Using Reinforcement Learning to Study Platform Economies under Market Shocks

Xintong Wang\textsuperscript{1}, Gary Qiurui Ma\textsuperscript{1}, Alon Eden\textsuperscript{1}, Clara Li\textsuperscript{1}, Alexander Trott\textsuperscript{2}, Stephan Zheng\textsuperscript{2}, and David C. Parkes\textsuperscript{1}

\textsuperscript{1}Harvard University
\textsuperscript{1}\{xintongw,qiurui_ma,aloneden\}@seas.harvard.edu, clarali@college.harvard.edu, parkes@eecs.harvard.edu
\textsuperscript{2}Salesforce Research
\textsuperscript{1}\{atrott,stephan.zheng\}@salesforce.com

March 28, 2022

Abstract
Driven by rapid digitization and expansive internet access, market-driven platforms (e.g., Amazon, DoorDash, Uber, TaskRabbit) are increasingly prevalent and becoming key drivers of the economy. Across many industries, platforms leverage digital infrastructure to efficiently match producers and consumers, dynamically set prices, and enable economies of scale. This increasing prominence makes it important to understand the behavior of platforms, which induces complex phenomenon especially in the presence of severe market shocks (e.g., during pandemics). In this work, we develop a multi-agent simulation environment to capture key elements of a platform economy, including the kinds of economic shocks that disrupt a traditional, off-platform market. We use deep reinforcement learning (RL) to model the pricing and matching behavior of a platform that optimizes for revenue and various socially-aware objectives. We start with tractable motivating examples to establish intuitions about the dynamics and function of optimal platform policies. We then conduct extensive empirical simulations on multi-period environments, including settings with market shocks. We characterize the effect of a platform on the efficiency and resilience of an economic system under different platform design objectives. We further analyze the consequences of regulation fixing platform fees, and study the alignment of a revenue-maximizing platform with social welfare under different platform matching policies. As such, our RL-based framework provides a foundation for understanding platform economies under different designs and for yielding new economic insights that are beyond analytical tractability.

1 Introduction

Market-driven platforms (e.g., Amazon, DoorDash, Uber, TaskRabbit) play an increasingly important role in today’s economy, bringing together parties from two or more sides of the market

\hspace{1cm} \*This research is funded in part by Defense Advanced Research Projects Agency under Cooperative Agreement HR00111920029. The content of the information does not necessarily reflect the position or the policy of the Government, and no official endorsement should be inferred. This is approved for public release; distribution is unlimited.
to facilitate trades. The platform-based economy introduces new ways to create value. It reduces search costs by introducing potential matches that were not considered before. Just as importantly, it reduces the effort made by the consumers to physically go to the provider, which might be time consuming and require scarce resources (e.g., having a car). In the meantime, the platform itself is a strategic entity that operates to achieve certain goals. This makes it important for the economy designer to understand a platform’s behavior as well as the complex phenomenon or outcomes induced by such behavior. For example, while a platform’s goal may be to maximize its own revenue, this may also incentivize the platform to increase market efficiency, e.g., through improved matching quality and lower fulfillment friction. As such, the platform may attract more consumers and providers, enhancing its network effect and increasing the total amount of service fee it collects.

The importance of platform economies was made even more apparent by the Covid-19 pandemic. Stay-at-home orders, and the increased caution due to potentially high-cost outcomes of physically visiting providers, have tremendously increased the cost of consumers transacting in the brick-and-mortar stores without using a platform. As a result, the network effect is enhanced, with the increasing number of users and share of transactions made via platforms. An empirical study indeed shows that low profit-margin restaurants which did not find platform services to be economical pre-pandemic, turned to these platforms in order to survive [29]. A further study by UberEats from February through May 2020 shows that restaurants that used delivery platforms saw “significant and economically meaningful” increases in orders, which may have kept some restaurants alive [22].

On the other hand, as an increasing number of customers turns to adopt platforms due to the higher off-platform transaction costs, platforms possess larger market power, and thus have driven up the on-platform service and commission fees. Such fee adjustments have received complaints and even lawsuits from restaurants and consumers alike [30].

Motivated by the significant impact, we investigate the effects of a platform on the overall economic system. In particular, we aim to study the economy during an economic shock that disrupts the traditional brick-and-mortar market. We focus on the efficiency, resilience, as well as surplus distribution among parties achieved by economic systems that are mediated by platforms. We consider revenue-maximizing platform design objectives along with modified goals that capture proxies of economic health. In addition, we analyze the possible effect of regulation that limits fees while leaving platforms with the freedom to set their own matching policies.

Analytical methods fall short in analyzing such complex and highly dynamic environments: there is uncertainty about economic shocks, multiple policy decision moments for the platform to set fees and consider matching, and dynamic joining and leaving decisions made by buyers and sellers. Indeed, even when all non-platform actors and fees are fixed, optimally matching buyers and sellers is computationally hard even for a single time period [18]. Moreover, setting optimal fees in dynamic settings is computationally hard, even for a one sided market with a single buyer and two time periods [21].

To make progress, we use reinforcement learning (RL) to optimize platform policies, motivated by its recent successes in other complex economic settings [2, 33, 6, 25, 34]. Our study uses a multi-agent simulation of a platform economy, in which we train RL platform policies. The simulation enables evaluating RL policies in diverse settings under different objectives and model assumptions. In particular, simulations are useful when data is scarce and hard to gather, or when real-world experiments are near-impossible, e.g., economies undergoing a large shock. To summarize, our contributions are threefold:
1) **RL for platform fee design and query matching.** We model the platform as a rational agent, who sets its fees and matching policy across each of multiple periods to mediate a two-sided market between buyers and sellers. The platform uses reinforcement learning (RL) to set registration and transaction fees, and also to decide how to match a buyer “query” (representing a particular interest at some moment in time) with an on-platform seller. Buyers and sellers, in turn, choose whether or not to join the platform, which acts to reduce search costs and fulfillment cost.

2) **A multi-agent simulation of a platform economy.** We developed an OpenAI Gym environment [3] to capture key aspects of a platform economy, and study the strategic interplay among the rules of the platform and a set of buyers and sellers. The simulator is configurable to represent different market structures, i.e., distributions of buyers and sellers in a latent space, knowledge levels of buyers about sellers, and the cost of off-platform fulfillment, e.g., the cost to visit a brick-and-mortar store. The simulator models buyers and sellers that respond to platform policies by joining or leaving the platform, and for buyers, which transactions to complete.

3) **Understanding platform economies under a dynamic environment with shocks.** Our particular focus is to use RL to model the behavior of an economic platform for different design objectives, including platform revenue and a combination of revenue and various suitable proxy metrics. We use this to study the effect of a platform on market efficiency and surplus to different parties, including in the presence of an economic shock during which the cost of off-platform fulfillment increases. We find that a revenue-maximizing platform, though helps to facilitate trades during the shock, tends to raise fees and extracts more surplus from buyers and sellers, leading to a decrease in social welfare at the post-shock period. Results for different socially-aware objectives indicate that a surplus-aware or seller-aware platform may help restore a comparable level of total welfare post-shock, reducing the impact of a market shock on the economic system. We also study a setting where platform fees are fixed (e.g., due to regulation), leaving the platform to decide how to match. In this case, we find that under most fee regimes, a platform’s revenue-maximizing incentive can generally align with welfare considerations of the overall economy.

### 2 Related Work

**Platform Design** Existing papers on platform design in the economics literature focus mainly on the effect of a single round of fee-setting with a fixed matching policy. The initial work of Caillaud and Jullien studied a market with competing platforms in the presence of homogeneous buyers and sellers [5]. Rochet and Tirole studied platform competition under various governance structures (e.g. revenue-maximizing platforms) [23]. Armstrong studied various model setups, ranging from a monopoly platform, to various forms of competing platforms [1]. Mladenov et al. studied maximizing social welfare by efficiently matching queries when all other parameters of the platform and the demand of agents are fixed [18]. Our work analyzes a single platform, with buyers having an option to complete transactions without using the platform, in a dynamic, multi-period environment. Our platform can set both fees and a matching policy in an evolving manner.

In our model, we also incorporate an inertia component, which describes agents’ tendency to stick with their current decision (say being registered to a platform) than change their mind. This behavior was observed in choosing consumer packaged goods [26, 9], health insurance [11] and auto
insurance [13], among others. We adopt a similar model to [8, 17, 10], where the inertia is modeled as an additional additive term to the agent’s surplus of the current decision. We differ from these models by considering an inertia that is increasing as the period in which the agent makes the decision increases. In our model, the decision whether to subscribe to the platform is done by taking probabilities over choices using the discrete-choice logit model, as done in [8, 17].

Reinforcement Learning for Economic System Design  Reinforcement learning methods have been extremely useful in the design of economic systems. Zhan et al. [33] and Chen et al. [6] show the usefulness of RL in designing recommender systems. Chen et al. [6] addresses the problem of inferring good recommender systems in the face of getting feedback only from the chosen recommendations. Zhan et al. [33] use RL to optimize the long-term satisfaction of both users and content providers in a dynamic setting. These differ from our setting in several ways. First, they use RL to optimize the social welfare, as oppose to platform’s own revenue. Second, they do not consider agents transacting directly without the platform, and do not consider the platform’s role in reducing the world friction during shocks.

RL was also used in mechanism design to design mechanisms to sell users impressions to online advertisers [4, 28], set reserve prices over time [25] and to design sequential price mechanisms [2].

Simulation Frameworks  One of the main contributions of our work is creating a simulation framework for the study and design of platforms under the presence of shock. Similar simulation environments were designed for the study of Recommender Systems [15] and and tax policies in dynamic economies [34].

3 Platform Economies: Motivating Examples

We study the role and the design of a platform that can set fees and match sellers to buyers’ queries. In this section, we show various ways in which a platform can improve the market’s efficiency: (i) by reducing world friction, (ii) by introducing possible matches for buyers queries that are unknown otherwise, and (iii) by using a matching policy that compensates sellers that do not get many transactions otherwise. We also show that adding market regulation, in the form of adding a surplus term to the objective of the platform can improve market efficiency.

Here, we demonstrate the basic mechanics of our economic model for a highly simplified setting. We model two buyers, each with fixed demands, and two sellers. Given its simplicity, we can analyze a *Stackelberg equilibrium* of this economy, where the platform first sets fees and also a matching policy, and the buyers and sellers adopt an equilibrium response (join or not). A seller who does not cover their costs goes bankrupt. World friction, which corresponds to the cost of transaction fulfillment off platform, is an important part of our general model. Here we assume it is constant. Another important part of our full model is the knowledge structure of buyers, which models which sellers they know and allows for transactions in the absence of the platform. Here we assume this knowledge structure is fixed for each buyer.

In the absence of a platform, the market is very inefficient, with low agent surplus and Seller 1 going bankrupt. We then consider a revenue-maximizing platform that can set fees and myopically matches queries of on-platform buyers with the on-platform seller who is the best fit for the buyer. Seller 1 still goes bankrupt, but the presence of the platform increases agents’ surplus by effecting transactions that are free of friction and also enabling more transactions. A platform without
myopic matching can also direct some queries towards Seller 1, with the effect of increasing the platform’s revenue and preventing Seller 1 from going bankrupt. We also show in this simple example that adding a surplus term to the platform’s objective results in the platform setting fees such that Seller 1 joins the platform and does not go bankrupt, and even with myopic matching, with the effect that the overall surplus increases.

Two buyer, two seller model See Figure 1. An important component of our general model is a latent structure, that represents the preferences of buyers and capabilities of sellers. Here, this latent structure is one-dimensional, with two sellers and two buyers, each with \(2m\) queries. A query represents a demand request from a buyer. For Buyer 2 these are at location \(Q_2\), at distance 1 from Seller 1 and distance 3 from Seller 2. For Buyer 1, we have \(m\) queries in location \(Q_{11}\), and at distance \(\epsilon\) from Seller 1 and distance \(2-\epsilon\) from Seller 2 and \(m\) queries in location \(Q_{12}\), and at distance \(1+\epsilon\) from Seller 1 and distance \(1-\epsilon\) from Seller 2. Buyer 1 knows Seller 2, while Buyer 2 knows both sellers.

We only consider registration fees, while the general model also includes a seller referral fee when an on-platform transaction is completed.

A buyer can transact off-platform with a known seller and also match on-platform with other sellers. Let \(I_w \in \{0, 1\}\) be an indicator variable as to whether a transaction is completed off-platform \((I_w = 1)\) or on-platform \((I_w = 0)\), and let \(d(q, s)\) be the distance in the latent space between a query \(q\) and a seller \(s\). Buyer \(i\)'s utility from transacting on query \(q\) with seller \(s\) is \(u_i(q, s, 1) = 2 - d(q, s) - 1 \cdot I_w\), with utility \(2 - d(q, s)\), so that smaller distance is preferred, and cost of 1 due to world friction in the case of an off-platform transaction.

