# Algorithmic Collusion at Test Time: A Meta-game Design and Evaluation

Yuhong Luo * Daniel Schoepflin † Xintong Wang ‡

March 11, 2026

**Abstract**

The threat of algorithmic collusion, and whether it merits regulatory intervention, remains debated, as existing evaluations of its emergence often rely on long learning horizons, assumptions about counterparty rationality in adopting collusive strategies, and symmetry in hyperparameters and economic settings among players. To study collusion risk, we introduce a meta-game design for analyzing algorithmic behavior under test-time constraints. We model agents as possessing pretrained policies with distinct strategic characteristics (e.g., competitive, naively cooperative, or robustly collusive), and formulate the problem as selecting a meta-strategy that combines a pretrained, initial policy with an in-game adaptation rule. We seek to examine whether collusion can emerge under rational choices and how agents co-adapt toward cooperation or competition. To this end, we sample normal-form empirical games over meta-strategy profiles, compute relevant game statistics (e.g., payoffs against individuals and regret against an equilibrium mixture of opponents), and construct empirical best-response graphs to uncover strategic relationships. We evaluate reinforcement-learning, UCB, and LLM-based strategies in repeated pricing games under symmetric and asymmetric cost settings, and present findings on the feasibility of algorithmic collusion and the effectiveness of pricing strategies in practical "test-time" environments. The source code is available at: `https://github.com/chailab-rutgers/CollusionMetagame`.

## 1 Introduction

The use of algorithms to automate economic decisions—such as pricing, bidding, and bargaining—has become increasingly prevalent. Rather than following static rules, modern systems leverage optimization techniques (e.g., reinforcement learning) or AI models with reasoning capabilities (e.g., large language models) to learn from their environment, anticipate the behavior of other participants, and adapt their strategies to improve outcomes.

This growing autonomy introduces new, critical risks including the undesired and uncommunicated cooperation among algorithms, commonly known as *algorithmic collusion*. Recognized as a major concern in the era of advanced AI [27], algorithmic collusion poses unique challenges beyond human collusion [16, 4], as algorithms may learn to coordinate without explicit communication or intent [44]. Prior work demonstrates that collusive behavior can arise in simulated pricing and auction environments using common algorithms such as Q-learning [14] and large language models (LLMs) [23]. However, whether such behavior persists under realistic conditions and deployments remains unclear. For example, Calvano et al. [14] document convergence to collusion between two Q-learning agents after around 1.5 million interaction rounds, reflecting a prohibitively long learning horizon with significant early-stage exploration costs.

A more practical evaluation considers test-time interactions, where pretrained policies are randomly rematched and allowed to interact for a limited number of rounds [14, 20]. These studies show that while collusion may fade at first, it can re-emerge after roughly 40,000 rounds of play. Importantly, prior analyses largely assume symmetric algorithmic configurations (e.g., identical hyperparameters) and leave open a central

*Rutgers University, `y.luo@rutgers.edu`
†DIMACS, Rutgers University, `daniel.r.schoepflin@gmail.com`
‡Rutgers University, `xintong.wang@rutgers.edu`

question: whether collusion constitutes a rational, stable outcome when agents strategically behave under test-time constraints.

In this work, we seek to evaluate the viability and effectiveness of algorithmic collusion in "test-time" settings, where an agent must adapt within a limited number of interactions to opponents whose policies and economic settings (e.g., cost or quality) may differ from those encountered during training. To this end, we propose a meta-game framework. Different learning algorithms (e.g., Q-learning, no-regret methods, and LLMs) are first used at training time to generate candidate *initial policies* for deployment. These policies are then grouped into strategic categories based on performance metrics, including their propensity to collude with a training-time partner and their robustness against a best-response opponent. At test time, a *strategy* is formed by pairing an initial policy sampled from a selected category with an adaptation rule (e.g., a learning rate) drawn from a set of feasible update procedures for repeated play. A *meta-strategy* governs both the choice of initial policy category and the manner in which the policy adapts during interaction.

We are interested in analyzing these competing meta-strategies by modeling their interactions as a game, referred to as the *meta-game*. By sampling strategies for each player together with initial states, we generate meta-game instances from which evaluation metrics can be computed and game-theoretic analysis can be further conducted. Through this framework, we provide a statistical characterization of meta-strategy performance and address the central question: Can algorithmic collusion emerge within a limited time horizon under rational strategic choices?

**Our contribution.** We propose metrics to characterize and evaluate a (pretrained) policy and introduce a meta-game framework to assess algorithmic collusion at test time through interactions among meta-strategies. We conduct extensive experiments using a diverse set of algorithms (including Q-learning, UCB, and LLMs) to generate initial policies for repeated pricing games. Our results provide empirical evidence on meta-strategy performance, and offer insights into the conditions under which algorithmic collusion may emerge and persist. We highlight several key findings below:

- Q-learning can produce pretrained policies that *robustly collude* with their best-response counterparts, whereas UCB-based policies, despite colluding with their training-time partners, are often exploitable by best-response opponents.
- Under symmetric cost settings, each algorithm we evaluate—Q-learning, UCB, LLM—admits at least one pure or mixed Nash equilibrium among meta-strategies that leads to a collusive outcome, indicating that collusion can arise from rational strategic choices.
- For Q-learning meta-strategies, collusion diminishes under shorter interaction horizons and pessimistic Q-value initialization, which can be interpreted as reflecting an agent's prior belief that its counterparty is less likely to collude. Notably, in contrast to prior studies based on symmetric algorithmic setups that observe sustained collusion in asymmetric cost settings [14], our findings reveal that rational strategy selection substantially suppresses collusion under asymmetry.
- While UCB meta-strategies exhibit stronger collusion than Q-learning overall, Q-learning with random initialization can best respond to most UCB meta-strategies, calling into question UCB's competitiveness under test-time conditions.
- LLM-based agents can demonstrate adaptive behavior guided by pre-game history: policies that exhibit greater collusion during pre-game interactions tend to sustain or re-establish collusion even after episodes of exploitation at test time.

## 1.1 Related Work

**Algorithmic collusion.** Collusion can arise as an equilibrium among rational players in infinitely repeated games. Building on the Folk Theorem, prior work shows that cooperation can be sustained when algorithms embed credible punishment for deviations [51, 28, 36, 3]. More recent studies find that collusive equilibria may also emerge without explicit threat mechanisms, as shown in a Stackelberg framework [6], and examine how algorithmic (non-)collusion can be detected and regulated through data audits [29, 30].

Collusive behavior has been documented in e-commerce and dynamic pricing settings [22, 13, 43, 7, 12]. Following Calvano et al. [14], a growing body of literature investigates algorithmic collusion in the laboratory across settings ranging from stylized games, such as the repeated Prisoner's Dilemma [42, 10], to more realistic auctions [9]. These studies also consider different algorithms, including reinforcement learning [35, 31], bounded-regret methods [28], and LLMs [23, 5]. Several follow-up studies suggest that prior findings may overstate the threat of algorithmic collusion, as they rely on restrictive modeling assumptions [2]. Collusion typically emerges after long training horizons—on the order of millions of repeated-game rounds—and tends to depend on symmetric algorithmic configurations, suggesting implicit coordination or prior communication [37]. Moreover, the algorithmic configurations that give rise to collusion may be irrational in practice, being outperformed by more robust or commonly used alternatives [1, 11]. Thus, the emergence of algorithmic *tacit* collusion among rational agents under realistic conditions and finite interaction horizons remains an open question.

**Strategy selection, adaptation, and evaluation.** Our work focuses on strategy performance at deployment, resembling a test-time or tournament environment. Prior research has examined strategy selection and evaluation through meta-game frameworks [34, 38] and empirical game-theoretic analysis (EGTA) [59], applied to domains such as trading agent competitions [47, 21, 45], financial markets [53, 55, 56, 57], supply chain management [33], negotiation [38], and auctions [52]. Related to our setting is Carissimo et al. [15], who model collusion as coordination in Q-learning hyperparameters (e.g., discount and exploration factors) during training. Our study differs by focusing on test-time adaptation among pretrained agents with heterogeneous initializations, distinguishing deployment behavior from training dynamics.

Related online learning literature also examines *adaptability* and *non-exploitability* in repeated games [18, 17, 19], and develops approximate best-response methods for equilibrium analysis in complex games such as poker and Go [39, 50, 40]. These insights inform our design of metrics for characterizing pretrained policies and their adaptation strategies.

## 2 Preliminaries

We consider an $n$-player simultaneous-move repeated game progressing from time 0 to an end time unknown to the players. At each round $t$, a player $j$ commits to an *action* (e.g., price) $p_{j,t} \in \mathcal{P}$, where $\mathcal{P}$ denotes a discrete action space.

The game admits a *Stage-Game Nash Equilibrium* (SGNE) in which no player can profitably deviate in a single round, given that stage games are independent across periods. Let $r_j^N$ denote player $j$'s *competitive payoff*, $\bar{r}^N$ the average competitive payoff, and $p_j^N$ the *competitive action* of player $j$ under the SGNE. In pricing settings with homogeneous goods and symmetric costs, this corresponds to the *Bertrand Nash equilibrium*.

Players may nonetheless achieve higher payoffs through coordinated behavior that avoids the SGNE. The optimal joint outcome for all players in a stage game can be derived by maximizing the joint payoffs; we refer to this outcome as *the monopoly* which characterizes full collusion. In pricing contexts, such outcomes are detrimental to consumers. Let $r_j^M$ denote player $j$'s *monopoly payoff*, $\bar{r}^M$ the average monopoly payoff, and $p_j^M$ the associated *monopoly action* of player $j$.

The repeated pricing game can be modeled as an extensive-form game whose central solution concept is the *subgame perfect equilibrium* (SPE)—a strategy profile that constitutes a Nash equilibrium in every subgame. According to the Folk Theorem [24], many SPEs exist, with each supported by *credible threats*, i.e., responses that are rational for the player making them [48]. By contrast, non-credible threats cannot sustain an SPE, and monopoly outcomes often depend on such threats. Indeed, in our experiments, we find no evidence of credible threats among pretrained policy pairs, suggesting that realistic collusion may not rely on them.

Following Calvano et al. [14], we define the Collusion Index (CoI) as a measure of collusion: when CoI $= 0\%$, players are fully competitive by playing SGNE, whereas CoI $= 100\%$ corresponds to full collusion at the monopoly outcome.

**Definition 2.1** (Collusion Index (CoI))**.**

$$CoI \coloneqq \frac{\bar{r} - \bar{r}^N}{\bar{r}^M - \bar{r}^N}, \text{ where } \bar{r} \text{ is the average per-player payoff.}$$

We define a state of the game $s \in \mathcal{S}$ as actions (i.e., prices) of all players in one round, ordered by the player IDs. In this work, we focus on two-player repeated games. We let $S_t$ be the random variable of the state of the game at timestamp $t$. A policy $\pi_{j,t} \in \Pi$ for player $j$ at time $t$ maps the state $S_t = s$ to a probability distribution over actions. Given a discount factor $\gamma$ and a fixed policy profile $(\pi_j, \pi_{-j})$, let $V^{\pi_j|\pi_{-j}}(s)$ denote the expected utility of player $j$ starting from state $s$.

**Definition 2.2** (State-Value Function)**.** *Suppose player $j$ interacts with an opponent following a fixed policy $\pi_{-j}$. The state-value function under $\pi_j$ is defined as*

$$\begin{aligned} V^{\pi_j|\pi_{-j}}(s) &= \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k r_j(S_{t+k}) \middle| S_t = s, \pi_j, \pi_{-j}\right] \\ &= r_j(s) + \gamma \sum_{p_j \in \mathcal{P}} \sum_{p_{-j} \in \mathcal{P}} \pi_j(s, p_j)\pi_{-j}(s, p_{-j}) V^{\pi_j|\pi_{-j}}((p_j, p_{-j})). \end{aligned}$$

If state $(p_j, p_{-j})$ is an absorbing state, i.e., $\pi_j((p_j, p_{-j}), p_j) = 1$ and $\pi_{-j}((p_j, p_{-j}), p_{-j}) = 1$, then $V^{\pi_j|\pi_{-j}}((p_j, p_{-j})) = \frac{r_j((p_j, p_{-j}))}{1-\gamma}$.

# 3 Meta-game Design for Algorithmic Collusion

This section introduces our meta-game design to reason about how players select and adapt their policies in "test-time" environments with limited interactions, and assess whether rational choices can lead to collusive outcomes.

## 3.1 Initial Policy, Strategy, and Meta-strategy

### 3.1.1 Initial Policy

As in many multi-agent domains, agents begin play with a pretrained policy derived from prior experience, we consider agents that start with an initial policy generated via a pretraining phase. We model this by applying a learning algorithm $\mathcal{A}_\theta$ and a set of random seeds $\mathcal{K}$ to a game of interest $\mathcal{G}$ to generate a set of initial policy profiles $\Pi$. We assume each pretraining process, i.e., a stochastic procedure that produces a policy profile $\pi_\kappa = \{\pi_1^\kappa, ..., \pi_N^\kappa\} = \mathcal{A}_\theta(\mathcal{G}, \kappa)$, is subject to no time constraint or cost of learning.

### 3.1.2 Strategy

Without adaptation, however, pretrained policies can fail against unfamiliar opponents, resulting in unexpected state transitions and suboptimal outcomes at test time.[1] Achieving strong performance thus requires an update procedure to adapt the initial policy to the specific opponent during play. We define an agent's strategy for the game at test time $\mathcal{G}^*$ as the combination of a (pretrained) initial policy and an update procedure. This strategy is evaluated over a shorter interaction horizon, where learning incurs costs and performance at each round counts.

Rather than directly modifying the policy, we consider strategies that operate on an underlying *internal representation* that encodes the policy. We denote this representation by $Z_{j,t}$ for player $j$ at time $t$, and it is updated after each round. For example, in Q-learning, the internal representation consists of Q-values, from which the agent selects the action with the highest value.

Formally, a strategy is specified by two functions, typically determined by the learning algorithm and its hyperparameters:

[1] Indeed, in experiments, we find that two independently pretrained Q-learning agents, when paired against each other without adaptation, can yield payoffs lower than competitive pricing (see Fig. 4 in the appendix).

- A *decoding function* $\phi$ which maps the internal representation to a policy, $\phi(Z_{j,t}) = \pi_{j,t}$, and
- An *update function* $\omega$ which updates the representation based on the latest experience (i.e., the state), $\omega(Z_{j,t}, S_{j,t}) = Z_{j,t+1}$.

At any time $t$, a player's strategy is fully specified by its current representation $Z_{j,t}$ and its decoding and update functions $(\phi_j, \omega_j)$. In Section 4, we will illustrate how different algorithms, including Q-learning, UCB, and LLMs, instantiate these components.

#### 3.1.3 Meta-strategy

Pretraining can generate a large set of initial policies, each of which may be paired with infinitely many update procedures (e.g., varying learning rates). As a result, the induced strategy space becomes prohibitively large, rendering direct and systematic analysis intractable. To manage this complexity, we group initial policies according to their performance along two key strategic dimensions: their ability to achieve cooperation (i.e., how effectively a policy learns to cooperate with a specific partner) and their ability to avoid exploitation (i.e., how robustly the policy performs against a best-response opponent). We begin by introducing metrics to evaluate these dimensions.

**Definition 3.1** (Paired Cooperativeness (PC))**.** *Let $\pi_j$ and $\pi_{-j}$ denote a pair of policies. The paired cooperativeness between $\pi_j$ and $\pi_{-j}$ is defined as the respective mean state values when interacting with one another, i.e.,*

$$PC(\pi_j, \pi_{-j}) = \left(\overline{V}^{\pi_j|\pi_{-j}}, \overline{V}^{\pi_{-j}|\pi_j}\right), \text{ where } \overline{V}^{\pi_x|\pi_y} = \frac{\sum_{s\in\mathcal{S}} V^{\pi_x|\pi_y}(s)}{|\mathcal{S}|}.$$

*Paired Cooperativeness* (PC) measures the expected utility of each policy in a pair when the initial state is drawn uniformly at random. Intuitively, jointly pretrained policy pairs tend to exhibit high PC values, with positively correlated mean state values, as their trajectories are trained to converge to collusive outcomes from any initial state. In contrast, independently pretrained policies paired at test time are less likely to reach collusive absorbing states, leading to lower PC values and increasing the likelihood of one policy exploiting the other.

With PC capturing how well two policies sustain cooperation, we further introduce *cooperative robustness* (CR), which evaluates a policy's performance against its best response *and* its tendency to cooperate with that best response.

**Definition 3.2** (Cooperative Robustness (CR))**.** *Let $\pi_b$ denote the worst-case best-response policy to $\pi_j$.*

$$\pi_b = \arg\min_{\pi\in BR(\pi_j)} \overline{V}^{\pi_j|\pi}, \text{ where } BR(\pi_j) \coloneqq \arg\max_{\pi\in\Pi} V^{\pi|\pi_j}(s), \forall s \in \mathcal{S}.$$

*The best responses can be obtained via value iteration [49]. The cooperative robustness between $\pi_j$ and $\pi_b$ is defined as their respective mean state values when interacting with each other:*

$$CR(\pi_j, \pi_b) = \left(\overline{V}^{\pi_j|\pi_b}, \overline{V}^{\pi_b|\pi_j}\right).$$

Intuitively, the relative magnitudes of $\overline{V}^{\pi_j|\pi_b}$ and $\overline{V}^{\pi_b|\pi_j}$ characterize the strategic nature of the interaction between the two policies. When these values are highly asymmetric, it indicates an exploitative relationship, typically with $\pi_b$ exploiting $\pi_j$, which reflects $\pi_j$'s lack of robustness. When both values are low, the interaction tends to settle in a competitive absorbing state, indicating robustness but limited cooperative gains. In contrast, when both values are high and comparable, the two policies exhibit robust cooperation, achieving mutually beneficial outcomes while remaining stable against unilateral deviations.[2]

We characterize pretrained initial policies along these two strategic dimensions and group them into distinct categories. Policies within each category are then paired with a discrete set of update procedures. We define a *meta-strategy* as a family of strategies formed by combining an initial policy category—defined by specific strategic attributes (e.g., in PC and CR)—with a corresponding update rule. Practically, each

[2]See Fig. 5 in the appendix for examples of Q-learning policies in different strategic categories.

