953-969.

[9] Zhang, Y., Ma, F., & Wang, Y. (2019). Forecasting crude oil prices with a large set of predictors: Can LASSO select powerful predictors? *Journal of Empirical Finance*, *54*, 97-117.

[10] Hong, C. S., & Kim, J. M. (2010). The SSR Plot: A Graphical Representation for Regression. *Communications in Statistics—Simulation and Computation®*, *39*(4), 726-735.

[11] Walter D. Pullen (2023). *Maze Creation Algorithms, Maze Algorithms.* Available at: https://www.astrolog.org/labyrnth.htm (Accessed: 10 October 2023).