Exploration of the Iterated Prisoner's Dilemma

Joshua Lebert, Julia Ducharme, Norika Upadhyay, and Tansh Koul

University of Windsor

COMP-3710: Artificial Intelligence Concepts

Dr. Robin Gras

March 19, 2025

Abstract – **Invented in the 1950s by Flood & Drescher, the Prisoner’s Dilemma is well known in many areas of study. This project aims to explore the Iterated Prisoner's Dilemma, where the agent has knowledge of previous rounds of the Prisoner’s Dilemma being played. We will examine various strategies and optimization methods that have been developed to determine the method that generates the best strategies and why. Experiments are performed to observe how different strategies perform against each other. We have implemented different optimization methods (genetic algorithm, hill climbing, and tabu search), changing variables such as number of generations, population size, and memory depth. In manipulating the variables, the experiments found that a memory depth of 2 yielded the highest average score, while a population size of 250 produced the greatest stability of results, and that 250 generations was sufficient to train the genetic algorithm models. Furthermore, the optimization method which returned the highest average score was genetic algorithms, with tabu search and hill-climbing following closely behind.**

Introduction

The Prisoners' Dilemma is a classic problem in game theory that demonstrates the conflict between individual and collective rationality. In the standard version, two criminals are arrested and are given the choice to either cooperate or defect. If both criminals confess, they receive a moderate sentence. Meanwhile, if one confesses and the other doesn't, the confessor goes free, and the other gets a severe sentence. If neither confesses, both will receive a minimal sentence. The two cannot communicate, eliminating opportunities for them to strategize together and maximizing likelihood of each criminal prioritizing individual wellness over global wellness.

The payoff structure encourages each both to confess, as this is the "rational" choice for the individual and minimizes their potential sentence regardless of what the other chooses; if both act rationally and confess, they both receive a moderate sentence. However, if they both acted irrationally and cooperated by not confessing, they would both receive the minimal sentence, achieving a better global output.

In real-world scenarios, interactions such as this are often repeated, leading to the Iterated Prisoners' Dilemma (IPD). In the IPD, players have knowledge of the previous three or more rounds and can base their decisions off this history, which opens up the possibility for the evolution of cooperative behavior. The strategies that emerge should be deterministic; the players will make the same moves given the same history.

Our project focuses on the IPD, aiming to explore and compare different strategies that players can employ in this repeated game. By simulating various strategies, we intend to analyze their performance in terms of accumulated payoffs and the level of cooperation they foster. We also intend to test these strategies against optimization methods. This analysis will help us understand the dynamics of cooperation and competition in repeated social interactions.

The motivation behind this project is to gain a deeper understanding of how cooperation can evolve in a world where individual self-interest might seem to dictate otherwise. By studying the IPD, we can draw parallels to real-world situations such as business partnerships, international relations, and

evolutionary biology, where repeated interactions play a crucial role.

Our contribution to this project includes:

1. Implementing a simulation framework for the IPD that can handle multiple strategies and a variable number of rounds.
2. Implementing a range of human based algorithms including always-cooperate, always-defect, random selection, tit for tat, suspicious tit for tat, and tit for 2 tat.
3. Testing the implemented human strategies against optimization methods such as hill climbing, tabu search and the genetic algorithm.
4. Analyzing the results to determine which strategies perform best under different conditions and how they influence the overall level of cooperation.
5. Providing recommendations based on our findings for scenarios where repeated interactions are common.

Through this project, we aim to contribute to the ongoing discussion on the evolution of cooperation and the strategic choices individuals make in repeated social dilemmas.

Literature Review

The Iterated Prisoners' Dilemma (IPD) serves as a cornerstone model in game theory, providing profound insights into how cooperation can emerge and persist in repeated strategic interactions despite immediate incentives to defect. Unlike the single-round Prisoners' Dilemma where defection dominates due to its immediate advantage, IPD involves multiple rounds where two participants simultaneously choose to cooperate (C) or defect (D). The reward structure follows four parameters: mutual cooperation (C,C) yields reward (R), mutual defection (D,D) results in punishment (P), unilateral defection (D,C) gives the defector a temptation payoff (T) while leaving the cooperator with a sucker's payoff (S). The constraints ($T > R > P > S$) and ($2R > T + S$) ensure that cooperation remains collectively beneficial over time while defection tempts individual gain.

