

Problem Chosen	2026 MCM/ICM Summary Sheet	Team Control Number
C		0000

Summary

Dancing with the Stars (DWTS) is a long-running show that has received significant attention. However, the methods of combining judges' scores and fan votes have sparked controversy, particularly since the specific fan vote tallies are kept secret.

To solve these problems, several models are established: Model 1 uses Bayesian inference and Glicko-2 ratings to estimate unobservable fan support; Model 2 uses decision consistency bias analysis framework to evaluate the structural fairness of different voting rules; Model 3 applies SHAP-based feature importance analysis to uncover non-linear dynamics between contestant traits and outcomes; finally, Model 4 integrates a Multi-objective Optimization function to provide a validated reform for the DWTS scoring system.

For Task 1,

For 2,

For 3,

For 4,

Finally, sensitivity analysis and a memo for the

Keywords: keyword1; keyword2

Contents

1 Introduction

1.1 Problem Background

With over 34 seasons and a global footprint spanning dozens of countries, Dancing with the Stars (DWTS) serves as a quintessential laboratory for dual-channel decision-making systems. The competition's integrity hinges on a delicate equilibrium between professional technicality—quantified by expert judges—and public sentiment, captured through massive fan participation. However, the systemic opacity of raw voting tallies creates a mathematical "black box," where the actual influence of the "popularity premium" remains latent. Historical anomalies, such as the Season 2 "ranking inversion" where high-performing professionals were eclipsed by contestants with inferior technical scores, underscore a critical vulnerability: when the weighted influence of public favorability deconstructs the meritocracy of professional standards, the competition's evaluative authority diminishes. This tension necessitates a rigorous quantitative framework to reconstruct hidden preferences and audit the robustness of the synthesis logic.

1.2 Restatement of the Problem

The challenge resides in reverse-engineering a multi-attribute decision-making (MADM) process under conditions of incomplete information. To systematically address the competition's internal logic, we decompose the task into the following mathematical objectives:

- **Latent Variable Estimation of Public Sentiment:** We must treat the confidential fan tallies as latent variables within a stochastic estimation model. The objective is to reconstruct the hidden weekly voting distribution ($V_{i,t}$) and quantify the inferential certainty, effectively transforming elimination outcomes into observable probability densities.
- **Comparative Sensitivity Analysis of Synthesis Mechanics:** The model evaluates the structural divergence between the ordinal rank-summation method (Seasons 1-2) and the cardinal percentage-weighting scheme (Season 3 onwards). By identifying the "bifurcation points"—where rule changes alone alter elimination results—we can mathematically diagnose the triggers of historical controversial outcomes.
- **Multivariate Decoupling of Performance and Profile:** We require a multi-factor regression framework to decouple the influence of exogenous "celebrity attributes" (e.g., industry, age, and initial public profile) from endogenous professional performance. This analysis quantifies the elasticity of final scores relative to a contestant's pre-existing social capital.
- **Axiomatic Design of an Optimized Evaluation Kernel:** Ultimately, the objective is to synthesize these insights into an optimized scoring framework. This involves constructing a new decision algorithm that maximizes the alignment between technical excellence and public engagement while minimizing systemic bias and the "popularity premium."

1.3 Our Work

To address these complexities, we first develop a model, which utilizes a Bayesian framework to map judge score distributions and elimination outcomes back to estimate the number of fan votes

that each contestant receives per week. Based on these estimations, we implement the Structural Divergence Audit (SDA), to compare two methods (rank versus percentage) used in the show to analyze controversial incidents, and then figure out how these different synthesis rules amplify the "popularity premium" across 34 seasons of data. We constructed the Attribute Influence Network (AIN), a high-dimensional regression model to quantifies the impact of pro dancers and celebrities' profiles on the judge scores and fan vote result. Finally, we propose a re-engineered synthesis protocol to ensure that professional benchmarks remain the primary determinant of progression without dampening audience enthusiasm.

2 Assumptions and Notations

Assumption 1: The authentic dance skills of the contestants are relatively stable in the short term, and the change is smooth.

Assumption 2: The audience recognition of the contestants' abilities is memorylessness. As a result, the performance in the current week is totally based on the fine-tuning of the last week, and is irrelevant to the results from two weeks ago.

Assumption 3: The impact of age on audience voting rates is not linear and follow an inverted U-shaped curve. For example, an elderly artist may be rich in emotion but lacking in skill, while younger contestants may excel at high-difficulty movements.