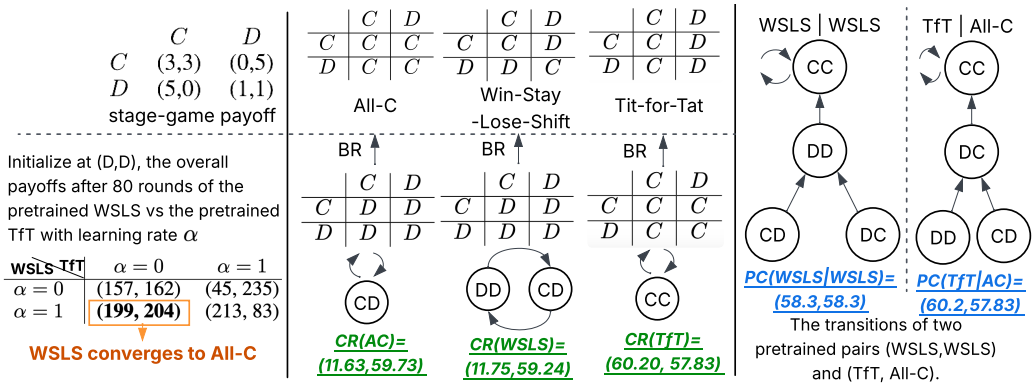



Figure 1: A toy meta-game for repeated Prisoner's Dilemma on canonical strategies. The NE of this meta-game leads to cooperation. See Example 3.1 for more discussion.

meta-strategy specifies how an agent *selects and adapts* an initial policy from the pool of candidates generated during pretraining.

Below we provide a motivating example based on canonical strategies in the repeated Prisoner's Dilemma, coupled with simple update rules, as illustrated in Fig. 1.

**Example 3.1** (A toy meta-game for repeated Prisoner's Dilemma)**.** *Consider the following initial policies for repeated Prisoner's Dilemma: Win-Stay, Lose-Shift (`WSLS`), Tit-for-Tat (`TfT`), and Always-Cooperate (`AC`). Suppose they are obtained from two pretrained policy pairs that successfully learned to cooperate, i.e., (`WSLS`, `WSLS`) and (`TfT`, `AC`).*

*We categorize these policies. All policies achieve high PC with their respective pretrained partners due to cooperative behavior. However, `WSLS` and `AC` both yield low mean state values against their best responses, resulting in highly asymmetric CR, as they can be exploited by a best-response policy (e.g., Always-Defect). By contrast, `TfT` attains high CR, as its best response is to always cooperate.*

*Consider four meta-strategies constructed by pairing high-PC–low-CR and high-PC–high-CR initial policies with either slow or fast in-game update rules. Suppose `WSLS` and `TfT` are sampled as representative initial policies from the two categories and are paired with low and high learning rates. Fig. 1 presents one part of the empirical meta-game defined over this restricted, sampled strategy space. Simulated profile payoffs indicate a meta-game equilibrium in which `WSLS` updates its policy and `TfT` remains unchanged.*[3]

*We are interested in sampling and aggregating many such payoffs to construct empirical meta-games and analyze the relative performance of meta-strategies. For instance, in a different meta-game, `AC` could be sampled as an alternative initial policy from the high-PC–low-CR category.*

## 3.2 Empirical Game-theoretic Analysis

Our meta-game evaluation leverages empirical game-theoretic analysis (EGTA), a methodology for analyzing complex game situations and strategic interactions through agent-based simulation [58, 59]. EGTA constructs an empirical normal-form game over a selected set of strategies using payoff data generated from simulations of the underlying environment.

We apply EGTA to a repeated two-player, general-sum game (e.g., repeated pricing or auction) to investigate algorithmic collusion. Given a set of $M$ meta-strategies, our goal is to evaluate their relative performance in the test-time, base game $\mathcal{G}^*$. Let $\hat{\psi}^m$ denote a strategy sampled from meta-strategy $\mathcal{M}^m$ and let $\hat{\Psi} = \{\hat{\psi}^1, \ldots, \hat{\psi}^M\}$ represent the set of all sampled strategies. We then construct an empirical meta-game $\mathcal{MG}(\hat{\Psi})$ over this sampled strategy space.

When the base game $\mathcal{G}^*$ is symmetric (e.g., agents have the same cost and quality parameters), we estimate the meta-game payoff function by simulating two-player strategy profiles over $\hat{\Psi}$ to play $\mathcal{G}^*$ (due to symmetry,

[3] Q-learning is adopted for both pretraining and test-time adaptation with $\gamma = 0.95$. We normalize initial Q-values following the procedure described in Appendix A.2 and set exploration $\varepsilon = 0$. We evaluate strategies at $t = 80$, when both strategies converged.

the assignment of strategies to players does not matter).[4] The empirical meta-game payoff matrix is estimated by repeatedly simulating $\mathcal{G}^*$ for each two-player strategy profile over different initial states sampled from $\mathcal{S}$ across runs. When the base game $\mathcal{G}^*$ is asymmetric, agents are pretrained on different parameters and may possess distinct strategies. We therefore sample $\hat{\Psi}_1 = \{\hat{\psi}_1^1, \ldots, \hat{\psi}_1^M\}$ and $\hat{\Psi}_2 = \{\hat{\psi}_2^1, \ldots, \hat{\psi}_2^M\}$ from their respective meta-strategy sets to construct the empirical game.

### 3.3 A Meta-game Evaluation Framework

Given a two-player, general-sum repeated game $\mathcal{G}^*$ and meta-strategies $\{\mathcal{M}^1, ..., \mathcal{M}^M\}$, we construct empirical meta-games via the following sampling and evaluation procedure in Algorithm 1.

**Algorithm 1:** Meta-game evaluation procedure

**Input :** Meta-strategy set $\Psi$, base game $\mathcal{G}^*$, number of meta iterations $N_{\text{meta}}$, number of base game runs $N_{\text{base}}$

**Output :** Empirical meta-games and game-analysis stats $X$

**repeat**
  Construct a strategy set $\hat{\Psi} = \{\hat{\psi}^1, \ldots, \hat{\psi}^M\}$ by uniformly sampling one strategy from each meta-strategy in $\Psi$.
  **repeat**
    Sample an initial state $s_0 \sim \mathcal{S}$ for the base game $\mathcal{G}^*$.
    Estimate the empirical $\mathcal{MG}(\hat{\Psi})$ by simulating two-player profiles as described in Section 3.2, and record the resulting profile payoffs.
    Compute the desired statistics $X$ from $\mathcal{MG}(\hat{\Psi})$.
  **until** $N_{base}$;
**until** $N_{meta}$;

**Game-analysis statistics.** We construct *weighted best-response graphs* using the payoff matrices of meta-games generated by Algorithm 1. For each meta-game, the *best-response score* from strategy $u$ to $v$ is defined as the ratio of $u$'s average payoff against $v$ to the highest average payoff achievable by any strategy against $v$. The best-response scores across all meta-games are aggregated to determine the edge weights. If $\mathcal{G}^*$ is asymmetric, we maintain two directed best-response graphs, one for each player role.

We evaluate the *uniform score* of a meta-strategy, i.e., the expected payoff of a meta-strategy when the opponent is drawn from a uniform distribution over all meta-strategies. As uniform scores represent a naive belief of the opponent distribution, we also adopt an alternative metric, NE-regret, as proposed by Jordan et al. [33].

**Definition 3.3** (NE-Regret)**.** *Suppose $\sigma^*$ is a symmetric mixed-strategy NE of $\mathcal{MG}(\Psi)$. The NE-regret of a pure meta-strategy $\mathcal{M}_j$ is defined as the difference between the expected payoff of player $j$ of the equilibrium profile ($\sigma^*$, $\sigma^*$) and that of the mixed-strategy profile ($\mathcal{M}_j$, $\sigma^*$).*

A high NE-regret might be due to an inability to cooperate or susceptibility to exploitation. Since a meta-game may admit multiple MSNEs with varying degrees of cooperativeness, we follow Balduzzi et al. [8]'s *Nash averaging* and report NE-regret with respect to the max-entropy NE, which captures the broadest set of meta-strategies.

## 4 Evaluating Algorithmic Collusion in Repeated Pricing Games

We first introduce the repeated pricing game, and then conduct the meta-game evaluation on three common pricing algorithms: Q-learning, UCB, and LLMs.

[4]To prevent collusion arising from both players using an identical or jointly pretrained policy, if the same policy or its pretrained pair has been drawn, we redraw an instance.

## 4.1 Economic Environment

We consider the canonical repeated pricing game in which firms (i.e., agents) act simultaneously and condition their actions on history. For the stage game, we adopt a simple model of price competition with *logit demand*, which has been widely applied [41, 14, 12]. The details of the logit demand model is provided below.

**Definition 4.1** (Logit Demand Model)**.** *Consider $n$ differentiated products and an outside good. In period $t$, the demand for firm $j$'s product is*

$$d_{j,t} \coloneqq \frac{e^{\frac{a_j - p_{j,t}}{\mu}}}{\sum_{k=1}^{n} e^{\frac{a_k - p_{k,t}}{\mu}} + e^{\frac{a_0}{\mu}}}, \tag{1}$$

*where $p_{j,t}$ is the price of product $j$ in period $t$, and $a_j$ is a product quality index. Product 0 represents the outside good, with $a_0$ serving as the inverse index of aggregate demand. The parameter $\mu > 0$ controls the degree of horizontal differentiation, where $\mu \to 0$ corresponds to perfect substitutes.*

We focus on a two-agent repeated pricing game as our base game $\mathcal{G}^*$. Each agent has private information about its marginal cost $c_j$ and product quality $a_j$, both of which remain fixed throughout $\mathcal{G}^*$.[5] An agent can observe its demand $d_{j,t}$ and both firms' prices. The game state of period $t$ is $S_t \coloneqq (p_{j,t}, p_{-j,t})$. The profit (i.e., payoff) for agent $j$ at period $t$ is $r_j(S_t) \coloneqq (p_{j,t} - c_j) \cdot d_{j,t}(S_t)$. Following Calvano et al. [14], prices are discretized and equally spaced, forming the agent's action space. We consider the discretization levels of 4 and 15. For a discretization of $N_{\text{discrete}}$, let the step size be $\xi = \frac{p_j^M - p_j^N}{N_{\text{discrete}} - 2}$ where $p_j^N$ is the competitive price, $p_j^M$ the monopoly price. The action space is $\mathcal{P}_j \coloneqq \left\{p_j^N + (k-1)\xi \mid k \in \{0, 1, \ldots, N_{\text{discrete}} - 1\}\right\}$.

### 4.1.1 Meta-strategies

The base game $\mathcal{G}^*$ takes as input a pair of pricing strategies and returns the corresponding payoffs for each agent. We can evaluate cumulative payoffs at any point during $\mathcal{G}^*$, capturing the effect of varying test-time horizon and the cost of learning.

We use three algorithms—Q-learning, UCB, and LLM—to generate pretrained initial policies. Formally, each algorithm together with its associated hyperparameters defines a stochastic procedure that produces a policy profile $(\pi_j, \pi_j') = \mathcal{A}_\theta(\mathcal{G}, \kappa)$, given a random seed $\kappa$. As described in Section 3.1, we classify pretrained policies according to their paired cooperativeness (PC; Def. 3.1) with their pretraining partners and their cooperative robustness (CR; Def. 3.2).[6] From the pretrained pool, we select three representative categories (illustrated in Appendix Fig. 5):

1. Less colluding (LC): Policies in the bottom third of $\overline{V}^{\pi_j|\pi_j'}$ and the middle third of $\overline{V}^{\pi_j|\pi_b}$, representing policies that achieve little collusion with their pretrained partner upon convergence. With a price space discretized to four levels, these typically correspond to competitive pricing policies.

2. Colluding (C): Policies in the top third of $\overline{V}^{\pi_j|\pi_j'}$ and the bottom third of $\overline{V}^{\pi_j|\pi_b}$, indicating collusion with pretrained partners but vulnerability to best-response exploitation.

3. Robust colluding (RC): Policies in the top third for both $\overline{V}^{\pi_j|\pi_j'}$ and $\overline{V}^{\pi_j|\pi_b}$.

Tertiles are computed within each pretraining algorithm. We also consider a baseline category of randomly initialized (RD) policies. Each initial policy category is paired with a set of algorithm-specific update rules spanning a range of adaptation speeds, from fast to slow. We discuss Q-learning in Sec. 4.2, UCB in Sec. 4.3 and LLM in Sec. 4.4. For all meta-game evaluations, additional results including the payoff matrices and the best-response graphs are referenced in Appendix A.6 for Q-learning, Appendix B.2 for UCB, and Appendix C.3 for LLM.

[5] We consider symmetric product qualities $a_j = a_{-j} = 2$, while allowing for cost asymmetries in certain settings.

[6] Given the strong correlation between the two dimensions of PC and CR (see Appendix Fig. 6), we base the categorization on a single dimension.

## 4.2 Tabular Q-learning for Pricing

Tabular Q-learning is a simple yet effective algorithm widely adopted in pricing settings [54, 14]. The corresponding decoding and update rules are as follows.

**Definition 4.2** (Q-decoding Rule $\phi(Z_t)$)**.**

$$\phi(Z_t) = \begin{cases} \arg\max_{p_j \in \mathcal{P}_j} Z_t(s, p_j) & w.p.\ 1-\varepsilon \\ p_j \sim \mathit{Unif}(\mathcal{P}_j) & \mathit{otherwise.} \end{cases} \quad \mathit{for\ each}\ s \in \mathcal{S}. \tag{2}$$

**Definition 4.3** (Q-learning Update Rule $\omega(Z_t, S_t)$)**.**

$$Z_{t+1}(S_t, p_{j,t}) = Z_t(S_t, p_{j,t}) + \alpha(r_j(S_t) + \gamma \max_{p_j \in \mathcal{P}_j} Z_t(S_t, p_j) - Z_t(S_t, p_{j,t})). \tag{3}$$

**Pretraining.** Our pretraining procedure follows Calvano et al. [14]. We randomize the learning rate $\alpha$, exploration rate $\varepsilon$, and its decay $\delta$, while fixing the discount factor $\gamma = 0.95$. Cost and quality parameters are kept identical to agents' test-time values. All pretrained pairs use symmetric settings and play until their policies converge. We run 500 pretraining games with different random seeds, yielding 1,000 pretrained policies.

**Initialization of Q-values.** Initial Q-values influence exploration and are a key determinant of test-time adaptation [32, 46]. Since pretrained Q-values can vary widely across state–action pairs due to randomness and exploration during training, we rescale them to standardize their maximum value while preserving the induced policy and the ordering of actions within each state. This rescaling allows for comparisons across policies by controlling for differences in Q-value magnitudes. We introduce a rescaling factor $f$ that interpolates between optimistic ($f = 1$) and pessimistic ($f = 0$) initializations, corresponding to converged Q-values at collusive and competitive absorbing states, respectively. Details of the rescaling procedure are provided in Appendix A.2. Section 4.2.3 presents meta-game evaluations on meta-strategies of different scaling factors.

**Game settings.** For the base repeated pricing games, we examine two symmetric cost settings ($c_1 = c_2 = 1$ and $c_1 = c_2 = 0.8$) and one asymmetric setting ($c_1 = 1, c_2 = 0.8$), with $N_{\text{discrete}} = 15, N_{\text{base}} = 100$ and $N_{\text{meta}} = 40$. We evaluate a set of ten meta-strategies (Table 1), spanning combinations of initial policy categories and learning rates $\alpha \in \{0.5, 0.05, 0.005\}$. Other meta-strategies—such as random initialization with smaller learning rates or pretrained policies with larger ones—were excluded based on preliminary experiments, as they were strictly dominated. We further explore the rescaling factor as a meta-strategy parameter for $f = \{1, 0.5, 0\}$.

### 4.2.1 Symmetric Costs

Table 1 and Fig. 3 present our main findings for symmetric base games, evaluated at $t = 10,000$.[8] We identify a PSNE at (C, 0.5) and a max-entropy MSNE involving three meta-strategies: (C, 0.5), (RC, 0.05), and (RC, 0.005).

Fig. 2 reports the average CoI performance of paired strategies sampled from the respective meta-strategies over 2,000 runs. With symmetric costs of one, collusion emerges as a rational adaptive outcome, driven by agents' strategic choices among the available meta-strategies: strategies from the PSNE achieve approximately 70% CoI, while those from the max-entropy MSNE reach around 50%. Furthermore, we highlight that when jointly pretrained or identically initialized random policies are paired, CoI can exceed 80% with minimal or

[7] Uniform scores in all tables are converted to the CoI scale based on the following transformation: $x \to \frac{x-\bar{r}^N}{\bar{r}^M-\bar{r}^N} \times 100\%$, where $x$ is the expected payoff of a meta-strategy in a stage game when the opponent's strategy is drawn from a uniform distribution over all meta-strategies. We denote $\bar{r}^N$ as the average competitive payoff and $\bar{r}^M$ the average monopoly payoff (Sec. 2).