An off-platform buyer will choose to transact query \(q\) to a known seller \(s\) to maximize \(u_i(q, s, 1)\), or not transact at all if this utility is negative. For an on-platform buyer, the platform will suggest a match between the buyer’s query and an on-platform seller, if any. The buyer will then choose the utility-maximizing option between this match \(s\) (obtaining utility \(u_i(q, s, 0)\)), the best, known off-platform seller \(s'\) (obtaining utility \(u_i(q, s', 1)\)), and not transacting at all. The overall utility of a buyer, without fee consideration, is the total utility from their queries.

Sellers receive a profit of 1 from each completed transaction (regardless of distance). We assume sellers have a fixed cost of \(m > 1\) which they need to cover in order to avoid going bankrupt. The overall utility of a seller, without fee consideration, is thus the number of completed transactions minus fixed cost \(m\). In the general dynamic model, \(m\) is replaced with the assumption that a seller that does not get a positive surplus for some number of time periods (epochs) goes bankrupt.

When subscribing to the platform, an agent’s surplus is then its utility minus the registration fee. No fee is incurred if not subscribing. The revenue of the platform is the sum of fees that it collects.

A platform with a myopic matching policy chooses an on-platform seller \(s\) that minimizes \(d(q, s)\), and thus maximizes the buyer’s utility. A platform may also want to consider a different matching policy; e.g., matching queries to the seller with the fewest transactions made so far, in order to
increase their profit and in turn the fees it might collect.

We compute the Stackelberg equilibrium for this simple economy under different scenarios. In a Stackelberg equilibrium, the platform first sets its policy (fees and matching policy). The sellers and buyers then play an equilibrium, where their strategy of each agent is whether to join or not to subscribe to the platform, and the platform cannot increase its objective value by choosing another policy. A formal description appears in Appendix A. We inspect four different scenarios, which vary depending on the presence of a platform, the matching policy of the platform, and whether or not surplus is part of its objective. We defer the proofs to Appendix B.

**Claim 1.** Without a platform, Seller 1 goes bankrupt and the total buyer and seller surplus is $c m$.

The next scenario shows that a revenue-maximizing platform, even with myopic matching, can substantially increase the surplus of the economic system.

**Claim 2.** When the platform uses a myopic matching policy and is revenue maximizing, in equilibrium, Buyers 1 and 2 and Seller 2 join the platform, Seller 1 goes bankrupt, and the total buyer and seller surplus is $(1 - \epsilon)m$.

The next two scenarios show possible ways to prevent Seller 1 from going bankrupt.

**Claim 3.** When the platform uses a non-myopic matching policy and is revenue maximizing, in equilibrium, all agents join the platform, and the total buyer and seller surplus is $(1 - 2\epsilon)m$.

**Claim 4.** Recall that $d(Q_{11}, S_1) = \epsilon$. When the platform’s matching policy is myopic and its objective is revenue plus $\alpha$ times total buyer and seller surplus, for $\alpha > 1/2 + \epsilon$, then no agent goes bankrupt, and the total agent surplus is $3m$.

4 Designing Platform Economies in a Multi-Period World

4.1 Simulation Environment for Designing Platform Economy

We build an agent-based Gym environment that captures the key aspects of a platform economy in a dynamic, multi-period setting. There are heterogeneous buyer agents $B$, heterogeneous seller agents $S$, and a single platform agent $p$. Throughout the paper, we use the food-service industry and a food ordering and delivery platform such as DoorDash as a motivating scenario.

Inspired by the embedding-based representation used in the design of recommender systems [24], we adopt a latent space to represent the buyers and sellers, $V \subseteq [0, 1]^2$, with the first dimension describing food features (e.g., Italian, Japanese; spicy or not) and the second dimension the (normalized) price level (e.g., $\ldots$ $$). For a buyer $b \in B$ with latent vector $v_b \in V$, $v^0_b$ is her preference for type of food and $v^1_b$ her preferred price point; for a seller $s$, $v^0_s$ is the type of food offered, and $v^1_s$ the price of food provided. We assume the latent locations of buyers and sellers do not change over time.

A buyer generates queries according to her taste and price preferences (e.g., $$ sushi or $$ pizza), representing demand in a particular moment, and has knowledge of some subset of sellers. Given a query $q \in V$, a buyer $b$ can choose to transact off-platform with her known sellers, denoted $S_b \in S$, or with a platform-recommended seller if subscribed to the platform.
A buyer with query \( q \) who transacts with seller \( s \) has \textit{matching utility}, \( U_B(q, s) \), reflecting the matching quality. A seller \( s \) sells their product at a price \( v_s \), of which an \( \omega \)-fraction is the production cost. Therefore, the seller gets a profit of \( (1 - \omega) v_s \) from selling their product. The price of a transaction is incorporated in utility \( U_B \), as the query model takes a buyer’s price preference into account. A seller, when chosen by a buyer either via the platform or in the world, cannot decline the transaction, and a seller’s utility is independent of the identity of the buyer. For every transaction matched via the platform, the seller also pays a \textit{referral fee}, which is a fraction of the food price. See Section 4.2 for further details.

We also model a time-varying \textit{world transaction friction}, representing a cost for buyers for completing a transaction off-platform (e.g., the cost of visiting a brick-and-mortar seller). For example, during Covid-19, especially the stay-at-home periods, the transaction friction for the foodservice industry was extremely high, due to fears of sharing indoor spaces with others and the absence of dine-in options. This captures an important economic factor that can affect the platform ecosystem, and we will use reinforcement learning to study platform design under \textit{economic shocks} that correspond to changes in this friction. The decisions facing the platform agent are: (1) how to set platform fees, including buyer and seller subscription fees and the per-transaction referral rate, and (2) how to match queries from on-platform buyers to on-platform sellers.

### 4.1.1 Simulation Dynamics.

We implement our model for the platform economy in a discrete-event simulation system, and formulate an \textit{epoch-based} decision problem for agents. Specifically, time steps are grouped into \textit{epochs} of fixed length \( T \geq 1 \) (e.g., a month of 30 days) and indexed \( k \). The world transaction friction, denoted \( \mu_k > 0 \), varies from epoch to epoch, and we assume that it is observable to all agents (including the platform agent). Each buyer has a \textit{per-epoch budget}, \( \psi_b > 0 \), linearly proportional to buyer’s price preference \( v_{b}^1 \), and prevents a buyer from frequently choosing unaffordable sellers.

At the start of an epoch \( k \), the platform agent sets the fees, including buyer and seller subscription fees, denoted \( P_{B,k} \geq 0 \) and \( P_{S,k} \geq 0 \), and the per-transaction seller referral rate \( P_{R,k} \in [0, 1] \), which denotes the fraction of the transaction paid to the platform. We discuss how the platform agent uses RL to set the three fees for epoch \( k \) in Section 5.1.

Buyers and sellers observe the fees and the world transaction friction, and decide whether to pay the subscription fee in order to be able to use the platform for epoch \( k \). We denote the sets of subscribed buy and sell agents in epoch \( k \) as \( B_k \) and \( S_k \), respectively. We assume that the platform knows the locations of on-platform sellers and the queries submitted by on-platform buyers.

Beyond setting fees, we also consider a parameterized matching policy for the platform that is jointly controlled by (1) a \textit{matching utility threshold} that specifies the minimum utility that a platform match should provide to the buyer, and (2) a \textit{matching rule} that directs how to pick a seller amongst those that meet the minimum utility threshold. We discuss how the platform sets these parameters in Section 5.2.

For each time step \( t \) in epoch \( k \), we follow the “query, match, transact” order:

- **Query:** a buyer \( b \in B \) is selected to issue a query, drawn from a Gaussian distribution around their latent location, \( q_{b,t} \sim \mathcal{N}(v_b, \sigma_b^2) \), where \( \sigma_b \) specifies one’s query variance.
- **Match:** for an on-platform buyer, the platform observes \( q_{b,t} \) and matches it to an on-platform seller \( s_{p,t} \in S_k \).
Figure 2: An illustration of the epoch-based decision making for buyers, sellers, and the platform agent.

- Transact: buyer $b$ picks a utility-maximizing seller $s$ from accessible sellers, which are $\{s_{p,t}\} \cup S_b$ if $b$ is on-platform and $S_b$ otherwise, or choose to not transact if the buyer’s utility is negative (see Section 4.2).

We refer to a transaction that is matched via the platform as a \textit{platform transaction}, and otherwise we refer to a transaction as a \textit{world transaction}. For each world transaction, the buyer suffers an additional cost of $\mu_k$, which reflects the world transaction friction. For each platform transaction, the seller also pays a \textit{referral fee}, defined to be a $P_{R,k}$ fraction of the seller’s price.

At the end of epoch $k$, the platform agent evaluates the revenue made through subscriptions and referral fees, and buyers and sellers evaluate their surplus from transactions and fees paid to the platform (see Section 4.3). Each seller also has a \textit{shutdown threshold}, $\lambda_s \in \mathbb{N} > 0$, and will go bankrupt if they cannot obtain positive surplus for a consecutive $\lambda_s$ epochs. Once bankrupt, a seller is unable to serve buyers in future epochs.

Figure 2 summarizes dynamics within an epoch. We describe the buyer choice model in regard to choosing a transaction in Section 4.2 and the way buyers and sellers decide to join the platform for the upcoming epoch in Section 4.4. We formalize the platform design problem in Section 5.

### 4.2 The Buyer’s Transaction-Level Decision Problem

We model buyers as choosing a transaction from those available to maximize matching surplus. For this, they have choices amongst sellers known to them off-platform, and if the buyer is on-platform, the buyer can also choose to transact with the seller matched by the platform to the submitted query.

#### 4.2.1 Choice in the world

In regard to world transactions, a buyer $b$ with query $q_{b,t}$ can choose from her known-sellers in the world whose prices are within her epoch-budget left at time $t$, denoted $\psi_{b,t}$, i.e., $S_{b,t} := \{s \in S_b : v_s^1 \leq \psi_{b,t}\}$. For $S_{b,t} \neq \emptyset$, the best off-platform choice is with the seller $s^*_w := \arg\max_{s \in S_{b,t}} u_B(q_{b,t}, s) - \mu_k$. But the world friction $\mu_k$ could be high enough to prevent buyer transacting even with the

---

1 An on-platform buyer may also choose to complete a world transaction if a platform’s suggested match is not good. Moreover, a world transaction does not require the seller not to be on-platform. Rather, we allow a buyer to transaction through a world transaction with a seller known to the buyer, whether or not the seller is on the platform. In this way, a seller on the platform is also able to complete off-platform transactions.
most suitable world seller: let $s^w_{b,t}$ be $s^*_{b,t}$ if $u_B(q_{b,t}, s^*_{b,t}) > \mu_k$, and $\phi$ otherwise or in case $S_{b,t} = \emptyset$. Writing $u_B(q_{b,t}, \phi) = 0$, the \textit{world surplus} available to buyer $b$ in period $t$ is

$$u^w_{b,t} = \max(u_B(q_{b,t}, s^w_{b,t}) - \mu_k, 0).$$

The above choice and corresponding surplus are the same regardless of whether buyer $b$ is on the platform or not.

\textbf{4.2.2 Choice on the platform}

An on-platform buyer $b$ first considers whether the platform-recommended seller $s^p_{p,t}$ is within her remaining budget $\psi_{b,t}$. Given $s^p_{p,t}$ is within $b$’s budget, the \textit{platform surplus} available to an on-platform buyer $b$ in time $t$, $u^p_{b,t}$, is $u_B(q_{b,t}, s^p_{p,t})$, and 0 otherwise.

\textbf{4.2.3 Overall buyer choice}

Putting this together, buyer $b$ chooses $s^w_{b,t}$ in the case it being off-platform, and the better one between $s^w_{b,t}$ and $s^p_{p,t}$ when it is on-platform. We assume that the buyer breaks ties in favor of the match offered by the platform. In either case, if no available seller grants buyer a positive utility, the will might be $\phi$, and a buyer who chooses not to transact.

We write $s_{b,t}$ to denote the choice of the buyer at time $t$, and denote a buyer’s query, options, and transaction as 4-tuple $(q_{b,t}, s^w_{b,t}, s^p_{p,t}, s_{b,t})$, where $s^p_{p,t}$ is $\phi$ in case of an off-platform buyer. The surplus to the buyer $b$ is $r_{b,t} = \max\{u^w_{b,t}, u^p_{b,t}\}$. Let $I^w_{b,t}$ be an indicator to whether buyer $b$ transacted with the world or not at time period $t$. We define the \textit{world matching surplus} and \textit{platform matching surplus} for buyer $b$ at timestep $t$ as $r^w_{b,t} = u^w_{b,t} \cdot I^w_{b,t}$ and $r^p_{b,t} = u^p_{b,t} \cdot (1 - I^w_{b,t})$, respectively.

We note the possibility that $s^p_{p,t}$ and $s^w_{b,t}$ end up being the same seller, and with the world transaction friction $\mu_k > 0$, the buyer will choose to transact via the platform.

A seller cannot decline a transaction, and seller $s$ has surplus (i.e., net profit) in period $t$ of,

$$r_{s,t} = \begin{cases} 
        v_1^s (1 - \omega_s - P_{R,K}) & \text{for a platform transaction,} \\
        v_1^s (1 - \omega_s) & \text{for a world transaction,} \\
        0 & \text{otherwise.} 
\end{cases}$$

Where again $\omega_s$ is and $P_{R,K}$ is seller’s cost fraction and platform’s seller referral rate respectively.