Assumption 4: The withdrawal from the competition is an exogenous event. We do not consider it a reason for elimination.

Assumption 5: During counterfactual reasoning, we assume that its "phantom" performance (judges' scores and fan votes) equals the average level of its actual appearances in the program during that season.

Assumption 6: In the Monte Carlo simulation, every contestant's fan vote is independent and identically distributed and everyone has the same noise distribution.

Assumption 7: The boost provided by professional partners to contestants is the average of their performance across all appearances, and that this bonus is transferable across seasons and between different contestants.

Assumption 8: The average judges' score of eliminated contestants each week represents the "minimum technical standard" for that week. Contestants who survive despite scoring below this threshold are considered to have received non-technical protection from fans.

We further assume contestants' weekly "undeserved scores" (the residuals where technical scores fall below the threshold) accumulate across weeks within a season. This explains why seeing someone consistently score low yet remain in the competition creates a perception of controversy.

3 Data Preprocessing and Feature Engineering

3.1 Data Cleaning and Normalization

1. By parsing the text information in the "results" field using regular expressions, we precisely defined the competition lifecycle for each contestant. For the special case of "Withdrew", the script

does not rely on static tags but instead iterates through the score column to dynamically find the contestant's last active week with a non-zero score. These contestants will be included in the judges' total score for their last episode in the subsequent estimated vote calculation, but will not affect their vote calculation for the next episode.

2. After the structural transformation, we calculated two relative indicators to eliminate the impact of inconsistent referee scoring standards across different seasons.

judges' score ratio:

$$P_{\text{judge}} = \frac{\text{Score}_{i,t}}{\sum_{j \in \text{Active}} \text{Score}_{j,t}}$$

By grouping and normalizing, the percentage of a contestant's score in the total score of all survivors for that week is calculated. This serves as the direct input for the "referee weights" in the subsequent Bayesian model.

weekly ranking:

Calculate the contestants' relative rankings to the judges for the week (the lower the value, the higher the ranking). This provides baseline data for subsequent verification of the discrepancies between the "Rank system" and the "Percentage system" of different rules.

To ensure consistency across the dataset, we standardized the birthplaces of celebrities. For U.S. based individuals, the birthplaces were aggregated to the state level, while others maintained their nationality.

3.2 Static Feature Engineering and Dimensionality Reduction

We transform the unstructured raw data into a high-density feature matrix suitable for computation.

Professional Dance Partner Effectiveness Index: To quantify the implicit boost a dance partner provides to a dancer, we calculated the average judge score for each professional dance partner across all historical seasons. This constructs a powerful proxy variable capable of capturing the uplift effect.

Industry category dimensionality reduction: To address the issue of high sparsity in the original "celebrity_industry" field. We use keyword mapping to categorize dozens of professions into five macro groups: Performing Arts, Sports, TV/Media, Model/Fashion, and Other. We then use One-Hot encoding to transform these into numerical features.

nonlinear age effects: To capture the balance between physiological function and experience, the Age_Squared term is introduced. This allows the model to fit an inverted U-shaped or U-shaped relationship between age and performance, rather than a simple linear assumption.

4 Task 1: Reconstructing Hidden Popularity

4.1 Monte Carlo Simulation with Heuristic Gradient Updates

For each week, the following procedure is repeated 50 times.

Step 1: Uniform Initialization. Initialize the vector of public voting share X uniformly, so that $\sum X_i = 1$ and $X_i = \frac{1}{N}$, where N denotes the number of active contestants.

Step 2: Monte Carlo Simulation. Based on the current parameter vector X , we generate $M = 1000$ sets of voting samples. For a given step size n , let f_i denote the simulated elimination probability of contestant i , derived from the frequency of their elimination. In each simulation trial, the contestant with the lowest total score is eliminated. In the event of a tie, the eliminated contestant is chosen at random.

Step 3: Estimation of Elimination Probability. Count the frequency each active contestant is eliminated in the simulations(f_i)

Step 4: Target Vector Construction. Construct the target vector according to the actual elimination outcome:

$$t_i = \begin{cases} 1, & \text{if contestant } i \text{ is eliminated in reality,} \\ 0, & \text{otherwise.} \end{cases}$$

Step 5: Error Computation. Define the error term as

$$e_i = t_i - f_i$$

Step 6: Heuristic Gradient Update. Update the support level of each contestant by

$$X_i \leftarrow X_i(1 - \eta e_i)$$

where η is the step size, for example, $\eta = 0.05$.