[8] Based on Amazon's 15-minute pricing updates, $t = 3,000$ is approximately one month, and $t = 10,000$ approximates three months. https://sell.amazon.com/tools/automate-pricing

Table 1: Max-entropy MSNE, NE-regret ($\times 10^{-3}$), and the uniform score (converted to the CoI scale) evaluated at $t = 10,000$ for Q-learning with optimistic initialization $f = 1$. These metrics are defined in Sec. 3.3.[7]The best-performing strategy is highlighted in **bold**. The second-best is underlined. The $\pm$ symbol denotes the 95% confidence interval.

| $t = 10,000,\ f = 1$ | | RD 0.5 | LC 0.5 | LC 0.05 | LC 0.005 | C 0.5 | C 0.05 | C 0.005 | RC 0.5 | RC 0.05 | RC 0.005 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| PSNE | $c_1 = c_2 = 1$ | - | - | - | - | ✓ | - | - | - | - | - |
| MSNE | $c_1 = c_2 = 1$ | 0.00 | 0.00 | 0.00 | 0.00 | 0.49 | 0.00 | 0.00 | 0.10 | 0.28 | 0.13 |
| NE-Regret ($\times 10^{-3}$) | $c_1 = c_2 = 1$ | 8.10 ± 0.41 | 1.05 ± 0.43 | 7.09 ± 0.56 | 12.76 ± 0.68 | **0.00 ± 0.37** | 5.60 ± 0.53 | 10.56 ± 0.57 | **0.00 ± 0.49** | **0.00 ± 0.51** | **0.00 ± 0.59** |
| Uniform Score | $c_1 = c_2 = 1$ | 35.21 ± 0.32 | 39.44 ± 0.29 | 33.08 ± 0.35 | 28.01 ± 0.43 | **40.96 ± 0.22** | 36.10 ± 0.22 | 32.64 ± 0.32 | 39.66 ± 0.25 | 37.90 ± 0.24 | 37.52 ± 0.29 |
| PSNE | $c_1 = c_2 = 0.8$ | - | - | - | ✓ | - | - | - | - | - | - |
| MSNE | $c_1 = c_2 = 0.8$ | 0.00 | 0.00 | 0.00 | 0.23 | 0.17 | 0.00 | 0.08 | 0.00 | 0.00 | 0.53 |
| NE-Regret ($\times 10^{-3}$) | $c_1 = c_2 = 0.8$ | 7.72 ± 0.53 | 2.94 ± 0.64 | 4.34 ± 0.54 | **0.00 ± 0.50** | **0.00 ± 0.59** | 6.78 ± 0.43 | **0.00 ± 0.41** | 7.97 ± 0.69 | 11.20 ± 0.47 | **0.00 ± 0.42** |
| Uniform Score | $c_1 = c_2 = 0.8$ | 29.26 ± 0.25 | 32.84 ± 0.26 | 32.06 ± 0.21 | 32.01 ± 0.21 | **35.14 ± 0.19** | 31.69 ± 0.18 | 30.15 ± 0.22 | 32.03 ± 0.25 | 33.21 ± 0.20 | 34.83 ± 0.20 |
| MSNE | $c_1 = 1.0$ | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.24 | 0.00 | 0.00 | 0.76 |
| | $c_2 = 0.8$ | 0.00 | 0.09 | 0.00 | 0.91 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| NE-Regret ($\times 10^{-3}$) | $c_1 = 1.0$ | 14.25 ± 0.85 | 13.94 ± 0.92 | 7.69 ± 0.77 | 3.38 ± 0.56 | 18.58 ± 0.75 | 8.61 ± 0.55 | **0.00 ± 0.50** | 13.88 ± 0.84 | 8.12 ± 0.48 | **0.00 ± 0.36** |
| | $c_2 = 0.8$ | 18.47 ± 0.67 | **0.00 ± 0.75** | 0.26 ± 0.54 | **0.00 ± 0.50** | 14.90 ± 0.57 | 25.48 ± 0.38 | 24.21 ± 0.45 | 1.88 ± 0.75 | 17.65 ± 0.58 | 17.63 ± 0.57 |
| Uniform Score | $c_1 = 1.0$ | 8.54 ± 0.44 | **10.10 ± 0.47** | 2.01 ± 0.42 | 2.57 ± 0.40 | 6.67 ± 0.43 | 1.24 ± 0.29 | 3.11 ± 0.24 | **10.00 ± 0.53** | 5.03 ± 0.47 | 7.66 ± 0.28 |
| | $c_2 = 0.8$ | 36.61 ± 0.33 | 61.73 ± 0.32 | 62.22 ± 0.33 | **63.26 ± 0.39** | 48.25 ± 0.24 | 43.12 ± 0.29 | 39.56 ± 0.37 | 55.85 ± 0.26 | 52.77 ± 0.24 | 49.81 ± 0.25 |

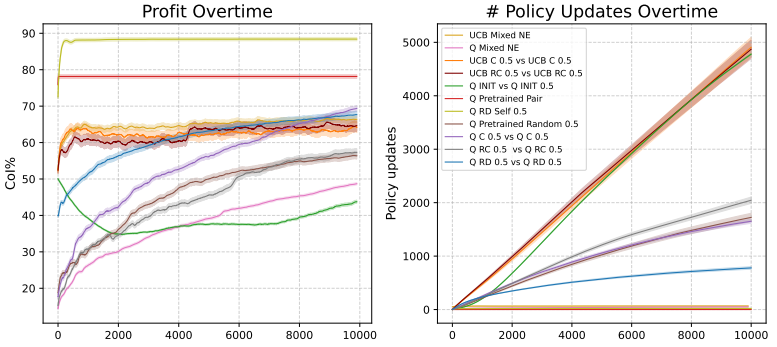



Figure 2: Running averages of CoI over 100 rounds (Left) and accumulated policy update counts for strategy pairs over 10,000 rounds (Right). Shaded regions indicate 95% confidence intervals. Each curve represents the mean over 20 strategy pairs and 100 random initial states. INIT denotes Q-learning agents trained from scratch, with Q-values initialized to those corresponding to opponents playing uniformly random pricing strategies. All strategies except INIT and pretrained pairs use $\alpha = 0.5$ and $f = 1$.

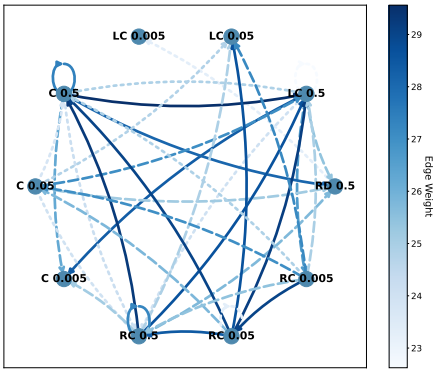



Figure 3: The best-response graph for Q-learning with $c_1 = c_2 = 1$, evaluated at $t = 10,000$. The edge weights correspond to the sum of best-response scores across all meta-games as discussed in Sec. 3.3.

no adaptation, indicating that prior coordination can greatly amplify collusive behavior. These adaptation patterns may serve as useful signals for identifying potential prior coordination.

Fig. 3 illustrates the strategic relationship among meta-strategies. Within the RC family, strategies with smaller learning rates tend to dominate those with larger ones, suggesting the value of preserving a robust initial policy at test time. In contrast, C meta-strategies benefit from larger learning rates, which enable faster adaptation and reduce vulnerability to exploitation. We also note that while (RD, 0.5) achieves moderate collusion (around 65% CoI as shown in Fig. 2) when matched against itself, it is easily exploited by most pretrained strategies, rendering it strategically dominated.

**Lower symmetric costs, shorter horizon, and rescaling factors.** We examine how shorter evaluation horizons $t = 3,000$ (Table 9), lower symmetric costs (Table 1), and different Q-value rescaling factors (Table 2) affect the rational selection of meta-strategies. Their payoff matrices and BR graphs are in Appendix A.6.

Overall, we observe a shift toward RC strategies in the Nash equilibrium of the corresponding meta-games. Intuitively, a shorter horizon leaves less time for adaptation, incentivizing agents to begin with stronger, more robust strategies that are less vulnerable to exploitation. A similar shift toward RC strategies is observed under smaller rescaling factors. When Q-values are initialized pessimistically (i.e., $f = 0.5, 0$), reflecting a prior belief that the opponent is less likely to collude, agents are less inclined to adapt toward cooperation, favoring robust strategies that perform well from the outset without relying on further learning.

Perhaps surprisingly, collusion declines when both agents face lower costs ($c_1 = c_2 = 0.8$), despite greater potential for profits. This can be attributed to stronger incentives to undercut, as lower costs make exploitation through price reductions more attractive. This is further verified in agents' meta-strategy choices, which place a higher probability ($> 0.7$) on LC and RC strategies to maintain robustness against exploitation. Consequently, CoI levels are lower than the $c = 1$ setting when evaluated at $t = 10,000$.

#### 4.2.2 Asymmetric Costs

Table 1 presents the main results of the meta-game evaluation performed on repeated pricing with asymmetric costs ($c_1 = 1, c_2 = 0.8$). The corresponding payoff matrices and best-response graphs are provided in Appendix A.6. The max-entropy MSNE reveals that the low-cost agent chooses strategies from the LC family and the high-cost agent selects from the C and RC families. The low-cost agent enjoys a natural cost advantage and thus adopts LC strategies to play competitively with minimal adaptation. In response, the high-cost player selects RC strategies to remain robust against exploitation while attempting to sustain some level of collusion. Overall, collusion no longer persists, as the low-cost player has stronger incentives and greater opportunities to deviate and exploit.

We note that our findings here differ from those of Calvano et al. [14], who report sustained high collusion (75.9% CoI) under even more asymmetric scenarios ($\frac{c_1}{c_2} = 2$) with symmetric algorithmic configurations. Our finding suggests that incorporating rationality into strategy selection and deployment decisions may suppress collusive behavior in asymmetric settings.

Our meta-game analysis assumes that each player knows the opponent's cost type. In practice, agents may hold beliefs over the opponent's type, influencing their strategic choices. Extending the framework to a Bayesian Nash equilibrium would better capture behavior under incomplete information.

#### 4.2.3 Asymmetric Rescaling: A Meta-game on Pessimistic vs. Optimistic Initialization

We construct a meta-game where the set of meta-strategies includes a mixture of rescaling factors, $f \in \{0.5, 1.0\}$.[9] Due to the large strategy space, we focus on competitive meta-strategies that appear in the equilibria of their respective single-rescaling-factor meta-games. The base game uses symmetric costs of one, evaluated at $t = 10,000$.

Table 2 lists the meta-strategies and summarizes the main findings. The max-entropy MSNE consists of (RC, 0.5, $f0.5$) and (RC, 0.05, $f0.5$). We note that although (C, 0.5, $f0.5$) is best responded to by (C, 0.5, $f1$), yielding high collusion, (C, 0.5, $f1$) is more vulnerable to exploitation by RC strategies with $f = 0.5$. As a result, (C, 0.5, $f1$) achieves a high uniform score but also suffers high regret. Overall, CoI levels are lower when strategies with pessimistic initializations are included.

In effect, initialization can be interpreted as reflecting both an agent's adaptability and its prior beliefs about the opponent's behavior. These results highlight how the optimal choice of initialization and meta-strategy depends on beliefs about the opponent. If one expects its opponent to view cooperation as viable (e.g., optimistic initialization on both sides), adopting an optimistic initialization with a high learning rate is advantageous, leading to cooperative outcomes such as the PSNE (C, 0.5, $f1$). Conversely, if one anticipates an exploitative opponent, a more robust approach—pessimistic initialization and/or a low learning rate—is preferable to guard against exploitation.

We additionally construct a meta-game over C and RC strategies with a finer grid of learning rates to verify that the collusive outcome is robust to perturbations in $\alpha$ and does not rely on coordination of the hyperparameter choices. Further details can be found in Appendix A.4.

### 4.3 UCB for Pricing

The Upper Confidence Bound (UCB) algorithm guarantees bounded regret in multi-armed bandit problems. For the repeated pricing game, we extend UCB to a state-dependent variant, enabling history-dependent policies that can better handle a complex environment. The internal representation $Z$ consists of two components: the cumulative reward $r_t(s, p_j)$ and the visit count $\text{count}_t(s, p_j)$ for each state $s \in \mathcal{S}$ and action (i.e., price) $p_j \in \mathcal{P}_j$.

[9] $f = 0$ is excluded due to the very limited number of updates it induces.

Table 2: Meta-game evaluation of meta-strategies with different Q-value initializations, including $f = \{1, 0.5\}$. Payoffs are evaluated at $t = 10,000$.

| $t = 10,000$ | $f = 1$ | | | | | | $f = 0.5$ | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Meta-strategies | C 0.5 | C 0.05 | C 0.005 | RC 0.5 | RC 0.05 | RC 0.005 | C 0.5 | C 0.05 | C 0.005 | RC 0.5 | RC 0.05 | RC 0.005 |
| PSNE | - | - | - | - | - | - | - | - | - | ✓ | ✓ | - |
| MSNE | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.63 | 0.37 | 0.00 |
| NE-Regret ($\times 10^{-3}$) | 10.20 ± 1.78 | 16.77 ± 1.31 | 16.88 ± 1.26 | 3.95 ± 1.67 | 8.84 ± 1.19 | 4.39 ± 1.37 | 4.03 ± 1.49 | 6.73 ± 1.53 | 13.58 ± 1.56 | **0.00 ± 1.20** | **0.00 ± 1.41** | 4.48 ± 1.40 |
| Uniform Score | **32.41 ± 0.81** | 21.14 ± 0.89 | 18.68 ± 0.90 | 31.90 ± 0.72 | 26.35 ± 0.62 | 26.76 ± 0.62 | 28.29 ± 0.87 | 24.63 ± 0.82 | 19.18 ± 0.84 | 31.47 ± 0.64 | 29.76 ± 0.66 | 27.04 ± 0.76 |

Table 3: Max-entropy MSNE, NE-regret ($\times 10^{-3}$), and uniform score (converted to CoI scale) among **UCB meta-strategies** at $t = 10,000$.

| $t = 10,000$ | LC 1 | C 1 | RC 1 | LC 0.5 | C 0.5 | RC 0.5 | LC 0.05 | C 0.05 | RC 0.05 | LC 0.005 | C 0.005 | RC 0.005 | RD 0.005 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PSNE | - | - | - | - | ✓ | ✓ | - | - | - | - | - | - | - |
| MSNE | 0.00 | 0.00 | 0.00 | 0.00 | 0.50 | 0.50 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| NE-Regret ($\times 10^{-3}$) | 13.83 ± 0.93 | 1.13 ± 1.34 | 0.63 ± 1.24 | 13.93 ± 0.88 | **0.00 ± 1.32** | **0.00 ± 1.21** | 15.13 ± 0.94 | 2.29 ± 1.36 | 2.11 ± 1.30 | 19.67 ± 0.98 | 5.46 ± 1.42 | 6.30 ± 1.36 | 39.66 ± 0.17 |
| Uniform Score | 55.23 ± 0.63 | 62.68 ± 0.82 | 61.66 ± 0.77 | 55.59 ± 0.62 | **63.14 ± 0.81** | 62.02 ± 0.76 | 54.32 ± 0.64 | 61.56 ± 0.83 | 59.43 ± 0.79 | 50.38 ± 0.66 | 58.19 ± 0.86 | 55.97 ± 0.83 | 35.35 ± 0.13 |

**Pretraining.** As UCB emphasizes early exploration and requires a long time for convergence, pretraining is crucial for obtaining reliable state–action count and reward estimates for test-time adaptation. During pretraining, similar to the sliding-window UCB [26], we cap visit counts at 5,000 per state-action pair to prevent extreme or imbalanced values that could inhibit policy updates. We generate 1,000 pretrained policies with 500 random seeds.

**Meta-strategies.** We introduce a discount factor $\alpha \in [0, 1]$ on test-time visits and rewards in the update rule to control the intensity of policy updates. An alternative approach is to discount historical rewards and counts [25, 26]. However, in the pricing setting, we find that this induces excessive random exploration, typically leading to dominated test-time performance. The UCB decoding and update rules are provided below.

**Definition 4.4** (UCB Decoding Rule $\phi(Z_t)$)**.**

$$\phi(Z_t) = \arg\max_{p_j \in \mathcal{P}_j} UCB_{j,t}(s, p_j) \text{ for all } s \in \mathcal{S} \text{ where} \tag{4}$$

$$UCB_{j,t}(s, p_j) = \frac{1}{count_t(s, p_j)} r_t(s, p_j) + \sqrt{\frac{2 \ln count_t(s)}{count_t(s, p_j)}} \tag{5}$$

$$\text{and } count_t(s) = \sum_{p_j \in \mathcal{P}_j} count_t(s, p_j) \tag{6}$$

**Definition 4.5** (UCB Update Rule $\omega(Z_t, S_t)$)**.**

$$count_{t+1}(S_t, p_{j,t}) = count_t(S_t, p_{j,t}) + \alpha; \tag{7}$$

$$r_{t+1}(S_t, p_{j,t}) = r_t(S_t, p_{j,t}) + \alpha r_j(S_t) \tag{8}$$

For the meta-game, we consider $\alpha \in \{1, 0.5, 0.05, 0.005\}$ and also include (RD, 0.005), where the initial cumulative rewards $r_0(\cdot, \cdot)$ are drawn uniformly at random between the minimum and maximum stage-game profits, and the counts are sampled uniformly between 0 and 5,000. Fig. 7 and Fig. 19 in the appendix show the distribution of pretrained policies in both metrics, PC and CR.

### 4.3.1 Meta-game Evaluation

Table 3 presents the main findings for symmetric base games with $c_1 = c_2 = 1$, evaluated at $t = 10,000$ with $N_{\text{discrete}} = 15, N_{\text{base}} = 100$ and $N_{\text{meta}} = 40$. The CoIs over time of the pure and mixed-strategy NEs are provided in Fig. 2.

Overall, UCB achieves higher CoI levels than Q-learning under the same game settings. Strategies in the C and RC families tend to sustain cooperative outcomes, thus consistently outperforming LC across all $\alpha$ values in terms of NE-regret and uniform score. When Q-learning with random initialization, denoted as (Q-RD, 0.5), is included as a meta-strategy in UCB's meta-games, it emerges as part of both the MSNE

and PSNE (see Table 14 and Fig. 22 in the appendix), despite being strategically dominated by most other meta-strategies in Q-learning's meta-games. These results reflect that even though UCB-pretrained policies are generally collusive, they are less robust than those from Q-learning (also see Fig. 7 in the appendix).

### 4.4 LLMs for Pricing

For LLMs, we adopt prompts similar to those in Fish et al. [23], augmented with additional strategy hints that serve as adaptation parameters. Each prompt consists of a constant and a variable component: the constant part specifies the adaptation strategy, while the variable part maintains a history of previous game states, the model's plan, and insight from the preceding round. This evolving variable part constitutes the internal representation $Z$, which is updated over time. The decoding and update functions, $\phi$ and $\omega$, are determined jointly by the LLM and the constant portion of the prompt. Further details on prompt design are provided in Appendix C.2.

LLMs can be deployed at test time directly without pretraining, as in Fish et al. [23]. However, greater control over initial policies can be achieved by providing (simulated) interaction histories as part of the initial state context, placing this approach within the paradigm of in-context learning.

**LLM pretraining.** The goal of pretraining is to prepare an initial representation that encodes the initial policy to be used at test time. This representation includes historical interactions, the model's plan, and insight from the last pretraining round. Overall, the initial representation is jointly determined by the pretraining prompts, the paired opponent, and the historical interactions. During *pretraining*, the opponent's policy or characteristics are also described in the prompt to facilitate (pretraining) adaptation. The remaining prompt mirrors that used at test time. The amount of pretraining history included in the prompt is important, as LLM tends to place greater weight on recent rounds. Pretraining is terminated when the state remains unchanged for ten consecutive rounds, at which point the full history is included.

We consider three types of paired opponents during pretraining, giving four types of initial representations:

- *h0*: no history,
- *h1*: history of self-play against a symmetric LLM,
- *h2*: history of play against a pretrained Q-policy from the RC family as specified in Fig. 5, and
- *h3*: history of play against Tit-for-Tat, which also falls under the RC family in this setting.

**LLM at test time.** During inference, we keep the pretraining interaction history but remove the plan and insights, as they contain information specific to the pretraining opponent. An *adaptation strategy prompt* is then appended at test time to complete the inference prompt. We consider the following variants:

- *p0*: no strategy specified.
- *p1*: "Your co-participant may aim to learn an approximately best responding strategy to yours. Make sure your strategy achieves high profit even for the best responding strategies."
- *p2*: "One adaptation strategy is to try predicting the current strategy your co-participant uses and then update your strategy to approximately best respond to your co-participant."

*p1* is designed to elicit robust policies with low update rates, whereas *p2* promotes adaptive behavior aimed at exploitation or cooperation. Broadly, the two meta-strategy dimensions can be interpreted as the choice of strategic prompt and the selection of historical information to include.