This repeated framework enables players to adapt their strategies based on interaction history, making IPD a rich environment for studying cooperation, competition, and strategic evolution. This review synthesizes IPD's historical development, provides comprehensive analysis of key strategies, explores optimization techniques with special focus on genetic algorithms, examines recent empirical findings, discusses interdisciplinary applications, addresses criticisms, and outlines future research directions, establishing a thorough foundation for a technical report.

Key Strategies and Their Dynamics

IPD strategies vary from simple to complex, addressing the dilemma. This analysis covers key strategies' mechanics, math, payoffs, evolution, strengths, weaknesses, theory, and performance across contexts (e.g., noise, game length, opponents), based on a payoff matrix (T=5, R=3, P=1, S=0), generalizable to $T > R > P > S$.

IPD Payoff Matrix

		B	
		cooperate	defect
A	cooperate	3, 3	0, 5
	defect	5, 0	1, 1

Fig.1: IPD Payoff Matrix (Gras, 2025).

Always Defect (ALLD): Defects every round, ignoring history. It scores 1 vs. itself, 5 vs. ALLC, and just over 1 vs. TFT after an initial 5. It exploits cooperators early but drops to 1 as defection spreads, losing to strategies scoring 3. Strong in one-shot or finite games, it avoids the worst payoff but lacks IPD adaptability. Self-interested and short-term, it fails as cooperation grows, excelling only in short games or vs. naive players

Always Cooperate (ALLC): Cooperates every round, ignoring opponent actions. It scores 3 vs. itself, 0 vs. ALLD (averaging 0), and 3 vs. TFT, depending on the opponent. It excels in cooperative groups but collapses against defectors, dropping to 0 with no defense. Strong in altruistic, noise-free settings, it's exploitable and lacks retaliation, failing in mixed environments. It models trust, thriving only in ideal conditions, and fades evolutionarily without support. Best in fully cooperative scenarios.

Tit-for-Tat (TFT): Starts by cooperating, then mimics the opponent's last move. It scores 3 vs. itself, drops to just above 1 vs. ALLD after an initial 0, and holds 3 vs. ALLC. It fosters cooperation evolutionarily, resisting ALLD in long games, but noise can spark defection cycles. Strong in its balance of niceness, retaliation, and forgiveness, it thrives in stable settings. Weak against noise (falling to 1) or complex strategies like ZD (scoring 2 vs. 4), it embodies reciprocal altruism, excelling in noise-free, adaptive contexts but faltering in noisy or short games.

Grim Trigger: Starts with cooperation but permanently switches to defection after a single opponent defection, making it a powerful deterrent in stable, noise-free environments. However, this unforgiving nature renders it highly vulnerable to errors and noise, unlike the more adaptable Tit-for-Tat and Pavlov strategies, and it can be exploited by sophisticated opponents, ultimately limiting its effectiveness in realistic, dynamic scenarios where forgiveness and adaptability are crucial.

Pavlov: Repeats its last move if rewarded (5 or 3), switches if not (1 or 0), focusing on outcomes. It oscillates then hits 3 vs. itself, averages near 1 vs. ALLD, and exceeds 3 vs. ALLC initially. Adapting via reinforcement, it exploits ALLC and resists ALLD in noise better than TFT. Strong in noisy settings, it corrects errors and stabilizes at 3, but slow against mixed strategies and weak vs. persistent defectors. A learning model, it shines where feedback aids cooperation, less so in short or defector-heavy games.

Generous Tit-for-Tat (GTFT): Mimics TFT but forgives defection with a small chance (e.g., 10%). It scores mostly 3 vs. itself, above 1 vs. ALLD (e.g., 1.4), and near 3 vs. TFT, beating TFT in noise. It breaks retaliation cycles evolutionarily, balancing resistance and tolerance. Strong in noisy games, it avoids TFT's drop to 1, but risks exploitation by defectors, slightly lowering noise-free scores (e.g., 2.9 vs. 3). Forgiveness aids stability, excelling in uncertain, long-term settings, though less against persistent defectors.

