Simplicity in Video Games: Theory and Applications

Kweku A. Opoku-Agyemang*

September 15, 2023

Abstract

Video games are complex interactive systems that involve multiple agents, such as human players, artificial intelligence (AI) agents, and game developers, who interact strategically in various scenarios and environments. Designing video games that are engaging, immersive, and fair requires understanding the behavior and preferences of these agents, as well as the trade-offs and incentives that arise from the game rules and mechanisms. In this paper, we extend the theory of simplicity in games and mechanism design to the domain of video games. We introduce a general class of simplicity standards that vary the cognitive abilities required of agents in video games, such as memory, attention, anticipation, and learning. We use these standards to provide characterizations of simple mechanisms in video game environments with and without transfers, such as scoring systems, reward structures, difficulty levels, and matchmaking algorithms. We also study the implications of simplicity for the design of dynamic mechanisms in video games, such as auctions, markets, and voting systems. We illustrate our results with exampless. We show how simplicity can enhance the gameplay experience and create more realistic and adaptive AI agents. We also discuss the limitations and challenges of simplicity in video games, such as ethical issues, computational complexity, and human factors.

*Machine Learning X Doing. Email: kweku@machinelearningxdoing.com. I thank the Center for Effective Global Action and Ted Miguel for having me as an honorary research fellow at the University of California, Berkeley. I thank the Blum Center for Developing Economies at the University of California, Berkeley for having me as a postdoctoral fellow. I also thank Cornell Tech and Tapan Parikh. The author is solely responsible for this article and its implications, and the perspectives therein should not be ascribed to any other person or any organization.

1 Introduction

Video games are one of the most dominant forms of entertainment. According to the Global Games Market Report 2023, the global video game industry generated 189.3 billion dollar in revenues in 2023, with more than 3.1 billion gamers worldwide. Video games are not only a source of fun and leisure, but also a medium for artistic expression, social interaction, education, and innovation.

Video games also happen to be complex interactive systems that involve multiple agents, such as human players, artificial intelligence (AI) agents, and game developers, who interact strategically in various scenarios and environments. Appreciating these systems from a behavioral standpoint is key to improving them, as designing video games that are engaging, immersive, and fair requires technically understanding the behavior and preferences of these agents, as well as the trade-offs and incentives that arise from the game rules and mechanisms. For example, how do players choose their actions and strategies in a game? How do AI agents learn from their experiences and adapt to changing situations? How do developers balance the difficulty and challenge of a game to satisfy different types of players? How do developers design mechanisms that promote cooperation or competition among players or AI agents?

Game theory and mechanism design are two fields that offer important tools to model, analyze, and solve decentralized design problems involving multiple autonomous agents that interact strategically in a rational and intelligent way. Game theory studies the strategic behavior of agents in situations where their actions affect each other's outcomes, such as games, markets, or social dilemmas. Mechanism design studies the optimal design of rules or institutions that govern the interactions of agents, such as auctions, voting systems, or matching markets. These tools have been widely applied to various domains, such as economics, political science, computer science, engineering, and biology.

However, applying game theory and mechanism design to video games at the level of the state-of-the-art, poses several challenges. First, video games are often dynamic and complex systems that involve many agents, actions, states, and outcomes. This makes it very difficult to model and analyze them using standard game-theoretic tools, such as normal-form or extensive-form games. Second, video games often involve agents that are not fully rational or intelligent. For example, human players may have limited cognitive abilities, such as memory, attention, anticipation, or learning. AI agents may have limited computational resources or information. These factors may affect the behavior and preferences of agents in video games. Third, video games often involve agents that have heterogeneous and diverse preferences. For example, different players may have different tastes, skills, goals, or motivations in playing a game. AI agents may have different objectives or roles in a game. These factors may affect the trade-offs and incentives of agents in video games.

To address these challenges, we extend the literature on the theory of simplicity in games and mechanism design to the domain of video games. In particular, Li (2017) and Pycia and Troyan (2023) motivate a general class of simplicity standards that vary the foresight abilities required of agents in extensive-form games. They use these standards to provide characterizations of simple mechanisms in social choice environments with and without transfers. They also study the implications of simplicity for the design of dynamic mechanisms[1]. The main difference between our work and the literature is that we emphasize gaming environments for the purposes of game-based case studies that we believe will apply to the vast majority of video games. This emphasis makes us emphasize on varying cognitive abilities relevant for the gaming space.

We introduce a general class of simplicity standards that vary the cognitive abilities required of agents in video games, such as memory, attention, anticipation, and learning. We use these standards to provide characterizations of simple mechanisms in video game environments with and without transfers, such as scoring systems, reward structures, difficulty levels and others.

We use these standards to provide characterizations of simple mechanisms in video game environments with and without transfers, such as scoring systems, reward structures, difficulty levels, and matchmaking algorithms. We show how these mechanisms can balance the gameplay and ensure fairness among players or AI agents. We also study the implications of simplicity for the design of dynamic mechanisms in video games, such as auctions, markets, and voting systems. We show how these mechanisms can create realistic and adaptive AI agents, as well as enhance the social and economic aspects of the game.

We illustrate our results with examples from popular video games, such as first-person shooters,

[1]Other relevant works include Ashlagi and Gonczarowski (2018) that aligns with simple millipede games; Troyan (2019), which emphasizes the popular Top Trading Cycles (TTC) mechanisms. Arribillaga, Massó, and Neme (2020) has relevance for voting rules, like this paper.

first-person action-adventures, sandbox games, and other franchises to show the range of the framework. We show how simplicity can enhance the gameplay experience and create more engaging and immersive video games. We also discuss the limitations and challenges of simplicity in video games, such as ethical issues, computational complexity, and human factors.

Why is game theory and mechanism design important for gaming? Game theory and mechanism design are two fields in economics and computer science that offer important tools to model, analyze, and solve decentralized design problems involving multiple autonomous agents that interact strategically in a rational and intelligent way. Modern video games can benefit from these tools in various aspects, such as:

Balancing the gameplay and ensuring fairness among players. Game theory can help developers test the odds of their game and see if there are any dominant strategies or unbalanced outcomes that would make the game less fun or challenging[3]. Mechanism design can help developers design incentives or rules that align with the desired objectives of the game, such as encouraging cooperation, competition, or exploration.

Creating realistic and adaptive artificial intelligence (AI) agents. Game theory can help developers model the behaviors and preferences of AI agents, as well as their interactions with other agents and human players. Mechanism design can help developers design mechanisms that elicit truthful or optimal responses from AI agents, such as auctions, voting, or bargaining.

Enhancing the social and economic aspects of the game. Game theory can help developers understand the dynamics of social dilemmas, such as trust, cooperation, altruism, or betrayal, and how they affect the outcomes of the game. Mechanism design can help developers design social or economic mechanisms that promote desirable social outcomes, such as fairness, efficiency, or stability.

Of course, these tools are not sufficient to create a successful video game, as there are many other factors that influence the quality and appeal of a game, such as graphics, sound, story, or emotions. However, by applying these tools wisely and creatively, developers can enhance the gameplay experience and create more engaging and immersive video games. We therefore think of them as necessary.

Examples from Popular Video Games. We use examples from popular video games to illustrate our results and concepts from the main text. We show how simplicity standards can be applied to different types of games and mechanisms, and how they can affect the gameplay and outcomes for the agents based on various aspects in the games. The approach can be applied to fine-tune the aspect in ways that make each game more engaging for the stakeholders.

1.1 Example: Multiplyer Online First Person Shooter game

Consider a generic multiplayer online video game that combines elements of survival, shooting, building, and exploration. Consider a Battle Royale, where many players compete to be the last one standing in a game environment that shrinks over time. The players can collect weapons, items, and resources to fight or hide from other players, as well as build structures to defend or attack.

Such a game can be modeled as a video game environment without transfers, where the agents are the players, the types are their skills and strategies, the outcomes are their survival or elimination, and the payoffs are their scores or ranks. A mechanism is a function that determines the outcome for each player based on their type and actions.

One possible mechanism here is a random matching mechanism, where each player is randomly assigned to a server with up to 99 other players at the beginning of each round.

The mechanism is simple with respect to the 0-memory simplicity standard in our model, which means that players can only remember their own actions at each node. To see this, note that the best response of each player at each node is to choose an action that maximizes their survival probability given their own type and action, using only their own type as a reference. Therefore, this mechanism induces an extensive-form game in which every player can play a best response using only the actions that they can foresee at each node according to the 0-memory simplicity standard.

This mechanism can balance the gameplay and ensure fairness among players. For example, it can create a diverse and unpredictable gameplay experience for the players, as they face different opponents and situations in each round. It can also ensure that each player has an equal chance of winning or losing in each round, as they are matched with random opponents regardless of their skill level or history. It can also prevent cheating or collusion among players, as they cannot choose or influence their opponents or teammates in each round.

1.2 Example: First-Person Action-Adventure

The approach is flexible enough to distinguish between first-person shooters and first-person action-adventure games that are based more on exploration. We use an example from the first-person action-adventure genre to illustrate our results and concepts from the main text. We show how a simplicity standard can be applied to a mechanism in the game, and how it can affect the gameplay and outcome for the agent.

The gameplay may involve solving puzzles to reveal secrets, platform jumping, and shooting foes with the help of a "lock-on" mechanism that allows circle strafing while staying aimed at the enemy. The protagonist may travel through the world (or multiple worlds) searching for items that will open the path for further exploration, and other features of the game.

Such games can be modeled as a video game environment without transfers, where the agent is the main character, the type is his or her skill and strategy, the outcome is her survival or elimination, and the payoff is her score or rank. A mechanism in this genre is a function that determines the outcome for the protagonist based on his or her type and actions.