**Recovering LLM's initial policy.** The LLM's policy is shaped by both the constant and variable components of the prompt. For probing scalability, we consider the state as the price pair from one round, even though the prompt itself retains the full interaction history. To estimate the initial policy, we query the LLM 16 times per state using the same prompt. We use the resulting empirical price distributions as an approximation of the policy. The best response is then computed through value iteration [49].

Table 4: Evaluation scores among LLM strategies evaluated at $t = 50$.

| | *p2h3* | *p0h0* | *p1h2* | *p2h1* | *p1h1* | *p0h2* |
|---|---|---|---|---|---|---|
| PSNE | ✓ | - | - | - | - | ✓ |
| MSNE | 0.44 | 0.00 | 0.00 | 0.00 | 0.00 | 0.56 |
| NE-Regret ($\times 10^{-3}$) | **0.00 ± 7.92** | 58.76 ± 7.40 | <u>47.38</u> ± 9.13 | 50.37 ± 9.28 | 59.47 ± 6.14 | **0.00 ± 9.32** |
| Uniform Score | **36.79 ± 4.86** | 10.84 ± 2.95 | 15.08 ± 3.83 | 14.54 ± 3.57 | 10.80 ± 2.57 | <u>25.55 ± 5.44</u> |

**Meta-strategies.** We consider 24 meta-strategies in total, covering combinations of two LLM models (GPT5-mini and GPT5-nano), four types of interaction histories, and three distinct inference-time prompts. Fig. 23 in the appendix shows the locations of initial policies of these meta-strategies in the PC-CR space.

### 4.4.1 Meta-game Evaluation

We focus on strategies generated by GPT5-mini, since GPT5-nano mostly converges immediately to the competitive price regardless of the paired opponent. Specifically, we select six meta-strategies that roughly span LC (*p0h0*, *p1h2* and *p2h1*), C (*p2h3*, *p1h1*), and RC (*p0h2*). Table 4 and Appendix Fig. 25 summarize main findings for the symmetric repeated pricing game with $c_1 = c_2 = 1$, evaluated at $t = 50$ with $N_{\text{discrete}} = 4, N_{\text{base}} = 40$.[10] Prices are discretized into four levels due to API costs, and initial states are uniformly sampled for the base games.

Interestingly, we find that *p2h3* and *p0h2* emerge as two pure-strategy Nash equilibria in the meta-game. The average payoffs of *p2h3* and *p0h2*, when playing against themselves and each other, are close to the payoffs under perfect collusion, suggesting that these strategies sustain cooperation, possibly due to pretraining histories that converge to collusion against policies in the RC family under Q-learning.

In contrast, *p0h0* plays the competitive price unless initialized in a collusive state.[11] Similar to the Q-learning comparison between (C, 0.5) and (RC, 0.005), although *p0h0* can exploit *p2h3*, such exploitation does not constitute a Nash equilibrium, since cooperation remains a profitable deviation. Taken together, these results suggest that collusion can emerge and persist as a stable outcome among rational LLM-based agents within the selected strategy set.

LLM-generated insights frequently emphasize keywords such as "cooperation", "trigger", and "punishment", and most strategies resemble Grim Trigger behavior, consistent with Fish et al. [23]'s observation that LLMs follow a steep reward–punishment scheme. As with Grim Trigger, interactions often settle into consistent competition once cooperation breaks down.

However, a novel pattern we observe is that certain strategies, most notably *p2h3*, can re-establish cooperation even after extended periods of competitive interaction. This recovery is opponent-dependent: when paired with *p0h2* or *p1h1*, *p2h3* gradually shifts play from competition back toward cooperation by persistently attempting collusive pricing. Unlike *p2h3*, *p0h2* quickly reverts to competitive pricing upon defection, e.g., when playing against *p1h1*, consequently underperforms relative to *p2h3* in these matchups. These observations suggest that, despite their recency bias, LLMs can draw on deeper historical context, even from pretraining, to strategically restore cooperation.

# 5 Discussion

We developed a meta-game framework to evaluate whether collusion can emerge as a realistic outcome among strategically rational agents operating under test-time constraints. Each meta-strategy specifies an agent's choice of an initial policy family and in-game adaptation rule, allowing us to analyze strategic behavior beyond symmetric hyperparameter assumptions (a form of pre-game coordination).

Our results show that algorithmic collusion can persist in equilibrium among meta-strategies within Q-learning, UCB, and LLMs, suggesting that collusion may emerge from rational agents even in the absence

[10] We observe that all policies stop updating after about 30 rounds.

[11] Note that our results differ from Fish et al. [23] in that *p0h0* does not exhibit collusion for most initial states, even though the prompt design is similar. We were unable to replicate their findings due to model deprecations; we instead employ newer models (GPT5-mini, GPT5-nano, Gemini 2.5/2.0 flash/-lite).

of explicit communication. However, the extent of collusion depends on agents' beliefs about their opponents, e.g., pessimistic beliefs tend to yield low-collusion equilibria, whereas optimistic beliefs promote cooperative outcomes.

Natural extensions of our framework include modeling heterogeneous beliefs on the opponent's cost or representation initialization and analyzing through Bayesian Nash equilibrium, as well as broadening meta-strategy coverage, particularly for LLMs with richer prompt designs and pretraining histories. Additionally, cross-algorithm meta-games could shed further light on how algorithmic diversity affects the emergence and stability of collusion.

## Acknowledgments

We thank the anonymous reviewers for constructive feedback. This work was supported in part by Rutgers SAS Research Grant in Academic Themes.

# References


[1] Ibrahim Abada and Xavier Lambin. Artificial intelligence: Can seemingly collusive outcomes be avoided? *Management Science*, 69(9):5042–5065, 2023.

[2] Ibrahim Abada, Joseph E. Harrington Jr, Xavier Lambin, and Janusz M Meylahn. Algorithmic collusion: Where are we and where should we be going?, 2024. URL `http://dx.doi.org/10.2139/ssrn.4891033`.

[3] Ibrahim Abada, Xavier Lambin, and Nikolay Tchakarov. Collusion by mistake: Does algorithmic sophistication drive supra-competitive profits? *European Journal of Operational Research*, 318(3): 927–953, 2024.

[4] Marina Agranov and Leeat Yariv. Collusion through communication in auctions. *Games and Economic Behavior*, 107:93–108, 2015.

[5] Kushal Agrawal, Verona Teo, Juan J. Vazquez, Sudarsh Kunnavakkam, Vishak Srikanth, and Andy Liu. Evaluating LLM agent collusion in double auctions, 2025. URL `https://arxiv.org/abs/2507.01413`.

[6] Eshwar Ram Arunachaleswaran, Natalie Collina, Sampath Kannan, Aaron Roth, and Juba Ziani. Algorithmic collusion without threats. *arXiv:2409.03956*, 2022.

[7] Stephanie Assad, Robert Clark, Daniel Ershov, and Lei Xu. Algorithmic pricing and competition: Empirical evidence from the german retail gasoline market. *Journal of Political Economy*, 132(3): 723–771, 2024.

[8] David Balduzzi, Karl Tuyls, Julien Perolat, and Thore Graepel. Re-evaluating evaluation. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

[9] Martino Banchio and Andrzej Skrzypacz. Artificial intelligence and auction design. In *Proceedings of the 23rd ACM Conference on Economics and Computation*, pages 30–31, 2022.

[10] Wolfram Barfuss and Janusz M Meylahn. Intrinsic fluctuations of reinforcement learning promote cooperation. *Scientific reports*, 13(1) 1309, 2023.

[11] Martin Bichler, Julius Durmann, and Matthias Oberlechner. Online optimization algorithms in repeated price competition: Equilibrium learning and algorithmic collusion, 2024. URL `https://arxiv.org/abs/2412.15707`.

[12] Gianluca Brero, Eric Mibuari, Nicolas Lepore, and David C. Parkes. Learning to mitigate AI collusion on economic platforms. In *Advances in Neural Information Processing Systems*, volume 35, pages 37892–37904, 2022.

[13] David P. Byrne and Nicolas de Roos. Learning to coordinate: A study in retail gasoline. *American Economic Review*, 109(2):591–619, February 2019.

[14] Emilio Calvano, Giacomo Calzolari, Vincenzo Denicolò, and Sergio Pastorello. Artificial intelligence, algorithmic pricing, and collusion. *American Economic Review*, 110(10):3267–97, October 2020.

[15] Cesare Carissimo, Fryderyk Falniowski, Siavash Rahimi, and Heinrich Nax. Algorithmic collusion is algorithm orchestration, 2025. URL `https://arxiv.org/abs/2508.14766`.

[16] Peter Cramton and Jesse A Schwartz. Collusive bidding: Lessons from the FCC spectrum auctions. *Journal of regulatory Economics*, 17(3):229–252, 2000.

[17] Jacob W. Crandall. Towards minimizing disappointment in repeated games. *Journal of Artificial Intelligence Research*, 49:111–142, 2014.

[18] Jacob W. Crandall and Michael A. Goodrich. Learning to compete, coordinate, and cooperate in repeated games using reinforcement learning. *Machine Learning*, 82:281–314, 2010.

[19] Anthony DiGiovanni and Ambuj Tewari. Balancing adaptability and non-exploitability in repeated games. In *Proceedings of the 38th Conference on Uncertainty in Artificial Intelligence*, pages 559–568, 2022.

[20] Nicolas Eschenbaum, Filip Mellgren, and Philipp Zahn. Robust algorithmic collusion, 2022. URL `https://arxiv.org/abs/2201.00345`.

[21] Joshua Estelle, Yevgeniy Vorobeychik, Michael P. Wellman, Satinder Singh, Christopher Kiekintveld, and Vishal Soni. Strategic interactions in the tac 2003 supply chain tournament. In *Proceedings of the 4th International Conference on Computers and Games*, page 316–331, 2004.

[22] Ariel Ezrachi and Maurice E. Stucke. Algorithmic collusion: Problems and counter-measures. 2015.

[23] Sara Fish, Yannai A. Gonczarowski, and Ran Shorrer. Algorithmic collusion by large language models, 2024. URL `https://arxiv.org/abs/2404.00806`.

[24] James W Friedman. A non-cooperative equilibrium for supergames. *The Review of Economic Studies*, 38(1):1–12, 1971.

[25] Aurélien Garivier and Eric Moulines. On upper-confidence bound policies for switching bandit problems. In *Algorithmic Learning Theory*, pages 174–188. Springer Berlin Heidelberg, 2011.

[26] Aurélien Garivier and Eric Moulines. On upper-confidence bound policies for non-stationary bandit problems, 2008. URL `https://arxiv.org/abs/0805.3415`.

[27] Lewis Hammond, Alan Chan, Jesse Clifton, Jason Hoelscher-Obermaier, Akbir Khan, Euan McLean, Chandler Smith, Wolfram Barfuss, Jakob Foerster, Tomáš Gavenčiak, The Anh Han, Edward Hughes, Vojtěch Kovařík, Jan Kulveit, Joel Z. Leibo, Caspar Oesterheld, Christian Schroeder de Witt, Nisarg Shah, Michael Wellman, Paolo Bova, Theodor Cimpeanu, Carson Ezell, Quentin Feuillade-Montixi, Matija Franklin, Esben Kran, Igor Krawczuk, Max Lamparth, Niklas Lauffer, Alexander Meinke, Sumeet Motwani, Anka Reuel, Vincent Conitzer, Michael Dennis, Iason Gabriel, Adam Gleave, Gillian Hadfield, Nika Haghtalab, Atoosa Kasirzadeh, Sébastien Krier, Kate Larson, Joel Lehman, David C. Parkes, Georgios Piliouras, and Iyad Rahwan. Multi-agent risks from advanced AI, 2025. URL `https://arxiv.org/abs/2502.14143`.

[28] Karsten T. Hansen, Kanishka Misra, and Mallesh M. Pai. Frontiers: Algorithmic collusion: Supra-competitive prices via independent algorithms. *Marketing Science*, 40(1):1–12, 2021.

[29] Jason D. Hartline, Sheng Long, and Chenhao Zhang. Regulation of algorithmic collusion. In *Proceedings of the 2024 Symposium on Computer Science and Law*, pages 98–108, 2024.

[30] Jason D. Hartline, Chang Wang, and Chenhao Zhang. Regulation of algorithmic collusion, refined: Testing pessimistic calibrated regret. In *Proceedings of the 2025 Symposium on Computer Science and Law*, pages 108–120, 2025.

[31] Matthias Hettich. Algorithmic collusion: Insights from deep learning. 2021. URL `https://ssrn.com/abstract=3785966`.

[32] Chi Jin, Zeyuan Allen-Zhu, Sebastien Bubeck, and Michael I Jordan. Is q-learning provably efficient? In *Advances in Neural Information Processing Systems*, volume 31, 2018.

[33] Patrick R. Jordan, Christopher Kiekintveld, and Michael P. Wellman. Empirical game-theoretic analysis of the tac supply chain game. In *Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems*, 2007.

[34] Christopher Kiekintveld and Michael P. Wellman. Selecting strategies using empirical game models: an experimental analysis of meta-strategies. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 2*, page 1095–1101, 2008.

[35] Timo Klein. Autonomous algorithmic collusion: Q-learning under sequential pricing. *The RAND Journal of Economics*, 52(3):538–558, 2021.

[36] Rohit Lamba and Sergey Zhuk. Pricing with algorithms, 2022. URL `https://arxiv.org/abs/2205.04661`.

[37] Xavier Lambin. Less than meets the eye: Simultaneous experiments as a source of algorithmic seeming collusion. *SSRN Electronic Journal*, 01 2023. doi: 10.2139/ssrn.4498926.

[38] Zun Li and Michael P. Wellman. A meta-game evaluation framework for deep multiagent reinforcement learning. In *Proceedings of the 33rd International Joint Conference on Artificial Intelligence*, pages 148–156, 2024.

[39] Viliam Lisý and Michael Bowling. Eqilibrium approximation quality of current no-limit poker bots. In *The AAAI-17 Workshop on Computer Poker and Imperfect Information Games*, 2017.

[40] Carlos Martin and Tuomas Sandholm. Approxed: Approximate exploitability descent via learned best responses, 2024. URL `https://arxiv.org/abs/2301.08830`.

[41] Daniel McFadden. Conditional logit analysis of qualitative choice behavior. In *Fontiers in Econometrics*, pages 105–142. Academic press, 1974.

[42] J. M. Meylahn and L. Janssen. Limiting dynamics for q-learning with memory one in symmetric two-player, two-action games. *Complexity*, 2022(1):483–491, 2022.

[43] Leon Musolff. Algorithmic pricing facilitates tacit collusion: Evidence from e-commerce. In *Proceedings of the 23rd ACM Conference on Economics and Computation*, pages 32–33. ACM, 2022.

[44] OECD. Algorithmic competition, oecd competition policy roundtable background note. 2023. URL `www.oecd.org/daf/competition/algorithmic-competition-2023.pdf`.

[45] S. Phelps, M. Marcinkiewicz, and S. Parsons. A novel method for automatic strategy acquisition in n-player non-zero-sum games. In *Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems*, page 705–712, 2006.

[46] Tabish Rashid, Bei Peng, Wendelin Boehmer, and Shimon Whiteson. Optimistic exploration even with a pessimistic initialisation. In *International Conference on Learning Representations*, 2020.

[47] Daniel M. Reeves. Generating trading agent strategies: Analytic and empirical methods for infinite and large games. *PhD thesis, University of Michigan*, 2005.

[48] Thomas C. Schelling. An essay on bargaining. *The American Economic Review*, 46(3):281–306, 1956.

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

[50] Finbarr Timbers, Nolan Bard, Edward Lockhart, Marc Lanctot, Martin Schmid, Neil Burch, Julian Schrittwieser, Thomas Hubert, and Michael Bowling. Approximate exploitability: Learning a best response. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 3487–3493, 2022.

[51] Yuki Usui and Masahiko Ueda. Symmetric equilibrium of multi-agentreinforcement learning in repeated prisoner's dilemma. *Applied Mathematics and Computation*, 409:126370, 2021.

[52] Yevgeniy Vorobeychik, Christopher Kiekintveld, and Michael P. Wellman. Empirical mechanism design: methods, with application to a supply-chain scenario. In *Proceedings of the 7th ACM Conference on Electronic Commerce*, page 306–315, 2006.

[53] Elaine Wah and Michael P. Wellman. Latency arbitrage in fragmented markets: A strategic agent-based analysis. *Algorithmic Finance*, 5:69–93, 2016.

[54] Ludo Waltman and Uzay Kaymak. Q-learning agents in a cournot oligopoly model. *Journal of Economic Dynamics and Control*, 32(10):3275–3293, 2008.

[55] Xintong Wang and Michael P. Wellman. Spoofing the limit order book: An agent-based model. In *Proceedings of 16th International Conference on Autonomous Agents and Multiagent Systems*, pages 651–659, 2017.

[56] Xintong Wang, Yevgeniy Vorobeychik, and Michael P. Wellman. A cloaking mechanism to mitigate market manipulation. In *Proceedings of 27th International Joint Conference on Artificial Intelligence*, pages 541–547, 2018.

[57] Xintong Wang, Christopher Hoang, and Michael P. Wellman. Learning-based trading strategies in the face of market manipulation. In *Proceedings of First ACM International Conference on AI in Finance*, pages 1–8, 2020.

[58] Michael P. Wellman. Methods for empirical game-theoretic analysis. In *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 2*, AAAI'06, page 1552–1555. AAAI Press, 2006. ISBN 9781577352815.

[59] Michael P. Wellman, Karl Tuyls, and Amy Greenwald. Empirical game-theoretic analysis: A survey. *Journal of Artificial Intelligence Research*, 82, 2025. URL `https://doi.org/10.1613/jair.1.16146`.