We let $n^p_{s,k}$ denote the number of transactions completed by seller $s$ via the platform during epoch $k$, and $n^w_{s,k}$ the number of transactions completed by seller $s$ in the world.

\textbf{4.3 Epoch Surplus and Epoch Platform Revenue}

We calculate agent surplus and platform profit at the end of every epoch. Buyer $b$’s \textit{epoch surplus}, $r_{b,k}$, in epoch $k$ is her total surplus from matching minus any subscription fee paid:

$$r_{b,k} = \sum_{t \in k} r_{b,t} - I^w_{b,k} P_{B,k} = \sum_{t \in k} r^w_{b,t} + \sum_{t \in k} r^p_{b,t} - I^w_{b,k} P_{B,k},$$

Where $r^w_{b,t}$ and $r^p_{b,t}$ are the world and platform surplus for buyer $b$ at timestep $t$, respectively.
where $I_{p.b,k} \in \{0, 1\}$ is an indicator to denote whether the buyer is on- or off-platform, $P_{B,k}$ being platform fee for buyer subscription. This surplus decomposes into surplus from world transactions $r_{b,k}^{w}$ and surplus from platform transactions $r_{b,k}^{p}$, and we define the total buyer surplus generated by the platform in epoch $k$ as $r_{B,k}^{p} = \sum_{b \in B} r_{b,k}^{p}$, and the total buyer surplus from the world in epoch $k$ as $r_{B,k}^{w} = \sum_{b \in B} r_{b,k}^{w}$.

A seller agent’s epoch surplus, denoted $r_{s,k}$, in epoch $k$ is her total net profit from transactions during epoch $k$ minus any subscription fee paid:

$$r_{s,k} = \sum_{t \in k} r_{s,t} - I_{p.s,k} P_{S,k} = n_{w,s,k} v_{s}^{1} (1 - \omega_{s}) + n_{p,s,k} v_{s}^{1} (1 - \omega_{s} - P_{R,k}) - I_{p,s,k} P_{S,k},$$

(4)

where $I_{p,s,k} \in \{0, 1\}$ is an indicator to denote whether the seller is on- or off-platform, $P_{S,k}$ being the platform fee for seller subscription. We define the total seller surplus from platform transactions in epoch $k$ as $r_{S,k}^{p} = \sum_{s \in S} r_{s,k}^{p}$ and the total seller surplus from world transactions in epoch $k$ as $r_{S,k}^{w} = \sum_{s \in S} r_{s,k}^{w}$.

The total platform profit in epoch $k$ is the sum of the subscription and referral fee it charges

$$r_{p,k} = \sum_{b \in B} I_{p.b,k} P_{B,k} + \sum_{s \in S} \left( r_{s,k}^{p} P_{S,k} + n_{p,s,k} v_{s}^{1} P_{R,k} \right).$$

(5)

4.4 Buyer and Seller Subscription-Level Decision Problem

At the start of each epoch $k + 1$, each buyer and seller chooses whether or not to subscribe to the platform. This decision is utility-theoretic, and depends on (1) estimating the surplus from joining vs. operating off platform, based on the observed new fees and world transaction friction, and (2) an agent-specific inertia that captures the extent to which an agent becomes affiliated with a particular transaction channel due to its previous decisions. As often seen in practice, an agent might stick to their current transaction channel even though it’s beneficial for them to move to a different one (see related works section for more details).

4.4.1 Estimating the Effect of a Subscription Decision

To reason about whether to be subscribe to the platform in epoch $k + 1$, each buyer and seller contemplates about the surplus that they would gain if on platform or off platform, for the newly proposed fees and the world transaction friction for the next epoch. For this, a buyer assumes that the queries submitted to the platform by other buyers, itself, and the on-platform sellers are the same as that in epoch $k$. Let $\xi_{b,k+1}^{w}$ and $\xi_{b,k+1}^{w}$ denote the predicted $k + 1$ epoch surplus values assuming the buyer does and does not subscribe, respectively. A seller assumes that the queries submitted to the platform by all buyers, and the on-platform sellers are the same as that in epoch $k$. The seller then calculates the surplus for possible subscription decisions, where $\xi_{s,k+1}^{w}$ and $\xi_{s,k+1}^{w}$ are the predicted $k + 1$ epoch surplus values when a seller subscribes and does not subscribe, respectively. Buyers and sellers might have access to the information either from previous trial periods where they gain platform experience, or by getting various statistics and estimates from the platform itself, which opts to improve the user experience. A complete description of the calculation of these counterfactual processes is deferred to Appendix C.
4.4.2 Agent-Specific Decision Inertia

We also model a buyer or seller’s decision inertia, in considering the preference for sticking with their same decision (whether off-platform or on-platform) as reflected in recent choices. A buyer, for example, becomes more likely to subscribe if they are already a subscriber. We model inertia logarithmically in the number of epochs a buyer or a seller sticks with their decision, and ‘resets’ when they change their decision. The inertia is added to the surplus term as common in the literature [8, 17, 10]. Based on this adjusted surplus, agents decide whether to subscribe or not according to probabilities inferred by the standard discrete-choice logit model [8, 17]. The exact way in which inertia is introduced into our model is depicted in Appendix D.

5 Solving the Platform Design Problem

Our modeled platform ecosystem include a common component, represented by the world transaction friction, and private components, represented by agent-specific knowledge and preferences. Many of these aspects are sequential in nature (e.g., shocks on transaction friction, agent inertia built from prior choices, seller shutdowns), which results in a sequential decision problem under uncertainty for the platform agent.

We study the pricing and matching decision problems faced by a platform agent at each epoch, using reinforcement learning to set platform fees and choose how to match queries to platform sellers. Buyers’ knowledge, as well as their transactions with world sellers, are private and thus not observable to the platform. For this reason, we formulate the platform design problem as a partially observable Markov decision process (POMDP), and learn the respective pricing and matching policies based on observations of on-platform agents, including their queries and transactions.

5.1 Learning to Set Platform Fees under Myopic Query Matching

We first model the problem of setting fees, for a platform that uses myopic query matching (i.e., recommend the closest on-platform seller to a query, and thus yield highest utility to the buyer).

A POMDP [16] can be formally described as a 6-tuple \((\mathcal{X}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \mathcal{O}, \mathcal{O})\), with the state space \(\mathcal{X}\), action space \(\mathcal{A}\), a Markovian state-action-state transition probability function \(\mathcal{P}\), and a reward function \(\mathcal{R}\) the same as for an MDP. Instead of seeing the true state \(x\), the agent receives an observation \(o \in \mathcal{O}\), generated from the underlying state according to the probability distribution \(o \sim \mathcal{O}(x)\). We describe each component of the POMDP and define the platform agent’s pricing policy. Here the platform is making decision for epoch \(k\) based on experience from epoch \(k - 1\).

- The state \(x_k \in \mathcal{X}\) at the start of epoch \(k\) (before the platform sets fees) is defined by
  
  1. buyer attributes: the latent location, epoch budget, query distribution, and knowledge of sellers,
  2. seller attributes: the latent location, cost fraction, and shutdown threshold,
  3. agent subscription states: either on- or off-platform for the past epoch, \(T_{B,k-1}^p\) and \(T_{S,k-1}^p\),
  4. the agent inertia levels: \(\chi_{b,k-1}\) and \(\chi_{s,k-1}\),
  5. a sequence of query, seller candidates, and buyer’s choices of previous epoch:

\[
Q_{k-1} = \{(q_{b,t}, s_{b,t}^p, s_{b,t}^w)\}_{t \in k-1, b \in B},
\]
6. the shutdown states for sellers: whether a seller has shut down at the end of epoch $k - 1$, $I_{S,k-1}$,
7. the platform fees for the past epoch: $P_{B,k-1}, P_{S,k-1}, P_{R,k-1}$, and
8. the world transaction friction for the current epoch: $\mu_k$.

- An action $a_k = (P_{B,k}, P_{S,k}, P_{R,k})$ defines the fees for the upcoming epoch $k$. We model a discrete action space $\mathcal{A}$ where fees take discrete values at integer multiples of a tick (or percentage) size.

- For the state transition $P : \mathcal{X} \times \mathcal{A} \rightarrow \Delta(\mathcal{X})$, we assume that agent attributes remain the same across epochs. Buyers and (viable) sellers follow their choice model to subscribe to the platform (Section 4.4), leading to new subscription states and inertia levels. For each time step $t \in k$, we follow the simulation dynamics in Section 4.1: (1) a buy agent generates a query, (2) if the buyer is on platform, the platform recommends the platform seller whose latent location is the closest to the query (i.e., myopic query matching), and (3) the buyer selects a seller to transact (Section 4.2). This gives a full sequence $Q_k$. Each viable seller may shut down based on the surplus received in epoch $k$ and her shutdown threshold. Fees follow naturally by the actions taken, and the world transaction friction evolves accordingly. The above altogether gives a new state $x_{k+1} \sim P(x_k, a_k)$.

- A reward $r_k \sim R(x_k, a_k)$ is provided at the end of an epoch $k$, when agent subscription and transaction outcomes are available. The reward is set to model different design objectives, e.g., platform revenue, a combination of revenue and other suitable proxy measures of the platform.

- The platform’s observation $o_{k+1} \in \Omega$ consists of most elements in a state, except for agent private knowledge on the set of known sellers $S_b$, and their matching and transacting experiences in the world (i.e., the full sequence of $Q_k$). Instead, the platform observes the sequence of queries generated by on-platform buyers, as well as their decisions on whether or not to transact via the platform (but not who they transact with if transacting off platform). We denote the platform observable sequence as $Q^p_k := \{(q_{b,t}, s^p_{b,t}, I\{s_{b,t} = s^p_{b,t}\})\}_{t \in k, b \in B_k}$.

The platform agent’s goal is to learn a pricing policy $\pi(a|o_k)$ over the action space to maximize the discounted cumulative reward across different episodes. An optimal policy in a POMDP requires the action to be taken depending on the entire history of observations, which we denote $h_k := \{o_0, a_0, ..., a_{k-1}, o_k\}$. Following the success of using representation learning to solve POMDPs [13, 31, 12], we use neural networks to learn sufficient statistics of the history, these denoted $T(h_k)$. Altogether, we use deep reinforcement learning to learn the platform’s pricing policy $\pi(a|o_k; \theta)$, specifically parameters $\theta$ of a neural network that extract $T(h_k)$ and map to actions to maximize the platform objective,

$$\max_{\pi} \quad \mathbb{E}_{o \sim \pi, x \sim P} \left[ \sum_{k=0}^{K} \gamma^k r_k \right],$$

where $\gamma$ is the discount factor, and $K = |\tau|$ is the total number of epochs, and $r_k$ the reward to the platform. In this work, we consider platform revenue as well as alternately various socially-aware objectives as the reward to the platform. We discuss the algorithm that we use to solve this POMDP and give additional implementation details in the experiments section.
5.2 Learning to Match Queries to Platform Sellers under Fixed Fee Schedules

Besides the myopic matching described above, which always match a query to the buyer’s utility-maximizing on-platform seller, here we model platform changing the way it matches queries: instead of benefiting the buyer, it may direct queries to benefit some specific sellers or the platform itself. For simplicity we assume platform fees are set through regulatory means and the platform is only able to change the way it matches.

To reflect the intuition of “matching in favor of one party in the economy”, we define the platform’s matching strategy by two parameters: (1) a matching utility threshold $\eta \in [0, 1]$, that specifies the minimum utility that a recommended seller should provide to the buyer, as a fraction of the buyer’s utility for the myopically-optimal match, and (2) a matching rule, which directs how to pick a seller amongst those that meet this utility threshold. We consider:

- **The seller-aware rule**: Among sellers that meet the utility threshold, match a query to the seller who has achieved the smallest surplus on the platform so far during the epoch, breaking ties in the buyer’s favor (for detailed implementation, see Algorithm 1).

- **The profit-driven rule**: Among candidates, sellers that meet the utility threshold, match a query to the seller who brings the largest revenue to the platform, breaking ties in the buyer’s favor (given referral fees, in our setting, this is the most expensive, on-platform seller).

These are two extreme matching policies, designed to be representative of a space of possibilities and also to complement the myopic matching policy (which can be considered “buyer-aware”). The seller-aware rule is designed to increase sales to sellers who have been benefiting less from the platform in order to promote a more diverse set of sellers and in the long-run drive more platform revenue as collected through fees. The profit-driven rule is at the other end of the spectrum, aiming to maximize the platform’s myopic transaction revenue without concern to longer-term effect.

Our goal is to learn a platform matching policy that chooses a matching rule and its corresponding matching utility threshold for each epoch (i.e., the choice of rule and $\eta$ can vary by epoch). Note whichever the rule, a matching policy with $\eta = 1$ is simply the myopic query matching. We are particularly interested to understand how a (long-term) profit-maximizing platform chooses its matching policy under different fee regimes, and in particular how well the chosen matching policy may align with the welfare and resilience considerations of the overall economy.