Optimization Techniques in IPD

Optimization fuels IPD strategy evolution, with genetic algorithms excelling in vast searches, alongside hill climbing and tabu search for refinement.

Genetic Algorithms (GAs)

Mechanism*:* GAs mimic natural selection to evolve IPD strategies, treating them as chromosomes within a population. The process unfolds as follows:

Initialization: A population (e.g., 50-200 strategies) is randomly generated, encoding rules like TFT or random FSMs.

Fitness Evaluation: Each strategy competes in IPD tournaments against a set of opponents (e.g., ALLD, TFT, Pavlov), with fitness defined as average payoff or tournament rank across games.

Selection: High-fitness strategies are chosen via methods like tournament selection or roulette-wheel selection, favoring those with higher payoffs.

Crossover: Strategy pairs exchange genetic material, blending traits to create offspring with hybrid behaviors.

Mutation: Random changes occur at specified rate, introducing novelty to prevent stagnation.

Iteration: Process repeats over generations (e.g., 100-500), refining population until convergence or stopping criterion (e.g., stable fitness plateau).

Parameters Research*:* Studies on GA parameters in IPD provide insight into effective configurations:

Population Size(N): Ranges of 50-200 are common. Smaller populations (50-100) suffice for simple strategies (e.g., rediscovering TFT), while larger ones (100-200) enhance diversity for complex FSMs or noisy environments (Nowak & Sigmund, 1993; Chong et al., 2008). Harper's 2024 study with 195 strategies suggests larger populations for broad exploration, balancing computational cost with solution variety.

Mutation Rate(μ): Typically 0.01-0.1. Low rates (0.01-0.05) preserve successful traits (e.g., TFT's reciprocity), minimizing disruption, while higher rates (0.05-0.1) introduce novelty, essential for adapting to noisy or diverse opponents (Fogel, 1994). Axelrod's experiments often used ($\mu = 0.05$), evolving robust variants over 200 generations.

Number of Generations: 100-500 generations are standard. Simpler settings converge by 100-200 (e.g., rediscovering TFT), while complex, noisy, or probabilistic ending scenarios require 300-500 to refine adaptive strategies like Evolved FSM 16 (Miller, 1996). Harper's simulations likely used 200-300 generations to stabilize 45,600 tournament outcomes.

Impact*:* GAs produced Evolved FSM 16, a 16-state machine excelling in standard tournaments by recalling multiple past moves, averaging near 3 against cooperative opponents while punishing defectors. In noisy settings, GA-evolved strategies adjust transition probabilities (e.g., increasing (p(C)) after D to mimic GTFT), outperforming TFT's rigid retaliation. Axelrod's early GA work rediscovered TFT-like behaviors, while later studies evolved memory-2 or probabilistic strategies for greater adaptability.

Hill Climbing: Begins with a strategy like TFT, tweaks it (e.g., defecting after two Ds instead of one), tests these changes in games, and keeps the best version until no better option appears. It refines TFT into tailored variants, boosting performance

against specific opponents by adjusting rules like retaliation. Fast and straightforward, it excels at fine-tuning responses (e.g., to ALLD) but often gets stuck in local optima, missing bigger leaps like shifting to memory-2 strategies, unlike genetic algorithms' broader search.

Hill Climbing with Random Restart: Boosts traditional hill climbing in the Iterated Prisoners' Dilemma (IPD), where players repeatedly cooperate (C) or defect (D) to maximize payoffs (T=5, R=3, P=1, S=0). Starting with a strategy like Tit-for-Tat, it refines through small tweaks (e.g., defect after two Ds) until stuck, then restarts from a new random point (e.g., "cooperate 70% of the time"), repeating 5-10 times to pick the best result. In IPD's vast strategy space, this escapes local optima—like refining Tit-for-Tat to 2.8 while missing Pavlov's 3.5; exploring diverse options and boosting payoffs (e.g., 3.2). Harper (2024) supports its edge in complex landscapes by highlighting adaptability and exploration in winning IPD strategies, and through the use of related optimization techniques like evolutionary algorithms. It balances precision and exploration, optimizing strategies against varied opponents.