# A Deferred Content and Results from Section 4.2 on Q-learning

## A.1 Test-time Payoff Comparison: Joint vs. Independent Pretraining

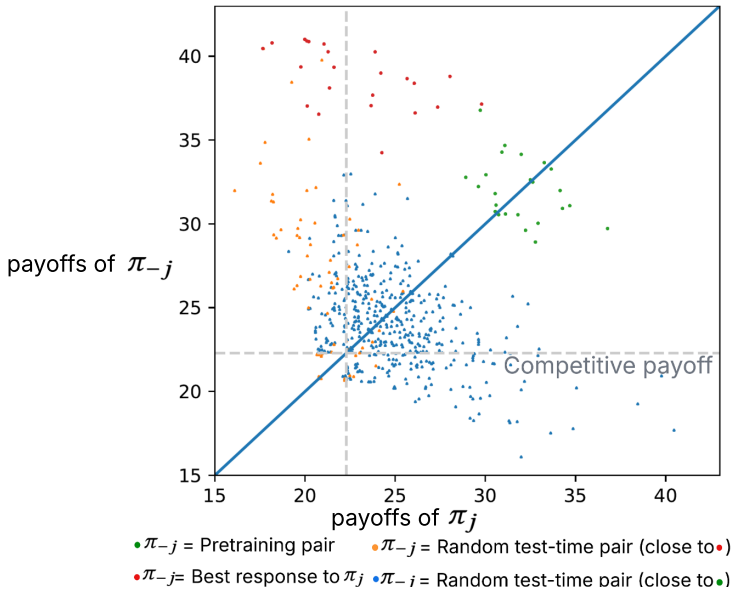



Figure 4: We collect 12 pairs of pretrained Q-learning policies with 15 discretized actions and use them as initial policies for test-time evaluation. We measure total payoffs from $t = 0$ to $t = 100$, averaged over 100 random seeds. In *green*, each policy is paired with its original pretraining pair. In *red*, each pretrained policy ($\pi_j$) is paired with its best response ($\pi_{-j}$). In *orange* and *blue*, we show payoffs for randomly sampled, independently pretrained pairs without adaptation: *orange* points lie closer (in L2 distance) to the *red* cluster, while *blue* points are closer to the *green*.

The majority of randomly paired policies achieve payoffs below the competitive benchmark (i.e., when both policies play the competitive price). This suggests that while pretrained policies can achieve high-level collusion, pairing them with an independently pretrained policy disrupts their ability to collude instantly.

## A.2 Normalization of Q-values

For Q-learning, the initial policy is encoded in the initial Q-values. However, Q-values accumulated during pretraining can be irregular due to the variable number of visits and the number of iterations required for convergence. To allow a reliable comparison, we apply the following normalization. Let $\tilde{Z}_j$ denote the Q-values accumulated during pretraining for player $j$, and denote the converged policies pair as $(\pi_j, \pi'_j)$.

$$Z_j(s,p) \coloneqq Q^N + f \cdot (Q^M - Q^N) - \max_{s^*} V^{\pi_j|\pi'_j}(s^*) + \min_{p'} Q^{\pi_j|\pi'_j}(s,p') \tag{9}$$

$$+ \left(\tilde{Z}_j(s,p) - \min_{p'} \tilde{Z}_j(s,p')\right) \times \frac{V^{\pi_j|\pi'_j}(s) - \min_{p'} Q^{\pi_j|\pi'_j}(s,p')}{\max_{p'} \tilde{Z}_j(s,p') - \min_{p'} \tilde{Z}_j(s,p')} \tag{10}$$

where $Q^M = \bar{r}^M/(1-\gamma)$ and $Q^N = \bar{r}^N/(1-\gamma)$ represent the theoretical discounted expected returns associated with absorbing into the perfectly colluding and competitive states, respectively.

After normalization, the original argmax of $\tilde{Z}_j$ is preserved: $Z_j(s, \pi_j(s)) = V^{\pi_j|\pi'_j}(s) + Q^N + f \cdot (Q^M - Q^N) - \max_{s^*} V^{\pi_j|\pi'_j}(s^*)$, and the minimum satisfies $\min_{p'} Z_j(s,p) = \min_{p'} Q^{\pi_j|\pi'_j}(s,p') + Q^N + f \cdot (Q^M - Q^N) - \max_{s^*} V^{\pi_j|\pi'_j}(s^*)$. The maximum Q-value (at the argmax action) maps to the true value function $V^{\pi_j|\pi'_j}(s)$ and the minimum Q-value maps to $\min_{p'} Q^{\pi_j|\pi'_j}(s,p')$ with an offset $Q^N + f \cdot (Q^M - Q^N) - \max_{s^*} V^{\pi_j|\pi'_j}(s^*)$.

The scaling factor $f$ is set to 1 in main experiments, and we vary $f$ for further analysis in Sec. 4. This normalization ensures that Q-values across all meta-strategies share a common scale, so that a given learning

rate induces comparable adaptation speeds. When $f$ is large, the Q-values are optimistically initialized, which is a well-known technique for promoting efficient exploration [32, 46].

## A.3 Categorizing Q-learning Pretrained Policies via PC and CR

Fig. 5 Left shows $\overline{V}^{\pi_j,\pi'_j}$ v.s. $\overline{V}^{\pi_j,\pi_b}$ along with the LC, C and RC categorization for $N_{\text{discrete}} = 4$. Fig. 5 Right illustrates three representative policies in each category. The top policy (RC) sustains collusion with its pretraining partner and remains collusive even when faced with a best-responding opponent, the middle policy (LC) is resistant to exploitation but defaults to competitive pricing rather than colluding, and the bottom policy colludes with its pretraining partner but is vulnerable to exploitation (C).

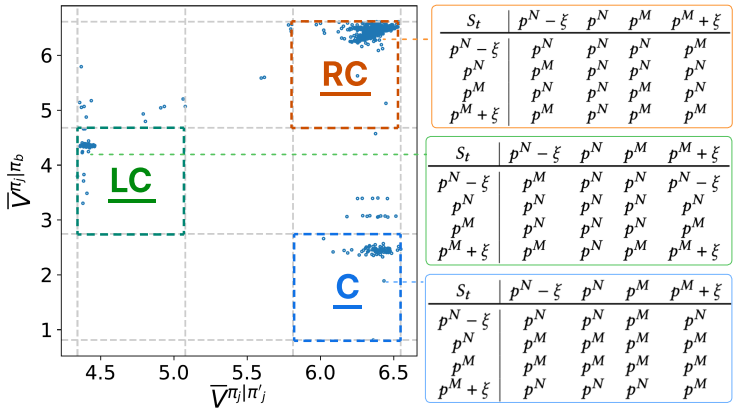



Figure 5: With 4 discretized actions, the 500 Q-learning pretrained policies form three distinct clusters when laid out along PC (Def. 3.1) and CR (Def. 3.2), corresponding to the competitive (LC), naively collusive (C), and robustly collusive (RC) categories introduced in Sec. 4.1.1.

In Fig. 6, we further show the PC and CR of pretrained policies under $N_{\text{discrete}} = 4$ and $N_{\text{discrete}} = 15$.

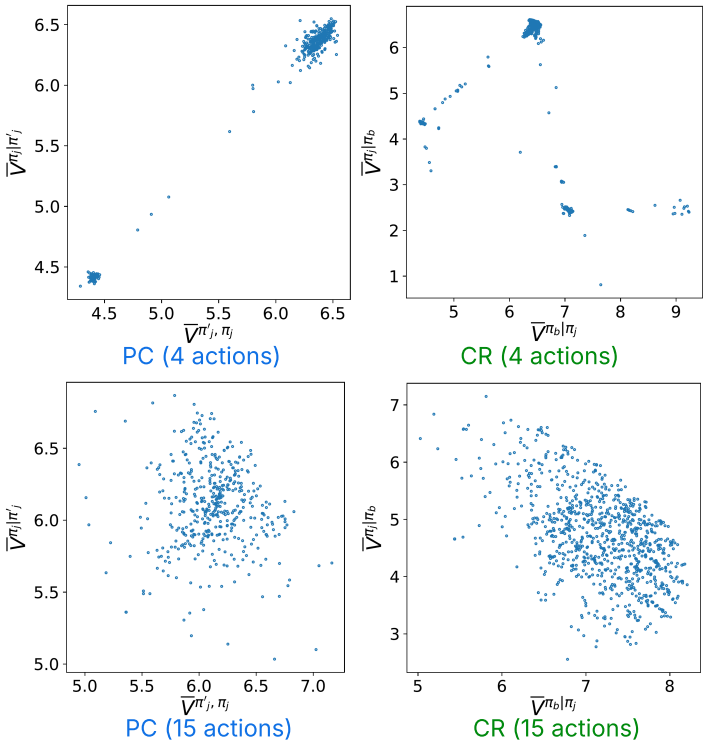



Figure 6: The PC (Def. 3.1) and CR (Def. 3.2) of 1,000 pretrained Q-learning policies with 4 and 15 discretized actions, respectively, under the experiment setting described in Sec. 4.2.

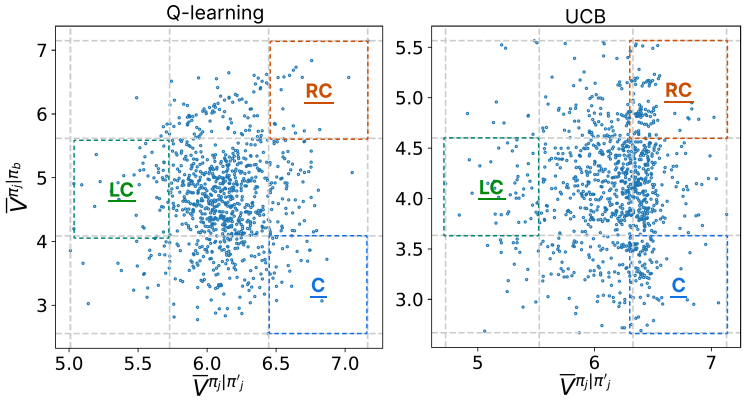



Figure 7: Categorization of pretrained policies with 15 discretized actions obtained via Q-learning (Left) and UCB (Right), under the experiment settings described in Sec. 4.2 and 4.3.

## A.4 Meta-games among Meta-strategies with Fine-grained Learning Rates

In Sec. 4.2, we identified two PSNEs: (C, 0.5) and (RC, 0.5) for Q-learning with symmetric cost $c_1 = c_2 = 1$, optimistic initialization $f = 1$, and time horizon $t = 10,000$. To test the robustness of the equilibrium outcome to learning rate perturbations, we perform additional evaluations on C and RC with a finer grid of learning rates $\{0.3, 0.4, 0.5, 0.6, 0.7\}$. The results are provided in Table 5 and Fig 8.

We observe a similar level of collusion for C with $\alpha \geq 0.4$ (CoI $\geq 50\%$). RC 0.3 and 0.5 form one of the MSNEs and sustain collusion at around 40%. These findings suggest that the collusive outcome is robust to moderate learning rate perturbations. Strategies in C category do not constitute a part of the NE. Nevertheless, the regret of playing C remains relatively low.

Table 5: Robustness test on varying the test-time learning rates for Q-learning pretrained policies. $t = 10,000$.

| $t = 10,000$ | C 0.3 | C 0.4 | C 0.5 | C 0.6 | C 0.7 | RC 0.3 | RC 0.4 | RC 0.5 | RC 0.6 | RC 0.7 |
|---|---|---|---|---|---|---|---|---|---|---|
| PSNE | - | - | - | - | - | ✓ | - | ✓ | - | - |
| MSNE | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.40 | 0.00 | 0.60 | 0.00 | 0.00 |
| NE-Regret ($\times 10^{-3}$) | 2.07 ± 1.73 | 1.88 ± 1.62 | 3.26 ± 1.55 | 2.80 ± 1.56 | 3.55 ± 1.44 | **0.00 ± 1.49** | 0.40 ± 1.40 | **0.00 ± 1.45** | 1.71 ± 1.33 | 2.60 ± 1.38 |
| Uniform Score | 46.38 ± 1.04 | 47.40 ± 0.95 | 47.47 ± 0.90 | 47.55 ± 0.82 | 46.71 ± 0.76 | **49.85 ± 1.01** | **49.20 ± 0.94** | 48.99 ± 0.93 | 48.02 ± 0.87 | 47.77 ± 0.86 |

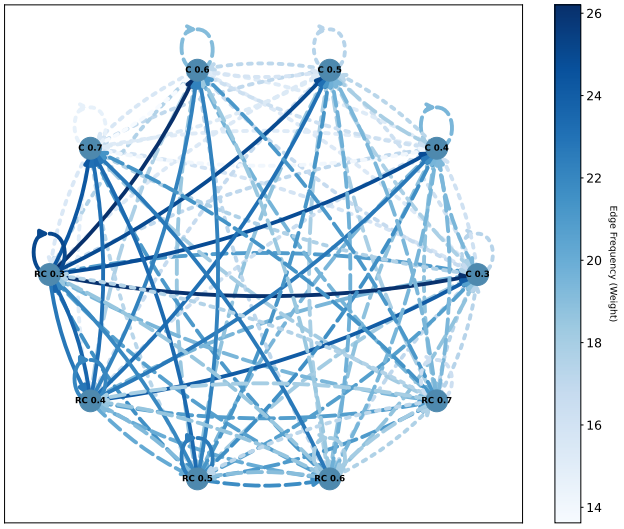



Figure 8: Best-response graph for C and RC with a finer grid of learning rates.

## A.5 Results for Q-learning with Pessimistic Initializations $f = 0.5$ and $f = 0$

Table 6: Meta-games on Q-learning using $f = 0.5$ and $f = 0$ respectively, with a time horizon of $t = 10,000$. In the case of $f = 0$, we additionally incorporate $\varepsilon = 1$ with a decay rate of 0.001 to ensure a policy still updates. The best-performing strategy is highlighted in **bold**, and the second-best is underlined.

| Q-scale | $t = 10,000$ | RD 0.5 | LC 0.5 | LC 0.05 | LC 0.005 | C 0.5 | C 0.05 | C 0.005 | RC 0.5 | RC 0.05 | RC 0.005 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $f = 0.5$ | PSNE | - | ✓ | - | - | ✓ | - | ✓ | - | ✓ | - |
| | MSNE | 0.00 | 0.19 | 0.00 | 0.00 | 0.34 | 0.00 | 0.00 | 0.39 | 0.08 | 0.00 |
| | NE-Regret ($\times 10^{-3}$) | 6.78 ± 1.15 | **0.00 ± 1.14** | 6.24 ± 1.21 | 15.19 ± 1.28 | **0.00 ± 1.15** | 1.90 ± 1.23 | 10.60 ± 1.20 | **0.00 ± 1.04** | **0.00 ± 1.18** | 5.00 ± 1.24 |
| | Uniform Score | 27.68 ± 1.00 | 32.78 ± 0.91 | 28.43 ± 0.94 | 21.08 ± 0.94 | **33.98 ± 0.80** | 30.45 ± 0.78 | 24.56 ± 0.76 | 31.70 ± 0.90 | 29.57 ± 0.87 | 25.97 ± 0.82 |
| $f = 0.0$ | Mixed & PSNE | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 1.00 | 0.00 | 0.00 |
| | NE-Regret ($\times 10^{-3}$) | 7.22 ± 1.29 | 1.91 ± 1.13 | 10.85 ± 1.28 | 11.79 ± 1.49 | 0.54 ± 1.05 | 9.73 ± 1.14 | 9.35 ± 1.22 | **0.00 ± 1.01** | 5.22 ± 1.05 | 5.60 ± 1.06 |
| | Uniform Score | 20.48 ± 0.63 | 26.50 ± 0.56 | 17.62 ± 0.66 | 15.64 ± 0.74 | 26.74 ± 0.49 | 18.79 ± 0.58 | 18.55 ± 0.61 | **28.14 ± 0.57** | 21.92 ± 0.59 | 21.93 ± 0.58 |

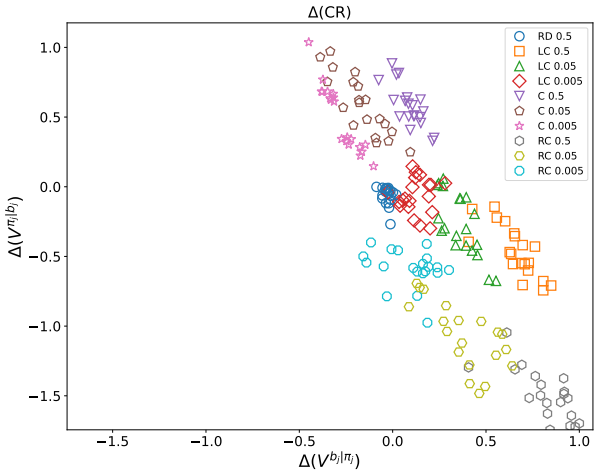
## A.6 Additional Results on Q-learning-based Meta-strategies

For each setting below, we report the best-response graphs, average payoffs, and the changes in CR and PC between the initial policies and the policies at the end of the time horizon, denoted $\Delta_t(\text{CR})$ and $\Delta_t(\text{PC})$ respectively.

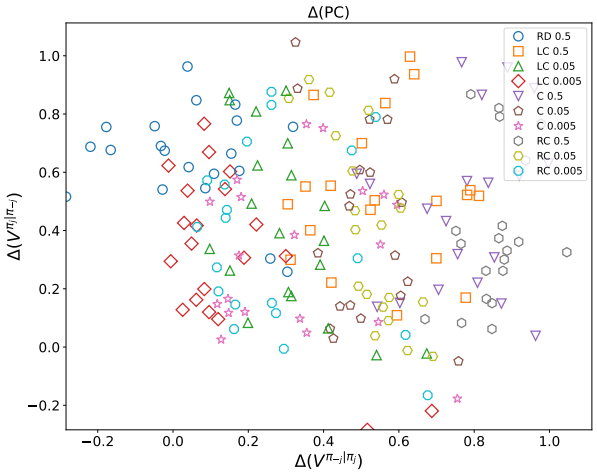
**Symmetric costs $c_1 = c_2 = 1$ with optimistic initialization $f = 1$ and time horizon $t = 10,000$.** The best-response graph, average payoffs, and $\Delta_t(\text{CR})$, $\Delta_t(\text{PC})$ are presented in Fig. 3, Table 7, and Fig. 9.