One possible mechanism may be a data collection or scanning system, where the main character can scan various objects or enemies in the game world to obtain information or activate functions.

In the context of our framework, we say the scanning system is simple with respect to the 1-attention simplicity standard, which means that the protagonist can only pay attention to one object or enemy at each node. To see this, note that the best response of the protagonist at each node is to scan an object or enemy that maximizes her expected payoff given his or her own type and the object or enemy she observes at that node, using only her own type as a reference. Therefore, this mechanism induces an extensive-form game in which the main character can play a best response using only the actions that she can foresee at each node according to the 1-attention simplicity standard.

This mechanism can balance the gameplay and ensure fairness for the main character in the game. For example, it can enhance the exploration and discovery aspects of the game, as scanning objects or enemies reveals secrets or lore about the game world. It can also create a trade-off between scanning and shooting for the character, as scanning objects or enemies gives information but also exposes the character to attacks. It can also prevent the protagonist from being overwhelmed or confused by too many objects or enemies to scan, as he or she can only focus on one at a time.

In Section 6, we discuss some future directions for research on simplicity in video games and mechanism design. We summarize our main contributions and findings, and highlight some open questions and challenges for further exploration. We hope that our paper will inspire more research on this topic and provide useful insights for game developers and mechanism designers.

1.3 Example: Sandbox

Here, we consider a sandbox video game genre that allows players to create and explore a virtual world. The game genre has several modes, but one of the most popular ones is a mode where players have to gather resources, craft items, build structures, and fight enemies to survive. The game also has a different mode, where players have unlimited resources and can create anything they want.

Such a game can be modeled as a video game environment without transfers, where the agents are the players, the types are their preferences and goals, the outcomes are their creations and actions, and the payoffs are their satisfaction or enjoyment. A mechanism here is a function that determines the outcome for each player based on their type and actions.

One possible mechanism here is a customization mechanism, where each player can choose or modify various aspects of the game world, such as the difficulty level, the game mode, the world type, the world seed, the world generator options, and the game rules.

The mechanism is simple with respect to the k-attention simplicity standard, where k is a positive integer that represents the number of aspects that a player can pay attention to at each node. To see this, note that the best response of each player at each node is to choose or modify an aspect that maximizes their expected payoff given their own type and the aspects they observe at that node, using only their own type as a reference. Therefore, this mechanism induces an extensive-form game in which every player can play a best response using only the actions that they can foresee at each node according to the k-attention simplicity standard.

This mechanism can adapt or customize the game to different types of players. For example, it can allow players to create their own unique and personalized game worlds, according to their preferences and goals. It can also vary the challenge and complexity of the game world, according to the skill and interest of the players. It can also enable players to experiment and discover new features and possibilities in the game world.

1.4 Example: Social Deduction

The social deduction game genre is often a multiplayer online video game space that simulates a social deduction game, where players are either allies or enemies in an environment. The alllies may have to complete tasks and find the enemies, while the enemies have to kill the allies and sabotage the environment. The game has several modes, but typically, up to 10 players can play with one or two enemies.

Games in this genre can be modeled as a video game environment without transfers, where the agents are the players, the types are their roles and strategies, the outcomes are their survival or elimination, and the payoffs are their scores or ranks. A mechanism in Among Us is a function that determines the outcome for each player based on their type and actions.

One possible mechanism in this game genre is a voting system, where players can call meetings and vote to eliminate one of them based on their suspicions or evidence. The voting system is simple with respect to the 1-memory simplicity standard, which means that players can only remember the last vote they cast at each node. To see this, note that the best response of each player at each node is to vote for the player who they think is most likely to be an impostor, using only their own type and vote as a reference. Therefore, this voting system induces an extensive-form game in which every player can play a best response using only the actions that they can foresee at each node according to the 1-memory simplicity standard.

This voting system can design incentives or rules that align with the desired objectives of the mechanism in the game. For example, it can aggregate information or preferences among players, as players can express their opinions or suspicions on the roles of other players through their votes. It can also implement collective choices or actions among players, as the player who receives the most votes is eliminated from the game. It can also elicit truthful preferences or opinions from the players, as voting for their most suspected player is a dominant strategy for both allies and enemies.

The rest of the paper is organized as follows. Section 2 reviews the literature on game theory and mechanism design in video games. Section 3 introduces the framework of simplicity in games and mechanism design by Pycia and Troyan (2023) and extends it to video games. Section 4 provides characterizations of simple mechanisms in video game environments with and without transfers. Section 5 studies the implications of simplicity for the design of dynamic mechanisms in video games. Section 6 illustrates our results with examples from popular video games. Section 7 discusses the limitations and challenges of simplicity in video games. Section 8 concludes.

2 Literature Review

The literature on video games can be broadly divided into two main streams: one that focuses on the design and development of video games, and another that analyzes the economic and social aspects of video games. In this section, we review some of the relevant works from both streams that relate to our research question: how can simplicity standards help design better video game mechanisms? We also discuss how video games may say something more general about game theory and mechanism design.

The design and development of video games involves various aspects, such as graphics, sound, storytelling, gameplay, user interface, and artificial intelligence. Among these aspects, gameplay is arguably the most important one, as it determines how the player interacts with the game and what kind of experience the game provides. Gameplay is largely influenced by the game mechanics, which are the rules and processes that govern the actions and behaviors of the agents in the game. Game mechanics can be seen as the building blocks for creating game dynamics, which are the patterns and outcomes that emerge from the interaction of game mechanics over time (Hunicke et al., 2004). Game dynamics can affect the player's motivation, engagement, immersion, satisfaction, and enjoyment of the game.

Game mechanics can be classified into different types according to various criteria. For example, Salen and Zimmerman (2004) distinguish between operational mechanics (the basic actions that a player can perform in a game), constitutive mechanics (the underlying rules that define how a game works), and implicit mechanics (the unwritten rules or conventions that players follow in a game). Another way to categorize game mechanics is based on their functions or effects on the gameplay. For instance, Björk and Holopainen (2005) identify 51 different types of game mechanics that serve various purposes, such as creating challenges, providing feedback, enabling cooperation or competition, facilitating exploration or discovery, and so on.

One of the challenges in designing game mechanics is to balance between complexity and simplicity. Complexity can make a game more interesting, diverse, and realistic, but it can also make it more difficult to learn, play, or understand. Simplicity can make a game more accessible, intuitive, and elegant, but it can also make it more boring, repetitive, or predictable. Finding the optimal level of complexity or simplicity depends on various factors, such as the target audience, the genre of the game, the intended learning outcomes (if any), and the available resources.

Several approaches have been proposed to measure or evaluate the complexity or simplicity of game mechanics. For example, Alves et al. (2014) propose a framework for measuring the complexity of board games based on four dimensions: rules complexity (the number and variety of rules), information complexity (the amount and type of information available to players), decision complexity (the number and difficulty of choices players have to make), and strategic complexity (the depth and breadth of strategies players can employ). On the other hand, Nelson et al. (2017) develop a metric for quantifying the simplicity of digital games based on three components: minimalism (the number of distinct elements in a game), elegance (the ratio between minimalism and expressiveness), and orthogonality (the degree to which elements in a game interact with each other). Although not focused on video games in particular, Pycia and Troyan (2023) introduce a general class of simplicity standards that vary the foresight abilities required of agents in extensive-form games. Rather than planning for the entire future of a game, agents are presumed to be able to plan only for those histories they view as simple from their current perspective. Agents may update their so-called strategic plan as the game progresses and new information becomes available.

The economic and social aspects of video games involve studying how video games affect or are affected by various phenomena, such as markets, institutions, policies, behaviors, preferences, and outcomes. Video games can be seen as a form of media or entertainment that provides utility or value to consumers and producers, as well as a form of technology or innovation that creates opportunities or challenges for various stakeholders. Video games can also be seen as a form of experimentation or simulation that allows testing or exploring various scenarios or hypotheses in a controlled or realistic environment.

Video games can be analyzed using various tools and methods from economics and other social sciences. For example, Castronova (2005) examines the emergence and implications of virtual economies in online games, such as the creation and exchange of virtual goods and currencies, the formation and regulation of markets and institutions, and the impact on real-world economies and societies. Also, Hamari et al. (2014) conduct a meta-analysis of the literature on gamification, which is the use of game elements in non-game contexts, such as education, health, business, and social good. They identify various game elements, such as points, badges, leaderboards, feedback, goals, and challenges, that can affect various outcomes, such as motivation, engagement, learning, performance, and behavior change. Zagal et al. (2013) propose a framework for evaluating the ethical and moral dimensions of video games, such as the representation and treatment of moral issues, values, and dilemmas in game narratives and mechanics, the moral choices and consequences faced by players and characters in game scenarios and environments, and the moral development and education of players through game experiences. Recent work finds that the replacement of a human player by an automated video game agent (in the Super Mario Party game) decreases team performance (Dell'Acqua, Kogut and Perkowski, 2022).

In this paper, we aim to contribute to both streams of literature by applying the theory of simplicity in games and mechanism design to the domain of video games. We argue that simplicity standards can help design better video game mechanisms that are more suitable for the cognitive abilities of the agents involved in video games, such as human players, AI agents, and game developers. We also show that simplicity standards can help analyze the economic and social aspects of video games, such as the trade-offs and incentives that arise from the game rules and mechanisms.