Previously, the fee-setting myopic matching platform decides the fee for an epoch before buyers and sellers join the platform. Here with fixed fees, the platform decides its matching policy for an epoch after buyers and sellers subscribe. This is sensible because we consider the strategy to be internal to the platform, whereas fees are public information. Given this change, several adjustments are made to define each component of the matching POMDP, relative to the pricing POMDP in Section 5.1. For details see Appendix E.

6 Empirical Study of Platform Dynamics

6.1 Environment Settings for the Platform Economy

We evaluate the effect of a platform under three types of market structures, represented by distinct latent locations of buyers and sellers:

\[\text{Here, we assume the platform knows a seller’s fractional production cost, and thus can calculate its surplus from transactions, given the price point and the set referral rate. In practice, this can be based on market estimations.}\]
- **Uniform:** Buyers and sellers are uniformly distributed, $v \sim U[0,1]^2$, representing diverse buyer interests and diverse seller attributes.

- **Core-and-Niche:** There is a propensity for buyers and sellers to locate towards the center, $v \sim \text{Truncated Gaussian}(\mu, \sigma, 0, 1)$ with $\mu = [0.5, 0.4]$ and $\sigma = 0.2$, and some niche agents to locate away from the market center. We choose the distribution to be slightly biased towards lower prices.

- **Two-Core:** One group of agents (buyers and sellers) is centered around relatively cheap options, Truncated Gaussian($\mu_1, \sigma_1, 0, 1$) with $\mu_1 = [0.7, 0.3]$ and $\sigma_1 = 0.17$, and another group around more expensive ones, Truncated Gaussian($\mu_2, \sigma_2, 0, 1$) with $\mu_2 = [0.3, 0.7]$ and $\sigma_2 = 0.17$. The two cores have different attributes, giving a right-down and left-up clustering structure.

Across all market structures, we consider environments with 10 buyers and 10 sellers. Figure 3 visualizes a representative latent location of buyers and sellers for each market typology. Each agent has an initial preference of joining the platform, sampled uniformly random from a range, $\chi \sim U[-2, 2]$. We vary the knowledge level in our experiments, with $p$ to represent the average fraction of sellers that a buyer knows in the world, and we use $\text{Bern}(\rho)$ for each buyer-and-seller pair to generate the set of known sellers $S_b$ for buyer $b$. Each seller has her fractional cost of transaction, $\omega_s \sim U[0.2, 0.4]$, and we set the shutdown threshold $\lambda = 2$ for all sellers, meaning that a seller will go out of business after two consecutive epochs with non-positive surplus.

Each simulation run, or episode, lasts $K = 12$ epochs, and each epoch contains $T = 100$ timesteps. Within each timestep $t$, a buyer $b$ arrives and submits a query around location $v_b$, $q_{b,t} \sim \mathcal{N}(v_b, \sigma^2_b)$, where $\sigma^2_b = 0.02$. Buyers arrive in a round robin fashion, so that each submits $N = 10$ queries per epoch. We set each buyer’s per-epoch budget proportional to $N$ and her price point in the latent space, i.e., $\psi_b = Nv_b$. When query $q$ is fulfilled by seller $s$, the buyer receives a matching utility $u_B(q,s) = \exp(-c\|q - v_s\|_2^2)$, where we choose $c = 2$ to have matching utilities span $[0,1]$.

We study platform design in the presence of an economic shock, where the cost of off-platform fulfillment (world transaction friction) surges, modeling an event such as the Covid-19 pandemic. World transaction friction $\mu_k \in [0,1]$ varies across the 12 epochs to represent the full cycle of an economic shock with three stages: pre-shock, shock, and post-shock. To facilitate comparing the platform performance, we fix the pre-shock and post-shock stages to each last for three epochs and have low world friction, $\mu_k = 0.1$. The shock stage is controlled by a shock intensity parameter $I \sim U[I_{\min}, I_{\max}]$, specifying the largest value that can be attained by $\mu$ during shock stage. We sample world transaction frictions, $\mu_k$, for epochs within the shock stage from Lognormal($\mu = 0, \sigma = 0.5$), and normalize these values according to the shock intensity $I$. Figure 5 (red line) shows the average shock schedule for intensity $I \sim U[0.8, 1]$.

We sample initial states for our experiments with a warm-up epoch (i.e., in addition to the twelve epochs), where the world transaction friction is 0.1, the platform charges no fees and uses myopic query matching, and buyers and sellers join the platform based on their initial preferences $\chi$. Agents’ experience in this warm-up epoch provides a basis for the platform to choose actions, and the buyers and sellers to form estimates to guide their subscription decisions. For a platform agent that sets fees, its action space is divided into three subspaces, one for each platform fee: both registration fees $P_{R,k}$ and $P_{S,k}$ range from 0 to 10 and discretized at 0.2, and the seller referral rate $P_{R,k}$ ranges from 0 to 1 and discretized at 0.1. For a platform agent that instead decides how to match queries to sellers, the action space is a combination of the choice of matching rule
Figure 3: An illustration of three representative latent locations of buyers and sellers for each market structure.

(a) Uniform.  
(b) Core-and-Niche.  
(c) Two-Core.

Figure 4: Total welfare and buyer and seller surplus achieved in environments that vary in $\rho$ and $\mu$, with and without a platform under the Core-and-Niche market structure. Results presented are the average of a hundred runs for each environment.

(a) Varying knowledge level about sellers $\rho$, with world transaction friction $\mu = 0.6$.  
(b) Varying the world transaction friction $\mu$, with knowledge level about sellers $\rho = 0.2$.  

6.2 Characterizing the Environment: Value Created by the Platform

We start by building some intuition about the basic economics of our simulation environments. We characterize the values a profit-maximizing platform generates across a range of single-epoch, no-shock environments that vary in (1) knowledge level about sellers $\rho$, and (2) the world transaction friction $\mu$. The warm-up epoch is retained to facilitate agents’ subscription decisions.

For each market structure, we generate three samples of latent locations of buyer and seller agents, and for each latent sample and knowledge level $\rho \in (0, 1)$, we sample ten different knowledge matrices, specifying which sellers are known by each buyer. For a given environment (defined by a sampled latent location of agents, a knowledge structure, and a world friction), we use Bayesian
Optimization [20] (BO) to find platform fees that maximize the platform’s revenue, and conduct control experiments on the same environment with and without a platform.

Figure 4 shows the total welfare, as well as buyer and seller surplus, achieved in environments with and without a platform under the Core-and-Niche market structure. We defer results for the other two market structures to the Appendix G. We normalize welfare and surplus by the total welfare achieved in an ideal world where buyers have complete knowledge about sellers and there is no world transaction friction. As one may expect, under no platform scenarios (Figure 4 red lines), total welfare increases as buyers’ knowledge level about sellers increase and the world friction decreases. Across all environments varying in $\rho$ and $\mu$, a revenue-maximizing platform consistently increases total welfare relative to the absence of a platform, creating value by reducing search costs (i.e., matching buyers to unknown sellers) and facilitating transactions (i.e., circumventing off-platform fulfilment costs). The amount of revenue a platform can extract (i.e., differences between the green and blue lines) increases, as buyers have less knowledge about sellers and as the world transaction friction increases. For scenarios when $\rho$ is extremely low or $\mu$ is very high, the platform possesses large market power and may end up extract all surplus from buyers and sellers.

6.3 Platform Design under Dynamic Environments with Shocks

This section investigates platform design in the presence of an economic shock. We use RL to model the behavior of a rational platform, optimizing for different design objectives, and evaluate its effect on the efficiency and resilience of the overall ecosystem. Following our POMDP formulation in Section 5, we make the following observations available to the platform agent:

• On-platform buyers and sellers, represented by two binary vectors, and their latent locations,

• Summary statistics of on-platform agents, including the number of platform transactions and platform surplus accumulated so far within an epoch,

• The platform matching and transaction matrix between on-platform buyers and sellers for the past epoch,

• The platform fees, the matching rule and utility threshold (if learn matching), and the current epoch’s world friction.

For both the pricing and matching policy, we use Advantage Actor-Critic (A2C) [27, 7] to optimize for a discounted sum of rewards (Eq. 6), whether this represents revenue or other blended objectives. Results of learned policies reported in this section are based on the average performance of two models trained from different torch seeds and 100 controlled test episodes. We defer a full description of the neural network structure and training hyperparameters to Appendix F.

6.3.1 Platform Fee Design

We consider a platform that uses RL to set fees under different design objectives, with their per-epoch rewards as follows:

• Platform revenue, i.e., $r_{p,k}$ as specified in Eq. (5).

• A combination of revenue and on-platform user surplus, i.e., $r_{p,k} + \alpha (r_{b,k}^p + r_{s,k}^p)$ with $\alpha = 0.5$. 

16
Figure 5: Total welfare, decomposed into platform revenue, buyer and seller surplus from platform transactions, and buyer and seller surplus from world transactions, across the 12 epochs in the Core-and-Niche market without a platform (Left), with a revenue-maximizing platform (Middle), and a platform that maximizes a combination of revenue and on-platform user surplus (Right). Revenue and surplus in the y-axis are normalized by the total welfare of an ideal world where buyers have complete knowledge and there is no friction or fees.

- A combination of revenue and number of platform buyers, i.e., \( r_{p,k} + \alpha |B_k| \) with \( \alpha = 3 \).
- A combination of revenue and number of platform sellers, i.e., \( r_{p,k} + \alpha |S_k| \) with \( \alpha = 3 \).
- A combination of revenue and number of platform transactions, i.e., \( r_{p,k} + \alpha \sum_{s \in S} n_{s,k}^p \) with \( \alpha = 0.3 \).

The last three objectives aim to provide suitable proxy measures for both the platform and regulator to consider, whereas the on-platform user surplus may be hard to estimate in practice and is used instead to provide a reference in our experiments. To facilitate the comparison, we choose \( \alpha \) to make these blended metrics have approximately equal weight across objectives. Each simulated episode includes a pre-shock stage (epoch 1-3), a shock stage (epoch 4-9) where we sample the shock intensity \( I \sim U[0.8, 1] \), and a post-shock stage (epoch 9-12).

To start, we examine the following three settings as depicted in Figure 5: (1) a world without a platform (as a baseline), (2) a world with a revenue-maximizing platform, and (3) a world with a platform that maximizes a combination of revenue and on-platform user surplus. The main observations are as follows. First, the presence of a platform during shock benefits the overall welfare. The platform opens up alternative channels of transaction during shock, which otherwise would not have been possible due to high world friction.

Second, a platform that cares about on-platform user surplus is able to maintain the overall welfare even after shock, while the purely revenue-maximizing platform is not. Higher post-shock welfare is due to more non-bankrupt sellers, as depicted in Figure 6. The figure captures the market situations in the last epoch of an 12-epoch episode, under the two platforms. For the revenue-maximizing platform, more cheap and niche sellers go bankrupt, resulting in buyer queries to be matched to less-preferred sellers (e.g., queries from \( b0 \) are matched to \( s2 \), queries from \( b8 \) are matched to \( s2 \)). It quite often turns out the less-preferred sellers are also of higher prices, and hence the platform makes higher revenue from the referral fees, at the cost of buyer and seller surplus. We seldom observe this phenomenon for the platform that cares about user surplus as well.
Compare Different Design Objectives We present our main findings on the effect of platform pricing policies induced by different design objectives in Figure 7. Figure 7(a) shows the welfare decomposition, grouped across epochs within each shock stage, achieved by platforms that optimize for different objectives. We find that except for the surplus-aware platform, a platform that also optimizes for number of on-platform sellers can retain a similar level of post-shock welfare compared to its pre-shock period. Interestingly, this also helps to restore economic health even off-platform: having benefited from the platform in avoiding bankruptcy, sellers who adopted the platform due to the surge in friction can leave the platform gain after shock.

Such observation can be verified in Figure 7(b), which plots the average number of on- and off-platform agents, as well as the number of bankrupt sellers. As we expect, across all settings, there are more buyers and sellers subscribed to the platform during the shock, reflecting the larger market power that the platform possesses and greater need on part of agents to avoid the surge in off-platform fulfillment cost. Depending on the design objective, a platform may set fees differently to take advantage of this market power, affecting buyers’ matching utility and sellers’ bankruptcy in the economic system (see Appendix Fig. 14 for registration fees and referral rates charged under different platform objectives). We further present an ablation analysis on how the platform has learned to raise fees in response to shock and the built-up agent inertia.

Overall, our simulation results establish that the surplus-aware and seller-aware platform designs may help restore a comparable level of total welfare after the shock, reducing the impact of a market shock on the economic system and protecting some of the sellers from bankruptcy.

Interpret Learned Pricing Policies We conduct simple ablation study to demonstrate what the revenue-maximizing platform agent has learned to respond to environments with two distinct sequential decision making factors: shock, and buyer and seller’s inertia of staying on or off platform. For simplicity, we consider a revenue-maximizing platform in the Core-and-Niche market that learns to set the buyer registration fee, while fixing seller fees ($P_{S,k} = 1.0$ and $P_{R,k} = 0.2$). The price set by platform per epoch is plotted in Figure 8.