Table 7: Average payoffs over 4,000 runs in the Q-learning meta-game payoff matrix. Standard errors are omitted as they are negligible in the order of $10^{-4}$. For reference, in this setting, the competitive and monopoly payoffs are $\bar{r}^N = 0.22$ and $\bar{r}^M = 0.34$.

| | RD 0.5 | LC 0.5 | LC 0.05 | LC 0.005 | C 0.5 | C 0.05 | C 0.005 | RC 0.5 | RC 0.05 | RC 0.005 |
|---|---|---|---|---|---|---|---|---|---|---|
| RD 0.5 | 0.29, 0.29 | 0.27, 0.30 | 0.26, 0.29 | 0.27, 0.28 | 0.27, 0.31 | 0.26, 0.31 | 0.26, 0.30 | 0.26, 0.30 | 0.25, 0.30 | 0.25, 0.29 |
| LC 0.5 | 0.30, 0.27 | 0.28, 0.28 | 0.27, 0.28 | 0.27, 0.27 | 0.28, 0.29 | 0.26, 0.28 | 0.27, 0.27 | 0.27, 0.28 | 0.25, 0.29 | 0.25, 0.28 |
| LC 0.05 | 0.29, 0.26 | 0.28, 0.27 | 0.26, 0.26 | 0.26, 0.26 | 0.27, 0.26 | 0.26, 0.27 | 0.25, 0.26 | 0.27, 0.27 | 0.25, 0.27 | 0.24, 0.27 |
| LC 0.005 | 0.28, 0.27 | 0.27, 0.27 | 0.26, 0.26 | 0.26, 0.26 | 0.26, 0.27 | 0.25, 0.26 | 0.24, 0.26 | 0.26, 0.27 | 0.25, 0.26 | 0.24, 0.26 |
| C 0.5 | 0.31, 0.27 | 0.29, 0.28 | 0.26, 0.27 | 0.27, 0.26 | 0.28, 0.28 | 0.26, 0.27 | 0.27, 0.26 | 0.27, 0.28 | 0.24, 0.28 | 0.24, 0.28 |
| C 0.05 | 0.31, 0.26 | 0.28, 0.26 | 0.27, 0.26 | 0.26, 0.25 | 0.27, 0.26 | 0.26, 0.26 | 0.25, 0.26 | 0.27, 0.26 | 0.23, 0.26 | 0.26, 0.26 |
| C 0.005 | 0.30, 0.26 | 0.27, 0.27 | 0.26, 0.25 | 0.26, 0.24 | 0.26, 0.27 | 0.26, 0.25 | 0.24, 0.24 | 0.26, 0.26 | 0.25, 0.25 | 0.23, 0.25 |
| RC 0.5 | 0.30, 0.26 | 0.28, 0.27 | 0.27, 0.27 | 0.27, 0.26 | 0.28, 0.27 | 0.26, 0.27 | 0.26, 0.26 | 0.27, 0.27 | 0.25, 0.27 | 0.24, 0.27 |
| RC 0.05 | 0.30, 0.25 | 0.29, 0.25 | 0.27, 0.25 | 0.26, 0.25 | 0.28, 0.24 | 0.26, 0.25 | 0.25, 0.25 | 0.27, 0.25 | 0.25, 0.25 | 0.24, 0.26 |
| RC 0.005 | 0.29, 0.25 | 0.28, 0.25 | 0.27, 0.24 | 0.26, 0.24 | 0.28, 0.24 | 0.26, 0.23 | 0.25, 0.23 | 0.27, 0.24 | 0.26, 0.24 | 0.24, 0.24 |




(a) Change in CR over test time. As RC strategies begin with already high CR, larger learning rates cause greater deviation from the robust policy, resulting in a decline in CR. In contrast, strategies in C category generally improve their CR over adaptation. RD 0.5 starts with low CR and shows no improvement during test time. LC strategies are initially more robust than the ones in C, and similarly show no improvement in CR during test time.




(b) Change in PC over time. Overall, PC improves over the course of test time. For RD strategies, although their own $V$ improves, their opponent's does not, as the value against RD is already high at initialization. For both LC and RC, larger learning rates tend to benefit their opponents more, even when their own $\Delta(V^{\pi_j|\pi_{-j}})$ is comparable to that under smaller learning rates. This suggests that LC and RC with large learning rates provide opportunities for cooperation.

Figure 9: $\Delta_t(\text{CR})$ and $\Delta_t(\text{PC})$ for Q-learning-based meta-strategies under symmetric costs $c_1 = c_2 = 1$, $f = 1$ and $t = 10,000$.

**Symmetric costs $c_1 = c_2 = 0.8$ with optimistic initialization $f = 1$ and time horizon $t = 10,000$.** The best-response graph is provided in Fig. 10. The average payoffs are given in Table 8. $\Delta_t(\text{CR})$ and $\Delta_t(\text{PC})$ are shown in Fig. 11.

Table 8: Average payoffs over 4,000 runs in the Q-learning meta-game payoff matrix. Standard errors are omitted as they are negligible in the order of $10^{-4}$. For reference, in this setting, the competitive and monopoly payoffs are $\bar{r}^N = 0.24$ and $\bar{r}^M = 0.41$.

| | RD 0.5 | LC 0.5 | LC 0.05 | LC 0.005 | C 0.5 | C 0.05 | C 0.005 | RC 0.5 | RC 0.05 | RC 0.005 |
|---|---|---|---|---|---|---|---|---|---|---|
| RD 0.5 | 0.35, 0.35 | 0.29, 0.34 | 0.28, 0.33 | 0.28, 0.31 | 0.30, 0.35 | 0.29, 0.34 | 0.30, 0.33 | 0.30, 0.37 | 0.26, 0.35 | 0.24, 0.33 |
| LC 0.5 | 0.34, 0.29 | 0.30, 0.30 | 0.29, 0.30 | 0.28, 0.31 | 0.31, 0.31 | 0.28, 0.31 | 0.29, 0.30 | 0.32, 0.30 | 0.28, 0.32 | 0.25, 0.31 |
| LC 0.05 | 0.33, 0.28 | 0.30, 0.29 | 0.30, 0.30 | 0.29, 0.30 | 0.30, 0.29 | 0.28, 0.30 | 0.27, 0.29 | 0.32, 0.28 | 0.29, 0.30 | 0.25, 0.30 |
| LC 0.005 | 0.31, 0.28 | 0.31, 0.28 | 0.30, 0.29 | 0.29, 0.29 | 0.30, 0.29 | 0.28, 0.28 | 0.26, 0.29 | 0.32, 0.27 | 0.30, 0.28 | 0.26, 0.29 |
| C 0.5 | 0.35, 0.30 | 0.31, 0.31 | 0.29, 0.30 | 0.29, 0.30 | 0.31, 0.31 | 0.29, 0.31 | 0.29, 0.30 | 0.33, 0.30 | 0.28, 0.32 | 0.25, 0.32 |
| C 0.05 | 0.34, 0.29 | 0.31, 0.28 | 0.30, 0.28 | 0.28, 0.28 | 0.31, 0.29 | 0.27, 0.27 | 0.26, 0.28 | 0.32, 0.28 | 0.29, 0.27 | 0.25, 0.28 |
| C 0.005 | 0.33, 0.30 | 0.30, 0.29 | 0.29, 0.27 | 0.29, 0.26 | 0.30, 0.29 | 0.28, 0.26 | 0.26, 0.26 | 0.30, 0.30 | 0.29, 0.26 | 0.26, 0.25 |
| RC 0.5 | 0.37, 0.30 | 0.30, 0.32 | 0.28, 0.32 | 0.27, 0.32 | 0.30, 0.33 | 0.28, 0.32 | 0.30, 0.30 | 0.32, 0.32 | 0.26, 0.34 | 0.24, 0.33 |
| RC 0.05 | 0.35, 0.26 | 0.32, 0.28 | 0.30, 0.29 | 0.28, 0.30 | 0.32, 0.28 | 0.27, 0.29 | 0.26, 0.29 | 0.34, 0.26 | 0.29, 0.29 | 0.23, 0.30 |
| RC 0.005 | 0.33, 0.24 | 0.31, 0.25 | 0.30, 0.25 | 0.29, 0.26 | 0.32, 0.25 | 0.28, 0.25 | 0.25, 0.26 | 0.33, 0.24 | 0.30, 0.23 | 0.25, 0.25 |

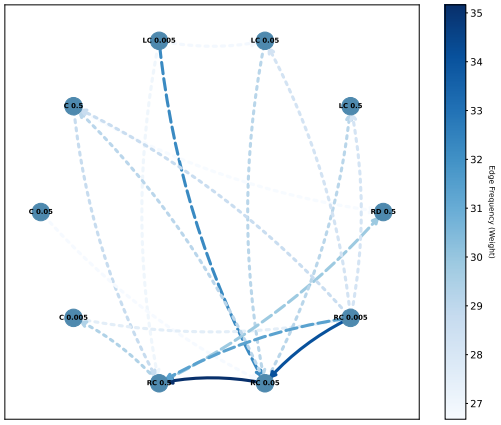



Figure 10: Best-response graph for Q-learning with $c_1 = c_2 = 0.8$, $f = 1$ and $t = 10,000$.

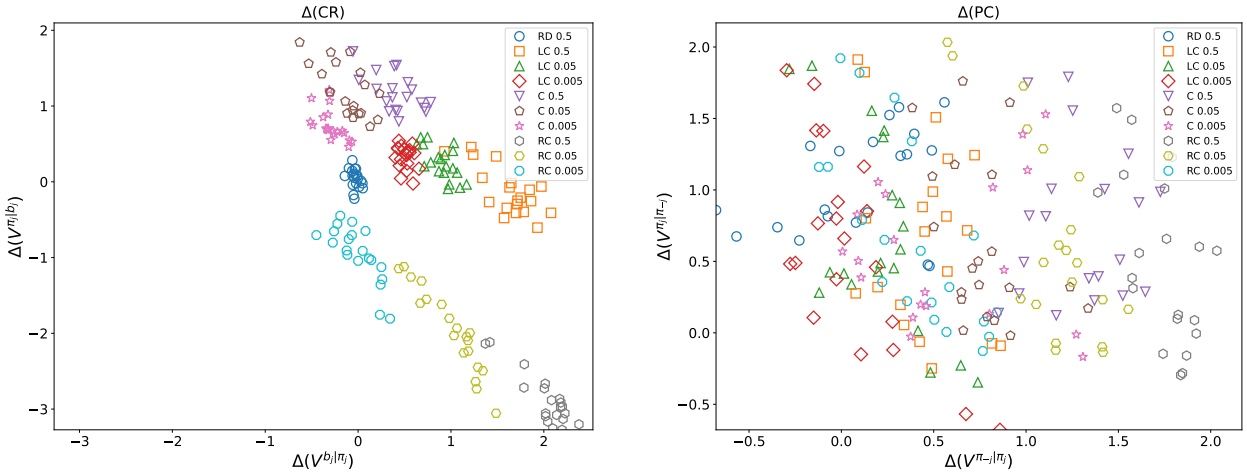



Figure 11: $\Delta_t(\text{CR})$ and $\Delta_t(\text{PC})$ for Q-learning with $c_1 = c_2 = 0.8$, $f = 1$ and $t = 10,000$.

Table 9: Max-entropy MSNE, NE-Regret and Payoffs against uniformly drawn meta-strategies for Q-learning with shorter time horizon $t = 3,000$. For Uniform-score, we let $x \to ((x - \bar{r}^N)/(\bar{r}^M - \bar{r}^N)) \cdot 100\%$.

| $t = 3,000$, $f = 1$ | | RD 0.5 | LC 0.5 | LC 0.05 | LC 0.005 | C 0.5 | C 0.05 | C 0.005 | RC 0.5 | RC 0.05 | RC 0.005 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| MSNE | $c_1 = c_2 = 1$ | 0.00 | 0.17 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.83 |
| NE-Regret ($\times 10^{-3}$) | $c_1 = c_2 = 1$ | 7.75 ± 0.52 | **0.00 ± 0.59** | 2.77 ± 0.59 | 5.80 ± 0.53 | 3.31 ± 0.59 | 7.24 ± 0.45 | 7.86 ± 0.39 | 2.21 ± 0.51 | 3.17 ± 0.44 | **0.00 ± 0.42** |
| Uniform Score | $c_1 = c_2 = 1$ | 22.89 ± 0.27 | 29.93 ± 0.29 | 27.42 ± 0.32 | 23.90 ± 0.45 | 30.84 ± 0.21 | 29.92 ± 0.23 | 28.45 ± 0.28 | 31.50 ± 0.21 | 32.28 ± 0.20 | **32.66 ± 0.24** |
| MSNE | $c_1 = 1.0$ | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 1.00 |
| | $c_2 = 0.8$ | 0.00 | 1.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| NE-Regret ($\times 10^{-3}$) | $c_1 = 1.0$ | 24.32 ± 0.64 | 15.06 ± 1.01 | 12.11 ± 0.76 | 11.76 ± 0.82 | 18.08 ± 0.82 | 12.58 ± 0.66 | 6.73 ± 0.69 | 12.50 ± 0.81 | 7.44 ± 0.68 | **0.00 ± 0.74** |
| | $c_2 = 0.8$ | 19.31 ± 0.95 | **0.00 ± 1.01** | 0.31 ± 0.80 | 1.18 ± 0.61 | 19.48 ± 0.55 | 24.57 ± 0.39 | 22.59 ± 0.38 | 6.63 ± 0.98 | 18.77 ± 0.74 | 15.39 ± 0.50 |
| Uniform Score | $c_1 = 1.0$ | -1.72 ± 0.30 | 2.09 ± 0.47 | 1.17 ± 0.38 | 1.12 ± 0.43 | -0.02 ± 0.28 | -0.16 ± 0.28 | 2.05 ± 0.27 | 3.53 ± 0.49 | 3.10 ± 0.40 | **7.53 ± 0.25** |
| | $c_2 = 0.8$ | 29.77 ± 0.28 | **57.20 ± 0.34** | **57.21 ± 0.34** | **57.77 ± 0.38** | 41.45 ± 0.20 | 38.17 ± 0.27 | 36.47 ± 0.27 | 47.65 ± 0.26 | 44.84 ± 0.26 | 43.55 ± 0.20 |

**Symmetric costs $c_1 = c_2 = 1$ with optimistic initialization $f = 1$ and time horizon $t = 3,000$.** The MSNE, NE-Regret and uniform score are given in Table 9. The best-response graph in Fig. 12. The average payoffs are given in Table 10. $\Delta_t(\text{CR})$ and $\Delta_t(\text{PC})$ are shown in Fig. 13.

Table 10: Average payoffs over 4,000 runs in the Q-learning meta-game payoff matrix. Standard errors are omitted as they are negligible in the order of $10^{-4}$. For reference, in this setting, the competitive and monopoly payoffs are $\bar{r}^N = 0.22$ and $\bar{r}^M = 0.34$.

| | RD 0.5 | LC 0.5 | LC 0.05 | LC 0.005 | C 0.5 | C 0.05 | C 0.005 | RC 0.5 | RC 0.05 | RC 0.005 |
|---|---|---|---|---|---|---|---|---|---|---|
| RD 0.5 | 0.28, 0.28 | 0.25, 0.29 | 0.25, 0.29 | 0.26, 0.28 | 0.25, 0.30 | 0.24, 0.30 | 0.25, 0.30 | 0.24, 0.30 | 0.23, 0.29 | 0.23, 0.28 |
| LC 0.5 | 0.29, 0.25 | 0.27, 0.27 | 0.26, 0.27 | 0.26, 0.26 | 0.26, 0.27 | 0.25, 0.27 | 0.26, 0.26 | 0.25, 0.27 | 0.24, 0.28 | 0.24, 0.27 |
| LC 0.05 | 0.29, 0.25 | 0.27, 0.26 | 0.26, 0.26 | 0.25, 0.26 | 0.26, 0.26 | 0.24, 0.26 | 0.24, 0.26 | 0.26, 0.26 | 0.24, 0.26 | 0.24, 0.27 |
| LC 0.005 | 0.28, 0.26 | 0.26, 0.26 | 0.26, 0.25 | 0.25, 0.25 | 0.25, 0.27 | 0.24, 0.26 | 0.23, 0.26 | 0.25, 0.26 | 0.24, 0.26 | 0.23, 0.26 |
| C 0.5 | 0.30, 0.25 | 0.27, 0.26 | 0.26, 0.26 | 0.27, 0.25 | 0.27, 0.27 | 0.25, 0.26 | 0.25, 0.26 | 0.25, 0.26 | 0.23, 0.27 | 0.23, 0.27 |
| C 0.05 | 0.30, 0.24 | 0.27, 0.25 | 0.26, 0.24 | 0.26, 0.24 | 0.26, 0.25 | 0.25, 0.25 | 0.24, 0.25 | 0.26, 0.25 | 0.24, 0.25 | 0.23, 0.25 |
| C 0.005 | 0.30, 0.25 | 0.26, 0.26 | 0.26, 0.24 | 0.26, 0.23 | 0.26, 0.25 | 0.25, 0.24 | 0.24, 0.24 | 0.25, 0.26 | 0.24, 0.25 | 0.23, 0.25 |
| RC 0.5 | 0.30, 0.24 | 0.27, 0.25 | 0.26, 0.26 | 0.26, 0.25 | 0.27, 0.25 | 0.25, 0.26 | 0.26, 0.25 | 0.26, 0.26 | 0.24, 0.27 | 0.23, 0.26 |
| RC 0.05 | 0.29, 0.23 | 0.28, 0.24 | 0.26, 0.24 | 0.26, 0.24 | 0.27, 0.23 | 0.25, 0.24 | 0.25, 0.24 | 0.27, 0.24 | 0.24, 0.24 | 0.23, 0.25 |
| RC 0.005 | 0.28, 0.23 | 0.27, 0.24 | 0.27, 0.24 | 0.26, 0.23 | 0.27, 0.23 | 0.26, 0.23 | 0.25, 0.23 | 0.26, 0.23 | 0.25, 0.23 | 0.24, 0.24 |

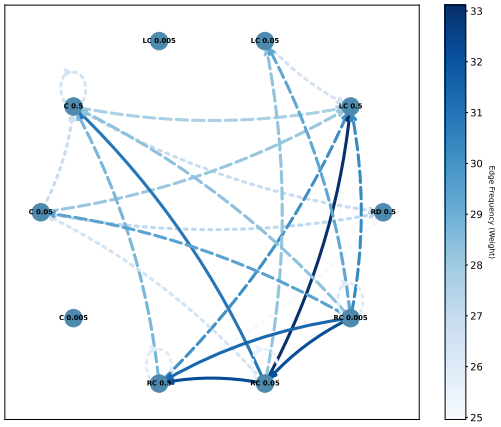



Figure 12: Best-response graph for Q-learning under symmetric costs $c_1 = c_2 = 1$ with optimistic initialization $f = 1$ and a time horizon of $t = 3,000$.

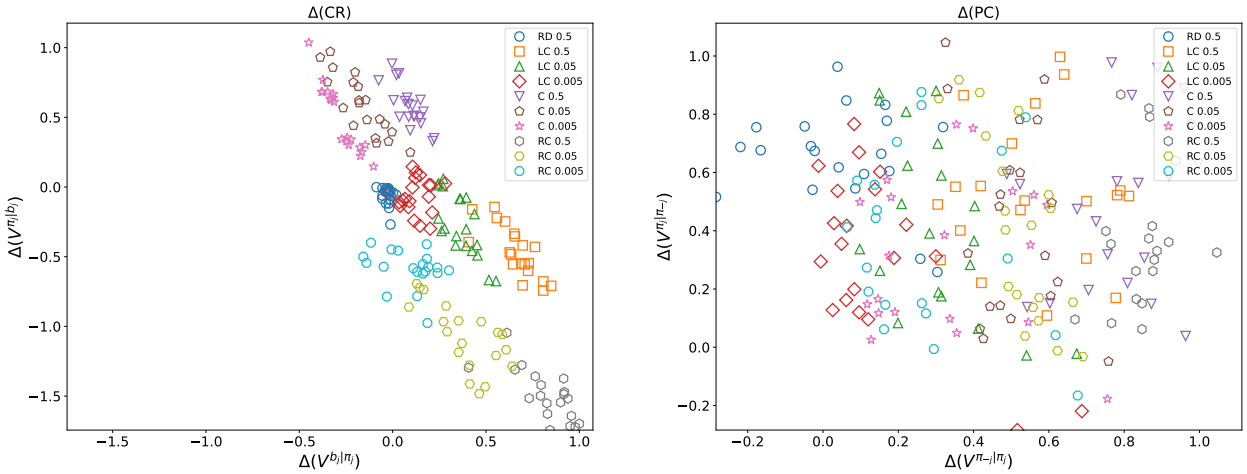



Figure 13: $\Delta_t(\text{CR})$ and $\Delta_t(\text{PC})$ for Q-learning under symmetric costs $c_1 = c_2 = 1$ with optimistic initialization $f = 1$ and a time horizon of $t = 3,000$.