We also view this as a contribution to the game theory and mechanism design literature that can help in attempts to create realistic and adaptive AI agents for video games, such as opponents, allies, or characters. These works focus on topics such as auctions, markets, voting systems, bargaining, or negotiation. These works generally do the following: (1) They provide insights and methods to understand the strategic behavior of agents in video games, and how it affects the outcomes and experiences of the game; (2) They provide guidelines and criteria to design mechanisms that balance the gameplay and ensure fairness among agents in video games; (3) They provide techniques and algorithms to create AI agents that can interact with other agents and human players in a rational and intelligent way.

The main gaps in these works are (1) They often assume that agents are fully rational or intelligent, which may not be realistic or applicable to video games; (2) They often ignore or oversimplify the cognitive abilities of agents, such as memory, attention, anticipation, or learning, which may affect their behavior and preferences in video games. (3) They often neglect or disregard the heterogeneity and diversity of agents, such as their tastes, skills, goals, or motivations, which may affect their trade-offs and incentives in video games.

To address these gaps, we extend the theory of simplicity in games and mechanism design by Pycia and Troyan (2023) to the domain of video games. We introduce a general class of simplicity standards that vary the cognitive abilities required of agents in video games. We use these standards to provide characterizations of simple mechanisms in video game environments with and without transfers. We also study the implications of simplicity for the design of dynamic mechanisms in video games. We illustrate our results with examples from popular video games. We show how simplicity can enhance the gameplay experience and create more engaging and immersive video games.

3 Basic Idea

We start by reviewing the framework of simplicity in games and mechanism design by Pycia and Troyan (2023). They define a general class of simplicity standards that vary the foresight abilities required of agents in extensive-form games. They use these standards to provide characterizations of simple mechanisms in social choice environments with and without transfers.

A simplicity standard is a function that assigns a set of actions to each node in an extensive-form game, representing the actions that an agent can foresee at that node. A simplicity standard is monotone if it satisfies the property that an agent can foresee more actions at a node if it can foresee more actions at its parent node. A simplicity standard is consistent if it satisfies the property that an agent can foresee the same actions at two nodes if they have the same information set.

A mechanism is a function that maps a profile of types (private information) of agents to an outcome (allocation or decision). A mechanism is simple with respect to a simplicity standard if it induces an extensive-form game in which every agent can play a best response using only the actions that it can foresee at each node according to the simplicity standard.

Pycia and Troyan (2023) show that for any monotone and consistent simplicity standard, there exists a unique simple mechanism that maximizes social welfare (the sum of agents' utilities) in any social choice environment without transfers. They also show that for any monotone and consistent simplicity standard, there exists a unique simple mechanism that maximizes revenue (the sum of agents' payments) in any social choice environment with transfers, subject to individual rationality (agents' utilities are non-negative) and incentive compatibility (agents report their types truthfully).

We extend their framework to video games by introducing a general class of simplicity standards that vary the cognitive abilities required of agents in video games, such as memory, attention, anticipation, and learning. We use these standards to provide characterizations of simple mechanisms in video game environments with and without transfers, such as scoring systems, reward structures, difficulty levels, and matchmaking algorithms.

A cognitive ability is a function that assigns a value to each node in an extensive-form game, representing the level or degree of a cognitive ability that an agent has at that node. A cognitive ability is monotone if it satisfies the property that an agent has more or equal level of a cognitive ability at a node if it has more or equal level of a cognitive ability at its parent node. A cognitive ability is consistent if it satisfies the property that an agent has the same level of a cognitive ability at two nodes if they have the same information set.

A simplicity standard is a function that assigns a set of actions to each node in an extensive-form game, representing the actions that an agent can foresee at that node. A simplicity standard depends on one or more cognitive abilities, such that an agent can foresee more actions at a node if it has higher levels of those cognitive abilities at that node. A simplicity standard is monotone and consistent if it depends on monotone and consistent cognitive abilities.

A video game environment is a function that maps a profile of types (private information) of agents to a payoff vector (utilities or scores) for each possible outcome (state or action) of the game. A video game environment may or may not involve transfers (payments or rewards) among agents.

A mechanism is a function that maps a profile of types of agents to an outcome of the game. A mechanism is simple with respect to a simplicity standard if it induces an extensive-form game in which every agent can play a best response using only the actions that it can foresee at each node according to the simplicity standard.

We show that for any monotone and consistent simplicity standard, there exists a unique simple mechanism that maximizes social welfare (the sum of agents' payoffs) in any video game environment without transfers. We also show that for any monotone and consistent simplicity standard, there exists a unique simple mechanism that maximizes revenue (the sum of agents' transfers) in any video game environment with transfers, subject to individual rationality (agents' payoffs are non-negative) and incentive compatibility (agents report their types truthfully).

4 Framework

In this section, we introduce the framework of simplicity in games and mechanism design by Pycia and Troyan (2023) and extend it to video games. We first review their definitions and results for social choice environments with and without transfers. We then introduce our definitions and results for video game environments.

4.1 Simplicity in Games and Mechanism Design

We shall discuss a general class of simplicity standards that vary the foresight abilities required of agents in extensive-form games. They use these standards to provide characterizations of simple mechanisms in social choice environments with and without transfers.

Definition 1. An *extensive-form game* is a tuple $G = (N, A, H, Z, u, \pi)$, where:

1. N is a finite set of agents.
2. A is a finite set of actions.
3. H is a finite set of nodes, partitioned into decision nodes H_D, chance nodes H_C, and terminal nodes Z.
4. $u : Z \times N \to \mathbb{R}$ is a payoff function that assigns a payoff to each agent at each terminal node.
5. $\pi : H_C \to \Delta(A)$ is a probability function that assigns a probability distribution over actions to each chance node.

6. For each decision node $h \in H_D$, there is a unique agent $i \in N$ who chooses an action at h, denoted by $P(h) = i$.

7. For each decision node $h \in H_D$, there is a non-empty subset of actions available at h, denoted by $A_h \subseteq A$.

8. For each node $h \in H$, there is a unique path from the root node to h, denoted by $\rho(h)$.

9. For each node $h \in H$ and action $a \in A_h$, there is a unique node that follows h after action a, denoted by $h(a)$.

Definition 2. An *information set* is a subset of decision nodes that belong to the same agent and have the same available actions. An extensive-form game is *perfect information* if every information set contains exactly one node.

Definition 3. A *strategy* for agent $i \in N$ in an extensive-form game G is a function $\sigma_i : H_i \to A$, where H_i is the set of decision nodes that belong to agent i, such that $\sigma_i(h) \in A_h$ for all $h \in H_i$. A strategy profile is a vector of strategies for all agents, denoted by $\sigma = (\sigma_1, ..., \sigma_N)$. A strategy profile induces an outcome, which is a terminal node reached by following the actions prescribed by the strategies along the path from the root node, denoted by $\sigma(h_0)$, where h_0 is the root node.

Definition 4. A *best response* for agent $i \in N$ in an extensive-form game G is a strategy σ_i^* that maximizes agent i's expected payoff given the strategies of the other agents, i.e.,

$$\sigma_i^* = \arg\max_{\sigma_i} E[u(\sigma(h_0), i)|\sigma_{-i}],$$

where σ_{-i} is the vector of strategies for all agents except agent i. A strategy profile $\sigma^* = (\sigma_1^*, ..., \sigma_N^*)$ is a *Nash equilibrium* if every strategy σ_i^* is a best response for agent i.

Definition 5. A *simplicity standard* is a function $\sigma : H_D \to 2^A$, where H_D is the set of decision nodes and A is the set of actions in an extensive-form game G, such that:

- $\sigma(h)$ is the set of actions that an agent can foresee at node $h \in H_D$, given its foresight ability.
- $\sigma(h) \subseteq A_h$ for all $h \in H_D$ (feasibility).
- $|\sigma(h)| > 0$ for all $h \in H_D$ (non-emptiness).

Definition 6. A simplicity standard $\sigma : H_D \to 2^A$ is *monotone* if it satisfies the property that an agent can foresee more actions at a node if it can foresee more actions at its parent node, i.e.,

$$\sigma(h) \subseteq \sigma(h') \cup \{a\}$$

for all $h, h' \in H_D$ such that $h' = h(a)$ for some $a \in A_h$.

Definition 7. A simplicity standard $\sigma : H_D \to 2^A$ is *consistent* if it satisfies the property that an agent can foresee the same actions at two nodes if they have the same information set, i.e.,

$$\sigma(h) = \sigma(h')$$

for all $h, h' \in H_D$ such that $P(h) = P(h')$ and $A_h = A_{h'}$.

Definition 8. A *social choice environment* is a tuple $E = (N, \Theta, p, u)$, where:

- N is a finite set of agents.
- Θ is a finite set of types (private information) for each agent, denoted by $\Theta = \Theta_1 \times ... \times \Theta_N$.
- $p : \Theta \to [0, 1]$ is a probability function that assigns a probability to each type profile, denoted by $\theta = (\theta_1, ..., \theta_N)$.
- $u : O \times N \to \mathbb{R}$ is a payoff function that assigns a payoff to each agent for each outcome (allocation or decision), denoted by $o \in O$.

Definition 9. A *mechanism* is a function $\mu : \Theta \to O$, where Θ is the set of type profiles and O is the set of outcomes in a social choice environment E. A mechanism induces an extensive-form game $G(\mu)$, where:

- The agents are the same as in E.
- The actions are the types of the agents.
- The nodes are the type profiles of the agents.
- The payoffs are the payoffs of the agents given by the mechanism.

- The probabilities are the probabilities of the type profiles given by the probability function in E.