---

4We defer results for the other two market structures to Appendix H.
(a) Welfare decomposition achieved by different learned pricing policies in each shock stage.

(b) The number of on- or off-platform agents and bankrupt sellers in each shock stage.

Figure 7: The welfare decomposition and buyer and seller states induced by different design objectives under the Core-and-Niche market structure.

The first set of experiment contrasts learning with or without inertia when no shock (i.e., $\mu_k = 0.1$) is present in the system. Represented by the green and blue lines, the platform sets an almost constant registration price when there is no inertia, while gradually increasing it when buyers become more reliant on the platform with inertia.

The second experiment reveals that the existence of shock not only influences the pricing behaviour during the shock periods, but also the pre-shock ones. The red line represents pricing policy trained in an environment always with shock. Knowing that buyers will go on platform when the shock occurs, it can afford to set a higher price in the pre-shock epochs. This behaviour contrasts drastically with the orange line, which is a pricing policy learned when shock occurs on one half of the training episodes, and tested in a shock environment. Not sure about the existence of future shock, it sets the pre-shock price to be almost equal to that of no-shock, and raises it when shock actually occurs.

Figure 8: Learned buyer registration fee under different sequential factors.
### Characterize Bankrupt Sellers

We next look in detail at which sellers are more likely to go out of business, and under what platform design objectives. For the Core-and-Niche market, we classify sellers into three groups: **core sellers** (within one standard deviation of the center and with at least two buyers nearby, e.g., s2, s3, s4, s7 in Figure 3b), **niche sellers** (beyond two standard deviation from the center and with at most one buyer nearby, e.g., s0, s5, s6 in Figure 3b), and **cheap sellers** (with price lower than 0.2, e.g., s1, s8, s9). For the Two-Core market, we simply group sellers according to the core from which a seller is sampled.

Table 1 summarizes the average shutdown frequency of each seller group under different platform pricing policies. We find that under many choice of design objectives, one group of sellers can suffer a substantially higher rate of bankruptcy than the other groups (e.g., especially cheap sellers in both market structures). In effect, a platform may only care about the group of sellers who can bring a large amount of demand, and thus revenue to the platform. This motivates the consideration of introducing diversity metrics into regulatory enforcement efforts in order to promote a diverse and healthy platform economy. In our simulation environments, we explore an additional design objective for the two-core market structure, which specifies a combination of platform revenue and the product of on-platform sellers from each core with $\alpha = 1.2$ (Table 1 Rev. + Diversity). We observe it reduces the bankrupt probability of both cheap and expensive sellers dramatically. The low bankrupt probability with high social welfare (See Appendix I) indicate seller diversity could be another favorable design objective in practice.

### 6.3.2 Platform Query Matching

We next examine the effect of a platform that uses RL to learn a matching policy under fixed fees. We are interested in (1) characterizing the matching policy (i.e., the matching rule and the matching utility threshold) of a revenue-maximizing platform under different fee regimes, (2) evaluating the effectiveness of this matching compared to myopic matching, for both profit and welfare, and (3) understanding to what extent revenue-maximizing matching is aligned with promoting efficiency and resilience for the overall economy, for example as regarding what a regulator might care about.

We focus on the same two-core environments in Section 6.3.1 and consider nine different fee regimes, that vary in the seller registration fees, $P_S \in \{2, 3, 4\}$, and the referral rates $P_R \in \{0.1, 0.3, 0.6\}$. These fee regimes are chosen to represent a range of possible fee structures in order to understand the effect of regulating fees while still allowing a platform to choose how to match. We fix the buyer registration to $P_B = 1.8$ to control the least amount of buyer satisfaction that the platform will need to provide.
Before using RL, it is instructive to compare the platform revenue and total welfare for different, fixed choices of matching utility threshold and for each of the seller-aware and profit-driven rule (i.e., fixing the matching strategy throughout an episode). Table 2 summarizes the results for each of the nine fee regimes, comparing the platform revenue and welfare for each of a revenue-maximizing and welfare-maximizing objectives, with the myopic matching as a baseline. At a high level, we see strong alignment between platform objectives (revenue) and broader objectives (welfare), as can be seen by the strong correlation between the revenue-maximizing and welfare-maximizing objective: they tend to yield the same parameterized matching policy. We also find that under all the low referral rate regimes, the platform chooses to adopt the seller-aware matching rule, which reflects the alignment between seller surplus and the platform’s revenue made from seller subscriptions. We also notice that the profit and welfare incentives may not align under the high referral rate regimes (bolded in Table 2), where the platform starts to choose the profit-driven rule, making more revenue from higher referral fees rather than an additional seller subscription.

Table 2: Platform revenue and total welfare achieved by platform economies that are respectively mediated by a revenue-maximizing, a welfare-maximizing, and a myopic platform matching policy. The s or p after \( \eta \) denotes whether the chosen matching policy follows the seller-aware or profit-driven rule. In most price regimes, we see strong alignment between platform objectives (revenue) and broader objectives (welfare). In bolded pricing regimes with high referral rates, misalignment can occur.

| Price regimes | Rev-max. obj | Welfare-max. obj | Myopic matching (\( \eta = 1 \)) |
|---------------|--------------|-----------------|-------------------------------|
|               | \( \eta \) & rule | Revenue | Welfare | Revenue | Welfare | Revenue | Welfare |
| low \( P_S \), low \( P_R \) | 0.9s | 285 (1.3) | 1040 (3.1) | 0.9s | 285 (1.3) | 1040 (3.1) | 282 (1.4) | 1036 (3.3) |
| low \( P_S \), med \( P_R \) | 0.9s | 311 (1.4) | 1000 (2.7) | 0.9s | 311 (1.4) | 1000 (2.7) | 307 (1.5) | 989 (3.3) |
| low \( P_S \), high \( P_R \) | 0.7s | 282 (2.4) | 818 (4.1) | 0.4p | 252 (2.1) | 823 (4.7) | 272 (2.6) | 770 (4.8) |
| med \( P_S \), low \( P_R \) | 0.8s | 303 (1.6) | 994 (3.2) | 0.8s | 303 (1.6) | 994 (3.2) | 298 (1.6) | 978 (4.1) |
| med \( P_S \), med \( P_R \) | 0.7s | 309 (1.8) | 941 (3.4) | 0.7s | 309 (1.8) | 941 (3.4) | 304 (1.9) | 922 (4.2) |
| med \( P_S \), high \( P_R \) | 0.8p | 259 (2.4) | 776 (6.7) | 0.5p | 246 (2.2) | 796 (3.9) | 258 (2.1) | 765 (6.5) |
| high \( P_S \), low \( P_R \) | 0.9s | 307 (1.7) | 941 (4.3) | 0.8s | 306 (1.9) | 942 (4.4) | 301 (1.8) | 932 (4.1) |
| high \( P_S \), med \( P_R \) | 0.7s | 301 (1.9) | 899 (3.9) | 0.7s | 301 (1.9) | 899 (3.9) | 291 (2.2) | 866 (6.2) |
| high \( P_S \), high \( P_R \) | 0.8p | 246 (1.8) | 772 (3.9) | 0.5p | 229 (1.9) | 780 (3.0) | 244 (1.9) | 765 (5.2) |

Figure 9 details the revenue and welfare outcomes under two of these nine fee regimes, where the platform now also makes use of RL to set its matching policy in each epoch. Under the high \( P_S \) low \( P_R \) regime, we observe perfect alignment between platform objectives (revenue) and broader objectives (welfare), even with the RL matching policy; whereas under the low \( P_S \) high \( P_R \), incentives may not align especially when a profit-driven matching rule is picked (i.e., right half of Figure 9b). For the learned RL matching policies, we find in both fee regimes before the shock, the platform agent learns to adopt a relatively low matching utility threshold with the seller-aware rule to attract sellers on to the platform. As the shock decays, the platform agent in the low referral regime tends to increase the matching utility threshold to retain buyers with better quality matches, whereas in the high referral regime, it is more inclined to use the profit-driven matching rule and attempts to extract revenue from high price sellers from epochs to epochs. We provide a visualization of the two learned matching policies in Appendix Fig. 10.
7  Future Directions

In this paper, we made a first stab of the study of platform economy in highly dynamic settings and in face of significant economic shock. While we tried to use a realistic and detailed model, we think there are plenty of extensions that can be made. First of all, we assumed the platform has perfect information regarding the location of sellers and queries in the latent space. It will be natural to consider settings where the platform does not hold such information, and needs to gather it as it learns. In addition, the underlying latent space need not be fixed, and can change over time, as buyers and sellers adjust their preferences. When training the platform’s matching policy, we restrict the learned matching to follow predetermined patterns and rules. This is done in order to reduce the search space of the policy, and increase the learning rate. An important extension will be to allow the RL agent to learn any arbitrary matching policy. Finally, we modeled buyers and sellers in our simulated environment as making utility comparisons with using the current fees, and previous epoch’s queries and matches. A more general way to model buyers and sellers is by modeling them as RL agents as well, and the system as a multi-agent economic system.

References

[1] Mark Armstrong. 2006. Competition in Two-Sided Markets. The RAND Journal of Economics 37, 3 (2006), 668–691. http://www.jstor.org/stable/25046266

[2] Gianluca Brero, Alon Eden, Matthias Gerstgrasser, David C. Parkes, and Duncan Rheingans-Yoo. 2021. Reinforcement Learning of Sequential Price Mechanisms. In Thirty-Fifth AAAI Conference on Artificial Intelligence. 5219–5227.
[3] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. 2016. OpenAI Gym. arXiv:arXiv:1606.01540

[4] Qingpeng Cai, Aris Filos-Ratsikas, Pingzhong Tang, and Yiwei Zhang. 2018. Reinforcement Mechanism Design for e-commerce. In Proceedings of the 2018 World Wide Web Conference, Pierre-Antoine Champin, Fabien Gandon, Mounia Lalmas, and Panagiotis G. Ipeirotis (Eds.). 1339–1348.

[5] Bernard Caillaud and Bruno Jullien. 2003. Chicken & Egg: Competition among Intermediation Service Providers. The RAND Journal of Economics 34, 2 (2003), 309–328. http://www.jstor.org/stable/1593720

[6] Minmin Chen, Alex Beutel, Paul Covington, Sagar Jain, Francois Belletti, and Ed Chi. 2019. Top-K Off-Policy Correction for a REINFORCE Recommender System. In Proceedings of the 12th ACM International Conference on Web Search and Data Mining. 456–464.

[7] Thomas Degris, Patrick M. Pilarski, and Richard S. Sutton. 2012. Model-Free reinforcement learning with continuous action in practice. In 2012 American Control Conference (ACC). 2177–2182.

[8] Jean-Pierre Dubé, Günter J Hitsch, and Peter E Rossi. 2009. Do switching costs make markets less competitive? Journal of Marketing research 46, 4 (2009), 435–445.

[9] Jean-Pierre Dubé, Günter J Hitsch, and Peter E Rossi. 2010. State dependence and alternative explanations for consumer inertia. The RAND Journal of Economics 41, 3 (2010), 417–445.

[10] Joseph Farrell and Carl Shapiro. 1988. Dynamic competition with switching costs. The RAND Journal of Economics (1988), 123–137.

[11] Benjamin R Handel. 2013. Adverse selection and inertia in health insurance markets: When nudging hurts. American Economic Review 103, 7 (2013), 2643–82.

[12] Matthew J. Hausknecht and Peter Stone. 2015. Deep Recurrent Q-Learning for Partially Observable MDPs. CoRR abs/1507.06527 (2015). arXiv:1507.06527 http://arxiv.org/abs/1507.06527

[13] Nicolas Heess, Jonathan J. Hunt, Timothy P. Lillicrap, and David Silver. 2015. Memory-based control with recurrent neural networks. CoRR abs/1512.04455 (2015). arXiv:1512.04455 http://arxiv.org/abs/1512.04455

[14] Elisabeth Honka. 2014. Quantifying search and switching costs in the US auto insurance industry. The RAND Journal of Economics 45, 4 (2014), 847–884.

[15] Eugene Ie, Chih-Wei Hsu, Martin Mladenov, Vihan Jain, Sanmit Narvekar, Jing Wang, Rui Wu, and Craig Boutilier. 2019. RecSim: A Configurable Simulation Platform for Recommender Systems. CoRR abs/1909.04847 (2019). http://arxiv.org/abs/1909.04847

[16] Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. 1998. Planning and acting in partially observable stochastic domains. Artificial Intelligence 101, 1 (1998), 99–134. https://doi.org/10.1016/S0004-3702(98)00023-X
[17] Alexander MacKay and Marc Remer. 2021. Consumer Inertia and Market Power. https://ssrn.com/abstract=3380390

[18] Martin Mladenov, Elliot Creager, Omer Ben-Porat, Kevin Swersky, Richard S. Zemel, and Craig Boutilier. 2020. Optimizing Long-term Social Welfare in Recommender Systems: A Constrained Matching Approach. In Proceedings of the 27th International Conference on Machine Learning. 6987–6998.