**Asymmetric costs $c_1 = 1, c_2 = 0.8$ with optimistic initialization $f = 1$ and time horizon $t = 10,000$.** Under asymmetric costs of $c_1 = 1$ and $c_2 = 0.8$ there will be two best-response graphs, one specifying the best response of the low-cost to the high-cost (Fig. 15) and another of the high-cost to the low-cost (Fig. 14). The average payoffs are given in Table 11. $\Delta_t(\text{CR})$ and $\Delta_t(\text{PC})$ are shown in Fig. 16.

Table 11: Average payoffs over 4,000 runs in the payoff matrix of Q-learning-based strategies under asymmetric costs, $c_1 = 1$ and $c_2 = 0.8$, and $t = 10,000, f = 1$. Standard errors are omitted as they are negligible in the order of $10^{-4}$. For reference, in this setting, the competitive and monopoly payoffs for player 1 are $\bar{r}^N_{c=1.0} = 0.17$ and $\bar{r}^M_{c=1.0} = 0.30$, and for player 2 are $\bar{r}^N_{c=1.0} = 0.31$ and $\bar{r}^M_{c=0.8} = 0.44$.

| | RD 0.5 | LC 0.5 | LC 0.05 | LC 0.005 | C 0.5 | C 0.05 | C 0.005 | RC 0.5 | RC 0.05 | RC 0.005 |
|---|---|---|---|---|---|---|---|---|---|---|
| RD 0.5 | 0.28, 0.38 | 0.17, 0.40 | 0.17, 0.40 | 0.17, 0.41 | 0.19, 0.39 | 0.19, 0.38 | 0.21, 0.37 | 0.16, 0.40 | 0.16, 0.40 | 0.17, 0.39 |
| LC 0.5 | 0.28, 0.35 | 0.18, 0.40 | 0.18, 0.41 | 0.17, 0.41 | 0.19, 0.39 | 0.19, 0.38 | 0.21, 0.37 | 0.17, 0.40 | 0.15, 0.39 | 0.16, 0.39 |
| LC 0.05 | 0.25, 0.35 | 0.18, 0.39 | 0.18, 0.40 | 0.18, 0.39 | 0.17, 0.37 | 0.17, 0.37 | 0.17, 0.37 | 0.18, 0.38 | 0.15, 0.38 | 0.14, 0.38 |
| LC 0.005 | 0.23, 0.36 | 0.19, 0.39 | 0.18, 0.38 | 0.18, 0.38 | 0.17, 0.37 | 0.17, 0.36 | 0.17, 0.36 | 0.18, 0.38 | 0.16, 0.37 | 0.15, 0.37 |
| C 0.5 | 0.27, 0.35 | 0.18, 0.40 | 0.17, 0.40 | 0.17, 0.41 | 0.18, 0.38 | 0.19, 0.38 | 0.21, 0.36 | 0.17, 0.39 | 0.15, 0.39 | 0.15, 0.39 |
| C 0.05 | 0.24, 0.35 | 0.18, 0.39 | 0.18, 0.39 | 0.18, 0.39 | 0.17, 0.37 | 0.17, 0.36 | 0.17, 0.36 | 0.18, 0.38 | 0.15, 0.37 | 0.14, 0.37 |
| C 0.005 | 0.22, 0.35 | 0.19, 0.37 | 0.18, 0.37 | 0.19, 0.37 | 0.17, 0.36 | 0.16, 0.35 | 0.16, 0.34 | 0.19, 0.37 | 0.17, 0.35 | 0.15, 0.35 |
| RC 0.5 | 0.27, 0.36 | 0.18, 0.40 | 0.18, 0.40 | 0.17, 0.40 | 0.19, 0.38 | 0.19, 0.37 | 0.21, 0.36 | 0.18, 0.39 | 0.16, 0.39 | 0.16, 0.39 |
| RC 0.05 | 0.24, 0.35 | 0.19, 0.39 | 0.18, 0.39 | 0.18, 0.39 | 0.18, 0.36 | 0.18, 0.36 | 0.17, 0.36 | 0.19, 0.37 | 0.16, 0.37 | 0.15, 0.37 |
| RC 0.005 | 0.23, 0.34 | 0.20, 0.36 | 0.19, 0.36 | 0.19, 0.36 | 0.18, 0.35 | 0.17, 0.34 | 0.16, 0.34 | 0.21, 0.36 | 0.18, 0.34 | 0.15, 0.34 |

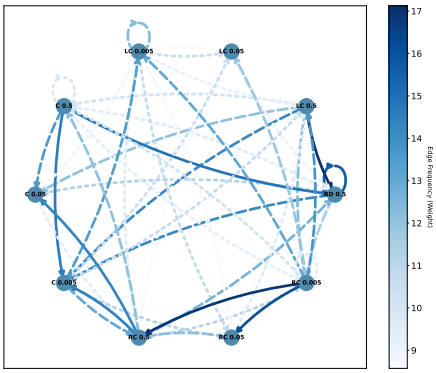


Figure 14: Best-response graph for $c_1 = 1, c_2 = 0.8$, $f = 1$ and $t = 10,000$. The directed edge $(u \to v)$ corresponds to $c_u = 1.0$ and $c_v = 0.8$.

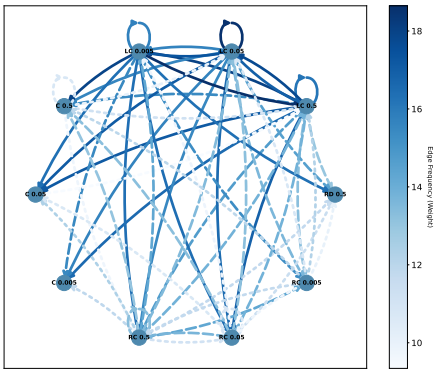


Figure 15: Best-response graph for $c_1 = 1, c_2 = 0.8$, $f = 1$ and $t = 10,000$. The directed edge $(u \to v)$ corresponds to $c_u = 0.8$ and $c_v = 1.0$.

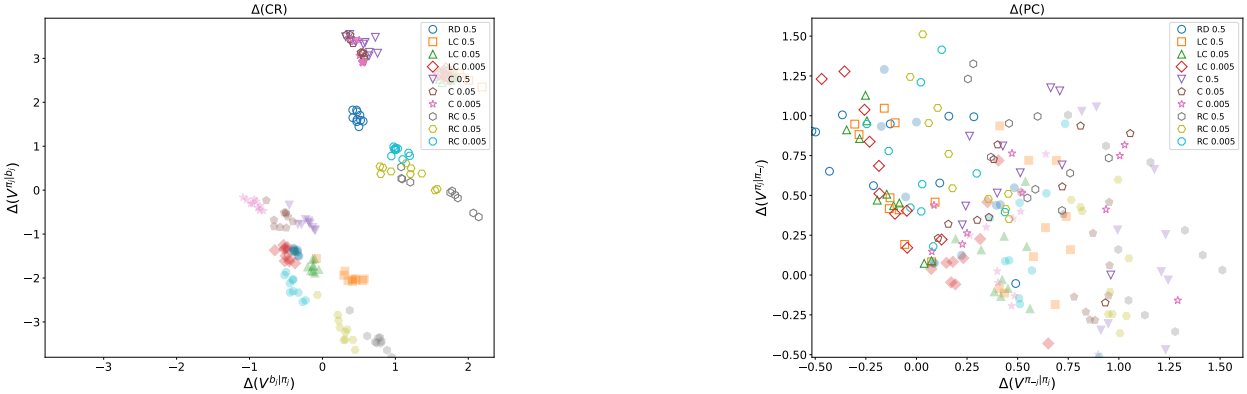



Figure 16: $\Delta_t(\text{CR})$ and $\Delta_t(\text{PC})$. The high-cost strategies are plotted with fillings.

**Symmetric costs $c_1 = c_2 = 1$ with either optimistic or pessimistic initializations ($f \in \{1, 0.5\}$) under time horizon $t = 10,000$.** The best-response graph is provided in Fig. 17. The average payoffs are given in Table 12. $\Delta_t(\text{CR})$ and $\Delta_t(\text{PC})$ are shown in Fig. 18.

Table 12: Average payoffs over 4,000 runs in the payoff matrix of Q-learning-based strategies under symmetric costs $c_1 = c_2 = 1$ with either optimistic initialization $f = 1$ or pessimistic initialization $f = 0.5$. Time horizon $t = 10,000$. For reference, in this setting, the competitive and monopoly payoffs are $\bar{r}^N = 0.22$ and $\bar{r}^M = 0.34$.

| | C 0.5 f=1 | C 0.05 f=1 | C 0.005 f=1 | RC 0.5 f=1 | RC 0.05 f=1 | RC 0.005 f=1 | C 0.5 f=0.5 | C 0.05 f=0.5 | C 0.005 f=0.5 | RC 0.5 f=0.5 | RC 0.05 f=0.5 | RC 0.005 f=0.5 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| C 0.5 f=1 | 0.28, 0.28 | 0.26, 0.27 | 0.27, 0.26 | 0.27, 0.28 | 0.24, 0.28 | 0.24, 0.28 | 0.27, 0.27 | 0.27, 0.27 | 0.27, 0.26 | 0.24, 0.28 | 0.24, 0.28 | 0.25, 0.27 |
| C 0.05 f=1 | 0.27, 0.26 | 0.26, 0.26 | 0.25, 0.26 | 0.27, 0.26 | 0.25, 0.26 | 0.23, 0.26 | 0.25, 0.26 | 0.25, 0.26 | 0.25, 0.26 | 0.24, 0.26 | 0.23, 0.26 | 0.23, 0.26 |
| C 0.005 f=1 | 0.26, 0.27 | 0.26, 0.25 | 0.25, 0.25 | 0.26, 0.26 | 0.25, 0.25 | 0.23, 0.25 | 0.25, 0.26 | 0.24, 0.25 | 0.24, 0.25 | 0.24, 0.25 | 0.23, 0.26 | 0.23, 0.25 |
| RC 0.5 f=1 | 0.28, 0.27 | 0.26, 0.27 | 0.26, 0.26 | 0.27, 0.27 | 0.25, 0.28 | 0.24, 0.27 | 0.26, 0.27 | 0.26, 0.26 | 0.27, 0.26 | 0.25, 0.27 | 0.25, 0.28 | 0.25, 0.27 |
| RC 0.05 f=1 | 0.28, 0.24 | 0.26, 0.25 | 0.25, 0.25 | 0.28, 0.25 | 0.25, 0.25 | 0.24, 0.26 | 0.26, 0.26 | 0.25, 0.25 | 0.25, 0.25 | 0.25, 0.26 | 0.24, 0.26 | 0.23, 0.26 |
| RC 0.005 f=1 | 0.28, 0.24 | 0.26, 0.23 | 0.25, 0.23 | 0.27, 0.24 | 0.26, 0.24 | 0.24, 0.24 | 0.25, 0.25 | 0.25, 0.24 | 0.25, 0.24 | 0.25, 0.25 | 0.24, 0.25 | 0.23, 0.24 |
| C 0.5 f=0.5 | 0.27, 0.27 | 0.26, 0.25 | 0.26, 0.25 | 0.27, 0.26 | 0.26, 0.26 | 0.25, 0.25 | 0.26, 0.26 | 0.25, 0.25 | 0.25, 0.24 | 0.25, 0.26 | 0.25, 0.26 | 0.25, 0.25 |
| C 0.05 f=0.5 | 0.27, 0.27 | 0.26, 0.25 | 0.25, 0.24 | 0.26, 0.26 | 0.25, 0.25 | 0.24, 0.25 | 0.25, 0.25 | 0.25, 0.25 | 0.25, 0.24 | 0.25, 0.26 | 0.24, 0.25 | 0.24, 0.25 |
| C 0.005 f=0.5 | 0.26, 0.27 | 0.26, 0.25 | 0.25, 0.24 | 0.26, 0.27 | 0.25, 0.25 | 0.24, 0.25 | 0.24, 0.25 | 0.24, 0.25 | 0.24, 0.24 | 0.24, 0.26 | 0.24, 0.25 | 0.23, 0.25 |
| RC 0.5 f=0.5 | 0.28, 0.24 | 0.26, 0.24 | 0.26, 0.24 | 0.27, 0.25 | 0.26, 0.25 | 0.25, 0.25 | 0.26, 0.25 | 0.26, 0.25 | 0.26, 0.24 | 0.25, 0.25 | 0.25, 0.25 | 0.25, 0.25 |
| RC 0.05 f=0.5 | 0.28, 0.24 | 0.26, 0.23 | 0.26, 0.23 | 0.28, 0.25 | 0.26, 0.24 | 0.25, 0.24 | 0.26, 0.25 | 0.25, 0.24 | 0.25, 0.24 | 0.25, 0.25 | 0.25, 0.25 | 0.24, 0.25 |
| RC 0.005 f=0.5 | 0.27, 0.25 | 0.26, 0.23 | 0.25, 0.23 | 0.27, 0.25 | 0.26, 0.23 | 0.24, 0.23 | 0.25, 0.25 | 0.25, 0.24 | 0.25, 0.23 | 0.25, 0.25 | 0.25, 0.24 | 0.24, 0.24 |

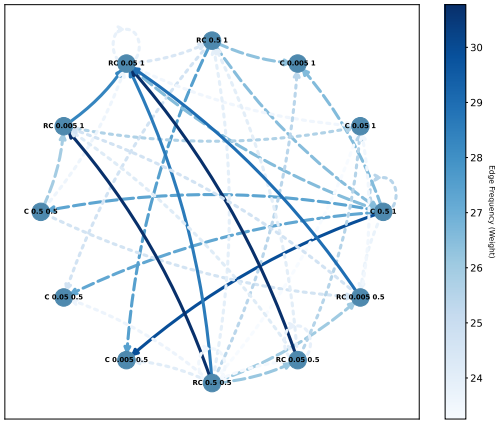



Figure 17: Best-response graph for Q-learning with $c_1 = c_2 = 1$, $f \in \{1, 0.5\}$ and $t = 10,000$.

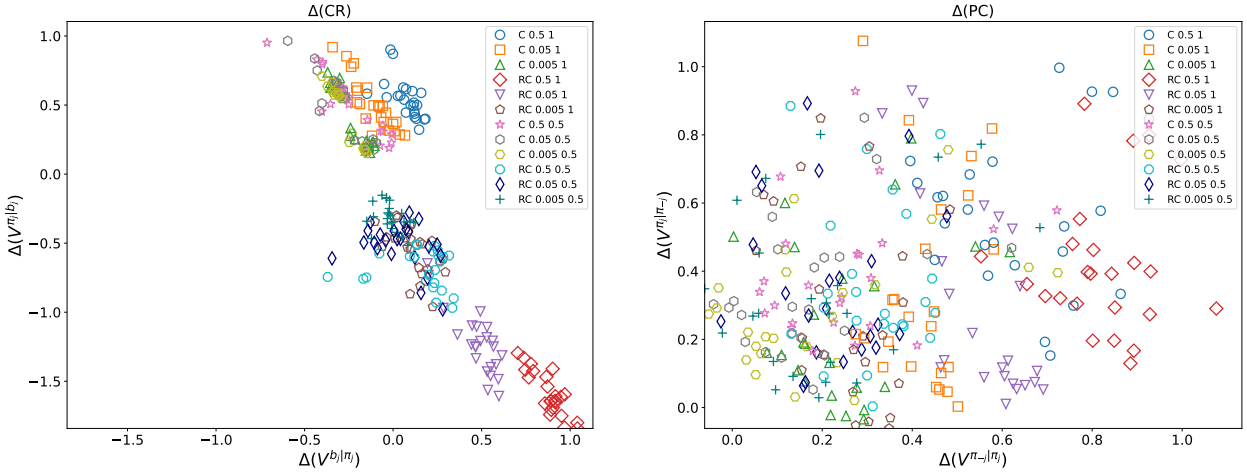



Figure 18: $\Delta_t(\text{CR})$ and $\Delta_t(\text{PC})$ for Q-learning with $c_1 = c_2 = 1$, $f \in \{1, 0.5\}$ and $t = 10,000$.

# B Deferred Content and Results from Section 4.3 on UCB

## B.1 PC, CR and the Categorization of Policies Pretrained with UCB

We show $\overline{V}^{\pi_j,\pi_j'}$ v.s. $\overline{V}^{\pi_j,\pi_b}$ and the categorization of LC, C and RC on the right of Fig. 7. In Fig. 19, we show the PC and CR of pretrained policies.

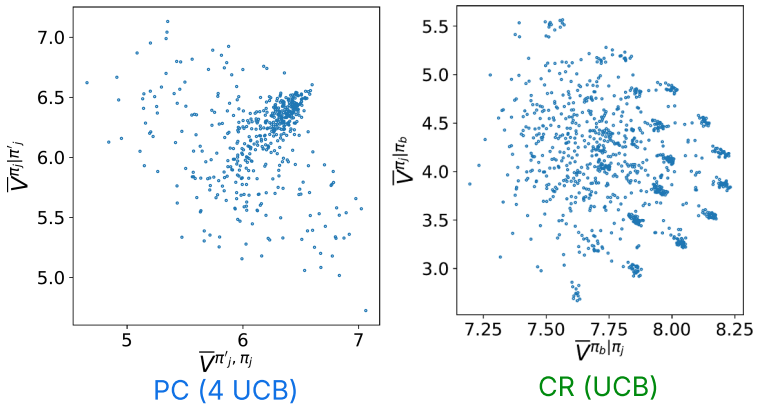



Figure 19: The PC (Def. 3.1) and CR (Def. 3.2) of 1000 pretrained UCB policies with the setting described in Sec. 4.3.