Tabu Search: enhances hill climbing by using a tabu list (e.g., 5-10 entries) to avoid revisiting recent strategies, escaping local optima by accepting worse moves temporarily, guided by aspiration criteria (e.g., overriding tabu for big payoff gains). It optimizes F(s) by exploring N(s), dodging tabu moves unless F(s') beats the best so far. More exploratory than hill climbing, its complexity ($O(I \cdot K \cdot M)$, with M as list size) suits noisy or dynamic IPD, though it's less scalable than genetic algorithms due to memory demands, balancing exploration and refinement effectively.

Experimental Setup and Methodology

We applied a systematic approach to evaluate the effectiveness of four optimization methods, Hill Climbing, Hill Climbing with Random Restart, Tabu Search, and Genetic Algorithm (with roulette wheel selection and random point crossover), in producing effective strategies for the Iterated Prisoner's Dilemma. For each optimization technique, we performed experiments to determine the best combination of input parameters, and then evaluated the performance of the best strategy produced by the optimal parameters against well known "human" strategies, which served as our baseline strategies.

Strategy Representation

To apply optimization techniques to the Iterated Prisoner's Dilemma, the strategies had to be represented in a way that would allow them to be optimized. For our experiments, we represented strategies as a string of bits, each bit representing a cooperate action (when the bit equals 0) or defect action (when the bit equals 1). The past moves (both the player's and its opponent's) are stored in a second bit array (of length 2n, where n is the memory depth, ie, the number of past moves being considered), which was used as an index to select which bit of the strategy string would be chosen to determine the current turn's action.

Performance Metrics

For our experiments, the fitness of a given strategy produced by a model is determined by the average score it achieves in a tournament against our baseline strategies. Each tournament consists of a 100-round match against each baseline strategy. The evaluation function of each model consisted of calculating this fitness

score. We chose nine common "human" (i.e., algorithmic, rather than trained by a model) strategies to serve as the baseline strategies we train our models against. These strategies consist of:

1) Always Cooperate: Cooperates every turn.
2) Always Defect: Defects every turn.
3) Tit for Tat: Begins by cooperating, and then mimics the opponent's previous move.
4) Suspicious Tit for Tat: Begins by defecting, and then mimics the opponent's previous move.
5) Tit for 2 Tat: Begins by cooperating, and then defects only when the opponent defected in the past two moves, otherwise cooperates.
6) Generous Tit for Tat: Begins by cooperating, and then mimics the opponent's previous move, with a small chance (10%) of forgiving a defection (cooperating when opponent defected).
7) Pavlov: Begins by cooperating. If the previous round resulted in a good payoff (both cooperate or successful defect), repeat the previous move. If the previous round resulted in a bad payoff (both defect or opponent defect) then switch to opposite move.
8) Grim Trigger: Cooperates until the opponent defects, after which it defects every turn.
9) Random: Chooses to cooperate or defect at random, with equal probability.

Parameter Experiments and Analysis

We began by performing experiments to determine the optimal input parameters for each model. Our methodology consisted of choosing a set of "fixed" parameters, and then varying each parameter (one at a time, while keeping the other parameters at their "fixed" values), over a chosen range of values. When choosing the fixed values for each model, the primary consideration was values that were suggested through our literature review, in addition to values that had performed well in past partial experiments. We then selected values greater than and less than the fixed values to vary over in our experiment. The average performance and average time required to train a model for a combination of parameters is averaged over a number of trained models, (typically 30 models, except where computationally prohibitive), in order to prevent outliers from skewing our results.

For the Hill Climbing method, we varied the memory depth and number of training iterations. We varied the memory depth over values of 1, 2, 3, 4, and 5 and chose a fixed value of 3. We varied the training iterations over values of 5, 8, 10, 25, and 50 and chose a fixed value of 10. The performance of each parameter combination was averaged over 30 models trained using the same parameters. Results demonstrated that a memory depth of 2 and 10 training iterations were the optimal parameters for this method. Hill Climbing rarely took more than 10 iterations to find a local maximum, so increasing the number of training iterations further has little impact.