[19] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous Methods for Deep Reinforcement Learning. In Proceedings of The 33rd International Conference on Machine Learning, Vol. 48. 1928–1937.

[20] Fernando Nogueira. 2014. Bayesian Optimization: Open source constrained global optimization tool for Python. https://github.com/fmfn/BayesianOptimization

[21] Christos H. Papadimitriou, George Pierrakos, Christos-Alexandros Psomas, and Aviad Rubinstein. 2016. On the Complexity of Dynamic Mechanism Design. In Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016, Robert Krauthgamer (Ed.). SIAM, 1458–1475. https://doi.org/10.1137/1.9781611974331.ch100

[22] Manav Raj, Arun Sundararajan, and Calum You. 2021. COVID-19 and Digital Resilience: Evidence from Uber Eats. https://ssrn.com/abstract=3625638

[23] Jean-Charles Rochet and Jean Tirole. 2003. Platform Competition in Two-Sided Markets. Journal of the European Economic Association 1, 4 (06 2003), 990–1029. https://doi.org/10.1162/154247603322493212 arXiv:https://academic.oup.com/jeea/article-pdf/1/4/990/10312916/jeea0990.pdf

[24] Ruslan Salakhutdinov and Andriy Mnih. 2007. Probabilistic Matrix Factorization. In Proceedings of the 20th International Conference on Neural Information Processing Systems. 1257–1264.

[25] Weiran Shen, Binghui Peng, Hanpeng Liu, Michael Zhang, Ruohan Qian, Yan Hong, Zhi Guo, Zongyao Ding, Pengjun Lu, and Pingzhong Tang. 2020. Reinforcement Mechanism Design: With Applications to Dynamic Pricing in Sponsored Search Auctions. In The Thirty-Fourth AAAI Conference on Artificial Intelligence. AAAI Press, 2236–2243.

[26] Matthew Shum. 2004. Does advertising overcome brand loyalty? Evidence from the breakfast-cereals market. Journal of Economics & Management Strategy 13, 2 (2004), 241–272.

[27] Richard S. Sutton and Andrew G. Barto. 1998. Reinforcement Learning: an Introduction. MIT Press.

[28] Pingzhong Tang. 2017. Reinforcement mechanism design. In Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017, Carles Sierra (Ed.). ijcai.org, 5146–5150. https://doi.org/10.24963/ijcai.2017/739
[29] Maureen Tkacik. 2020. Restaurants are barely surviving. Delivery apps will kill them. *The Washington Post* (2020). [https://www.washingtonpost.com/outlook/2020/05/29/delivery-apps-restaurants-coronavirus/](https://www.washingtonpost.com/outlook/2020/05/29/delivery-apps-restaurants-coronavirus/)

[30] Tanay Warerkar. 2020. Diners Sue Grubhub and DoorDash Over “Shocking” Restaurant Fees. *Eater New York* (2020). [https://ny.eater.com/2020/4/14/21220591/nyc-delivery-grubhub-doordash-uber-postmates-lawsuit](https://ny.eater.com/2020/4/14/21220591/nyc-delivery-grubhub-doordash-uber-postmates-lawsuit)

[31] Daan Wierstra, Alexander Foerster, Jan Peters, and Jürgen Schmidhuber. 2007. Solving Deep Memory POMDPs with Recurrent Policy Gradients. In *Artificial Neural Networks – ICANN 2007*, Joaquim Marques de Sá, Luís A. Alexandre, Wlodzislaw Duch, and Danilo Mandic (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 697–706.

[32] Ronald J. Williams and Jing Peng. 1991. Function Optimization using Connectionist Reinforcement Learning Algorithms. *Connection Science* 3 (1991), 241–268.

[33] Ruohan Zhan, Konstantina Christakopoulou, Ya Le, Jayden Ooi, Martin Mladenov, Alex Beutel, Craig Boutilier, Ed Chi, and Minmin Chen. 2021. Towards Content Provider Aware Recommender Systems: A Simulation Study on the Interplay between User and Provider Utilities. In *Proceedings of the Web Conference 2021*. 3872–3883.

[34] Stephan Zheng, Alexander Trott, Sunil Srinivasa, David C. Parkes, and Richard Socher. 2021. The AI Economist: Optimal Economic Policy Design via Two-level Deep Reinforcement Learning. *CoRR* abs/2108.02755 (2021).
A Stackelberg Equilibrium Definition for Section 3

The *Stackelberg equilibrium* for the platform economy in Section 3 is:

1. The platform first decides upon a buyers’ fee $P_B$ and a sellers’ fee $P_S$. If the platform is non-myopic, it also decides upon a matching policy.
2. Some sellers and buyers join the platform.
3. The buyers make their queries. The on-platform buyers get their suggestions from the platform, and each buyer decides upon their transactions according to the above reasoning.
4. The surplus an off-platform buyer gets in an epoch is the sum of utilities the buyer gets from their completed transactions, where the surplus of an on-platform buyer is the sum of utilities minus $P_B$.
5. An off-platform seller’s surplus in an epoch equals the number of transactions they participate in minus $m$, where an on-platform seller’s surplus follows the same expression but also subtracting the fee $P_S$.
6. Given the platform’s fees and matching policy, the sellers and buyers are playing an equilibrium: no agent can improve their surplus by making a different decision in regard to joining the platform.
7. The platform cannot increase their objective’s value (e.g., revenue) by changing their fees or matching policy, and allowing the agents to adapt their behavior.

B Proofs of Section 3

*Proof of Claim 1.* When there’s no platform, Buyer 1 who does not know Seller 1 can only match their queries to Seller 2. But because of world friction, matching queries of type $Q_{11}$ with Seller 2 provide the buyer with utility $2 - d(Q_{11}, S_2) - 1 = \epsilon - 1 < 0$, and so Buyer 1 will only match queries of type $Q_{12}$. Buyer 1’s surplus is

$$m \cdot u_1(Q_{12}, S_2, 1) = m(2 - d(Q_{12}, S_2) - 1) = m\epsilon.$$  

Considering world friction, Buyer 2 has zero utility from matching its queries to Seller 2 and negative utility from matching them to Seller 1. We assume the buyer does not complete any transactions, and either way, Buyer 2 has zero surplus.

As for the sellers, Seller 1 doesn’t get matched, and has a negative surplus of $-m$, causing the seller to go bankrupt, while Seller 2 transacts $m$ times, obtaining a zero surplus. Overall, the total surplus to the agents is $m\epsilon$.

*Proof of Claim 2.* Since the matching is myopic, the platform’s choice is which agents to keep on the platform such that in equilibrium, the maximal fees it can set while keeping the agents on the platform, maximizes its revenue.

We consider several options:
1. All agents join the platform. In this case, the myopic matching ensures that $Q_{11}$ type queries are matched to Seller 1, and $Q_{12}$ and $Q_2$ type queries are matched to Seller 2. Since Seller 1 gets zero surplus considering cost $m$, the platform must set $P_S = 0$ in order to keep this seller on the platform. Without fees, Buyer 1’s surplus is $(2 - \epsilon)m + (1 + \epsilon)m = 3m$, while without a platform, this buyer has surplus $em$; similarly, Buyer 2’s surplus is $2m$, while without a platform, their surplus is zero. Therefore, the platform can set a fee $P_B = \min\{2m, 3m - em\} = 2m$. The overall revenue of the platform in this case is $4m$.

2. All but Seller 1 join the platform. The myopic policy matches all $4m$ queries to Seller 2 (recall that Buyer 1 cannot transact with Seller 1 off-policy). Without fees, Seller 2 has surplus of $3m$, compared with a surplus of zero without joining, and the platform can set $P_S = 3m$. Since Seller 1 doesn’t transact, they go bankrupt. Without fees, Buyer 1’s surplus is $em + (1 + \epsilon)m = (1 + 2\epsilon)m$, compared with $em$ from not joining; similarly, Buyer 2’s surplus is $2m$, while the off-platform surplus of this buyer is zero. Therefore, the platform can set fee $P_B = \min\{(1 + 2\epsilon)m - em, 2m\} = (1 + \epsilon)m$, obtaining revenue of $2(1 + \epsilon)m + 3m = (5 + 2\epsilon)m$.

3. Seller 2 does not join the platform. In this case, Buyer 2 cannot get a positive utility on or off platform, and will also stay off the platform. Without fees, Seller 1’s surplus is $m$, while their off-platform surplus is 0, and $P_S = m$. In this case, the platform’s revenue is $(4 - 3\epsilon)m$.

Inspecting all options, a revenue-maximizing platform will choose the second one, with all but Seller 1 joining the platform, and setting fees in a way that only Buyer 2 has a positive surplus of $2m - (1 + \epsilon)m = (1 - \epsilon)m$.

Proof of Claim 3. In this case, the platform needs to decide who joins the platform, and how many queries of each type to match to which on-platform seller. We first notice that if the platform has three agents or less joining it, then the optimal revenue it can make is strictly less than $6m$. If there’s only one seller on joining the platform, then the platform’s policy is necessarily myopic, and from Claim 2 the revenue is $< 6m$. For the case of a single buyer joining the platform, then since there are only $2m$ queries to be matched, the sellers’ total surplus is at most $m$, and the buyer’s surplus is at most $4m$, therefore the revenue of the platform cannot be greater than $5m$.

In fact, there is a non-myopic matching policy with revenue $6m$, achieved when all agents join the platform. Notice that all queries of type $Q_2$ must be matched to Seller 2, otherwise Buyer 2 will reject the matches. Given this, the platform aims to increase Seller 1’s surplus from matching, in order to increase the fee to sellers. To this end, the platform diverts queries $x < m$ of queries of type $Q_{12}$ to Seller 1, with $m - x$ matched to Seller 2. Without fees, the relative surplus of each agents on and off platform is:

- **Buyer 1.** On-platform: $(2 - \epsilon)m + x(1 + \epsilon) + (m - x)(1 - \epsilon) = 3m - 2\epsilon m + 2\epsilon x$, and off-platform: $em$.
- **Buyer 2.** On-platform: $2m$, and off-platform: 0.
- **Seller 1.** On-platform: $2m - x - m = m - x$, and off-platform: 0.
- **Seller 2.** On-platform: $2m + x - m = m + x$, and off-platform: 0.
Based on this, the platform can set \( P_B = \min\{3m - 3e\varepsilon + 2x, 2m\} = 2m \) and \( P_S = \min\{m - x, m + x\} = m - x \), resulting in a revenue of \( 2 \cdot (2m + m - x) = 6m - 2x \). The choice \( x = 0 \) is optimal, and the resulting revenue is \( 6m \), with all agents joining the platform, total surplus to buyers and sellers of \( 3m - 2e\varepsilon - 2m = m - 2e\varepsilon \).

\[ \text{Proof of Claim 4.} \] Consider a fee \( p \geq 0 \) for an on-platform agent, this set strictly less than the difference between their (without fee) on-platform surplus and off-platform surplus. But increasing this to \( p + \Delta \) for some infinitesimal \( \Delta > 0 \) increases the platform’s objective by \((1 - \alpha)\Delta\), which is strictly larger than 0 since \( \alpha < 1 \). Thus, for whichever agents join the platform, the platform should then set fees to maximize its own revenue (under constraints that the agents don’t leave). Therefore, we can use the prices calculated in Claim 2. More generally, the platform again needs to consider different sets of agents who might join the platform. We consider the following cases:

1. All agents join the platform. To maximize its revenue, the platform sets fees \( P_B = 2m \) and \( P_S = 0 \), and the platform’s objective value is \( 4m + 3e\varepsilon m \).

2. All but Seller 1 join the platform. To maximize its revenue, the platform sets fees \( P_S = 3m \) and \( P_B = (1 + \varepsilon)m \), and the platform’s objective value is \( 5m + 2\varepsilon + 3e\varepsilon \).

3. Seller 2 does not join the platform. In this case, by Case 3 of Claim 2, both the surplus and the revenue are lower than the other two cases, and so is the objective value.

Since \( 4m + 3e\varepsilon m > 5m + 2\varepsilon + 3e\varepsilon \), whenever \( \alpha > 1/2 + \varepsilon \), the best option is for all buyers and sellers to join the platform, which also leads no seller to go bankrupt, and to a total buyer and seller surplus of \( 3m \).

\[ \text{C Complete Description of Subscription Effect Estimation} \]

Below we make precise these estimates, for buyers and then sellers. We first explain the calculations in the absence of decision inertia.

- **A buyer \( b \), on-platform in epoch \( k \).** The estimate of epoch surplus if not subscribing to the platform is denoted \( \xi^w_{b,k+1} \). For this, for each query of \( b \) in epoch \( k \), the choice in the world under new friction \( \mu_{k+1} \) (Section 4.2.1) is reevaluated, with \( \xi^w_{b,k+1} = \sum_{t \in k} u^w_{b,t}(\mu_{k+1}) = \sum_{t \in k} \max\{u^w_b(q_{b,t}, s^*_{b,t}) - \mu_{k+1}, 0\} \). Considering the on-platform scenario, the new friction may also affect the choice between a platform seller and a world seller. This surplus is estimated based on updated decisions under \( \mu_{k+1} \), with \( \xi^p_{b,k+1} = -P_{b,k+1} + \sum_{t \in k} \max\{u^w_{b,t}(\mu_{k+1}) - \mu_{k+1}, u^p_{b,t}\} \).