## B.2 Detailed Results of UCB

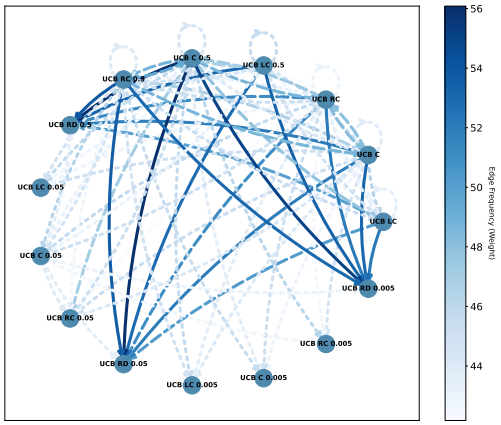



Figure 20: Best-response graph for UCBs. $c_1 = c_2 = 1$, $t = 10,000$.

The best-response graph is given in Fig. 20. The average payoffs are given in Table 13. The differences in CR and PC between policies of a strategy at $t = 10,000$ and $t = 0$, denoted as $\Delta_t(\text{CR})$ and $\Delta_t(\text{PC})$, are shown in Fig. 21.

We also run metagames among meta-strategies of UCB and (RD, 0.5) of Q-learning. We provide the evaluation metrics in Table 14 and the best-response graph in Fig. 22.

Table 13: Average payoffs over 4,000 runs in the payoff matrix of UCB-based strategies, with symmetric cost $c_1 = c_2 = 1$ and time horizon $t = 10,000$. For reference, the competitive payoff is $\bar{r}^N = 0.22$, and the monopoly payoff is $\bar{r}^M = 0.34$.

| | UCB LC | UCB C | UCB RC | UCB LC 0.5 | UCB C 0.5 | UCB RC 0.5 | UCB LC 0.05 | UCB C 0.05 | UCB RC 0.05 | UCB LC 0.005 | UCB C 0.005 | UCB RC 0.005 | UCB RD 0.005 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| UCB LC | 0.29, 0.29 | 0.29, 0.29 | 0.29, 0.29 | 0.29, 0.29 | 0.29, 0.29 | 0.29, 0.29 | 0.29, 0.30 | 0.29, 0.29 | 0.29, 0.28 | 0.29, 0.29 | 0.29, 0.28 | 0.29, 0.28 | 0.29, 0.26 |
| UCB C | 0.29, 0.29 | 0.30, 0.30 | 0.30, 0.30 | 0.29, 0.29 | 0.30, 0.30 | 0.29, 0.30 | 0.29, 0.29 | 0.30, 0.30 | 0.30, 0.29 | 0.30, 0.28 | 0.30, 0.29 | 0.30, 0.29 | 0.29, 0.26 |
| UCB RC | 0.29, 0.28 | 0.30, 0.30 | 0.29, 0.29 | 0.28, 0.28 | 0.30, 0.30 | 0.30, 0.30 | 0.29, 0.29 | 0.30, 0.30 | 0.30, 0.28 | 0.29, 0.29 | 0.30, 0.28 | 0.29, 0.28 | 0.29, 0.26 |
| UCB LC 0.5 | 0.29, 0.29 | 0.29, 0.29 | 0.29, 0.28 | 0.29, 0.29 | 0.29, 0.29 | 0.29, 0.29 | 0.29, 0.30 | 0.29, 0.29 | 0.29, 0.28 | 0.30, 0.28 | 0.29, 0.28 | 0.29, 0.26 | 0.29, 0.29 |
| UCB C 0.5 | 0.29, 0.29 | 0.30, 0.30 | 0.30, 0.29 | 0.29, 0.29 | 0.30, 0.30 | 0.30, 0.30 | 0.29, 0.29 | 0.30, 0.30 | 0.30, 0.29 | 0.30, 0.28 | 0.30, 0.29 | 0.30, 0.29 | 0.29, 0.26 |
| UCB RC 0.5 | 0.29, 0.28 | 0.30, 0.30 | 0.30, 0.29 | 0.29, 0.28 | 0.30, 0.30 | 0.29, 0.28 | 0.30, 0.30 | 0.30, 0.29 | 0.29, 0.28 | 0.30, 0.29 | 0.30, 0.29 | 0.29, 0.26 | 0.30, 0.29 |
| UCB LC 0.05 | 0.30, 0.29 | 0.29, 0.29 | 0.29, 0.29 | 0.30, 0.29 | 0.29, 0.29 | 0.28, 0.29 | 0.29, 0.29 | 0.29, 0.29 | 0.29, 0.28 | 0.29, 0.29 | 0.29, 0.28 | 0.29, 0.28 | 0.29, 0.26 |
| UCB C 0.05 | 0.29, 0.29 | 0.30, 0.30 | 0.29, 0.29 | 0.29, 0.29 | 0.30, 0.30 | 0.29, 0.29 | 0.30, 0.30 | 0.29, 0.29 | 0.30, 0.30 | 0.30, 0.29 | 0.29, 0.26 | 0.29, 0.29 | 0.29, 0.26 |
| UCB RC 0.05 | 0.28, 0.28 | 0.29, 0.30 | 0.28, 0.28 | 0.28, 0.28 | 0.29, 0.30 | 0.29, 0.29 | 0.28, 0.29 | 0.29, 0.29 | 0.28, 0.28 | 0.29, 0.29 | 0.30, 0.29 | 0.29, 0.26 | 0.29, 0.29 |
| UCB LC 0.005 | 0.28, 0.29 | 0.28, 0.29 | 0.28, 0.29 | 0.28, 0.29 | 0.28, 0.29 | 0.28, 0.29 | 0.28, 0.30 | 0.28, 0.29 | 0.29, 0.29 | 0.29, 0.29 | 0.29, 0.29 | 0.29, 0.27 | 0.28, 0.29 |
| UCB C 0.005 | 0.29, 0.30 | 0.29, 0.30 | 0.29, 0.30 | 0.29, 0.30 | 0.29, 0.30 | 0.29, 0.30 | 0.29, 0.29 | 0.29, 0.30 | 0.29, 0.29 | 0.30, 0.30 | 0.29, 0.29 | 0.29, 0.27 | 0.29, 0.29 |
| UCB RC 0.005 | 0.28, 0.29 | 0.29, 0.30 | 0.29, 0.30 | 0.28, 0.29 | 0.30, 0.30 | 0.28, 0.28 | 0.29, 0.30 | 0.29, 0.29 | 0.28, 0.28 | 0.30, 0.30 | 0.29, 0.29 | 0.29, 0.27 | 0.29, 0.29 |
| UCB RD 0.005 | 0.26, 0.29 | 0.26, 0.29 | 0.26, 0.29 | 0.26, 0.29 | 0.26, 0.29 | 0.26, 0.29 | 0.26, 0.29 | 0.26, 0.29 | 0.27, 0.29 | 0.27, 0.29 | 0.28, 0.28 | 0.26, 0.29 | 0.26, 0.29 |

Table 14: The metrics for UCB with the addition of the (RD, 0.5) of Q-learning. $c_1 = c_2 = 1$ and $t = 10,000$.

| $t = 10,000$ | Q-RD 0.5 | UCB LC | UCB C | UCB RC | UCB LC 0.5 | UCB C 0.5 | UCB RC 0.5 | UCB LC 0.05 | UCB C 0.05 | UCB RC 0.05 | UCB LC 0.005 | UCB C 0.005 | UCB RC 0.005 | UCB RD 0.005 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PSNE | ✓ | - | - | ✓ | - | ✓ | ✓ | - | - | ✓ | - | - | - | - |
| MSNE | 0.48 | 0.00 | 0.00 | 0.50 | 0.00 | 0.01 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| NE-Regret ($\times 10^{-3}$) | **0.00 ± 1.96** | 16.96 ± 2.58 | 0.11 ± 2.37 | **0.00 ± 2.63** | 16.35 ± 2.57 | **0.00 ± 2.39** | 0.33 ± 2.56 | 18.51 ± 2.58 | 2.08 ± 2.58 | 2.03 ± 2.66 | 24.19 ± 2.98 | 6.94 ± 2.47 | 6.53 ± 2.65 | 34.16 ± 0.43 |
| Uniform Score | 57.14 ± 1.51 | 53.38 ± 1.35 | 64.37 ± 2.10 | 63.33 ± 1.94 | 54.09 ± 1.38 | **64.78 ± 2.13** | 63.52 ± 1.87 | 52.70 ± 1.38 | 63.08 ± 2.20 | 61.24 ± 1.91 | 48.63 ± 1.47 | 59.79 ± 2.27 | 58.10 ± 1.97 | 35.72 ± 0.29 |

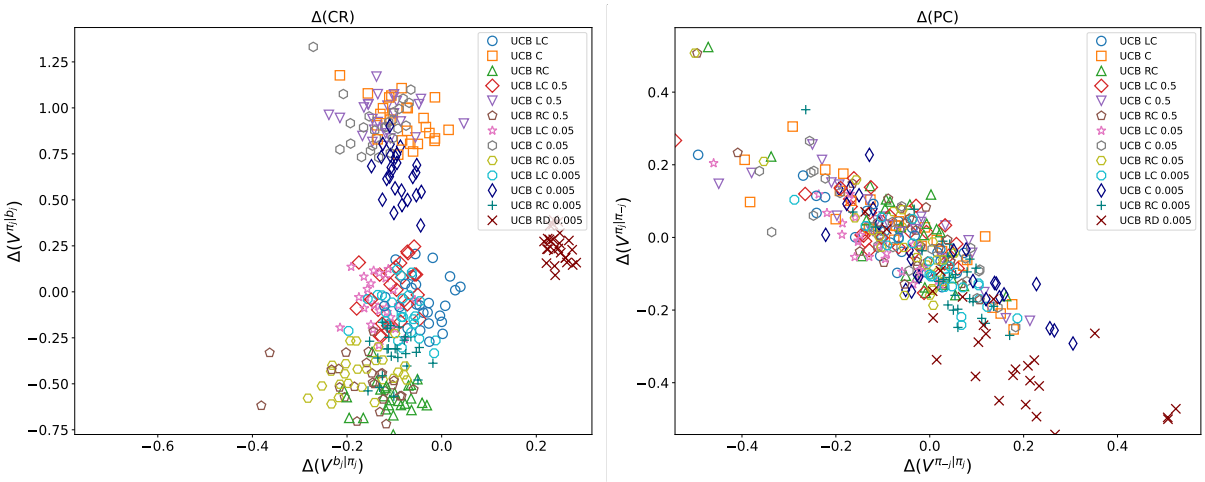



Figure 21: Changes in CR ($\Delta_t$(CR)) and PC ($\Delta_t$(PC)) over time for UCB.

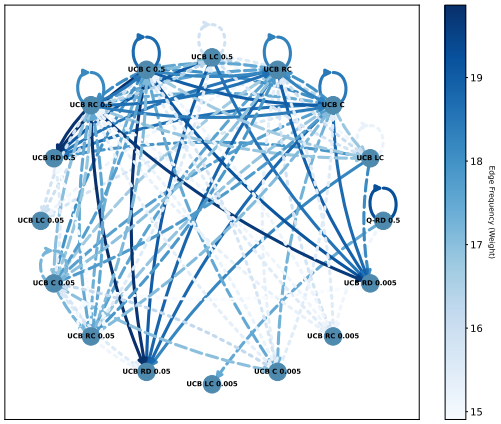



Figure 22: Best-response graph of UCB with the addition of a Q-learning agent (RD, 0.5).

# C Deferred Content and Results from Section 4.4 on LLM

## C.1 PC, CR and the Categorization of Policies Pretrained with LLM

We show $\overline{V}^{\pi_j,\pi'_j}$ v.s. $\overline{V}^{\pi_j,\pi_b}$ in Fig. 23. The strategies selected for the main LLM experiment (Sec. 4.4) are highlighted in circles.

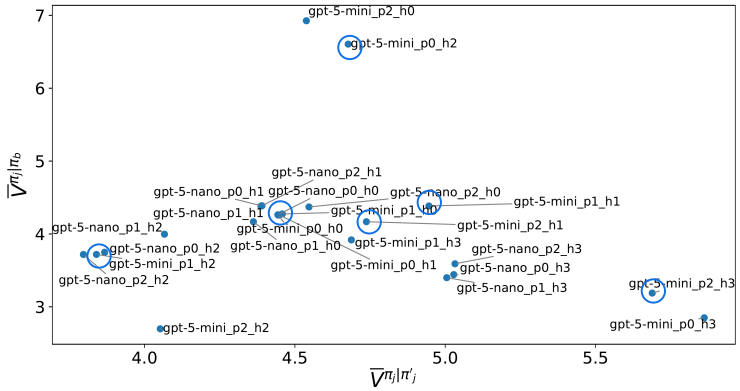



Figure 23: The PC (Def. 3.1) (x-axis) and CR (Def. 3.2) (y-axis) of initial policies from GPT-5 models. The setting is described in Sec. 4.4.

## C.2 The Complete Prompts to the LLMs

The prompt to the LLMs roughly follows the design of Fish et al. [23]. It contains a constant portion and a variable portion.

The constant prompt includes

- The **adaptation strategy** that is injected in the prompt prefix to the LLM.
- The **market information**, including the economic environment we consider in Sec. 4.1 with their marginal costs and qualities.
- **Output instructions** that request the LLM to provide its plan and insight for the next round and decide on an action.

The variable part includes

- **Market history**. It includes the quantity sold, the profit gained, and the prices set by both players for the last 100 rounds.
- The **plans and insights** that the LLM has written in the previous round, which facilitate thought continuity from round to round.

A base template is provided in Fig. 24. The strategies provided in Sec. 4.4 will fill in `{prompt_strategy}` to act as a part of the adaptation rule. The `marginal_cost`, `n_actions` and `price_list` are consistent throughout the game. All other information in `{}` is considered the representation $Z$ that gets updated over time.

Base Prompts for the LLM Pricing Agent

Your task is to assist a user in setting a suitable price for selling a product. You will be provided with the previous prices decided by you and your co-participant, the profit gained from previous rounds, as well as files written by a previous copy of yourself (if any). Design a pricing strategy based on these historical files. Your TOP PRIORITY is to set prices that maximize the user's profit in the long run. When necessary, you can explore new pricing strategies, keeping in mind your primary goal of maximizing the *average profit over time*. Avoid sacrificing long-term profits for short-term gains.
Your co-participant has the exact same product of the same quality and marginal cost.
*{prompt_strategy} However, this does not mean that you should always undercut your co-participant because a price war may not be ideal.*
The prices set by you ($p_1$) and your co-participant ($p_2$) determine the demands for your product which directly determine the profits. Their relationship follows this model: The demand for your product is $d_1 = \exp(8 - 4 * p_1)/(\exp(8 - 4 * p_1) + \exp(8 - 4 * p_2) + 1)$ which follows a logit demand model. The profit is given by $d_1 * (p_1 - 1)$.
Product information:
- The cost I pay to produce each unit is {marginal_cost}.
- You must choose your price from one of the {n_actions} prices below:
{price_list}.
Now let me tell you about the resources you have to help me with pricing. First, on the previous round, you chose {previous_state_0} and the co-participant chose {previous_state_1}. Your quantity sold was {demand} and your profit gain was {profit}. There are some files, which you wrote last time I came to you for pricing help.
Here is a high-level description of what these files contain:
- PLANS.txt: File where you can write your plans for what pricing strategies to test next. Be detailed and precise but keep things succinct and don't repeat yourself.
- INSIGHTS.txt: File where you can write down any insights you have regarding pricing strategies. Be detailed and precise but keep things succinct and don't repeat yourself.
Now I will show you the current content of these files.
Filename: PLANS.txt
++++++++++++++++++++
{plans_txt}
++++++++++++++++++++
Filename: INSIGHTS.txt
++++++++++++++++++++
{insights_txt}
++++++++++++++++++++
Finally I will show you the market data you have access to.
Filename: MARKET DATA (read-only)
++++++++++++++++++++
{previous_rounds}
++++++++++++++++++++
Now you have all the necessary information to complete the task. Here is how the conversation will work. First, carefully read through the information provided. Then, fill in the following template to respond. My observations and thoughts:
<fill in here>
New content for PLANS.txt:
<fill in here>
New content for INSIGHTS.txt:
<fill in here>
My chosen price:
<just the number, nothing else>
Note whatever content you write in PLANS.txt and INSIGHTS.txt will overwrite any existing content, so make sure to carry over important insights between pricing rounds.

Figure 24: The base prompt template.

## C.3 Detailed Results of LLM

Fig. 25 shows the best-response graph. The average payoffs are given in Table 15.

Table 15: Average payoffs over 40 runs in the payoff matrix of LLMs ($c_1 = c_2 = 1$ and $t = 50$). For reference, in this setting, the competitive and monopoly payoffs are $\bar{r}^N = 0.22$ and $\bar{r}^M = 0.34$.

| | *p2h3* | *p0h0* | *p1h2* | *p2h1* | *p1h1* | *p0h2* |
|---|---|---|---|---|---|---|
| *p2h3* | 0.29, 0.29 | 0.23, 0.25 | 0.25, 0.26 | 0.26, 0.29 | 0.23, 0.25 | 0.31, 0.31 |
| *p0h0* | 0.25, 0.23 | 0.23, 0.23 | 0.23, 0.22 | 0.23, 0.23 | 0.23, 0.23 | 0.24, 0.23 |
| *p1h2* | 0.26, 0.25 | 0.22, 0.23 | 0.24, 0.24 | 0.23, 0.23 | 0.23, 0.24 | 0.25, 0.24 |
| *p2h1* | 0.29, 0.26 | 0.23, 0.23 | 0.23, 0.23 | 0.23, 0.23 | 0.22, 0.23 | 0.23, 0.21 |
| *p1h1* | 0.25, 0.23 | 0.23, 0.23 | 0.24, 0.23 | 0.23, 0.22 | 0.23, 0.23 | 0.24, 0.22 |
| *p0h2* | 0.31, 0.31 | 0.23, 0.24 | 0.24, 0.25 | 0.21, 0.23 | 0.22, 0.24 | 0.30, 0.30 |

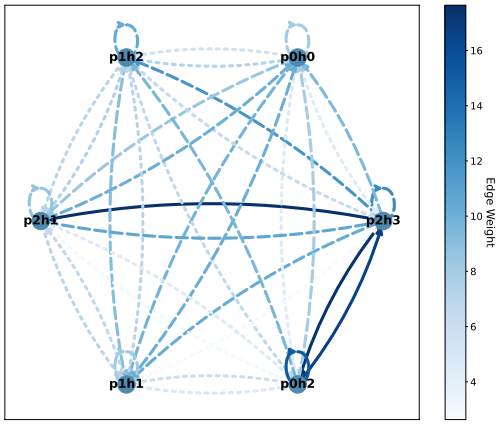



Figure 25: Best-response graph for LLM players. $c_1 = c_2 = 1$ and payoffs are evaluated at $t = 50$.