Step 8: Normalization. Renormalize the vector X to satisfy $\sum X_i = 1$

Step 9: Memory Effect Across Weeks. In the next week,a memory coefficient $a = 0.7$ is introduced to evaluate audience memory.

Let

$$X_t = aX_{t-1} + (1 - a)X_{\text{base}}$$

then renormalize to make $\sum X_i = 1$ and repeat the procedure above.

4.2 Bayesian Inference and Stratified Monte Carlo Simulation

1. Model Description

The difficulty of the problem is that the fan vote data is not visible. By combining Bayesian Inference and Monte Carlo Simulation, we construct a system that can reverse the potential popularity from the elimination results. At the same time, we introduce the ELO module to capture the changes in the players' abilities as the competition progresses.

The traditional ELO system assumes that the performance variance is fixed. But the data in the question is extremely sparse (one game per week) and high uncertainty, which is difficult to handle by the traditional methods. For this reason, we build a Glicko-2 style dynamic state space model.

2. Model Building

We define two parameters of the contestants. The state of each player i in week t is determined by two variables: θ_ELO (player skill) and RD (rating deviation).

A contestant versus group model is constructed instead of the traditional 1v1 winning rate formula.

We calculate the contestant's total score (judges' score + fan votes) relative to the group's average score for the week (Z-Score)

$$Z_i = \frac{Score_{total,i} - \mu_{field}}{\sigma_{field}}$$

And then mapping the Z-score to theoretical survival rate using the Logistic function

$$E_{survival,i} = \frac{1}{1 + \exp\left(-2.5 \cdot \frac{S_{total,i} - \mu_{field}}{\sigma_{field}}\right)}$$

We apply hierarchical random noise and employ the Monte Carlo method to simulate the voting process for 3,000 times. We record the elimination frequency f_i for each contestant.

We compare the eliminators model predicted with the actual outcomes to calculate the error $e_i = t_i - f_i$, where t_i denotes the target vector. Subsequently, we update the Elo ratings and Rating Deviations for each contestant. For eliminated contestants, we decrease their Elo ratings and increase their RD. For surviving contestants, we increase their Elo ratings and decrease their RD. The update formula is according to the Glicko-2 system.

In particular, Uncertainty-driven: The larger the RD of a player, the greater the weight. The logic is: "I don't know you, so every performance can drastically change my opinion."

$$g(RD_{i,t}) = \frac{1}{\sqrt{1 + \frac{3 \cdot q^2 \cdot RD_{i,t}^2}{\pi^2}}}$$

Surprise-driven: $|S_{Actual} - E_{Survival}|$ represent the degree of surprise of the result. The logic is: "If a high-scoring contestant is unexpectedly eliminated (a major upset), the system needs to significantly adjust the score; if a low-scoring contestant is eliminated (as expected), the score only needs to be slightly adjusted."

$$K_{adaptive} = K_{base} \cdot \frac{RD_{i,t}}{350} \cdot (1 + 0.5 \cdot |S_{actual} - E_{survival,i}|)$$

The final score update incorporates the learning rate, uncertainty scaling, and the residuals between the actual and expected results:

$$ELO : \theta_{i,t+1} = \theta_{i,t} + K_{adaptive} \cdot g(RD_{i,t}) \cdot (S_{actual} - E_{survival,i})$$

$$RD : RD_{i,t+1} = \max(30.0, RD_{i,t} \cdot \gamma_{decay} - 0.1 \cdot |\Delta\theta_{i,t}|)$$

In the formula, γ_{decay} is the time decay factor, and $\Delta\theta_{i,t}$ is the change in rating.

And finally we map it to fan support rate

$$\hat{P}_{fan,i} = \frac{\exp(\theta_{i,t}/\tau)}{\sum_{j \in Survivors} \exp(\theta_{j,t}/\tau)}$$

Where τ is the Softmax temperature parameter that dynamically adjusts over time.

4.3 Estimation and Consistency Verification

5 Task 2: Comparative Analysis of Voting Rules

5.1 1

We constructed a decision consistency bias analysis framework and an evaluation system based on ordinal consistency. By comparing the ranking results under different simulation rules with the original preferences of judges and fans, we quantified the degree of bias in different voting mechanisms (such as Rank and Percentage systems) towards skill level (judges) or popularity (fans).

1. Decision Consistency Evaluation Indicators

- Ranking Reconstruction: Since the competition results are presented in the form of elimination weeks, we define the simulated ranking R_{sim} as the difference between the total number of participants in the season and the number of elimination weeks. This transformation converts the survival length in the time dimension into competitive performance in the ordinal dimension, ensuring the consistency between the simulated results and the original preferences in mathematical expression.

- Consistency Measure: We use the Spearman rank correlation coefficient (ρ) to measure the monotonic correlation between the simulated output and the preferences of different evaluators.

Judge Consistency (ρ_{Judge}): Quantifies the degree of agreement between the simulated ranking and the expert technical rating ranking.

Fan Consistency: Quantifies the degree of agreement between simulated rankings and popularity preference rankings based on Bayesian inference.

2. Bias Analysis Framework Based on the above consistency indicators, we propose the core concept of decision bias bias to identify potential skews in rule design.

- Bias is defined as the difference between judge consistency and fan consistency: Bias = Judge - Fan. This indicator intuitively reflects the trade-off between "technology-oriented" and "popularity-oriented" rules.

- Propensity Identification: If $B > 0$, it indicates that the voting mechanism is more sensitive to technical skill during the decision-making process, tending to protect players with superior professional performance; if $B < 0$, it indicates that the mechanism is highly susceptible to fan preference, resulting in a significant "popularity premium."

3. Multi-Scenario Rule Evaluation To comprehensively evaluate the evolution of the DWTS voting system, we conducted sensitivity analysis on four rule combinations:

Rank vs. Percentage: This study compares the rank-based combination logic introduced early in the season with the percentage-based weighting logic introduced later. The focus is on whether the percentage system effectively corrects the "rank inversion" problem caused by extreme popularity interference.

- Judges' Save: By introducing or removing a "judges' save" mechanism in the simulator, we quantified the impact of this human intervention on system robustness. This mechanism is considered a "quality brake," designed to prevent players with extremely low skill ratings but extremely high popularity from excessively distorting match results.

Results:

Figure 1 presents a scatter plot comparison between the estimated fan shares derived from the Percentage Rule (x-axis) and the Rank Rule (y-axis). The red dashed line represents the locus of perfect agreement ($y=x$).

While the two methods exhibit a positive correlation ($R^2 > 0.9$), a distinct sigmoidal deviation pattern is observable, revealing the inherent "equalizing bias" of the Rank Rule.

Subsidization of the Tail (Low-end Bias): In the lower quadrant (fan share <0.15), the data points predominantly lie above the reference line. This indicates that the Rank Rule systematically overestimates the support for less popular contestants. By converting continuous vote counts into discrete ordinal ranks, the algorithm artificially inflates the value of marginal survival, effectively providing a "safety net" for weaker contestants.

Suppression of the Head (High-end Compression): Conversely, in the upper quadrant, the Rank Rule estimates consistently fall below the reference line. This phenomenon, known as variance compression, demonstrates that the Rank mechanism fails to capture the magnitude of "superstar" popularity. A contestant with 50% of the total votes receives the same rank score (Rank 1) as one with 25% in a tighter race, thereby discarding significant voter preference information.

Conclusion: Mathematically, the Rank Rule acts as a low-pass filter, smoothing out extreme variations in public opinion. While this may increase competition suspense, it statistically distorts the true "Voice of the People," making it harder for dominant talent to secure a mathematically justified lead.

5.2 2

For the counterfactual simulation, we employed a 'Shadow History' approach. In each week t , we included all contestants who historically competed in that week. We then applied the alternative voting rule (e.g., Judges' Save). If our target contestant (e.g., Bobby Bones) was identified for elimination under the new rule, the simulation terminated, marking that week as their 'Counterfactual Elimination Point'. This method utilizes the ground-truth technical scores from history, avoiding speculative data generation for contestants who did not actually perform.

5.3 3

We introduced a consistency coefficient

$$\rho = 1 - \frac{(6 \cdot d_i^2)}{n(n^2 - 1)}$$

where n is the number of contestants in the season, and d_i = final ranking–fan ranking (or referee ranking). Bias = $\rho_{\text{referees}} - \rho_{\text{fans}}$. If the calculation shows that the average $|\Delta\rho|$ of method A is significantly lower than that of method B, it proves that method A achieves a better balance between skill and popularity by compressing the influence of extreme values. Based on this logic, using the simulator from question 2, we simulate the rankings of the fifty most controversial figures with the highest controversy scores, calculate their bias, and quantify the performance of each method.

6 Task 3: Uncovering the Drivers of Success

We established a feature contribution comparison framework based on ensemble learning, aiming to quantify and compare the core driving factors influencing "professional judge scores (technical scores)" and "fan votes (popularity scores)."

During the data cleaning phase, we completed the construction of a multi-dimensional feature matrix.

We adopted a gradient boosting regression tree as the core engine to perform parallel modeling for two evaluation systems ($M_{\text{Fans}}/M_{\text{judge}}$). The judge model uses the total judge score obtained by each participant as the target variable to fit the mapping relationship between features and professional judging criteria. The fan model uses the Bayesian estimated fan vote rate as the target variable, aiming to fit the correlation between features and public aesthetics.

By calculating the decrease in MSE within the model, we quantified the contribution of each feature to the prediction of the target variable and normalized the feature importance of the two models respectively. This allows us to compare the weight differences of the same feature under the "judge's perspective" and "fan's perspective" on the same scale. Simultaneously, we grouped and aggregated age data (*age_groups*) and used second-order polynomial trend fitting to quantify the

natural decay of technical performance and the dynamic evolution of fan sympathy votes/emotional connections with age.

Results:

Based on the feature importance comparison of the model output, the dance partner is a decisive factor, but judges value it more (The "Pro" Effect). This explains 54.3% of the judges' scoring variation, but only 44.5% of the fan voting variation. Judges' scores are highly dependent on the dance partner's technical merit, while fans, although they also value performance, are less influenced by the dance partner (approximately 10 percentage points lower), leaving more room for "celebrity personal charm."

The Age Paradox: Age (and age squared) ranks 2nd and 3rd in importance in both models (approximately 38%-41% combined). The difference: the influence of age on fans (41.3% combined) is surprisingly slightly higher than its influence on judges (38.0% combined). It is generally believed that judges will punish older dancers for stiffness, but data shows that fans are very sensitive to age. Combined with SHAP analysis, this may mean that fans not only vote for younger idols but also show strong emotional preference for older, inspirational figures, or conversely, they may be extremely averse to certain age groups. The weak industry influence and low overall contribution from industry characteristics suggest that "who you are (actor/athlete)" is less important than "how well you perform (dance partner/age/status)."

7 Task 4: Proposed an Alternative System

7.1 System Redesign: Considering Fairness and Entertainment

Solving this problem requires attention to multiple factors:

Fairness: Trust the judges' professionalism and ensure that their scores carry the majority weight; fan voting, intended for entertainment purposes, should not excessively influence the outcome.

Entertainment: Fan voting can affect the results, retaining popular contestants/generating controversy to increase the show's popularity.

$$S_{\text{total}} = w_j \cdot P_{\text{judge}} + w_f \cdot \frac{\ln(1 + \lambda \cdot P_{\text{fan}})}{\ln(1 + \lambda)} + I_{\text{bonus}} \cdot \beta$$

$$w_j = 0.8, w_f = 0.2, \lambda = 100, \beta = 0.05$$

The reasons are as follows:

Fairness. Judges' scores carry significant weight, ensuring professionalism; fan votes are logarithmically compressed to prevent popularity from excessively influencing the results.

Fun. Fan votes have influence and can help maintain the popularity of contestants.

Other Additional Measures:

Sudden Death: After calculating the total scores each week, the two lowest-scoring contestants are selected for an elimination round, where judges and fans re-vote to calculate the total score.

Improvement Bonus: Each week, the difference between each contestant's judges' score and the previous week's score is calculated; the contestant with the greatest improvement receives a 0.05 bonus.

7.2 Simulation Validation and Improvement Assessment

We implemented the above scoring system in python. We used the players' Elo ratings as the mean of a Gaussian distribution, with the standard deviation related to the RD uncertainty, and simulated 2000 sudden death matches, calculating the win rate for each player.

We also simulated historical data. Among them:

138 players (46.78%) who were originally eliminated ranked higher than the second-to-last player under the new rules, meaning they did not need to participate in sudden death mode.

42 players (14.24%) who were eliminated in reality were revived by winning sudden death matches under the new rules.

8 Sensitivity Analysis

Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio.

9 Strengths and weaknesses

Etiam euismod. Fusce facilisis lacinia dui. Suspendisse potenti. In mi erat, cursus id, nonummy sed, ullamcorper eget, sapien. Praesent pretium, magna in eleifend egestas, pede pede pretium lorem, quis consectetur tortor sapien facilisis magna. Mauris quis magna varius nulla scelerisque imperdiet. Aliquam non quam. Aliquam porttitor quam a lacus. Praesent vel arcu ut tortor cursus volutpat. In vitae pede quis diam bibendum placerat. Fusce elementum convallis neque. Sed dolor orci, scelerisque ac, dapibus nec, ultricies ut, mi. Duis nec dui quis leo sagittis commodo.

9.1 Strengths

9.2 How to cite?

bibliography cite use [?, ?, ?]

AI cite use [AI: ?, ?, ?]

References

- [1] D. E. KNUTH The \TeX book the American Mathematical Society and Addison-Wesley Publishing Company , 1984-1986.
- [2] Lamport, Leslie, \LaTeX : “ A Document Preparation System ”, Addison-Wesley Publishing Company, 1986.
- [3] <https://www.latexstudio.net/>

Appendices

Appendix A First appendix

In addition, your report must include a letter to the Chief Financial Officer (CFO) of the Good-grant Foundation, Mr. Alpha Chiang, that describes the optimal investment strategy, your modeling approach and major results, and a brief discussion of your proposed concept of a return-on-investment (ROI). This letter should be no more than two pages in length.

Dear, Mr. Alpha Chiang

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Sincerely yours,

Your friends

Here are simulation programmes we used in our model as follow.

Input matlab source:

```

function [t,seat,aisle]=OI6Sim(n,target,seated)
pab=rand(1,n);
for i=1:n
    if pab(i)<0.4
        aisleTime(i)=0;
    else
        aisleTime(i)=trirnd(3.2,7.1,38.7);
    end
end

```

Appendix B Second appendix

some more text **Input C++ source:**

```

//=====
// Name      : Sudoku.cpp
// Author    : wzlf11
// Version   : a.0
// Copyright : Your copyright notice
// Description : Sudoku in C++.
//=====

#include <iostream>
#include <cstdlib>
#include <ctime>

using namespace std;

int table[9][9];

int main() {
    for(int i = 0; i < 9; i++){
        table[0][i] = i + 1;
    }

    srand((unsigned int)time(NULL));

    shuffle((int *)&table[0], 9);

    while(!put_line(1))
    {
        shuffle((int *)&table[0], 9);
    }

    for(int x = 0; x < 9; x++){
        for(int y = 0; y < 9; y++){
            cout << table[x][y] << " ";
        }
        cout << endl;
    }
}

```

```
        return 0;  
}
```

Report on Use of AI

1. Bing AI

Query1: write a sort algorithm

Output: Sure, I can help you with that. Here is a simple implementation of the selection sort algorithm in Python:

```
def selection_sort(arr):
    for i in range(len(arr)):
        min_idx = i
        for j in range(i + 1, len(arr)):
            if arr[min_idx] > arr[j]:
                min_idx = j
        arr[i], arr[min_idx] = arr[min_idx], arr[i]
    return arr
```

This algorithm sorts an array by repeatedly finding the minimum element from the unsorted part of the array and putting it at the beginning. The time complexity of this algorithm is $O(n^2)$.

There are many other sorting algorithms available such as bubble sort, insertion sort, merge sort, quick sort, heap sort, and more. Each of these algorithms has its own advantages and disadvantages. You can learn more about them on GeeksforGeeks and Programiz.

I hope this helps!

2. OpenAI Enie (Nov 5, 2023 version, Ernie 4.0)

Query1: <insert the exact wording of any subsequent input into the AI tool>

Output: <insert the complete output from the second query>

3. Github CoPilot (Feb 3, 2024 version)

Query1: <insert the exact wording you input into the AI tool>

Output: <insert the complete output from the AI tool>

4. Google Bard (Feb 2, 2024 version)

Query: <insert the exact wording of your query>

Output: <insert the complete output from the AI tool>

5. Bing AI

Query: write a FFT algorithm

Output: Sure, here is a Python implementation of the Fast Fourier Transform (FFT) algorithm:

```
import numpy as np

def fft(x):
    n = len(x)
    if n == 1:
        return x
    even = fft(x[0::2])
    odd = fft(x[1::2])
    T = [np.exp(-2j * np.pi * k / n) * odd[k] for k in range(n // 2)]
    return [even[k] + T[k] for k in range(n // 2)] + [even[k] - T[k] for k
→ in range(n // 2)]
```

The FFT is a widely used algorithm for computing the **Discrete Fourier Transform (DFT)** of a sequence. It reduces the number of computations needed from $O(n^2)$ to $O(n \log n)$ [12]. The basic idea of the FFT is to apply divide and conquer. We divide the coefficient vector of the polynomial into two vectors, recursively compute the DFT for each of them, and combine the results to compute the DFT of the complete polynomial [3].

I hope this helps!