For the Hill Climbing with Random Restarts method, we varied the memory depth, number of iterations, and the number of random restarts. We varied the memory depth over values of 1, 2, 3, 4, and 5, fixed at 2. We varied the number of iterations over

10, 25, 50, 100, 250, and 500, fixed at 100. We varied the number of random restarts over 5, 10, 25, 50, and 100, fixed at 10. Results demonstrated that a memory depth of 2, 100 training iterations, and 50 random restarts is the optimal combination of parameters. In

For the Tabu Search method, we varied the memory depth, number of training iterations and the tabu length. We varied the memory depth over 1, 2, 3, 4, 5 and chose a fixed value of 3. We varied the training iterations over values of 20, 100, 250, 500 and 1000 and chose a fixed value of 250. We varied tabu length over values of 20. 50, 100, 200, 250 and chose a fixed value of 100. The performance of each parameter combination was averaged over 30 models trained using the same parameters. Results demonstrated that a memory depth of 2 was optimal for this method. Varying iterations and tabu length did not show any significant difference in performance.

For the Genetic Algorithms, we varied the memory depth, the training iterations (number of generations), the population size, and the mutation rate. We varied the memory depth over values of 1, 2, 3, 4, and 5 and chose a fixed value of 3. The number of generations (training iterations) varied over 20, 100, 250, 500, and 1000, with a fixed value of 250. We varied the population size over 20, 100, 250, 500, and 1000, with a fixed value of 250. We varied the mutation rate over values of 0.0001, 0.001, 0.01, 0.1, and 0.5, fixed at 0.001. Parameter performance was averaged over 20 models trained with identical parameters. Results demonstrated that a memory depth of 2, 250 training iterations, a population size of 250, and a mutation rate of 0.001 produced the most effective models. For both population size and the number of training iterations, we found that although values lower than 250 perform worse, increasing the values beyond 250 had little impact, but carried immense computation cost. We found that extreme values produced reduced performance and moderate mutation rates between 0.001 and 0.1 produced the best performance, reinforcing Fogel's (1994) of balancing exploration and exploitation in evolutionary algorithms.

Graphs showing the results of the parameter experiments can be found in the Appendix.

Results and discussion

Results from our experiments on model parameters reveal that a memory depth of two (4 bit index string and a 16 bit strategy string) universally and consistently outperforms all other memory depth values that were tested in our experiments (1, 3, 4, and 5).

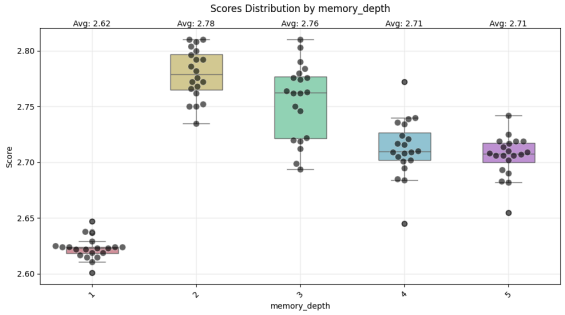


Fig.2: Scores Distributed by Memory Depth.

We theorize that these results arise because a memory depth of two provides a balance between the ability to recognize patterns and strategy complexity. A smaller memory depth (i.e. a memory depth of one) lacks the ability to recognize patterns, so it is unable to form advanced strategies and produces consistently poor results which Harper (2024) mentions. Memory depths greater than two can achieve similar performance, but with much less consistency, which implies that an optimal strategy for the Iterated Prisoner's Dilemma can be achieved with a memory depth of 2. As such, greater

depths exponentially increase the state space, and thus reduce the chance of finding the optimal strategy, while providing minimal possible benefit compared to a memory depth of 2.

The model which consistently performed best is the Genetic Algorithm, usually followed closely by Tabu Search and Hill Climbing with Random Restart, and then Hill Climbing followed by the baseline ("human") algorithms.

