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" problems 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 proble as an **Undirected Divisible Graph**:and employed:

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

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

For the competitive scenario, we model 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 imprctical, we propose a RM framework 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 $X = (X_1, X_2, \ldots, X_m)$, $k = 1, 2, \ldots, m$ 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 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 $S_{2B} = \sum_{evenj} X_j$
- overall goal: make sure: $S_{2A} > S_{2B}$

2.4 Unified Graph Mapping

All game elements map to an undirected divisibility graph $G = (V, E)$: $V = v_1, v_2, \ldots, v_{100}$, where V is the node set, with v_i representing $i \in \{1, \ldots, 100\}$; E contains $e_{i,j}$, which exists if $i|j$ or $j|i$. A valid chain $X = \{X_1, X_2, \ldots, X_m\}$ corresponds to a simple path $P = (v_{X_1}, v_{X_2}, \ldots, v_{X_m})$ in G, The "simple path" property (no repeated nodes) derectly 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 collapse the game's complexityinto a single, computable structure by translationgevery rules into topological constraints, making the vague "chain building" task into a well-defined " path-finding" 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 natrual 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	70.26%
Avg. Running Time	16.65s
Avg. Convergence Iterations	262.86 (of 2000)

So we can draw the following conclusions:

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

- **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 leverages the divisibility graph's topological traits—node 1 as a universal hub and large integers with high connectivity—and implicitly enforces "adjacent divisibility" and "non-repetition" constraints via traversal, removing redundant validation for streamlined computation. 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, it should satisfies:

- 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 second condition, we notice that some "small numbers "(especially like 2,3,8.....) are key points to connect two series of numbers, making the sequence as long as possible.

Consider a kind of numbers x for Alice to choose at the beginning of the game, satisfying:
(1) $x \in [51, 100]$;
(2) x is a semiprime, $x = p * q$, where p and q are both prime numbers.
(3)x, p, $q$shoul be as large as possible.
(4)p, q should not be 2 or 3.
This way, for Bob's pursuit of rewards, Bob will be forced to choose bigger prime factors of x. Then, Alice can choose another big semiprime in the next step. Repeating this process, Alice can gain a first-mover advantage. During our exploration, on the one hand, we discovered that this strategy is not applicable when Alice encounters "small numbers". So,
(5)we stops that strategy when Alice meets "small numbers".

Within these law, we get down to find the best opening move for Alice. For example, in the sequece 77, 7, 91, 13, 65, 5, 95, 19, 57, 3..... $S_{2A} = 385$, $S_{2B} = 47$, Alice wins easily. During our exploration, on the other hand, suppose Alice's opening number is a semiprime $x_0 = p_0 * q_0$, where p_0 or q_0 are the small number.No matter which factor for Bob to choose, Alice will quickly meet the "small number" and quit the existing strategy, causing a low reward ultimately.

Now, we will get down to the discussion of the best semiprime for Alice to choose at the beginning of this strategy. For p, q should not be 2 or 3, Alice chooses between 95,91,85,77,65,55, making the sequence listable and long. Calculationg the average S_{2A}, 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 next task to make the sequence long enough. We will always use numbers like 2,3,8,9 to let the items to switch between different series. (we define an p series is an series, whose items are multiples of p, like 5 series: 5,10,15,.....) Unfortunately, to give a detailed method is rather hard.
Our general advice is: **start with 91, continue with big numbers;; use small numbers like 2,3,8,9 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 machanism", 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

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
$\pi_\theta(a_t\|s_t)$	Action probability given state
$\log \pi_\theta(a_t\|s_t)$	Log probability of action

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 $u_i \in \{0, 1\}$ indecates 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 $\{1, ..., 100\}$ satisfying divisibility constraints and non-repetition. We employ action masking to ensure legal moves.

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

Network Architecture:

- **Input Layer**: 102-dimensional state vector encoding number usage patterns, last selected number, and current player identity
- **Hidden Layers**: Two fully-connected layers with 128 neurons each, utilizing ReLU activation for non-linear transformation
- **Output Layer**: 100-dimensional logits corresponding to integer selections 1-100, with Softmax normalization for probability distribution
- **Action Masking**: Automatic constraint enforcement through large negative penalties (-10^9) applied to illegal moves during forward propagation

Training Configuration:

- **Episodes**: 5,000 complete game trajectories for comprehensive strategy development
- **Algorithm**: Policy gradient optimization with discounted returns
- **Discount Factor**: $\gamma = 0.99$ to encourage long-term strategic planning
- **Optimizer**: Adam with learning rate 3×10^{-4} for stable convergence
- **Batch Processing**: Full episode rollouts for gradient estimation

Exploration Strategy: We implement an adaptive ϵ-greedy approach with linear decay:

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

This schedule ensures:

- **Initial Exploration** ($\epsilon = 0.2$): Broad strategy discovery in early training
- **Gradual Exploitation**: Smooth transition toward refined policy execution
- **Final Refinement** ($\epsilon = 0.05$): Minimal exploration for strategy polishing
- **Stable Convergence**: 3,000-episode decay period for stable learning dynamics

(full implementation in Appendix .2)

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

Above are the performance metrics showing the perfection of our MAARF; and the strategy it gives out as below: **Control-over-Score Principle**: Opening: large primes (> 50) or 64. Mid-game: hub nodes with multiple divisors. End-game: force Bob to play 1. Core: maximize future moves, not immediate gains. As we can see the strategy is barely satisfactory.

4.3.2 Reflection

The model effectively learns competitive strategies in the 1-100 divisibility chain game, outperforming random baselines. Limitations exist:The learned policy may converge to local suboptimal due to reward bias and limited exploration. It is specific to the 1–100 setting and lacks theoretical optimality guarantees.

5 Conclusions and Appendixs

5.1 Conclusion

This work presents a dual approach to the divisibility chain game. For cooperation, our ACO algorithm achieves 3551 total reward by balancing chain length and value. For competition, our MARL framework reveals that strategic control—prioritizing primes and power-of-two numbers—yields 78.6% win rates for Alice. The core insight across both scenarios is that optimal play requires sacrificing immediate gains for long-term structural advantages.

Cooperative Scenario: Our ACO algorithm achieves **3551 total reward** by balancing chain length and numerical value, outperforming traditional approaches through superior exploration-exploitation balance.

Competitive Scenario: MAARF reveals that strategic control yields **71% win rates** for Alice, with emergent strategies validating mathematical analysis while discovering novel tactics.

Key Insights:

- Structural control dominates immediate point maximization
- Graph modeling transforms constraints into path optimization
- Adversarial self-play discovers non-obvious strategies
- Reward design critically guides strategy convergence

Future Work: Extending to larger ranges, theoretical guarantees, and real-world applications.

The core principle: optimal play sacrifices 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.