- **A buyer \( b \), off-platform in epoch \( k \).** For this, we need to estimate the epoch surplus if \( b \) is subscribed to the platform. We assume that buyer \( b \) can observe platform-recommended sellers, e.g., this can be from trial periods to gain platform experience or a platform’s estimate of costs and benefits based on past orders. For a query sequence \( \{q_{b,t}\}_{t \in k} \), denote the corresponding best platform-recommended sellers as \( \{s^p_{b,t}\} \). The estimated surplus if subscribing to the platform is \( \xi^p_{b,k+1} = -P_{b,k+1} + \sum_{t \in k} \max\{u^w_{b,t}(\mu_{k+1}), u^p_{b,t}(s^p_{b,t})\} \). Further, the surplus from remaining off-platform depends on the new friction, i.e., \( \xi^w_{b,k+1} = \sum_{t \in k} u^w_{b,t}(\mu_{k+1}) \).

- **A seller \( s \), on-platform in epoch \( k \).** For this, we need to estimate the epoch surplus if \( s \) is not subscribed to the platform, by reasoning about (1) how many more world transactions
would happen if not on the platform, and (2) how buyers’ transaction decisions may be affected by the epoch \( k + 1 \) world friction. To facilitate a precise estimate, we assume that seller \( s \) can observe the sequence of query and seller candidate tuples \( \{(q_{b,t}, s, s_{b,t}^s)\}_{t \in k} \) in which seller \( s \) is chosen as the best world option. For the case where \( s_{b,t}^p = s \), and the seller was also recommended by the platform, the seller calculates their potential surplus by fixing the platform’s previous matching policy. Given \( q_{b,t} \) and \( S_k \setminus s \), we denote the updated, best-platform seller as \( s_{b,t}^{p*} \). Given this modified sequence \( \{(q_{b,t}, s, s_{b,t}^{p*})\} \), we then consider the choices buyers will make under the new friction \( \mu_{k+1} \), and estimate the number of transactions seller \( s \) will receive without being on the platform, denoted \( n_{s,k}^{w'} \). Thus, the estimated surplus if seller \( s \) is off platform is \\
\[ \xi^{w}_{s,k+1} = n_{s,k}^{w'} v_{s}^{1}(1 - \omega_{s}) \] \\

For remaining on-platform, we need to reason about how the new friction affects the number of transactions. Given the sequence \( \{(q_{b,t}, s_{b,t}^{w}, s)\}_{t \in k} \) where \( s \) is picked as the best platform seller, we consider each buyer’s choice and estimate the number of platform visits under \( \mu_{k+1} \), denoted \( n_{s,k}^{p*} \). Given the sequence \( \{(q_{b,t}, s, s_{b,t}^{p*})\} \) where \( s \) is picked as the best world seller, we estimate the number of world transactions under \( \mu_{k+1} \) (even if seller \( s \) chooses to stay on the platform), denoted \( n_{s,k}^{w*} \). Thus, the estimated surplus that seller \( s \) would receive by subscribing to the platform is \\
\[ \xi^{p}_{s,k+1} = -P_{S,k+1} + n_{s,k}^{p*} v_{s}^{1}(1 - \omega_{s} - P_{R,k+1}) + n_{s,k}^{w*} v_{s}^{1}(1 - \omega_{s}) \]

- **A seller \( s \), off-platform in epoch \( k \).** For this, we need to reason about the surplus from subscribing to the platform, e.g., by asking the platform for an estimate on the number of platform transactions. Given the sequence of queries on the platform, i.e., \( \{q_{b,t}\}_{b \in B_{k}, t \in k} \), the platform can update the matches it would suggest \( s \) also on-platform by following its matching policy. Denote the sequence of query and updated seller matches with tuples \( \{(q_{b,t}, s_{b,t}^{w}, s, s_{b,t}^{p*})\}_{b \in B_{k}, t \in k} \). With \( \mu_{k+1} \), we estimate the numbers of platform transactions \( n_{s,k}^{p*} \) and world transactions \( n_{s,k}^{w*} \). Then, the estimated surplus if seller \( s \) subscribes to the platform is \\
\[ \xi^{p}_{s,k+1} = -P_{S,k+1} + n_{s,k}^{p*} v_{s}^{1}(1 - \omega_{s} - P_{R,k+1}) + n_{s,k}^{w*} v_{s}^{1}(1 - \omega_{s}) \]

Seller \( s \) also adjusts the surplus that she would get by remaining off platform by reasoning about how buyers’ transaction decisions will be affected by the new world friction. Given the sequence of query and matched seller tuples \( \{(q_{b,t}, s, s_{b,t}^{w})\}_{b \in B_{k}, t \in k} \) in which seller \( s \) was chosen as the best world seller, we re-evaluate each buyer’s choice to get \( n_{s,k}^{w'} \). Thus, the estimated off-platform surplus for \( s \) is \\
\[ \xi^{w}_{s,k+1} = n_{s,k}^{w'} v_{s}^{1}(1 - \omega_{s}) \]

### D Modeling Buyers’ and Sellers’ Inertia

There is a rich body of literature that establishes this kind of decision inertia, modeling the presence of such inertia across different markets (see Section 2). In our setting an agent, either a buyer or seller, in subscription state \( T^b_k \in \{0, 1\} \), is prone to stay in the same state in epoch \( k + 1 \), due to habit formation, loyalty, or inattention. We treat buyers and sellers in the same way and illustrate the concept with a buyer \( b \) for simplicity. Each buyer starts with an initial preference in regard to adopting the platform or not, denoted by an integer \( \chi_{b,1} \sim U[-\chi, \chi] \) for some integer \( \chi \). Larger positive values indicate a stronger initial preference for the platform, whereas larger negative values indicate a stronger initial preference for not using the platform. Zero indicates no bias. If the initial \( \chi_{b,1} \) is positive, the buyer subscribes at the first epoch, otherwise they do not.
The inertia parameter is updated after each decision in the following way. If $\chi_{b,k} > 0$ and the buyer subscribes, then $\chi_{b,k+1} := \chi_{b,k} + 1$, and otherwise $\chi_{b,k+1} := -1$ (i.e., it resets). Similarly if $\chi_{b,k} < 0$ and the buyer decides to stay off-platform, then $\chi_{b,k+1} := \chi_{b,k} - 1$, and otherwise $\chi_{b,k+1} := 1$. The inertia $\chi_{b,k}$ maps into an additive bonus to either the surplus for joining the platform or remaining in the world through the functional form:

$$
\sigma_{b,k+1}^p := T_{b,k}^p \log(\chi_{b,k}) + 1, \quad \sigma_{b,k+1}^w := (1 - T_{b,k}^p) \log(-\chi_{b,k} + 1);
$$

that is, the longer the agent stays sticks to their decision, the larger the bias term gets, which increases in a concave way (logarithmically) over time.

This interpretation of decision inertia as an additive bonus is common in the literature [17, 8, 10]. Based on this adjusted utility, agents decide whether to subscribe or not according to probabilities inferred by the standard discrete-choice logit model [17, 8]. In the discrete-choice logit model, the probability of subscribing to the platform is $\delta_{b,k+1}^p := \exp\left(\xi_{b,k+1}^p + \sigma_{b,k+1}^p\right)/\left[\exp\left(\xi_{b,k+1}^p + \sigma_{b,k+1}^p\right) + \exp\left(\xi_{b,k+1}^w + \sigma_{b,k+1}^w\right)\right]$.

E Deferred Materials for Matching Queries to Platform Sellers

**Algorithm 1** The seller-aware matching rule with a utility threshold $\eta_k \in [0, 1]$.

**Input:** Platform fees $P_{B,k}$, $P_{S,k}$, and $P_{R,k}$. On-platform sellers $S_k$ and their observable attributes. A buy query $q_{b,t}$ for $t \in k$ and $b \in B_k$.

**Output:** A recommended seller $s^* \in S_k$.

1: Initialize a vector of estimated platform surplus for sellers $r_{s,t}^p$ at the start of epoch $k$:

$$
r_{s,t}^p = \begin{cases} -P_{S,k} & \text{if } s \in S_k, \\ 0 & \text{otherwise.} \end{cases}
$$

2: Given $q_{b,t}$, compute $u^* \leftarrow \max_{s \in S_k} u_B(q_{b,t}, s)$.

3: $S_u \leftarrow \{s : s \in S_k \text{ and } u_B(q_{b,t}, s) \geq \eta_k u^*\}$ \quad $\triangleright$ candidate sellers yield surplus at least $\eta_k u^*$.

4: if $|S_u| = 1$ then

5: \quad $s^* \leftarrow s \in S_u$ \quad $\triangleright$ the same seller as myopic matching.

6: else

7: \quad $S_r \leftarrow \{s : s \in S_u \text{ and } r_{s,t}^p \leq 0\}$

8: \quad if $S_r = \emptyset$ then

9: \quad \quad $r_{\min}^p \leftarrow \min_{s \in S_u} r_{s,t}^p$, $S^* \leftarrow \{s : s \in S_u \text{ and } r_{s,t}^p = r_{\min}^p\}$

10: \quad \quad $s^* \leftarrow \arg\max_{s \in S^*} u_B(q_{b,t}, s)$ \quad $\triangleright$ the best candidate with lowest platform surplus.

11: \quad else

12: \quad \quad $r_{\max}^p \leftarrow \max_{s \in S_r} r_{s,t}^p$, $S^* \leftarrow \{s : s \in S_r \text{ and } r_{s,t}^p = r_{\max}^p\}$

13: \quad \quad $s^* \leftarrow \arg\max_{s \in S^*} u_B(q_{b,t}, s)$ \quad $\triangleright$ the best candidate who is closest to break-even.

14: \quad \quad $r_{s^*,t}^p \leftarrow r_{s^*,t}^p + v_1^s (1 - \omega s^* - P_{R,k})$

15: \quad return $s^*$

Based on the pricing POMDP (Section 5.1), we make the following adjustments to define each component of the matching POMDP:

30
Here, we define $x_k \in \mathcal{X}$ for epoch $k$ as the state of the system after the platform has set fees and agents have chosen to subscribe, but still before the first query is submitted. We have agent-subscription states $I_{pB,k}$ and $I_{pS,k}$, and platform fees, all for the epoch that is about to take place, i.e., $P_{B,k}, P_{S,k}, P_{R,k}$. We also include the matching utility threshold for the previous epoch, $\eta_{k-1}$, to state $x_k$. All other elements of the state remain the same.

Here, the platform’s action $a_k$ chooses (i) the matching utility threshold for epoch $k$, and (ii) the matching rule for the epoch, whether seller-aware or profit-driven. For the threshold, we consider $\eta_k$ that takes discrete values in $[0, 1]$.

Different from the pricing POMDP, where a new epoch $k$ starts with agent subscription decisions, for matching, it starts with buyer queries: (1) a buy agent generates a query, (2) if the buyer is on platform, the platform recommends a seller following $\eta_k$ and its matching rule, and (3) the buyer selects a seller with whom to transact (Section 4.2). This gives the full sequence of queries, $Q_k$. At the end of epoch $k$, buyers and sellers observe the new fees $P_{B,k+1}, P_{S,k+1}$ and $P_{R,k+1}$, as given by a fixed fee schedule, and decide whether or not to subscribe to the platform for the next epoch.

Here, we include in the reward $r_k \sim \mathcal{R}(x_k, a_k)$ to the platform for epoch $k$ both the referral fees from epoch $k$ and the subscription fees from buyers and sellers as reflect the decision they make at the end of the epoch in regard to whether or not to join the platform for epoch $k + 1$. This avoids delayed reward for the platform, reflecting that the registration decisions and thus fees registration fees for the next epoch are influenced by the platform’s matching policy during epoch $k$, and thus by action $a_k$.

The platform’s observability of information in the state follows in the same way as for the pricing POMDP.

Figure 10: The probability density of the matching rule and utility threshold chosen by the learned matching policy pre-shock, during shock, and post-shock.
Based on preliminary explorations, we choose to have the actor and critic share a fully-connected layer, LSTM cells of size 128, and again a fully-connected layer to recover sufficient statistics of the history, using this to in effect infer the knowledge structure of buyers and the demand elasticity of agents to platform fees. Each network also has its own two fully-connected layers. The critic outputs the value $V_\psi(o)$ of an observation $o$, and the actor gives policy $\pi_\theta$ for an observation $o$. For the pricing actor, this includes three separate output layers, with each returning a vector of probabilities for one type of platform fee. For the matching actor, this is a vector of probabilities over the matching utility thresholds that are applicable to both matching rules. Figure 11 illustrates the neural network structure we implement.

Besides the policy gradient loss, we apply entropy regularization [19, 32] to the policy network to encourage exploration. The respective losses for the policy network and the value network are,

$$L_\pi = -\log \pi(a_k|o_k; \theta)(R_k - V_\psi(o_k)) - \beta H(\pi(A_k|o_k; \theta)) \quad \text{and} \quad L_V = (R_k - V_\psi(o_k))^2,$$

where $H$ denotes the entropy over learned action probabilities. We tune the platform agent with various combinations of learning rates $\{0.0001, 0.0005, 0.001\}$, batch sizes $\{4, 16, 32, 64, 128\}$, and entropy weights $\{0.001, 0.01, 0.05\}$, and select hyperparameters that maximize the objective function. Results of learned policies reported in Section 6.3.1 are based on the average performance of two models trained from different torch seeds. All training parameters are displayed in Table 3.

Figure 12 depicts the learning curve of reinforcement learning platform agents of different reward objectives during training processes for pricing. The horizontal line depicts the reward of a set of the best fixed prices obtained by Bayesian Optimization (BO). Theoretically the reinforcement learner, with the ability to set prices every epoch, should outperform the fixed prices submitted by Bayesian optimization. We note for the purely revenue maximizing case, it is not always the case: BO sets seller registration price to be almost the maximum and seller referral price to be zero, and hence creates an oscillation effect (one epoch buyers subscribe but no sellers, the next epoch sellers subscribe but no buyers). Thus the platform could profit primarily through charging extremely high seller registration prices from only a few of the non-bankrupt sellers. This is an inherent limitation of the monopoly platform market and the counterfactual modeling: in real world with fierce platform competitions, sellers will soon go to the platform’s competitor if the platform charges a very high registration fee and yet provides no buyers. When the reward include some other metrics like the number of on-platform transactions, creating buyer-seller subscription oscillation have a low number of transactions, and thus RL will outperform BO easier by setting varying prices at different epochs.
Figure 11: An illustration of the neural network structure for the policy network and the value network.

| Parameters                               | Value          |
|------------------------------------------|----------------|
| RL Algorithm                             | A2C            |
| Optimizer                                | Adam           |
| Learning Rate                            | 0.0001         |
| Batch Size                               | 4              |
| Reward Discount Factor                    | 0.99           |
| Entropy Weight                           | 0.01           |
| Training Episodes                        | 25000          |
| Number of Epochs per Episode             | 12             |
| Linear Layer before LSTM Dimension       | 256            |
| LSTM Cell Size                           | 128            |
| Linear Layers for Policy and Value Networks | 128         |
| Observation Size                         | 282            |
| Buyer Registration Max                   | 10             |
| Seller Registration Max                  | 10             |
| Seller Referral Max                      | 1              |
| Buyer Registration Discretization        | 0.2            |
| Seller Registration Discretization       | 0.2            |
| Seller Referral Discretization           | 0.1            |
| Bayesian Optimization                    |                |
| Rounds                                   | 64             |
| Initial points                           | 10             |
| Iterations in round                      | 50             |

Table 3: Training Hyperparameters.
Figure 12: Reinforcement Learning training Curves and Bayesian Optimization Baseline. The horizontal line depicts the reward of a set of the best fixed prices obtained by Bayesian Optimization (BO).
G  Platform Value Plots Under Different Latent Structures

Supplementing Figure 4 in Section 6.2, Figure 13 shows the total welfare, as well as buyer and seller surplus, achieved in environments with and without a platform under the Uniform and Two-Core market structures. As in the previously-seen graph for the Core-and-Niche structure, we normalize welfare and surplus by the total welfare achieved in an ideal world where buyers have complete knowledge about sellers and there is no world transaction friction.

Figure 13: Total welfare and buyer and seller surplus, achieved in environments that vary in knowledge level and world transaction friction, with and without a platform under the Uniform, Core-and-Niche, and Two-Core market structures. Results presented are the average of a hundred runs for each environment.
H  Design Objectives for Two-Core and Uniform Market Structure

Figure 14: Platform fees set by platforms induced by different design objectives under the Core-and-Niche market structure. Results are grouped across epochs within each shock stage and are averaged on a hundred controlled test episodes with two training seeds.

Supplementing the Core latent structure in Section 6.3.1, we present results for the two core and uniform latent structure. Just as the core environment, on-platform user surplus and the number of platform buyers remains the two desirable design objective to keep the economy healthy after the shock. In the two-core environment, seller diversity compared to the number of sellers reduces the number of bankrupt sellers, while maintaining a similar level of post-shock welfare.
Figure 15: The welfare decomposition, buyer and seller states, and platform fees of pricing policies induced by different design objectives under the two-core market structure. Results are grouped across epochs within each shock stage and are averaged on a hundred controlled test episodes with two training seeds.
Figure 16: The welfare decomposition, buyer and seller states, and platform fees of pricing policies induced by different design objectives under the uniform market structure. Results are grouped across epochs within each shock stage and are averaged on a hundred controlled test episodes with two training seeds.
# Model Notation & Summary

| Model Parameters                  | Notation and Value |
|-----------------------------------|--------------------|
| Latent Space                      | $\mathcal{V} \in [0, 1]^n$ |
| Uniform Latent Structure          | $v \sim U[0, 1]^2$ |
| Core Latent Structure             | $v \sim \text{Truncated Gaussian} (\mu = [0.5, 0.4], \sigma = 0.2, 0.1)$ |
| Two-Core Latent Structure         | $v \sim \text{Truncated Gaussian} (\mu_1 = [0.7, 0.3], \sigma_1 = 0.17, 0.1)$ or $\mu_2 = [0.3, 0.7], \sigma_2 = 0.17, 0.1$ |
| Timesteps                         | $T = 100$ |
| Epochs                            | $K = 12$ |
| World Friction                    | $\mu_k \in [0, 1]$ |
| Shock Intensity Parameter         | $I \sim U[I_{\text{min}} = 0.8, I_{\text{max}} = 1]$ |
| Pre/Post-Shock Epochs             | $\mu_{1,2,3,10,11,12} = 0.1$ |
| Shock Epochs                      | $\mu_{3,4,5,6,7,8,9} \sim \text{Lognormal}(\mu = 0, \sigma = 0.5)$; normalized to $I$ |
| On-Platform Agent Inertia         | $\sigma_{p,b,k+1}^p := I_{b,k} \log(\chi_{b,k}) + 1$; initial inertia $\chi_{b,0} \sim U[-2, 2]$ |
| Off-Platform/World Agent Inertia | $\sigma_{p,b,k+1}^w := (1 - I_{b,k}) \log(-\chi_{b,k} + 1)$ |
| Probability of Subscribing to Platform | $\delta_{b,k+1}^p := \exp(\chi_{b,k+1}^p + \sigma_{b,k+1}^p)/\left[\exp(\chi_{b,k+1}^p + \sigma_{b,k+1}^p) + \exp(\chi_{b,k+1}^w + \sigma_{b,k+1}^w)\right]$ |
| Buyers                            | $b \in \mathcal{B}$; $|\mathcal{B}| = 10$ |
| Buyer Known Fraction              | $\rho \in [0, 1]$ |
| Buyer Known Sellers               | $\mathcal{S}_b \subseteq \mathcal{S}$; $p(s \in \mathcal{S}_b) \sim \text{Bern}(\rho)$ |
| Buyer Query                       | $q_b \in \mathcal{V}$; $q_{b,t} \sim N(v_b, \sigma_b^2 = 0.02)$ |
| Per-Epoch Budget                  | $\psi_b = \text{N}(v_b, \sigma_b^2)$; $N = 10$ queries per epoch |
| Buyer Matching Utility            | $u_{B}(q, s) = \exp(-c \cdot ||q - v_s||^2)$; where $c = 2, u_{B} \in [0, 1]$ |
| Best World Match                  | $s_{b,t}^w := \text{arg max}_{s \in \mathcal{S}_t} u_{B}(q_{b,t}, s) - \mu_k \text{ or } 0$; $S_{b,t} := \{s \in \mathcal{S}_b : v_s^1 \leq \psi_b, s \in \mathcal{S}_t\}$ |
| World Transaction Surplus         | $r_{b,t}^w = u_{b,t}^w \cdot \psi_b^I$; $u_{b,t}^w = u_{B}(q_{b,t}, s_{b,t}^w) - \mu_k$ or 0 |
| Platform-Provided Match           | $s_{b,t}^p = 0$; $u_{b,t}^p = u_{B}(q_{b,t}, s_{b,t}^p) \text{ or } 0$ |
| Platform Transaction Surplus      | $r_{b,t}^p = u_{b,t}^p \cdot (1 - \psi_b)$; $u_{b,t}^p = u_{B}(q_{b,t}, s_{b,t}^p)$ or 0 |
| Final Transaction Surplus         | $r_{b,t} = \max \left\{u_{b,t}^w, u_{b,t}^p, 0\right\}$ |
| Final Epoch Surplus               | $r_{b,k} = \sum_{t \in K} r_{b,t} - \psi_{b,k}^{P_{B,k}}$ |
| Counterfactual World Surplus      | $\xi_{b,k+1}^w = \sum_{t \in K} \max\{u_{b,t}^w, 0\}$ |
| Counterfactual Platform Surplus   | $\xi_{b,k+1}^p = -P_{B,k+1} + \sum_{t \in K} \max\{u_{b,t}^w, u_{b,t}^p, 0\}$ |
| Sellers                           | $s \in \mathcal{S}$; $|\mathcal{S}| = 10$ |
| Production Cost                   | $\omega_s \sim U[0.2, 0.4]$ |
| Number of World Transactions      | $n_{s,k}^w$ |
| World Transaction Surplus         | $r_{s,t}^w = v_s^1(1 - \omega_s) \text{ or } 0$ |
| Number of Platform Transactions   | $n_{s,k}^p$ |
| Platform Transaction Surplus      | $r_{s,t}^p = v_s^1(1 - \omega_s - P_{r,ef,k}) \text{ or } 0$ |
| Final Epoch Surplus               | $r_{s,k} = \sum_{t \in K} r_{s,t} - \psi_{s,k}^{P_{S,k}}$ |
| Shutdown Threshold                | $\lambda_s = 2$ |
| Counterfactual World Surplus      | $\xi_{s,k+1}^w = n_{s,k}^w v_s^1(1 - \omega_s)$ |
| Counterfactual Platform Surplus   | $\xi_{s,k+1}^p = -P_{S,k+1} + n_{s,k}^p v_s^1(1 - \omega_s) + n_{s,k}^w v_s^1(1 - \omega_s)$ |
| Model Parameters                        | Notation and Value |
|----------------------------------------|--------------------|
| **Platform**                           | $\mathcal{P}$     |
| Buyer Registration Fee                 | $P_{B,k} \in [0,10]$; discretized at 0.2 |
| Seller Registration Fee                | $P_{S,k} \in [0,10]$; discretized at 0.2 |
| Seller Referral Fee                    | $P_{ref,k} \in [0,1]$; discretized at 0.1 |
| Matching Utility Threshold             | $\eta \in [0,1]$; |
| **Matching Rule**                      | Seller-Aware or Profit-Driven |
| **Platform Epoch Revenue**             | $r_{p,k} = \sum_{b \in \mathcal{B}}^{p} P_{B,b,k} + \sum_{s \in \mathcal{S}}^{p} (n_{r,s,k}^{p} P_{S,s,k} + v_{s}^{p} P_{ref,k})$ |
| **Revenue Objective Reward**           | $r_{k} = r_{p,k}$ |
| Revenue + Surplus Objective Reward     | $r_{k} = r_{p,k} + \alpha (r_{b,k}^{p} + r_{s,k}^{p})$; where $\alpha = 0.5$ |
| Revenue + # Platform Buyers Objective Reward | $r_{k} = r_{p,k} + \alpha |\mathcal{B}|$; where $\alpha = 3$ |
| Revenue + # Platform Sellers Objective Reward | $r_{k} = r_{p,k} + \alpha |\mathcal{S}|$; where $\alpha = 3$ |
| Revenue + Platform Seller Diversity Objective Reward | $r_{k} = r_{p,k} + \alpha |\mathcal{S}_{\mu_1,k}| |\mathcal{S}_{\mu_2,k}|$; where $\alpha = 1.2$ |
| Revenue + # Platform Transactions Objective Reward | $r_{k} = r_{p,k} + \alpha \sum_{s \in \mathcal{S}}^{p} n_{s,k}$; where $\alpha = 0.3$ |
| **Learned Policy**                     | $\pi^{*}(a|o_k; \theta) = \max_{a} \mathbb{E}_{a \sim \pi, x \sim \mathcal{P}} \left[ \sum_{k=0}^{K} \gamma^{k} r_{k} \right]$; where $\gamma = 0.99$ |
| **A2C Policy Loss**                    | $\mathcal{L}_{\pi} = -\log \pi(a_k|o_k; \theta) (R_k - V_{\psi}(o_k)) - \beta \mathbb{H}(\pi(A_k|o_k; \theta))$ |
| **A2C Value Loss**                     | $\mathcal{L}_{V} = (R_k - V_{\psi}(o_k))^2$ |
| **Total Epoch Welfare**                | $r_{k} = \sum_{b \in \mathcal{B}} r_{b,k} + \sum_{s \in \mathcal{S}} r_{s,k} + r_{p,k}$ |