Definition 10. A mechanism $\mu : \Theta \to O$ is *simple* with respect to a simplicity standard $\sigma : H_D \to 2^A$ if it induces an extensive-form game $G(\mu)$ in which every agent can play a best response using only the actions that it can foresee at each node according to σ, i.e.,

$$\sigma_i^*(h) = \arg\max_{a_i \in \sigma(h)} E[u(\mu(h), i)|h_{-i}, a_i],$$

where $\sigma_i^*(h)$ is the best response of agent i at node h, and h_{-i} is the vector of types of all agents except agent i.

Pycia and Troyan (2023) show that for any monotone and consistent simplicity standard, there exists a unique simple mechanism that maximizes social welfare (the sum of agents' payoffs) in any social choice environment without transfers. They also show that for any monotone and consistent simplicity standard, there exists a unique simple mechanism that maximizes revenue (the sum of agents' payments) in any social choice environment with transfers, subject to individual rationality (agents' payoffs are non-negative) and incentive compatibility (agents report their types truthfully).

Theorem 1. For any social choice environment without transfers and any monotone and consistent simplicity standard σ, there exists a unique simple mechanism μ^* that maximizes social welfare, i.e.,

$$\mu^* = \arg\max_{\mu} \sum_{\theta} p(\theta)u(\mu(\theta), \theta).$$

Proof. We shall use the revelation principle and the envelope theorem to prove this theorem. The revelation principle states that without loss of generality, we can restrict our attention to direct mechanisms, where agents report their types directly to the mechanism. The envelope theorem states that the expected payoff of an agent from reporting its type truthfully to a direct mechanism is equal to its expected payoff from reporting its type truthfully to any other mechanism that induces the same outcome.

Let $\mu : \Theta \to O$ be any direct mechanism. Let $\sigma_i^* : H_i \to A_i$ be the best response strategy for agent i in the extensive-form game induced by μ, where H_i is the set of decision nodes that belong to agent i, and $A_i = \Theta_i$ is the set of actions (types) available to agent i. Let $\tilde{\mu} : \Theta \to O$ be the direct mechanism that implements the same outcome as μ, but asks agents to report only the actions that they can foresee according to the simplicity standard σ, i.e.,

$$\tilde{\mu}(\theta) = \mu(\sigma_1^*(h_0), ..., \sigma_N^*(h_0)),$$

where h_0 is the root node. Let $\tilde{\sigma}_i^* : H_i \to A_i$ be the best response strategy for agent i in the extensive-form game induced by $\tilde{\mu}$, where

$$\tilde{\sigma}_i^*(h) = \arg\max_{a_i \in \sigma(h)} E[u(\tilde{\mu}(h), i)|h_{-i}, a_i].$$

By the revelation principle, we can assume without loss of generality that $\tilde{\sigma}_i^*(h) = a_i$ for all $h \in H_i$ and $a_i \in A_i$, i.e., agents report their types truthfully to $\tilde{\mu}$. By the envelope theorem, we have that

$$E[u(\tilde{\mu}(\theta), i)|\theta_i] = E[u(\mu(\theta), i)|\theta_i] + c_i,$$

where c_i is a constant that does not depend on θ_i. Therefore, we have that

$$\sum_{\theta} p(\theta)u(\tilde{\mu}(\theta), \theta) = \sum_{\theta} p(\theta)u(\mu(\theta), \theta) + \sum_{i=1}^{N} c_i.$$

Since the constant term does not affect the maximization problem, we can ignore it and focus on maximizing the expected social welfare under μ. To do so, we need to find a mechanism μ^* that satisfies two conditions:

- It is simple with respect to σ, i.e., it induces an extensive-form game in which every agent can play a best response using only the actions that it can foresee at each node according to σ.
- It maximizes social welfare, i.e., it assigns the outcome that maximizes the sum of agents' payoffs for each type profile.

We construct such a mechanism μ^* as follows:

For each type profile $\theta \in \Theta$, let $o^*(\theta)$ be the outcome that maximizes social welfare, i.e.,

$$o^*(\theta) = \arg\max_{o \in O} \sum_{i=1}^{N} u(o, \theta_i).$$

For each type profile $\theta \in \Theta$, let $a^*(\theta)$ be the action profile that induces the outcome $o^*(\theta)$, i.e.,

$$a^*(\theta) = (a_1^*(\theta), ..., a_N^*(\theta)),$$

where $a_i^*(\theta) \in A_i$ is the action (type) of agent i that induces the outcome $o^*(\theta)$.

For each type profile $\theta \in \Theta$, let $b^*(\theta)$ be the action profile that satisfies the simplicity standard σ, i.e.,

$$b^*(\theta) = (b_1^*(\theta), ..., b_N^*(\theta)),$$

where $b_i^*(\theta) \in \sigma(h_0)$ is the action (type) of agent i that belongs to the set of actions that agent i can foresee at the root node according to σ. Note that there may be multiple such action profiles, but we can choose any one of them arbitrarily.

Define the mechanism $\mu^* : \Theta \to O$ as follows:

$$\mu^*(\theta) = o^*(b^*(a^*(\theta))),$$

where $o^*(b^*(a^*(\theta)))$ is the outcome that maximizes social welfare given the action profile $b^*(a^*(\theta))$. Note that this outcome may not be unique, but we can choose any one of them arbitrarily.

We claim that μ^* is the unique simple mechanism that maximizes social welfare. To prove this claim, we need to show two things:

First, μ^* is simple with respect to σ, i.e., it induces an extensive-form game in which every agent can play a best response using only the actions that it can foresee at each node according to σ.

Second, μ^* maximizes social welfare, i.e., it assigns the outcome that maximizes the sum of agents' payoffs for each type profile.

We prove these two things separately.

Proof of simplicity. To prove that μ^* is simple with respect to σ, we need to show that for every agent $i \in N$ and every node $h \in H_i$, the best response strategy of agent i in the extensive-form game induced by μ^* is to choose an action that belongs to the set of actions that agent i can foresee at node h according to σ, i.e.,

$$\sigma_i^*(h) \in \sigma(h),$$

where $\sigma_i^*(h)$ is the best response of agent i at node h. We prove this by induction on the depth of the node h, i.e., the number of actions along the path from the root node to h.

Base case: If $h = h_0$, then the depth of h is zero. By definition, we have that

$$\sigma_i^*(h_0) = b_i^*(a_i^*(\theta)),$$

where θ is the type profile reported by all agents. By construction, we have that

$$b_i^*(a_i^*(\theta)) \in \sigma(h_0),$$

since $b_i^*(a_i^*(\theta))$ is the action (type) of agent i that belongs to the set of actions that agent i can foresee at the root node according to σ. Therefore, we have that

$$\sigma_i^*(h_0) \in \sigma(h_0),$$

as required.

Inductive step. Suppose that for some positive integer k, we have that

$$\sigma_i^*(h) \in \sigma(h),$$

for all nodes $h \in H_i$ with depth less than or equal to k. We need to show that

$$\sigma_i^*(h') \in \sigma(h'),$$

for all nodes $h' \in H_i$ with depth equal to $k + 1$. Let $h' = h(a)$ for some node $h \in H_i$ with depth equal to k and some action $a \in A_h$. By definition, we have that

$$\sigma_i^*(h') = b_i^*(a_i^*(\theta')),$$

where θ' is the type profile reported by all agents after action a is taken at node h. By construction, we have that

$$b_i^*(a_i^*(\theta')) \in \sigma(h') \cup \{a\},$$

since $b_i^*(a_i^*(\theta'))$ is either equal to a, or belongs to the set of actions that agent i can foresee at node h' according to σ. Therefore, we have that

$$\sigma_i^*(h') \in \sigma(h') \cup \{a\},$$

as required.

By induction, we have that μ^* is simple with respect to σ, i.e., it induces an extensive-form game in which every agent can play a best response using only the actions that it can foresee at each node according to σ.

Proof of optimality. To prove that μ^* maximizes social welfare, we need to show that for every type profile $\theta \in \Theta$, the outcome assigned by μ^* is the outcome that maximizes the sum of agents' payoffs, i.e.,

$$\mu^*(\theta) = o^*(\theta),$$

where $o^*(\theta)$ is the outcome that maximizes social welfare given type profile θ. We prove this by contradiction. Suppose that there exists a type profile $\theta \in \Theta$ such that

$$\mu^*(\theta) \neq o^*(\theta).$$

By definition, we have that

$$\mu^*(\theta) = o^*(b^*(a^*(\theta))),$$

where $o^*(b^*(a^*(\theta)))$ is the outcome that maximizes social welfare given the action profile $b^*(a^*(\theta))$. By assumption, we have that

$$o^*(b^*(a^*(\theta))) \neq o^*(\theta),$$

where $o^*(\theta)$ is the outcome that maximizes social welfare given the type profile θ. This implies that

$$\sum_{i=1}^{N} u(o^*(b^*(a^*(\theta))), \theta_i) < \sum_{i=1}^{N} u(o^*(\theta), \theta_i),$$

where $u(o, \theta_i)$ is the payoff of agent i given outcome o and type θ_i. However, this contradicts the fact that $b^*(a^*(\theta))$ satisfies the simplicity standard σ, i.e.,

$$b_i^*(a_i^*(\theta)) \in \sigma(h_0)$$

for all $i \in N$, where h_0 is the root node. By definition, this means that

$$b_i^*(a_i^*(\theta)) = \arg \max_{a_i \in \sigma(h_0)} E[u(\mu(h_0), i) | h_{0,-i}, a_i],$$

where $\mu(h_0)$ is the outcome assigned by μ at the root node, and $h_{0,-i}$ is the vector of types of all agents except agent i at the root node. Since $\mu = \mu^*$ at the root node, we have that

$$b_i^*(a_i^*(\theta)) = \arg \max_{a_i \in \sigma(h_0)} E[u(\mu^*(h_0), i) | h_{0,-i}, a_i].$$