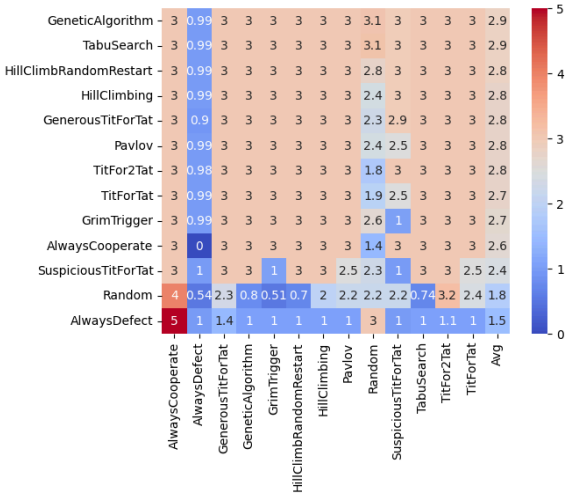


Fig.3: Heatmap of competing strategies.

The performance of the Genetic Algorithm method is much more dependent on optimized input parameters than the other methods, but performs better when provided with such parameters. For Hill Climbing, Hill Climbing with Random Restart, and Tabu Search, the input parameters often had little impact beyond a certain threshold, likely because these methods can terminate early.

Analysis of the successful strategies created by the optimization methods reveals that they all share common traits: retaliation against betrayal, eagerness to cooperate, and forgiveness of betrayal. This aligns with Axelrod's (1984) findings that strategies that are 1) nice (i.e., eager to cooperate, and cooperate first), 2) provocable (i.e., retaliate against betrayal, and 3) forgiving (i.e. attempt to return to cooperation after they retaliate) are most successful.

Conclusion

The Prisoner's Dilemma has been studied in many fields, and the Iterated Prisoner's Dilemma adds complexity by introducing layers of strategy. Through our experiments, we found that a memory depth of 2 yielded the highest average scores and the genetic algorithm worked best out of the optimization methods.

Future Work

While our experiments were expansive and yielded many practical results, there are areas we are interested in exploring further.

Agent-Based Modeling: Use agent-based modeling to simulate the Iterated Prisoner's Dilemma in more complex environments. This could involve introducing spatial structure, social networks, or other factors that influence how strategies interact and evolve.

Varying Payoff Structures*:* Explore the impact of different payoff structures. Your current study uses a specific payoff matrix. How do the results change if the rewards and penalties for cooperation and defection are altered? This could reveal insights into how the incentive structure influences the evolution of cooperation.

Machine Learning: Using Machine learning to predict the fitness of a strategy. Some considerations would include how to build the train set, what size to build it, and how the fitness itself will be evaluated.

References

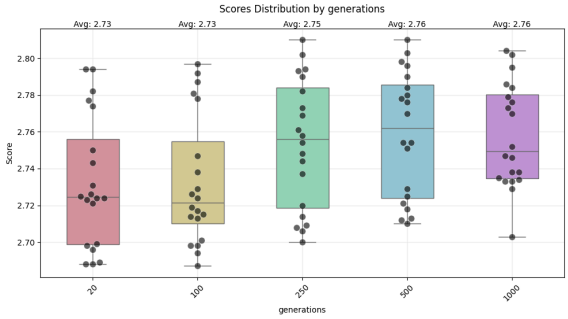
1. Axelrod, R. (1984). The Evolution of Cooperation. Link
2. Press, W. H., & Dyson, F. J. (2012). Iterated Prisoner's Dilemma contains strategies that dominate any evolutionary opponent. Proceedings of the National Academy of Sciences, 109(26), 10409-10413. Link
3. Harper, M. (2024). Properties of winning Iterated Prisoner's Dilemma strategies. PLOS Computational Biology. Link
4. Nowak, M. A., & Sigmund, K. (1993). A strategy of win-stay, lose-shift that outperforms titfor-tat in the Prisoner's Dilemma game. Nature, 364(6432), 56-58. Link
5. S. Y. Chong, P. TiÑo and X. Yao, "Measuring Generalization Performance in Coevolutionary Learning," in IEEE Transactions on Evolutionary Computation, vol. 12, no. 4, pp. 479-505, Aug. 2008, doi: 10.1109/TEVC.2007.907593.
6. D. B. Fogel, "An introduction to simulated evolutionary optimization," in IEEE Transactions on Neural Networks, vol. 5, no. 1, pp. 3-14, Jan. 1994, doi: 10.1109/72.265956.
7. Raihani, N. J., & Bshary, R. (2011). Resolving the iterated prisoner's dilemma: theory and reality. Journal of Evolutionary Biology, 24(8), 1628-1639. Link
9. Nowak, M. A., & May, R. M. (1992). Evolutionary games and spatial chaos. Nature, 359(6398), 826-829. Link
10. Miller, J. H. (1996). The coevolution of automata in the repeated Prisoner's Dilemma. Journal of Economic Behavior & Organization, 29(1), 87-112. Link
11. A.J. Stewart, & J.B. Plotkin, From extortion to generosity, evolution in the Iterated Prisoner's Dilemma, Proc. Natl. Acad. Sci. U.S.A. 110 (38) 15348-15353, Link (2013).
12. Chen, J., & Zinger, A. (2014). The robustness of zero-determinant strategies in Iterated Prisoner's Dilemma games. Journal of Theoretical Biology, 357, 46-54. Link
13. R. Gras (2025), COMP-3710 Project Slides. Link
14. OpenAI. (2023). ChatGPT (o4 model). Link
15. J. Lebert, & J. Ducharme. (2025). 3710-prisoners-dilemma. Link

