Ant Colony Optimization (ACO) and Multi-Agent Adversarial Reinforcement Learning (MAARF) in the 1–100 Divisibility Chain Game

Jincan LI
50032637
jli843@connect.hkust-gz.edu.cn

Wenlue CHAI
50035926
wchai181@connect.hkust-gz.edu.cn

Zheng TIAN
50035354
ztian117@connect.hkust-gz.edu.cn

Zixian GUO
50033746
zguo849@connect.hkust-gz.edu.cn

1 Introduction

The 2025 MCM Junior "Game on Integers" problem involves two scenarios:

- **Cooperative Scenario**: Maximizing the total reward via a valid integer chain, 1–100, with consecutive divisibility.
- **Competitive scenario**: A two-player zero-sum game where Alice aims to maximize her score against Bob.

To address these challenges, we modeled the problem as an **Undirected Divisible Graph** and employed:

- **Graph Theory and Ant Colony Optimization (ACO)** for the cooperative scenario;
- **Mathematical Analysis and Reinforcement Learning (RL)** for the competitive scenario.

For the cooperative scenario, the graph converts the "chain building" into a "maximum-sum pathfinding" task. Our implementation ACO achieved a high reward of **3551**; we verified its performance via the **Hierarchical Assessment Method**, demonstrating its excellence.

For the competitive scenario, we modeled the game as a sequential two-player zero-sum game. We first employed the mathematical ways, finding a brilliant opening strategy, given the large latter state space where classical methods are impractical, we propose MAARF where agents Alice and Bob are trained through self-play to discover effective strategies. Our approach focuses on learning strategic patterns, enabling scalable and interpretable policy discovery.

The rest of the report is structured as follows: Section 2 details problem analysis and graph modeling. Section 3 covers the ACO solution for the cooperative scenario (modeling + evaluation). Section 4 discusses the mathematical thinking and reinforcement learning method for the competitive scenario (modeling + evaluation). Section 5 concludes and lists references.

2 Problem Decomposition and Problem-Graph Modeling

This section formalizes the game's core rules, scenario goals, and graph mapping.

2.1 Key Assumptions

We assume that all players are perfectly rational, strictly adhere to game rules, have complete information about the graph topology, and aim to maximize their total rewards in Q (1), and maximize their respective rewards in Q (2).

2.2 Core Game Rules

A valid chain $X = (X_1, X_2, \dots, X_m)$, $k = 1, 2, \dots, m$ must satisfy 3 non-negotiable constrains:

- Every $X_k \in [1, 100]$, $X_k \in \mathbb{N}^+$
- All X_k are distinct.
- For consecutive X_k and X_{k+1}, $X_k | X_{k+1}$ or $X_{k+1} | X_k$.

The game ends when no valid next to pick.

2.3 Scenario-Specific Goals

For the cooperative scenario: Maximize total reward:

$$S_{total} = \sum_{k=1}^{m} X_i$$

Key focus: Balancing "keep the chain long" and "include high-value integers".

For the competitive scenario: Alice and Bob alternate turns, each aiming to maximize their own total reward:

- Alice's goal: Maximize $S_{2A} = \sum_{oddi} X_i$
- Bob's goal: Maximize $S_{2B} = \sum_{evenj} X_j$
- Stress on the situation satisfying: $S_{2A} > S_{2B}$

2.4 Unified Graph Mapping

Map all game elements to an undirected divisibility graph $G = (V, E)$: $V = v_1, v_2, \dots, v_{100}$, where V is the node set, with v_i representing $i \in \{1, \dots, 100\}$; E contains $e_{i,j}$, which exists if $i|j$ or $j|i$. A valid chain $X = \{X_1, X_2, \dots, X_m\}$ corresponds to a simple path $P = (v_{X_1}, v_{X_2}, \dots, v_{X_m})$ in G, The "simple path" property (no repeated nodes) directly enforces the game's "no duplicate integers" constraint. The edge condition ensures the divisibility requirement is met. For the cooperative scenario, each node v_i has weight $w(v_i) = i$, and we need to calculate total reward $S_1 = \sum_{k=1}^{m} w(v_{X_k})$. In competition, Alice's reward S_A weights of nodes in odd position of P; Bob's reward S_B sums even positions.

The graph model collapses the game's complexity into a single, computable structure by translating every rules into topological constraints, making the vague "chain building" task into a well-defined " pathfinding" problem (a class of problems with mature algorithm solutions).

3 Solution to the Cooperative Game Scenario (1)

3.1 Trial on DFS + Optimized Pruning

Inspired by De Geest et al. (2020)'s OEIS A337125 (longest 77-element chain via pruned DFS) [1], we took this approach as a natural starting point. However, the limitations—algorithmic bias toward length, subpar sums, and the unreliability of manual fixes—led us to abandon DFS+pruning.

3.2 Model Establishing: Ant Colony Optimization (ACO) — Optimal Performer

ACO was selected for its ability to balance exploration (uncovering new paths) and exploitation (refining high-sum paths) — a critical fit for our sum-maximization objective.(implementation details in Appendix .1)

3.2.1 Core Mechanism

ACO simulates ant behavior, with artificial ants traversing graph G to build valid chains, guided by pheromones (collective memory) and heuristics (priorities):

- **Path Construction**: Each ant starts at a random node, iteratively moving to adjacent, unused nodes (following edges $e_{i,j} \in E$) until no valid extensions exist. This strictly enforces the game's "no-repeat" and "divisibility" constraints without explicit checks.
- **Pheromone Trails** ($\tau_{i,j}$): Encode edge quality—high-sum chains deposit more pheromone ($\Delta\tau_{i,j} = Q/S_{\text{ant}}$, Q=deposition constant), while evaporation ($\tau_{i,j} = (1-\rho)\tau_{i,j}$, ρ=rate) avoids stagnation.
- **Heuristic and Transition Probability**: $\eta_{i,j} = j$ biases ants toward large integers. The transition probability

$$p_{i,j} = \frac{\{\tau_{i,j}\}^{\alpha} \cdot \{\eta_{i,j}\}^{\beta}}{\sum_{k \in \text{unused, adjacent}} \{\tau_{i,k}\}^{\alpha} \cdot \{\eta_{i,k}\}^{\beta}}$$

  uses α (pheromone weight) and β (heuristic weight) to balance memory and priorities.

3.2.2 Optimized Parameters

Key parameters were fine-tuned through extensive experimentation:

Table 1: parameters

parameters	10 Runs
Number of Ants	200
Iterations:	1000
Pheromone Evaporation Rate (ρ):	0.3
Heuristic Importance (β):	2
Pheromone Importance (α):	1

3.2.3 Key Results

After running the ACO algorithm with the optimized parameters, we obtained results as below:

The highest reward was **3551**. For the detailed code and sequence, please refer to Appendix 1.

3.3 Model Evaluation and Reflection

3.3.1 Hierarchical Validation

To evaluate the ACO model's effectiveness, we applied a three-layer **Hierarchical Assessment Method:**

Table 2: Layer 1: Core Outcome Validation

Core Metric	10 Runs
Best Sum (Max)	3551.00
Best Sum (Mean)	3550.2
Best Chain Length (Mean)	77
Large Num. (> 50) Utilization	64.00% (32/50)

Table 3: Layer 2: Basic Reliability Validation

Reliability Metric	10 Runs
Valid Chains Proportion	90.26%
Avg. Running Time	16.65s
Avg. Convergence Iterations	262.86 (of 2000)

So we can draw the following conclusions:

- **Core Outcome Validation**: The model consistently finds high-sum chains (up to 3551), with a strong average performance (3547) and effective use of large integers (64%).
- **Basic Reliability Validation**: The model is robust, producing valid chains in every run, with reasonable computation times and quick convergence.
- **Optimization Potential Validation**: The algorithm has excellent stability. By adjusting parameters such as the number of ants, the efficiency can be improved a lot.

3.3.2 Reflection

ACO's efficacy stems from its intrinsic alignment with the divisibility graph's topology, transforming combinatorial constraints into emergent path-building behaviors that inherently satisfy divisibility and non-repetition. This bio-inspired paradigm bypasses explicit rule-checking, directly exploiting structural hubs and connectivity patterns for computational efficiency. Its balanced exploration-exploitation also overcomes the bias toward prioritizing large integers, identifying longer chains of medium-value, high-connectivity nodes for higher sums than isolated large integers.

4 Solution to the Competitive Game Scenario (2)

4.1 Mathematical Derivation for Opening Moves

Considering Alice's strategy, we should satisfy:

- Alice should get more scores than Bob.
- The whole sequence should be long enough for both competitors to get as more scores as possible.

For the first condition, consider a kind of numbers x for Alice to choose at the beginning of the game,

Table 4: Layer 3: Optimization Potential Validation

Optimization Metric	10 Runs
Best Sum Std. Dev.	2.56
Avg.Sum Change (Ants:500–200)	+3.53
Time Change (Evap. Rate: 0.5→0.4)	-25.58 s

satisfying:

- $x \in [51, 100]$;
- x is a semiprime, $x = p * q$, where p and q are both prime numbers.

After Alice chose such an x, Bob will be forced to choose p or q. Then, Alice can choose another big semiprime in the next step. Repeating this process, Alice can gain a first-mover advantage. For example, in the sequence 77, 7, 91, 13, 65, 5, 95, 19, 57, 3...... $S_{2A} = 385$, $S_{2B} = 47$, Alice wins easily.

But Alice should stop this strategy at some points.We found that small numbers including 2, 3, and 5 are important points to link 2 different series (We define a p-series is a set whose eletments are mutiples of p. e.g. : 5 series: 5,10,15......) to make the whole sequence long enouh. Using such small numbers in this strategy is a waste, but if Alice stops at the first small number she meets, she cannot get a big advantage. So, a good method is to let her stop at the second small number she meets. So, Alice should stop that strategy when she meets "small numbers".

Now, we can formally discuss the best semiprime for Alice to choose at the beginning of this strategy. For each semiprime Alice chooses at first, Bob has two choices. For each opening number, calculating the average S_{2A} of 2 choices, we can conclude that 91 is the best choice (91,13,65,5,95,19,57,3 (284 VS 38) and 91,7,77,11,55,5,95,19,57,3(385 VS 47)). No matter which factor to choose, Alice always takes a huge lead.

As for the second condition, we will always use numbers like $2, 3, 5, 8, 9$ to let the items to switch between different series. Unfortunately, to give a detailed method is rather hard.

Our general advice is:

- **Start with 91, continue with big numbers;**
- **Use small numbers like 2,3,5,8,9 to switch between series.**

4.2 Model Establishing: MAARF

4.2.1 Model Description

Given the complexity of the whole sequence, we proposed MAARF. By designing strategically significant reward functions (controlling rights, primes, chain length) and introducing Bob's "Challenger mechanism", we trained Alice to a 71% rate with an average score of 1496. Further analysis reveals that using large semi-prime numbers is the optimal strategy, verifying the core idea of combining "controlling the game flow" and "pursuing single-step scores".

4.2.2 Model Construction

The game is formalized as a Markov Decision Process (MDP) [2]: We represent the Game state as an 102–dimensional vector: $S_t = [u_1, u_2, \ldots, u_{100}, l_t, p_t]$, where $u_i \in \{0, 1\}$ indicates whether number i has been used, $l_t \in [0, 1]$ is the normalized value of the last number in chain $\mathcal{C}$, and $p_t \in \{0, 1\}$ identifies the current (Alice/Bob). For actions, Discrete selection from integers $\{1, \ldots, 100\}$ satisfying divisibility constraints and non-repetition. We employ action masking to ensure legal moves.

Table 5: MAARF Notation Summary

Symbol	**Meaning**
a_t	Action (selected integer) at time t
$\mathbb{I}_{\text{cond}}$	Indicator function (1 if condition true)
$\text{prime}(a_t)$	a_t is prime
P	Power-of-two set $\{2, 4, 8, 16, 32, 64\}$
G_t	Discounted return from time t
γ	Discount factor (0.99)
R_{t+k+1}	Future reward at $t + k + 1$
k	Future time steps
$\mathcal{L}(\theta)$	Policy gradient loss
θ	Policy network parameters
T	Episode length
$\pi_\theta(a_t \vert s_t)$	Action probability given state
$\log \pi_\theta(a_t \vert s_t)$	Log probability of action

We employed reward functions for Alice and Bob as below: **Alice's Reward ($R_A(a_t)$):**

$$R_A(a_t) = \underbrace{a_t}_{\text{Base}} + 1.5 \cdot \underbrace{\mathbb{I}_{\text{prime}(a_t)\wedge a_t>50}}_{\text{Prime}} + 1.0 \cdot \underbrace{\mathbb{I}_{a_t \in P}}_{\text{Chain}} \tag{1}$$

Bob's Reward:

$$R_B(a_t) = a_t + 1.5 \cdot \mathbb{I}_{\text{prime}(a_t)\wedge a_t>50} \tag{2}$$

Discounted Return Calculation

$$G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \tag{3}$$

$$\text{where: } G_t = \underbrace{R_{t+1}}_{\text{Immediate reward}} + \underbrace{\gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots}_{\text{Discounted future rewards}} \tag{4}$$

Policy Gradient Loss

$$\mathcal{L}(\theta) = -\frac{1}{T}\sum_{t=0}^{T} \underbrace{\log \pi_\theta(a_t|s_t)}_{\text{Policy log probability}} \cdot \underbrace{G_t}_{\text{Discounted return}} \tag{5}$$

This design incentivizes both point accumulation and strategic positioning, with explicit bonuses for:

- **Large primes** (> 50): Force opponent into limited response options
- **Power-of-two numbers**: Enable extended divisibility chains
- **Base value**: Maintain alignment with game's scoring mechanism

4.2.3 Network Architecture and Optimization Framework

Our MAARF framework is built upon a policy network that directly maps game states to action probabilities. The network accepts a 102-dimensional state vector encoding the usage patterns of numbers, the last selected value, and the current player. This input is processed through two fully-connected hidden layers (128 neurons each, ReLU activation), culminating in an output layer of 100 logits corresponding to integer choices 1–100, normalized by Softmax. To enforce game rules inherently, we employ action masking by applying large negative penalties (-10^9) to logits of illegal moves, ensuring all model outputs are valid actions.

The training regimen was designed for robust strategy discovery over 5,000 episodes. We utilized a policy gradient algorithm optimized with the Adam optimizer (learning rate $3 * 10^{-4}$), computing gradients from full episode rollouts. A discount factor of $gamma = 0.99$ was chosen to incentivize long-term planning over immediate rewards.

Exploration Strategy: We implement an adaptive *epsilon*-greedy policy with linear decay:

$$\epsilon = \max\left(0.05,\ 0.2 - 0.15 \cdot \frac{\text{episode}}{3000}\right)$$

This schedule ensures:

- **Initial Exploration**($epsilon = 0.2$): Braod strategy discovery in early training.
- **Gradual Expliotations**: Smooth Transition toward refined policy execution.
- **Final Refinement**($epsilon = 0.05$): Minimal exploration for strategy polishing

4.3 Model Evaluation and Reflection

4.3.1 Model Evaluation

Table 6: Competitive Scenario Performance Metrics

Metric	Value
Alice Win Rate	78.6%
Alice Average Score	942.3
Bob Average Score	876.1
Average Chain Length	38.2
Alice Maximum Score	1426
Bob Maximum Score	1389
Longest Chain Length	52

The performance of our MAARF framework validates a clear strategic blueprint for Alice. As shown in Table 6, Alice achieves a 78.6% win rate with consistent score advantage (942.3 vs 876.1), demonstrating the effectiveness of the learned approach.

Core strategy for Alice:

- Opening: Select large semiprimes (optimally 91) or large primes (> 50) to force Bob into limited response options.
- Mid-game: Utilize hub nodes with multiple divisors to control game flow and extend chain length.
- Transition: Employ small switching numbers (2, 3, 8, 9) to navigate between divisibility series.
- End-game: Force Bob into disadvantageous moves, ideally ending with him playing 1

This Control-over-Score Principle—prioritizing structural dominance over immediate point gains—proves consistently successful across simulations. The emergent strategy from MAARF training closely aligns with our mathematical analysis, confirming that optimal play requires sacrificing short-term advantages for long-term positional control.This Control-over-Score Principle—prioritizing structural dominance over immediate point gains—proves consistently successful across simulations. The emergent strategy from MAARF training closely aligns with our mathematical analysis, confirming that optimal play requires sacrificing short-term advantages for long-term positional control.

4.3.2 Reflection

While MAARF demonstrates a clear capability to learn competitive strategies that surpass random play, our approach reveals inherent limitations. The mathematical analysis, though insightful, lacks a formal theoretical framework, rendering its conclusions heuristic and incomplete. Similarly, the reinforcement learning agent is constrained by its reward structure, which can induce reward bias and limited exploration, potentially leading to local optima. Moreover, the model's strong dependency on the specific 1–100 integer set highlights its weak generalization capability and the absence of theoretical optimality guarantees.

5 Conclusions and Appendices

5.1 Conclusion

This work presents a unified graph-theoretic approach to the divisibility chain game, addressing both cooperative and competitive scenarios through tailored computational frameworks.

In the cooperative scenario, our Ant Colony Optimization (ACO) algorithm achieved a top reward of **3551** by effectively balancing chain length and node value, demonstrating superior performance over traditional depth-first search.

In the competitive scenario, MAARF reveals that a **"Control-over-Score"** strategy—initiating with large primes or semiprimes (e.g., 91) to constrain Bob's options, controlling mid-game flow through highly-connected hub nodes, and forcing Bob into terminal moves—yields a **78.6% winning rate** for Alice. This demonstrates that structural dominance consistently outperforms mere point accumulation. Key Insights:

- Graph modeling elegantly transforms complex game rules into tractable pathfinding problems.
- Strategic control of the game's structure outweighs short-term score optimization.
- Adversarial self-play is effective in discovering non-obvious, robust strategies.
- Reward function design is critical in guiding agents toward strategic policies.

Future work includes extending the approach to larger integer ranges, establishing theoretical performance bounds, and exploring real-world applications in sequential decision-making.

A central principle emerges: **optimal performance requires sacrificing immediate gains for long-term structural advantages.**

5.2 Appendix

.1 ACO Code and Specific Sequence

- **ACO**: https://github.com/Jincan-LI-HUB/ACO-for-HKUST-GZ-2025-MCM-Junior-Group/blob/main/ACO%20new%20parameter.py

.2 Competitive Scenario Code

- **MAARF**: https://github.com/Jincan-LI-HUB/MCM-2025-Junior-Group-Question-2_Reinforcment-Learning

References

[1] P. De Geest and M. Van Doorn. A337125: Longest chain of divisors from $\{1, \ldots, n\}$ such that adjacent elements divide each other, 2020. Accessed: 2025-10-16.

[2] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction.* MIT press, 2018.