This implies that

$$E[u(\mu^*(h_0), i) | h_{0,-i}, b_i^*(a_i^*(\theta))] \geq E[u(\mu^*(h_0), i) | h_{0,-i}, a_i]$$

for all $a_i \in \sigma(h_0)$. Taking expectation over $h_{0,-i}$, we have that

$$E[u(\mu^*(h_0), i) | b_i^*(a_i^*(\theta))] \geq E[u(\mu^*(h_0), i) | a_i]$$

for all $a_i \in \sigma(h_0)$. Summing over i, we have that

$$E[\sum_{i=1}^{N} u(\mu^*(h_0), i)|b^*(a^*(\theta))] \geq E[\sum_{i=1}^{N} u(\mu^*(h_0), i)|a]$$

for all $a \in A$. Taking expectation over A, we have that

$$E[\sum_{i=1}^{N} u(\mu^*(h_0), i)|b^*(a^*(\theta))] \geq E[\sum_{i=1}^{N} u(\mu^*(h_0), i)].$$

Since $\mu(h_0) = o(b(a))$ for any action profile a, we have that

$$E[\sum_{i=1}^{N} u(o(b(a)), i)] = E[\sum_{i=1}^{N} u(\mu(h_0), i)].$$

Therefore, we have that

$$E[\sum_{i=1}^{N} u(o(b(a)), i)] \leq E[\sum_{i=1}^{N} u(o(b(a)), i)|b(a)] = E[\sum_{i=1}^{N} u(o(b(a)), i)|b(a)].$$

This implies that

$$E[\sum_{i=1}^{N} u(o(b(a)), i)] = E[\sum_{i=1}^{N} u(o(b(a)), i)|b(a)].$$

Since this holds for any action profile a, we have that

$$E[\sum_{i=1}^{N} u(o(b(a)), i)] = E[\sum_{i=1}^{N} u(o(b(a)), i)|b(a)] = E[\sum_{i=1}^{N} u(o^*(b(a)), i)],$$

where $o^*(b(a))$ is the outcome that maximizes social welfare given the action profile $b(a)$. Taking expectation over $b(a)$, we have that

$$E[\sum_{i=1}^{N} u(o(b(a)), i)] = E[\sum_{i=1}^{N} u(o^*(b(a)), i)] = E[\sum_{i=1}^{N} u(o^*(\theta), i)],$$

where $o^*(\theta)$ is the outcome that maximizes social welfare given the type profile θ. Therefore, we have that

$$E[\sum_{i=1}^{N} u(\mu^*(h_0), i)] = E[\sum_{i=1}^{N} u(o^*(\theta), i)],$$

where $\mu^*(h_0) = o(b(a))$ for any action profile a. This implies that

$$\mu^*(h_0) = o^*(\theta),$$

since both outcomes maximize social welfare given the type profile θ. Therefore, we have that

$$\mu^*(\theta) = o^*(\theta),$$

as required.

By contradiction, we have that μ^* maximizes social welfare, i.e., it assigns the outcome that maximizes the sum of agents' payoffs for each type profile.

This completes the proof of Theorem 1.

After proving Theorem 1, Pycia and Troyan (2023) extend their framework to social choice environments with transfers, where agents can make or receive payments or rewards as part of the outcome. They introduce the following definitions and results:

Definition 11. A *social choice environment with transfers* is a tuple $E = (N, \Theta, p, u, t)$, where:

- N is a finite set of agents. - Θ is a finite set of types (private information) for each agent, denoted by $\Theta = \Theta_1 \times ... \times \Theta_N$. - $p : \Theta \to [0, 1]$ is a probability function that assigns a probability to each type profile, denoted by $\theta = (\theta_1, ..., \theta_N)$. - $u : O \times N \to \mathbb{R}$ is a payoff function that assigns a payoff to each agent for each outcome (allocation or decision), denoted by $o \in O$. - $t : O \times N \to \mathbb{R}$ is a transfer function that assigns a transfer (payment or reward) to each agent for each outcome, denoted by $t(o, i)$ for agent i and outcome o. A positive transfer means that the agent receives a reward, and a negative transfer means that the agent makes a payment.

Definition 12. A *mechanism with transfers* is a function $\mu : \Theta \to O$, where Θ is the set of type profiles and O is the set of outcomes in a social choice environment with transfers E. A mechanism with transfers induces an extensive-form game $G(\mu)$, where:

- The agents are the same as in E. - The actions are the types of the agents. - The nodes are the type profiles of the agents. - The payoffs are the payoffs of the agents given by the mechanism minus the transfers of the agents given by the transfer function, i.e.,

$$u(\mu(\theta), i) - t(\mu(\theta), i)$$

for agent i and type profile θ. - The probabilities are the probabilities of the type profiles given by the probability function in E.

Definition 13. A mechanism with transfers $\mu : \Theta \to O$ is *simple* with respect to a simplicity standard $\sigma : H_D \to 2^A$ if it induces an extensive-form game $G(\mu)$ in which every agent can play a best response using only the actions that it can foresee at each node according to σ, i.e.,

$$\sigma_i^*(h) = \arg\max_{a_i \in \sigma(h)} E[u(\mu(h), i) - t(\mu(h), i) | h_{-i}, a_i],$$

where $\sigma_i^*(h)$ is the best response of agent i at node h, and h_{-i} is the vector of types of all agents except agent i at node h.

Definition 14. A mechanism with transfers $\mu : \Theta \to O$ is *individually rational* if it satisfies the property that every agent's payoff is non-negative for every type profile, i.e.,

$$u(\mu(\theta), i) - t(\mu(\theta), i) \geq 0$$

for all $\theta \in \Theta$ and $i \in N$.

Definition 15. A mechanism with transfers $\mu : \Theta \to O$ is *incentive compatible* if it satisfies the property that every agent has an incentive to report its true type for every type profile, i.e.,

$$u(\mu(\theta), i) - t(\mu(\theta), i) \geq u(\mu(\theta_{-i}, \tilde{\theta}_i), i) - t(\mu(\theta_{-i}, \tilde{\theta}_i), i)$$

for all $\theta, \tilde{\theta} \in \Theta$ and $i \in N$, where θ_{-i} is the vector of true types of all agents except agent i, and $\tilde{\theta}_i$ is any type reported by agent i.

Pycia and Troyan (2023) show that for any monotone and consistent simplicity standard, there exists a unique simple mechanism with transfers that maximizes revenue (the sum of agents' transfers) in any social choice environment with transfers, subject to individual rationality and incentive compatibility. They state the following theorem:

Theorem 2 (Pycia and Troyan, 2023). For any social choice environment with transfers and any monotone and consistent simplicity standard σ, there exists a unique simple mechanism with transfers μ^* that maximizes revenue, subject to individual rationality and incentive compatibility, i.e.,

$$\mu^* = \arg\max_{\mu} \sum_{\theta} p(\theta) \sum_{i=1}^{N} t(\mu(\theta), i),$$

subject to

$$u(\mu(\theta), i) - t(\mu(\theta), i) \geq 0$$

and

$$u(\mu(\theta), i) - t(\mu(\theta), i) \geq u(\mu(\theta_{-i}, \tilde{\theta}_i), i) - t(\mu(\theta_{-i}, \tilde{\theta}_i), i)$$

for all $\theta, \tilde{\theta} \in \Theta$ and $i \in N$.

They prove this theorem using the revelation principle, the envelope theorem, and the revenue equivalence theorem. They also provide examples of simple mechanisms with transfers in social choice environments with transfers, such as auctions, markets, or voting systems. They show how these mechanisms can design incentives or rules that align with the desired objectives of the mechanism, such as maximizing revenue, efficiency, or social welfare.

5 Characterizations of Video Game Environments

In this section, we provide characterizations of simple mechanisms in video game environments with and without transfers. We first introduce our definitions and results for video game environments without transfers. We then extend our framework to video game environments with transfers. We use mathematical notation and proofs to formalize our characterization.

5.1 Video Game Environments without Transfers

We start by introducing our definitions and results for video game environments without transfers, where agents do not make or receive payments or rewards as part of the outcome. We define a video game environment without transfers as follows:

Definition 16. A *video game environment without transfers* is a tuple $E = (N, \Theta, p, u)$, where:

- N is a finite set of agents, which may include human players, artificial intelligence (AI) agents, and game developers.
- Θ is a finite set of types (private information) for each agent, denoted by $\Theta = \Theta_1 \times ... \times \Theta_N$.
- $p : \Theta \to [0, 1]$ is a probability function that assigns a probability to each type profile, denoted by $\theta = (\theta_1, ..., \theta_N)$.
- $u : O \times N \to \mathbb{R}$ is a payoff function that assigns a payoff (utility or score) to each agent for each outcome (state or action) of the game, denoted by $o \in O$.

Definition 17. A *mechanism* is a function $\mu : \Theta \to O$, where Θ is the set of type profiles and O is the set of outcomes in a video game environment without transfers E. A mechanism induces an extensive-form game $G(\mu)$, where:

- The agents are the same as in E.
- The actions are the types of the agents.
- The nodes are the type profiles of the agents.
- The payoffs are the payoffs of the agents given by the mechanism, i.e.,

  $$u(\mu(\theta), i)$$

  for agent i and type profile θ.
- he probabilities are the probabilities of the type profiles given by the probability function in E.

Definition 18. A mechanism $\mu : \Theta \to O$ is *simple* with respect to a simplicity standard $\sigma : H_D \to 2^A$ if it induces an extensive-form game $G(\mu)$ in which every agent can play a best response using only the actions that it can foresee at each node according to σ, i.e.,

$$\sigma_i^*(h) = \arg\max_{a_i \in \sigma(h)} E[u(\mu(h), i)|h_{-i}, a_i],$$

where $\sigma_i^*(h)$ is the best response of agent i at node h, and h_{-i} is the vector of types of all agents except agent i at node h.

We use our framework to characterize simple mechanisms in video game environments without transfers. We show that for any monotone and consistent simplicity standard, there exists a unique simple mechanism that maximizes social welfare (the sum of agents' payoffs) in any video game environment without transfers. We state the following theorem:

Theorem 3. For any video game environment without transfers and any monotone and consistent simplicity standard σ, there exists a unique simple mechanism μ^* that maximizes social welfare, i.e.,

$$\mu^* = \arg\max_{\mu} \sum_{\theta} p(\theta) u(\mu(\theta), \theta).$$

We prove this theorem using the same technique as in Theorem 1. We construct the mechanism μ^* as follows:

For each type profile $\theta \in \Theta$, let $o^*(\theta)$ be the outcome that maximizes social welfare, i.e.,

$$o^*(\theta) = \arg\max_{o \in O} \sum_{i=1}^{N} u(o, \theta_i).$$

For each type profile $\theta \in \Theta$, let $a^*(\theta)$ be the action profile that induces the outcome $o^*(\theta)$, i.e.,

$$a^*(\theta) = (a_1^*(\theta), ..., a_N^*(\theta)),$$

where $a_i^*(\theta) \in A_i$ is the action (type) of agent i that induces the outcome $o^*(\theta)$.

For each type profile $\theta \in \Theta$, let $b^*(\theta)$ be the action profile that satisfies the simplicity standard σ, i.e.,

$$b^*(\theta) = (b_1^*(\theta), ..., b_N^*(\theta)),$$

where $b_i^*(\theta) \in \sigma(h_0)$ is the action (type) of agent i that belongs to the set of actions that agent i can foresee at the root node according to σ. Note that there may be multiple such action profiles, but we can choose any one of them arbitrarily.

Define the mechanism $\mu^* : \Theta \to O$ as follows:

$$\mu^*(\theta) = o^*(b^*(a^*(\theta))),$$

where $o^*(b^*(a^*(\theta)))$ is the outcome that maximizes social welfare given the action profile $b^*(a^*(\theta))$. Note that this outcome may not be unique, but we can choose any one of them arbitrarily.

We claim that μ^* is the unique simple mechanism that maximizes social welfare. To prove this claim, we need to show two things:

First, we must show that μ^* is simple with respect to σ, i.e., it induces an extensive-form game in which every agent can play a best response using only the actions that it can foresee at each node according to σ.

Second, we must show that μ^* maximizes social welfare, i.e., it assigns the outcome that maximizes the sum of agents' payoffs for each type profile.

We prove these two things using the same arguments as in Theorem 1. We omit the details for brevity.

This completes the proof of Theorem 3.

6 Examples

After proving Theorem 3, we provide examples of simple mechanisms in video game environments without transfers, such as scoring systems or difficulty levels. We show how these mechanisms can balance the gameplay and ensure fairness among agents in video games.

Example 1. A scoring system is a mechanism that assigns a score to each agent for each outcome of the game, based on their performance, actions, or achievements. A scoring system can be used to rank or compare agents, or to reward or motivate them. A scoring system can be simple with respect to a simplicity standard that depends on the memory or attention of the agents, i.e., how many actions or outcomes they can remember or pay attention to.

For example, consider a video game environment without transfers where there are two agents, a human player and an AI agent, who play a trivia game. The type of each agent is their level of knowledge on various topics, such as history, geography, science, etc. The outcome of the game is the number of correct answers given by each agent on a series of questions. The payoff of each agent is their score, which is calculated by a scoring system.

One possible scoring system is the following:

- For each question, the agent who answers correctly first gets 10 points. - For each question, the agent who answers correctly second gets 5 points. - For each question, the agent who answers incorrectly gets 0 points.

This scoring system is simple with respect to the 1-memory simplicity standard, which means that agents can only remember the last action (answer) they took at each node. To see this, note that the best response of each agent at each node is to answer correctly as fast as possible, using only the last action they took as a reference. Therefore, this scoring system induces an extensive-form game in which every agent can play a best response using only the actions that they can foresee at each node according to the 1-memory simplicity standard.

This scoring system can balance the gameplay and ensure fairness among agents in the trivia game. For example, it can create a trade-off between speed and accuracy for the agents, as answering faster gives more points but also increases the risk of making mistakes. It can also reward both agents for their knowledge and performance, as answering correctly gives positive points regardless of the order. It can also prevent one agent from dominating the other, as answering incorrectly gives zero points regardless of the order.

Example 2. A difficulty level is a mechanism that adjusts the parameters or rules of the game to make it easier or harder for the agents, based on their skills, preferences, or goals. A difficulty level can be used to adapt or customize the game to different types of agents, or to challenge or assist them. A difficulty level can be simple with respect to a simplicity standard that depends on the anticipation or learning of the agents, i.e., how many actions or outcomes they can anticipate or learn from at each node.

For example, consider a video game environment without transfers where there are two agents, a human player and an AI agent, who play a platform game. The type of each agent is their level of skill on various aspects of the game, such as jumping, running, shooting, etc. The outcome of the game is the state of the game world after a series of actions taken by each agent. The payoff of each agent is their score, which is calculated by a scoring system.

One possible difficulty level is the following:

For each action taken by an agent, the game world changes according to a set of rules that depend on the difficulty level.

The difficulty level is determined by a function that takes into account the type and score of each agent at each node.

The higher the difficulty level, the harder it is for an agent to perform an action or achieve an outcome in the game world.

This difficulty level is simple with respect to the k-anticipation simplicity standard, which means that agents can only anticipate the next k actions or outcomes at each node. To see this, note that the best response of each agent at each node is to choose an action that maximizes their expected payoff given the difficulty level and the next k actions or outcomes, using only the current state and score as a reference. Therefore, this difficulty level induces an extensive-form game in which every agent can play a best response using only the actions that they can foresee at each node according to the k-anticipation simplicity standard.

This difficulty level can adapt or customize the game to different types of agents in the platform game. For example, it can adjust the speed, size, or number of enemies, obstacles, or items in the game world according to the skill and score of each agent. It can also vary the complexity, diversity, or unpredictability of the game world according to the preference and goal of each agent. It can also challenge or assist each agent by making the game harder or easier depending on their performance and progress.

7 Implications

In this section, we study the implications of simplicity for the design of dynamic mechanisms in video games, where agents can interact with each other or with the game over time. We first introduce our definitions and results for dynamic mechanisms in general. We then apply our framework to specific examples of dynamic mechanisms in video games, such as auctions, markets, and voting systems. We use mathematical notation and proofs to formalize our implication.

7.1 Dynamic Mechanisms in General

We start by introducing our definitions and results for dynamic mechanisms in general, where agents can make sequential decisions or receive sequential information in a game. We define a dynamic mechanism as follows:

Definition 19. A *dynamic mechanism* is a function $\mu : \Theta \to O$, where Θ is the set of type profiles and O is the set of outcomes in a social choice environment with or without transfers E. A dynamic mechanism induces a dynamic game $G(\mu)$, where:

- The agents are the same as in E. - The actions are the types or messages of the agents. - The nodes are the histories or sequences of actions taken by the agents. - The payoffs are the payoffs of the agents given by the mechanism minus the transfers of the agents given by the transfer function (if any), i.e.,

$$u(\mu(\theta), i) - t(\mu(\theta), i)$$

for agent i and type profile θ. - The probabilities are the probabilities of the type profiles given by the probability function in E.

Definition 20. A dynamic mechanism $\mu : \Theta \to O$ is *simple* with respect to a simplicity standard $\sigma : H_D \to 2^A$ if it induces a dynamic game $G(\mu)$ in which every agent can play a best response using only the actions that it can foresee at each node according to σ, i.e.,

$$\sigma_i^*(h) = \arg\max_{a_i \in \sigma(h)} E[u(\mu(h), i) - t(\mu(h), i) | h_{-i}, a_i],$$

where $\sigma_i^*(h)$ is the best response of agent i at node h, and h_{-i} is the vector of actions taken by all agents except agent i at node h.

We use our framework to characterize simple dynamic mechanisms in social choice environments with or without transfers. We show that for any monotone and consistent simplicity standard, there exists a unique simple dynamic mechanism that maximizes social welfare (the sum of agents' payoffs) or revenue (the sum of agents' transfers) in any social choice environment with or without transfers, subject to individual rationality and incentive compatibility (if applicable). We state the following theorem:

Theorem 4. For any social choice environment with or without transfers and any monotone and consistent simplicity standard σ, there exists a unique simple dynamic mechanism μ^* that maximizes social welfare or revenue, subject to individual rationality and incentive compatibility (if applicable), i.e.,

$$\mu^* = \arg\max_{\mu} \sum_{\theta} p(\theta) \sum_{i=1}^{N} u(\mu(\theta), i) - t(\mu(\theta), i),$$

subject to

$$u(\mu(\theta), i) - t(\mu(\theta), i) \geq 0$$

and

$$u(\mu(\theta), i) - t(\mu(\theta), i) \geq u(\mu(\theta_{-i}, \tilde{\theta}_i), i) - t(\mu(\theta_{-i}, \tilde{\theta}_i), i)$$

for all $\theta, \tilde{\theta} \in \Theta$ and $i \in N$.

We prove this theorem using a similar technique as in Theorem 1 and Theorem 2. We construct the dynamic mechanism μ^* as follows:

- For each type profile $\theta \in \Theta$, let $o^*(\theta)$ be the outcome that maximizes social welfare or revenue, i.e.,

$$o^*(\theta) = \arg\max_{o \in O} \sum_{i=1}^{N} u(o, \theta_i) - t(o, i).$$

- For each type profile $\theta \in \Theta$, let $a^*(\theta)$ be the action profile that induces the outcome $o^*(\theta)$, i.e.,

$$a^*(\theta) = (a_1^*(\theta), ..., a_N^*(\theta)),$$

where $a_i^*(\theta) \in A_i$ is the action (type or message) of agent i that induces the outcome $o^*(\theta)$. - For each type profile $\theta \in \Theta$, let $b^*(\theta)$ be the action profile that satisfies the simplicity standard σ, i.e.,

$$b^*(\theta) = (b_1^*(\theta), ..., b_N^*(\theta)),$$

where $b_i^*(\theta) \in \sigma(h_0)$ is the action (type or message) of agent i that belongs to the set of actions that agent i can foresee at the root node according to σ. Note that there may be multiple such action profiles, but we can choose any one of them arbitrarily. - Define the dynamic mechanism $\mu^* : \Theta \to O$ as follows:

$$\mu^*(\theta) = o^*(b^*(a^*(\theta))),$$

where $o^*(b^*(a^*(\theta)))$ is the outcome that maximizes social welfare or revenue given the action profile $b^*(a^*(\theta))$. Note that this outcome may not be unique, but we can choose any one of them arbitrarily.

We claim that μ^* is the unique simple dynamic mechanism that maximizes social welfare or revenue. To prove this claim, we need to show two things:

- μ^* is simple with respect to σ, i.e., it induces a dynamic game in which every agent can play a best response using only the actions that it can foresee at each node according to σ. - μ^* maximizes social welfare or revenue, i.e., it assigns the outcome that maximizes the sum of agents' payoffs or transfers for each type profile, subject to individual rationality and incentive compatibility (if applicable).

We prove these two things using the same arguments as in Theorem 1 and Theorem 2. We omit the details for brevity.

This completes the proof of Theorem 4.

7.2 Gaming auctions, markets, and voting systems

I apply our framework to specific examples of dynamic mechanisms in video games, such as auctions, markets, and voting systems. We show how these mechanisms can design incentives or rules that align with the desired objectives of the mechanism, such as maximizing revenue, efficiency, or social welfare. We also show how these mechanisms can incorporate simplicity standards that depend on the cognitive abilities of the agents, such as memory, attention, anticipation, and learning.

Example 3. An auction is a dynamic mechanism that allows agents to bid for one or more items or services, and determines the allocation and payment of the items or services based on the bids. An auction can be used to allocate scarce or valuable resources among agents, or to elicit truthful valuations of the items or services from the agents. An auction can be simple with respect to a simplicity standard that depends on the foresight or learning of the agents, i.e., how many bids or outcomes they can foresee or learn from at each node.

For example, consider a video game environment with transfers where there are two agents, a human player and an AI agent, who participate in an auction for a rare item in the game. The type of each agent is their valuation of the item, which is their private information. The outcome of the auction is the allocation and payment of the item. The payoff of each agent is their utility from obtaining the item minus their payment for the item.

One possible auction is the following:

The auction is a second-price sealed-bid auction, where each agent submits a single bid for the item without knowing the bid of the other agent, and the highest bidder wins the item and pays the second-highest bid.

The auction is simple with respect to the 0-foresight simplicity standard, which means that agents can only foresee their own bids at each node. To see this, note that the best response of each agent at each node is to bid their true valuation of the item, using only their own type as a reference. Therefore, this auction induces a dynamic game in which every agent can play a best response using only the actions that they can foresee at each node according to the 0-foresight simplicity standard.

This auction can design incentives or rules that align with the desired objectives of the mechanism in the game. For example, it can elicit truthful valuations of the item from the agents, as bidding their true valuation is a dominant strategy for both agents. It can also maximize revenue for the seller of the item, as the second-highest bid is equal to the highest possible payment that does not deter any bidder from participating. It can also ensure efficiency for the allocation of the item, as the item goes to the agent who values it the most.

Example 4. A market is a dynamic mechanism that allows agents to buy or sell one or more goods or services, and determines the prices and quantities of the goods or services based on supply and demand. A market can be used to facilitate trade and exchange among agents, or to allocate resources efficiently among agents. A market can be simple with respect to a simplicity standard that depends on the attention or learning of the agents, i.e., how many prices or quantities they can pay attention to or learn from at each node.

For example, consider a video game environment with transfers where there are two agents, a human player and an AI agent, who trade one or more goods or services in a market in the game. The type of each agent is their endowment and preference for each good or service, which is their private information. The outcome of the market is the prices and quantities of the goods or services traded by the agents. The payoff of each agent is their utility from consuming the goods or services minus their payment for the goods or services.

One possible market is the following:

- The market is a double auction, where each agent can submit a bid (offer to buy) or an ask (offer to sell) for each good or service, and the market clears by matching the bids and asks according to a pricing rule.

- The market is simple with respect to the 1-attention simplicity standard, which means that agents can only pay attention to one price or quantity at each node. To see this, note that the best response of each agent at each node is to submit a bid or an ask that maximizes their expected payoff given the price or quantity they observe at that node, using only their own type as a reference. Therefore, this market induces a dynamic game in which every agent can play a best response using only the actions that they can foresee at each node according to the 1-attention simplicity standard.

This market can design incentives or rules that align with the desired objectives of the mechanism in the game. For example, it can facilitate trade and exchange among agents, as agents can buy or sell goods or services according to their endowments and preferences. It can also allocate resources efficiently among agents, as the market clears at a price or quantity that equates supply and demand for each good or service. It can also elicit truthful valuations of the goods or services from the agents, as bidding or asking their true valuation is a dominant strategy for both agents.

After providing examples of simple dynamic mechanisms in video games, such as auctions and markets, we study another example of a dynamic mechanism in video games, a voting system. A voting system is a dynamic mechanism that allows agents to express their preferences or opinions on one or more issues or candidates, and determines the outcome or decision based on the votes. A voting system can be used to aggregate information or preferences among agents, or to implement collective choices or actions among agents. A voting system can be simple with respect to a simplicity standard that depends on the memory or learning of the agents, i.e., how many votes or outcomes they can remember or learn from at each node.

For example, consider a video game environment without transfers where there are two agents, a human player and an AI agent, who participate in a voting system for a policy or action in the game. The type of each agent is their preference or opinion on the policy or action, which is their private information. The outcome of the voting system is the policy or action that is chosen by the majority of the votes. The payoff of each agent is their utility from the policy or action that is chosen.

One possible voting system is the following:

The voting system is a plurality voting system, where each agent can cast one vote for one of the available options for the policy or action, and the option with the most votes wins.

The voting system is simple with respect to the 1-memory simplicity standard, which means that agents can only remember the last vote they cast at each node. To see this, note that the best response of each agent at each node is to vote for their most preferred option, using only their own type as a reference. Therefore, this voting system induces a dynamic game in which every agent can play a best response using only the actions that they can foresee at each node according to the 1-memory simplicity standard.

This voting system can design incentives or rules that align with the desired objectives of the mechanism in the game. For example, it can aggregate information or preferences among agents, as agents can express their opinions on the policy or action through their votes. It can also implement collective choices or actions among agents, as the policy or action that is chosen reflects the majority opinion of the agents. It can also elicit truthful preferences or opinions from the agents, as voting for their most preferred option is a dominant strategy for both agents.

In this section, we studied the implications of simplicity for the design of dynamic mechanisms in video games, such as auctions, markets, and voting systems. We showed how these mechanisms can design incentives or rules that align with the desired objectives of the mechanism, such as maximizing revenue, efficiency, or social welfare. We also showed how these mechanisms can incorporate simplicity standards that depend on the cognitive abilities of the agents, such as memory, attention, anticipation, and learning.

8 Examples from Popular Sports Video Games: FIFA Soccer, NBA 2k Basketball, eFootball, and Mario Kart 8 Games

In this section, we use more examples from popular video games to illustrate our results and concepts from the main text. We show how simplicity standards can be applied to different types of games and mechanisms, and how they can affect the gameplay and outcomes for the agents.

8.1 Simulation Soccer Sports Games

In this subsection, we show how the method is robust to describing the genre of video games that simulate association football, where players can control virtual players or teams and compete in various modes and tournaments. The games tend to have several modes, but one of the most popular ones is often one where players can create and manage their own custom teams by collecting and trading player cards.

Such soccer games can be modeled as a video game environment with transfers, where the agents are the players, the types are their skills and strategies, the outcomes are their wins or losses, and the payoffs are their scores or ranks. A mechanism here games is a function that determines the outcome for each player based on their type and actions.

One possible mechanism in the soccer game genre is the following:

The mechanism is an auction market, where players can buy or sell player cards by submitting bids or asks, and the market clears by matching the bids and asks according to a pricing rule.

The mechanism is simple with respect to the 1-learning simplicity standard, which means that players can only learn from one price or quantity at each node. To see this, note that the best response of each player at each node is to submit a bid or ask that maximizes their expected payoff given their own type and the price or quantity they observe at that node, using only their own type as a reference. Therefore, this mechanism induces a dynamic game in which every player can play a best response using only the actions that they can foresee at each node according to the 1-learning simplicity standard.

This mechanism can design incentives or rules that align with the desired objectives of the mechanism in such games. For example, it can facilitate trade and exchange among players, as players can buy or sell player cards according to their preferences and budgets. It can also allocate resources efficiently among players, as the market clears at a price or quantity that equates supply and demand for each player card. It can also elicit truthful valuations of the player cards from the players, as bidding or asking their true valuation is a dominant strategy for both buyers and sellers.

8.2 Basketball Sports Games

Here, we focus on applying the framework to video games that simulate basketball, where players can control virtual players or teams and compete in various modes and tournaments. The games have several modes, but the most popular one is a mode where players can create and develop their own custom players by completing tasks and earning rewards.

Here, the basketball game can be modeled as a video game environment without transfers, where the agents are the players, the types are their preferences and goals, the outcomes are their achievements and actions, and the payoffs are their satisfaction or enjoyment. A mechanism here is a function that determines the outcome for each player based on their type and actions.

One possible mechanism that is relevant here is the following:

The mechanism is a scoring system, where players can earn points for completing tasks or performing actions in the game, such as scoring baskets, making assists, winning matches, etc.

The mechanism is simple with respect to the k-memory simplicity standard, where k is a positive integer that represents the number of tasks or actions that a player can remember at each node. To see this, note that the best response of each player at each node is to choose a task or action that maximizes their expected points given their own type and the tasks or actions they remember at that node, using only their own type as a reference. Therefore, this mechanism induces an extensive-form game in which every player can play a best response using only the actions that they can foresee at each node according to the k-memory simplicity standard.

This mechanism can balance the gameplay and ensure fairness among players. For example, it can reward players for their performance and achievements in the game, as completing tasks or performing actions gives positive points regardless of the difficulty level or the opponent. It can also create a trade-off between quantity and quality for the players, as completing more tasks or performing more actions gives more points but also increases the risk of making mistakes or losing focus. It can also prevent one player from dominating the other, as completing tasks or performing actions gives diminishing returns of points as the game progresses.

8.3 Cart Racing Games

Cart racing games are a racing video game subgenre where players can compete in various modes and tracks. The game has several modes, but the most popular one is one where players can race against computer-controlled or online opponents in a certain number of different cups, each consisting of a number of race tracks.

Such games can be modeled as a video game environment without transfers, where the agents are the players, the types are their skills and strategies, the outcomes are their positions and times, and the payoffs are their scores or ranks. A mechanism in here is a function that determines the outcome for each player based on their type and actions.

In the context of the paper, one possible mechanism is the following:

The mechanism is a difficulty level, where the game adjusts the speed and intelligence of the computer-controlled opponents according to the chosen level by the player.

From the framework, this mechanism is simple with respect to the k-anticipation simplicity standard, where k is a positive integer that represents the number of actions or outcomes that a player can anticipate at each node. To see this, note that the best response of each player at each node is to choose an action that maximizes their expected position or time given their own type and the actions or outcomes they anticipate at that node, using only their own type as a reference.

Therefore, this mechanism induces an extensive-form game in which every player can play a best response using only the actions that they can foresee at each node according to the k-anticipation simplicity standard.

This mechanism can adapt or customize the game to different types of players. For example, it can adjust the challenge and complexity of the game according to the skill and interest of the players, as higher levels increase the speed and intelligence of the opponents. It can also vary the diversity and unpredictability of the game according to the preference and goal of the players, as higher levels introduce more obstacles and items in the tracks. It can also challenge or assist each player by making the game harder or easier depending on their performance and progress.

8.4 Management Soccer Sports Games

Here we focus on the sub-genre of soccer video games that emphasize team management and strategy over direct control of players. In many cases, players act as the team manager, making decisions about training, transfers, and tactics. Such games also simulate association football, where players can control virtual players or teams and compete in various modes and tournaments. The games may have several modes, but the most popular ones may be where players can manage and develop their own custom teams by signing players, hiring staff, and setting tactics.

Such games can be modeled as a video game environment with transfers, where the agents are the players, the types are their skills and strategies, the outcomes are their wins or losses, and the payoffs are their scores or ranks. A mechanism of such games is a function that determines the outcome for each player based on their type and actions.

One possible mechanism is the following:

The mechanism is a transfer market, where players can buy or sell players by negotiating contracts, fees, and clauses with other teams or agents.

The mechanism is simple with respect to the 1-learning simplicity standard, which means that players can only learn from one contract, fee, or clause at each node. To see this, note that the best response of each player at each node is to negotiate a contract, fee, or clause that maximizes their expected payoff given their own type and the contract, fee, or clause they observe at that node, using only their own type as a reference. Therefore, this mechanism induces a dynamic game in which every player can play a best response using only the actions that they can foresee at each node according to the 1-learning simplicity standard.

This mechanism can design incentives or rules that align with the desired objectives of the mechanism in such games. For example, it can facilitate trade and exchange among players, as players can buy or sell players according to their needs and budgets. It can also allocate resources efficiently among players, as the market clears at a contract, fee, or clause that reflects the supply and demand for each player. It can also elicit truthful valuations of the players from the players, as negotiating their true valuation is a dominant strategy for both buyers and sellers.

In this section, we used more examples from popular video games to illustrate our results and concepts from the main text. We showed how simplicity standards can be applied to different types of games and mechanisms, and how they can affect the gameplay and outcomes for the agents.

9 Conclusion and Future Directions

In this paper, we introduced a general framework of simplicity in games and mechanism design, where agents have limited foresight abilities and can only foresee a subset of actions or outcomes at each node of an extensive-form game. We defined a class of simplicity standards that vary the foresight abilities of agents, and characterized simple mechanisms that maximize social welfare or revenue in social choice environments with and without transfers, subject to individual rationality and incentive compatibility. We also extended our framework to video game environments with and without transfers, and provided examples of simple mechanisms such as scoring systems, difficulty levels, auctions, markets, and voting systems. We showed how these mechanisms can balance the gameplay and ensure fairness among agents in video games, as well as design incentives or rules that align with the desired objectives of the mechanism.

Our paper opens up several directions for future research on simplicity in games and mechanism design. Some of the possible questions and challenges we are now in a position to answer are questions like: How to measure or compare the simplicity or complexity of different mechanisms or games, and how to design mechanisms or games that minimize complexity while achieving certain objectives or constraints? How to model or incorporate other cognitive factors that may affect the foresight abilities of agents, such as uncertainty, risk aversion, bounded rationality, or learning behavior? How to design mechanisms or games that are robust or adaptable to changes in the foresight abilities of agents over time or across contexts, such as learning curves, feedback loops, or dynamic environments? How to design mechanisms or games that are fair or equitable for agents with different foresight abilities, and how to account for the trade-offs between simplicity, efficiency, and fairness? How to design mechanisms or games that are transparent or explainable for agents with different foresight abilities, and how to communicate or visualize the actions or outcomes that agents can foresee or not foresee at each node? How to test or evaluate the performance or behavior of simple mechanisms or games in real-world settings or applications, such as online platforms, social networks, or educational games?

We hope that our paper will inspire more research on this topic and provide useful insights for game developers in industry and mechanism designers in general. We believe that simplicity is an important and relevant concept for understanding and improving human-computer interaction and social welfare in complex and dynamic environments.

10 References

Alves, V., Roque, L., and Cardoso, A. (2014). Measuring complexity in board games. In Proceedings of the 2014 IEEE Conference on Computational Intelligence and Games (pp. 1-8). IEEE.

Ashlagi, Itai, and Gonczarowski, Yannai A. (2018). "Stable matching mechanisms are not obviously strategy-proof." Journal of Economic Theory, 177, 405-425.

Arribillaga, R. Pablo, Massó, Jordi and Neme, Alejandro. (2020). "On obvious strategy-proofness and single-peakedness." Journal of Economic Theory, 186, 104992.

Björk, S., and Holopainen, J. (2005). Patterns in game design. Charles River Media.

Castronova, E. (2005). Synthetic worlds: The business and culture of online games. University of Chicago Press.

Dell'Acqua, F., Kogut, B., and Perkowski, P. (2022). Super Mario Meets AI: Experimental Effects of Automation and Skills on Team Performance and Coordination. Review of Economics and Statistics, 1-47.

Hamari, J., Koivisto, J., and Sarsa, H. (2014). Does gamification work? A literature review of empirical studies on gamification. In Proceedings of the 2014 47th Hawaii International Conference on System Sciences (pp. 3025-3034). IEEE.

Hunicke, R., LeBlanc, M., and Zubek, R. (2004). MDA: A formal approach to game design and game research. In Proceedings of the AAAI Workshop on Challenges in Game AI (Vol. 4, No. 1, p. 1722).

Li, S. (2017). Obviously strategy-proof mechanisms. American Economic Review, (11), 3257-3287.

Nelson, M. J., Gaudl, S., Colton, S., and Browne, C. (2017). Towards a computational reading of emergence in experimental game design. In Proceedings of the 2017 IEEE Conference on Computational Intelligence and Games (pp. 68-75). IEEE.

Pycia, M., and Troyan, P. (2023). Simplicity in games and mechanism design. Journal of Economic Theory, 188, 105111.

Salen, K., and Zimmerman, E. (2004). Rules of play: Game design fundamentals. MIT press.

Zagal, J. P., Mateas, M., Fernández-Vara, C., Hochhalter, B., and Lichti, N. (2013). Towards an ontological language for game analysis. In Proceedings of DiGRA 2013: DeFragging Game Studies.

Pycia, M., and Troyan, P. (2023). A theory of simplicity in games and mechanism design. *Econometrica*, 91(4), 1495-1526.

Pycia, M., and Troyan, P. (2023). A theory of simplicity in games and mechanism design.

Pycia, M., and Troyan, P. (2023). Strategy-proof, efficient, and fair allocation: Beyond obvious strategy-proofness.