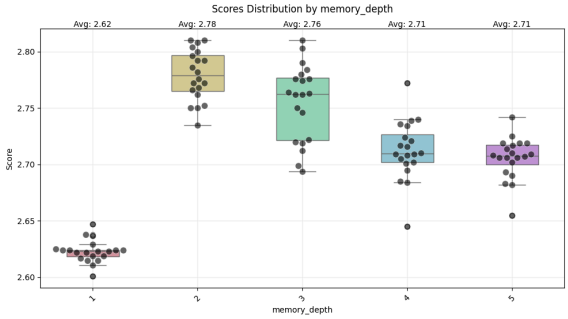
Contributions

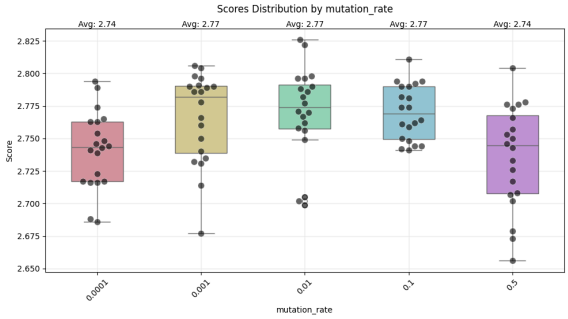
Norika Upadhyay contributed the abstract, introduction, and future work.
Tansh Koul contributed the literature review.
Joshua Lebert and Julia Ducharme collaborated equally on the experimental setup and methodology, and results and discussion. Joshua focused primarily on implementation of optimization methods and simulation of IPD, while Julia focused primarily on experimentation infrastructure, data analysis, and graphing.

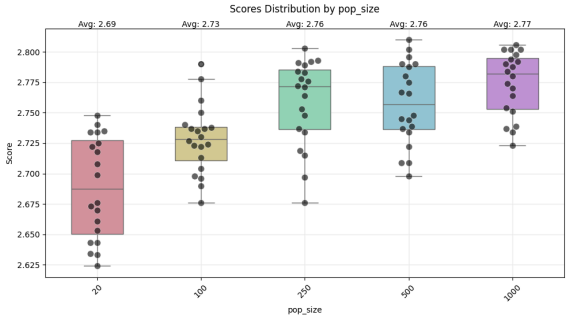
Appendix

Genetic Algorithm Experiments:

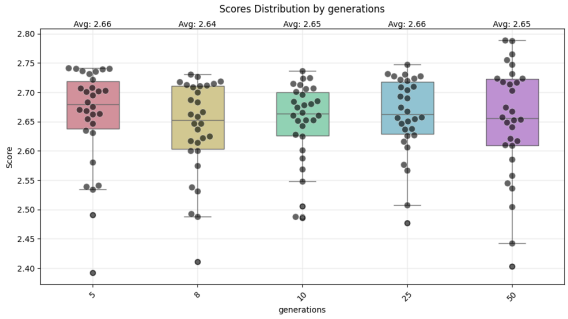


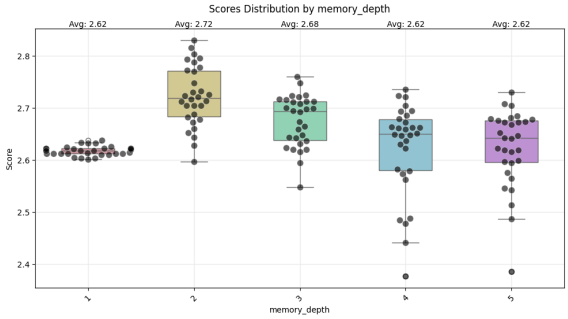






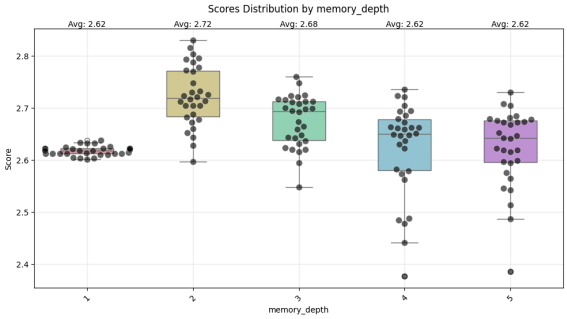
Hill Climbing Experiments:

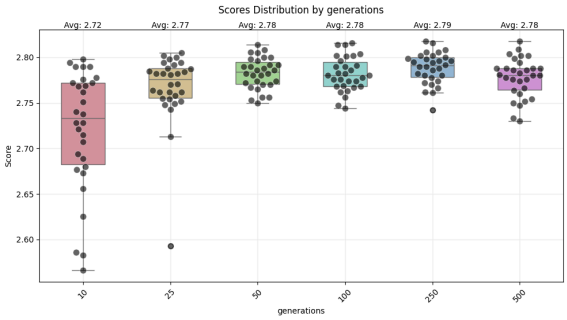


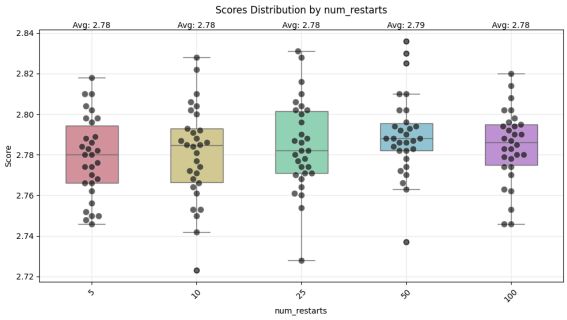


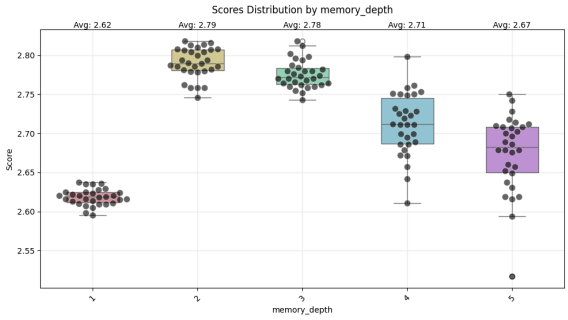
Appendix

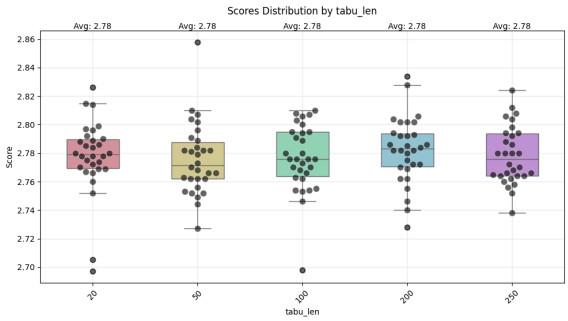
Hill Climbing With Random Restart Experiments:











Tabu Search Experiments: