Game Implementation: What Are the Obstructions?

Jiehua Chen, Seyedeh Negar Layegh Khavidaki,
Sebastian Vincent Haydn, Sofia Simola, Manuel Sorge
TU Wien, Austria
{jiehua.chen, seyedehngar.khavidaki, sofia.simola, manuel.sorge}@tuwien.ac.at,
e51832021@student.tuwien.ac.at

Abstract

In many applications, we want to influence the decisions of independent agents by designing incentives for their actions. We revisit a fundamental problem in this area, called GAME IMPLEMENTATION: Given a game in standard form and a set of desired strategies, can we design a set of payment promises such that if the players take the payment promises into account, then all undominated strategies are desired? Furthermore, we aim to minimize the cost, that is, the worst-case amount of payments.

We study the tractability of computing such payment promises and determine more closely what obstructions we may have to overcome in doing so. We show that GAME IMPLEMENTATION is NP-hard even for two players, solving in particular a long-standing open question and suggesting more restrictions are necessary to obtain tractability results. We thus study the regime in which players have only a small constant number of strategies and obtain the following. First, this case remains NP-hard even if each player's utility depends only on three others. Second, we repair a flawed efficient algorithm for the case of both small number of strategies and small number of players. Among further results, we characterize sets of desired strategies that can be implemented at zero cost as a generalization of Nash equilibria.

1 Introduction

Nudge theory (Thaler and Sunstein 2008), gamification (Hamari 2019), and the design of blockchain systems (Buterin et al. 2020) are just a few areas in which we apply incentives in order to coax agents towards behaving in a desirable way. In these general settings, agents select strategies on their own volition, but we may add incentives (or incur penalties) that increase (resp. decrease) the salience or utility of particular strategies in situations of our choice. The goal is to implement a desired set of strategies or strategy profiles, that is, to ensure that undesired strategies entail smaller utility than desired ones.

With the advent of blockchain systems, we feel that this topic has gained renewed relevance. First, the design of a blockchain system itself, such as Bitcoin or Ethereum, involves the design of a protocol that rewards intended behavior (e.g., validating transactions by mining blocks for block rewards in Bitcoin) or penalizes unintended behavior (e.g., by slashing the stake of validators that deviate from a consensus in the recent upgrade of Ethereum). The latter is a form of enforcing the existence of a Schelling point via incentives. Second, there are now base-layer systems like Ethereum in place that allow world-wide consistent general-purpose computations and thus the straightforward creation of new moneys (called tokens) that can be made to behave in new ways: generated, burned, exchanged, locked, etc. Thus an immense design space for incentive-based protocols was opened up and we witness its continued exploration. There are for example stablecoins, which are tokens that use incentive-based mechanisms to try and reflect the value of some underlying security (Maker DAI, Terra USD, FRAX Shares, and many more)[1]. Another direction are incentive-based consensus mechanisms for adjudication, moderation, and transferring real-world information onto blockchains (such as Kleros, UMA, Chainlink oracles, and again many more)[2].

In all of the above design problems, there are independent actors that we want to incentivize to behave in a certain desired way. A fundamental underlying problem herein is GAME IMPLEMENTATION (Monderer and Tennenholtz 2004), stated as follows: We are given a game in standard form (a set of players, strategies for each player, and their utility) and for each player a set of desired strategies. We want to specify a set of payment promises that define additional utilities that we give to players for playing certain strategies. These payment promises shall *implement* our desired sets of strategies, that is, when taking the payments into account, no player wants to play an undesired strategy. In technical terms, each strategy that is not dominated by any other strategy is desired (see Section 2 for the formal definitions).[3] Furthermore, we want to minimize the *cost*

[1]See https://makerdao.com/en/whitepaper/, https://terra.money/Terra_White_paper.pdf, and https://docs.frax.finance/overview.

[2]See https://kleros.gitbook.io/docs/, https://docs.umaproject.org/, and https://chain.link/whitepaper.

[3]We focus here only on pure strategies. Furthermore, GAME IMPLEMENTATION implements a set of strategy profiles rather than a set of strategies for each player. Implementing sets of strategies corresponds to implementing so-called rectangular strategy profiles, see the formal definitions in Section 2.

of the implementation, that is, the amount paid in the worst case. More precisely, we want to minimize, over all strategy profiles that consist of undominated strategies, the sum of payment promises to all players. In this work, we explore the question "How difficult is it to implement a desired set of strategies?"

Contribution. We obtain the following results. We first show that GAME IMPLEMENTATION is NP-hard, even if there are only two players and even if our budget for the cost is 0 (Theorem 3.1). This strengthens two results by Deng, Tang, and Zheng (2016) who showed that GAME IMPLEMENTATION is NP-hard for six players, and that GAME IMPLEMENTATION is NP-hard for two players with mixed strategies, both with positive budgets.[4] We note that hardness for mixed-strategies or positive budgets is less surprising because there are a priori more possibilities for encoding combinatorial structure into the solutions. Instead, our reduction shows that the difficulty lies already and mainly in selecting, for each undesired strategy x, a desired strategy that dominates x.

We then study a variant of GAME IMPLEMENTATION that was supposedly more tractable (Monderer and Tennenholtz 2004), called EXACT GAME IMPLEMENTATION: In addition to requiring undominated strategies to be desired, no desired strategy may be dominated by another strategy. We show that also this a priori simpler-looking problem is NP-hard even for two players (Theorem 5.1); this answers an open question by Eidenbenz et al. (2011). Indeed Monderer and Tennenholtz (2004) gave a polynomial-time algorithm for EXACT GAME IMPLEMENTATION which was shown to produce suboptimal results by Eidenbenz et al. (2011).

The above hardness results do not apply in scenarios in which players have only a small constant number of strategies to choose from. We hence consider this regime next. If both the number of players and the number of strategies are small constants, then the only part of the input that may be of unbounded size are the quantities specified in the utility functions. Eidenbenz et al. (2011) showed that in this case EXACT GAME IMPLEMENTATION can be solved efficiently; however, as we observe here there is a flaw in the algorithm. We simplify the algorithm and repair the flaw for a large though not universal class of problem instances, providing the first nontrivial algorithm for implementing strategies that is formally proven to be correct (Theorem 6.2).

As we increase the number of players, the size of the input (the number of utility values we have to specify) scales exponentially in the number of players (and strategies). A common way to deal with this explosion is to instead consider the relevant special case of graphical games (Kearns, Littman, and Singh 2001), where the players are situated in a graph and the utility of a player depends only on its neighbors. We hence study this case next, that is, GAME IMPLEMENTATION on graphical games with small constant number of strategies per player. We show that even the case where each player's utility depends only on three others and each player has only two strategies remains NP-hard (Theorem 4.1). As the case where each player has only one strategy is trivial, a promising future direction is to consider the case where each player depends only on two others or tree-structured games.

Finally, before discovering our NP-hardness reduction for GAME IMPLEMENTATION we believed that zero-cost implementation could be solved efficiently. As a tool towards this we characterized strategy sets that can be implemented at cost 0 as a form of generalized Nash equilibria. We believe that this characterization is of independent interest, in particular because it generalizes the result of Monderer and Tennenholtz (2004) that states that Nash equilibria can be implemented at cost 0. Moreover, it captures a fundamental property of self-enforcing sets of strategies, such as morality, which we are not aware of having been formally defined before.

Further related work. Implementation theory (Maskin 1999; Maskin and Sjöström 2002) generally studies the implementation of social-choice rules with incentives and it is impossible to give an overview over the large body of work here. Conitzer and Sandholm (2014) studied the complexity of implementing social-choice rules. The main difference to GAME IMPLEMENTATION is that the payment promises to the players that we may choose from are restricted and given in the input. Such restrictions give significantly more leeway for designing hardness reductions. Brill, Freeman, and Conitzer (2015) considered a problem related to GAME IMPLEMENTATION in which some of the utility values are missing and we are to complete the missing values, possibly with negative ones. The goal is to ensure that strategies participating in some Nash equilibrium are desired. In GAME IMPLEMENTATION we have much more freedom in designing our solutions. Wooldridge et al. (2013) studied implementation questions for Boolean games, that is, where the strategies of the players correspond to a selection of truth values of some variables intrinsic to the game. They aimed at implementing Boolean formulas on the variables in some or all Nash equilibria. Because of the relation to Boolean satisfiability, implementation for Boolean games is situated higher in the polynomial hierarchy. Zero-cost implementation has been studied for routing games by Moscibroda and Schmid (2009). They gave bounds on the difference between anarchistic equilibria and those achievable by zero-cost implementation. Finally, Letchford and Conitzer (2010); Deng and Conitzer (2017, 2018) considered the complexity of committing to certain behaviors as a way for one player to change the outcome of a game to his favor.

2 Preliminaries

(Full) proofs for results marked by $\star$ are deferred to the full version (Chen et al. 2022). Throughout, for $t \in \mathbb{N}$ we use $[t]$ to denote the set $\{1, 2, \ldots, t\}$.

A *game* G is a tuple $(N, \mathcal{X}, \mathcal{U})$ where N is the set of *players*; usually $N = [n]$. We specify for each player $i \in N$ a set X_i of *strategies* available to i. Then the set $\mathcal{X}$ equals $X_1 \times X_2 \times \ldots \times X_n$. We call elements of $\mathcal{X}$ *strategy profiles*. Finally, $\mathcal{U} = \{U_1, U_2, \ldots, U_n\}$, where each U_i is a

[4]Monderer and Tennenholtz (2004) claimed NP-hardness of GAME IMPLEMENTATION, but the proof was erroneous (Eidenbenz et al. 2011).

function $\mathcal{X} \to \mathbb{R}$, called *utility function* for player i.

As a notational shorthand, for any $i \in N$ we use $\mathcal{X}_{-i}$ to denote $X_1 \times X_2 \times \ldots \times X_{i-1} \times X_{i+1} \times \ldots \times X_n$.

Let $\mathbf{x} = \{x_1, \ldots, x_n\}$ be a strategy profile. Then we use $\mathbf{x}_{-i}$ to refer to $\mathbf{x}$ without its ith element, i.e., $\mathbf{x}_{-i} = (x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$. Similarly, we use $\mathbf{x}_i$ to refer to its ith element, i.e., $\mathbf{x}_i = x_i$.

We sometimes write the value of a utility function U_i such that the strategy played by player i comes first in the argument of U_i and the remaining strategies second. For $\mathbf{x}$ above and player $i \in N$, we could write $U(x_i, \mathbf{x}_{-i})$. However, if there are only two players, then the first argument x of $U_i(x, y)$ always refers to the strategy of player 1 and the second argument y always refers to the strategy of player 2.

Let $x, y \in X_i$ be two strategies of player i. We say that x *dominates* y if for each $\mathbf{x}_{-i} \in \mathcal{X}_{-i}$ we have $U_i(x, \mathbf{x}_{-i}) \geq U_i(y, \mathbf{x}_{-i})$ and there exists $\mathbf{x}_{-i} \in \mathcal{X}_{-i}$ such that $U_i(x, \mathbf{x}_{-i}) > U_i(y, \mathbf{x}_{-i})$.[5] We say that $x \in X_i$ is *undominated* if no other strategy of player i dominates x. For each player $i \in N$ we denote by $X_i^\star$ the set of undominated strategies in X_i. For a game G, by $\mathcal{X}_G^\star$ we denote the set of strategy profiles that consist entirely of undominated strategies, that is, $\mathcal{X}_G^\star = X_1^\star \times X_2^\star \times \ldots \times X_n^\star$. We omit the index G if it is clear from the context.

Let $G = (N, \mathcal{X}, \mathcal{U})$ be a game. We now define the modified game obtained from G after payments are promised to the players. A *payment promise* to player i in G is a function $\mathcal{X} \to \mathbb{R}$, usually denoted by V_i. A *payment promise* for game G is the set of payment promises of the players: $\mathcal{V} := \{V_1, V_2, \ldots, V_n\}$. The *modified game* $G[\mathcal{V}]$ obtained from G with a payment promise $\mathcal{V}$ is the game $(N, \mathcal{X}, [\mathcal{U} + \mathcal{V}])$, wherein $[\mathcal{U} + \mathcal{V}] := \{[U_i + V_i] \mid i \in N\}$ and for each $i \in N$ the function $[U_i + V_i]$ is defined as $[U_i + V_i](\mathbf{x}) := U_i(\mathbf{x}) + V_i(\mathbf{x})$ for all $\mathbf{x} \in \mathcal{X}$. The *cost*, denoted as $\operatorname{cost}(\mathcal{V})$, of a payment promise $\mathcal{V}$ is $\max_{\mathbf{x} \in \mathcal{X}^\star_{G[\mathcal{V}]}} \sum_{i \in N} V_i(\mathbf{x})$.

We consider the following decision problem.

GAME IMPLEMENTATION
Instance: A game $G = (N, \mathcal{X}, \mathcal{U})$, a set of strategy profiles $\mathcal{O} \subseteq \mathcal{X}$, and a real budget $\delta \in \mathbb{R}_{\geq 0}$.
Question: Is there a payment promise $\mathcal{V}$ such that $\operatorname{cost}(\mathcal{V}) \leq \delta$ and $\mathcal{X}^\star_{G[\mathcal{V}]} \subseteq \mathcal{O}$?

A payment promise $\mathcal{V}$ as above *implements* $\mathcal{O}$. The following is a variant of GAME IMPLEMENTATION where we want a given set of strategy profiles to be undominated and say that a payment promise $\mathcal{V}$ as below *implements* $\mathcal{O}$ *exactly*.

EXACT GAME IMPLEMENTATION
Instance: A game $G = (N, \mathcal{X}, \mathcal{U})$, a set of strategy profiles $\mathcal{O} \subseteq \mathcal{X}$, and a real budget $\delta \in \mathbb{R}_{\geq 0}$.
Question: Is there a payment promise $\mathcal{V}$ such that $\operatorname{cost}(\mathcal{V}) \leq \delta$ and $\mathcal{X}^\star_{G[\mathcal{V}]} = \mathcal{O}$?

[5]This notion of domination is commonly referred to as weak domination. Alternative notions of domination are also studied, such as strict domination in which we require $U_i(x, \mathbf{x}_{-i}) > U_i(y, \mathbf{x}_{-i})$ for all $\mathbf{x}_{-i} \in \mathcal{X}_{-i}$. In keeping with the literature on implementation (Eidenbenz et al. 2011; Deng, Tang, and Zheng 2016) we focus on weak domination; the results are usually transferable.

We give an example in the full version. We mainly focus on the special case where the strategy profile sets $\mathcal{O}$ are rectangular. A strategy profile set $\mathcal{Y}$ for a game with player set N is *rectangular* if for each $i \in N$ there is $Y_i \subseteq X_i$ such that $\mathcal{Y} = Y_1 \times Y_2 \times \ldots \times Y_n$. For a given player $i \in N$, we use $\mathcal{Y}_{-i}$ to denote $Y_1 \times \cdots \times Y_{i-1} \times Y_{i+1} \times \ldots \times Y_n$. All our hardness results indeed hold even for rectangular strategy profile sets $\mathcal{O}$.

Graphical games. For a more succinct representation we also use the concept of a *graphical game*. This is a tuple (G, H), where $G = (N, \mathcal{X}, \mathcal{U})$ is a game and H an undirected graph with vertex set N and edge set E. Let us use $\mathrm{N}_H(i)$ to denote the neighborhood of a vertex $i \in N$, i.e., the set of all the vertices that are adjacent to i. For every $i \in N$, the utility function U_i and the potential payment promise V_i map from $X_{j_1} \times \cdots \times X_{j_k}$ to $\mathbb{R}$, where $j_1, \ldots, j_k$ is the canonical ordering of the vertices restricted to $\mathrm{N}_H(i) \cup \{i\}$. In other words, the utility of player i only depends on its own actions and the actions of its neighbors in H. The *degree* of (G, H) is the maximum vertex degree.

Properties of the domination relation. We now describe a few simple properties of the dominance relation, which are useful in our proofs.

Observation 2.1 ($\star$). *Domination is transitive.*

Observation 2.2 ($\star$). *Domination is asymmetric. In other words, if x dominates y, then y does not dominate x.*

Observation 2.3 ($\star$). *Every dominated strategy is dominated by some undominated strategy.*

This immediately implies the following observation:

Observation 2.4. *The set of undominated strategies is nonempty.*

3 GAME IMPLEMENTATION is NP-hard for Two Players and Zero Budget

In this section we prove that GAME IMPLEMENTATION is NP-hard even in the very restricted case where we have two players and the budget is zero.

Theorem 3.1. GAME IMPLEMENTATION *is NP-hard, even for two players and zero budget.*

We reduce from the following NP-hard problem (Schaefer 1978):

X3C
Instance: A set of $3\hat{n}$ elements $\mathcal{A}$ and a collection of $3\hat{n}$ sets $\mathcal{C}\}$ such that $C_j \subset \mathcal{A}$ and $|C_j| = 3$ for every $j \in \{0, \ldots, 3\hat{n} - 1\}$ and every element $a_i \in \mathcal{A}$ appears in exactly three sets, i.e., $|\{C \in \mathcal{C} \mid a_i \in C\}| = 3$.
Question: Is there an *exact cover* of $\mathcal{A}$, i.e., a subcollection $\mathcal{S} \subset \mathcal{C}$ s.t. $|\mathcal{S}| = \hat{n}$ and $\mathcal{A} = \bigcup_{C \in \mathcal{S}} C$?

Let $\mathcal{I} = (\mathcal{A} = \{a_0, \ldots, a_{3\hat{n}-1}\}, \mathcal{C} = \{C_0, \ldots, C_{3\hat{n}-1}\})$ be an instance of X3C, where $|\mathcal{A}| = 3\hat{n}$. We create an instance $\mathcal{I}'$ of GAME IMPLEMENTATION with two players p_1 and p_2. Let $X_1 = X_2 = \mathcal{A} \cup \{c_j^{a_i} \mid C_j \in \mathcal{C}, a_i \in C_j\}$ and $O_1 = O_2 = \{c_j^{a_i} \mid C_j \in \mathcal{C}, a_i \in C_j\}$. Throughout, we take

$i+1$ and $i-1$ modulo $3\hat{n}$. For each $i \in [3\hat{n}]$ and each set $C_j, C_p \in \mathcal{C}$ with $a_i \in C_j, a_{i-1} \in C_p$, we define the utilities as

$$U_1(a_i, c_j^{a_i}) = 2, \qquad U_1(a_i, c_j^{a_z}) = 1,$$
$$U_1(c_j^{a_i}, c_j^{a_i}) = 2, \qquad U_1(c_j^{a_i}, c_j^{a_z}) = 1 \; \forall\, a_z \in C_j \setminus \{a_i\},$$
$$U_2(c_p^{a_{i-1}}, a_i) = 1, \quad U_2(c_p^{a_{i-1}}, c_j^{a_i}) = 1.$$

The undefined utilities are 0 and the budget δ is also 0. The utilities of p_1 and p_2 are shown in Section 3.

Before continuing with the proof, let us explain some intuition. For each $a_i \in \mathcal{A}$ we denote the sets containing a_i as $C_{i,1}, C_{i,2}, C_{i,3}$. The utilities of value 2 for p_1 and the zero budget enforce that for every $a_i \in \mathcal{A}$ exactly one of the strategies $c_{i,1}^{a_i}, c_{i,2}^{a_i}, c_{i,3}^{a_i}$ can be undominated for p_2: To dominate a_i for p_1 with, say, $c_{i,2}^{a_i}$, we must promise a positive amount for playing $c_{i,2}^{a_i}$ whenever p_2 plays $c_{i,3}^{a_i}$ or $c_{i,1}^{a_i}$. Thus we must have that neither of those latter strategies is undominated for p_2 in order to stay within the budget.

The utilities of value 1 for p_1 enforce consistency, i.e., if $a_s \in \mathcal{A}$ is covered by C_r, then every $a_{i'} \in C_r$ must also be covered by C_r. If $c_r^{a_s}$ is undominated for p_2, then if we were to dominate $a_i \in C_r$ with $c_{i,2}^{a_i}$ where $C_{i,2} \neq C_r$, we would have to pay p_1 least 1 for the strategy $c_{i,2}^{a_i}$ when p_2 plays $c_r^{a_s}$. But since these are both undominated strategies, we would exceed the budget.

The utilities for p_2 enforce that we cover every element $a_i \in \mathcal{A}$. Player p_1 always tries to match the element p_2 is playing, whereas p_2 tries to be one ahead. This prevents the two players from picking some element $a_z \in \mathcal{A}$ and only playing the strategies related to that.

Formally, we claim that $\mathcal{I}$ is a positive instance of X3C if and only if $\mathcal{I}'$ is a positive instance of GAME IMPLEMENTATION.

For the forward direction, assume that $(\mathcal{A}, \mathcal{C})$ admits an exact cover $\mathcal{S}$. For each element $a_i \in \mathcal{A}$ and two distinct sets $C_j, C_p \in \mathcal{C}$ with $a_i \in C_j \cap C_p$, where $C_j \in \mathcal{S}$ and for each element $a_z \in C_j$, we define $V_1(c_j^{a_i}, c_p^{a_z}) = \infty$. For each element $a_i \in \mathcal{A}$ and two distinct sets $C_j, C_p \in \mathcal{C}$ with $a_i \in C_j \in \mathcal{S}$, while $a_{i-1} \in C_p$ but $C_p \notin \mathcal{S}$, we define $V_2(c_p^{a_{i-1}}, c_j^{a_i}) = \infty$. To show that this is a valid implementation, we will show that $\mathcal{X}^\star \subseteq \mathcal{O}$ and $\mathrm{cost}(\mathcal{V}) = 0$.

Claim 3.2 ($\star$)**.** *For each $C_j \in \mathcal{S}$, each $a_i \in C_j$ and each $C_\ell \in \mathcal{C} \setminus \{C_j\}$ with $a_i \in C_\ell$, we have that $c_j^{a_i}$ dominates both a_i and every $c_l^{a_i}$ for both p_1 and p_2.*

Proof of Claim 3.2. Observe that for p_1, strategy a_i dominates $c_j^{a_i}$ for every $C_j \in \mathcal{C}$ where $a_i \in C_j$ in G. Since $\mathcal{S}$ is an exact cover, no $C_l \in \mathcal{C} \setminus \{C_j\}$ such that $a_i \in C_l$ is in $\mathcal{S}$. Thus payment promise V_1 is 0 when p_1 plays $c_l^{a_i}$. Thus a_i also dominates $c_l^{a_i}$ in $G[\mathcal{V}]$. Because dominance is transitive (Observation 2.1), it is enough to show that $c_j^{a_i}$ dominates a_i.

To see that $c_j^{a_i}$ dominates a_i for p_1, we show that the utility for playing $c_j^{a_i}$ is always at least that of playing a_i.

Case 1: p_2 **plays** $c_j^{a_i}$**.**
Then, $[U_i + V_i](c_j^{a_i}, c_j^{a_i}) = 2 = [U_i + V_i](a_i, c_j^{a_i})$.

Case 2: p_2 **plays** $c_l^{a_i}$ **for some** $C_l \in \mathcal{C} \setminus \{C_j\}$ **s.t.** $a_i \in C_l$**.**
Then, since a_i is covered by C_j, we know that $C_l \notin \mathcal{S}$. Therefore, $[U_1 + V_1](c_j^{a_i}, c_l^{a_i}) = \infty > 2 = [U_1 + V_1](a_i, c_l^{a_i})$.

Case 3: p_2 **plays** $c_j^{a_z}$ **for some** $a_z \in C_j \setminus \{a_i\}$**.**
Then, $[U_1 + V_1](c_j^{a_i}, c_j^{a_z}) = 1 = [U_1 + V_1](a_i, c_j^{a_z})$.

Case 4: p_2 **plays** $c_l^{a_z}$ **for some** $a_z \in \mathcal{A} \setminus \{a_i\}, C_l \in \mathcal{C} \setminus \{C_j\}$ **where** $a_i, a_z \in C_l$**.**
Then, since element a_i is covered by C_j we know that $C_l \notin \mathcal{S}$. Therefore $[U_1 + V_1](c_j^{a_i}, c_l^{a_z}) = \infty > 1 = [U_1 + V_1](a_i, c_l^{a_z})$.

Case 5: p_2 **plays** $c_l^{a_z}$ **for some** $a_z \in \mathcal{A} \setminus \{a_i\}, C_l \in \mathcal{C} \setminus \{C_j\}$**, where** $a_i \notin C_l, a_z \in C_l$**.**
Then, $[U_1 + V_1](c_j^{a_i}, c_l^{a_z}) = 0 = [U_1 + V_1](a_i, c_l^{a_z})$.

Case 6: p_2 **plays** $a_z \in \mathcal{A}$ (a_i, a_z **not necessarily distinct).**
Then, $[U_1 + V_1](c_j^{a_i}, a_z) = 0 = [U_1 + V_1](a_i, a_z)$.

In all cases, the new utility of $c_j^{a_i}$ is higher than or equal to the utility of a_i, and in the Cases 2 and 4 the utility is strictly higher. Thus $c_j^{a_i}$ dominates a_i for p_1. The analogous observation for p_2 is proved in the full version. $\diamond$

Since the set of undominated strategies is non-empty (Observation 2.4), Claim 3.2 implies that for every $i \in [2]$, $X_i^\star \subseteq \{c_j^{a_i} \mid C_j \in \mathcal{S}, a_i \in C_j\}$ as all other strategies are dominated. Therefore $\mathcal{X}^\star \subseteq \{c_j^{a_i} \mid C_j \in \mathcal{S}, a_i \in C_j\} \times \{c_j^{a_i} \mid C_j \in \mathcal{S}, a_i \in C_j\} \subseteq \mathcal{O}$, as required. For every $j \in [2]$, we have that $V_j(s_1, s_2) > 0$ only when s_1 or s_2 is in $\{c_j^{a_i} \mid C_j \notin \mathcal{S}, a_i \in C_j\}$. Since none of these strategies is in $X_{j'}^\star$, we have that $\mathrm{cost}(\mathcal{V}) = \max_{\mathbf{x} \in \mathcal{X}^\star_{G[\mathcal{V}]}} \sum_{i \in [2]} V_i(\mathbf{x}) = 0$, as required. This concludes the forwards direction.

For the backward direction, assume that we have a payment promise $\mathcal{V}$ such that $\mathrm{cost}(\mathcal{V}) = 0$ and $\mathcal{X}^\star \subseteq \mathcal{O}$ in the modified game $G[\mathcal{V}]$.

Claim 3.3. *Let $C_j \in \mathcal{C}$, $a_i \in \mathcal{A}$. If $c_j^{a_i} \in X_2^\star$, then in $G[\mathcal{V}]$*

(i) $c_j^{a_i} \in X_1^\star$,
(ii) $c_j^{a_i}$ dominates a_i for p_1,
(iii) $c_l^{a_i} \notin X_2^\star$ where $C_l \in \mathcal{C} \setminus \{C_j\}$ and $a_i \in C_l$.

Proof of Claim 3.3. We first show (i). Since $\mathcal{V}$ implements $\mathcal{O}$, by Observation 2.3 there is an undominated strategy that dominates a_i for p_1. For a strategy $s \in X_1^\star$ to dominate a_i for p_1 we need that $V_1(s, c_j^{a_i}) \geq U_1(a_i, c_j^{a_i}) - U_1(s, c_j^{a_i}) = 2 - U_1(s, c_j^{a_i})$. Because s is undominated, $(s, c_j^{a_i}) \in \mathcal{X}^\star_{G[\mathcal{V}]}$. By the definition of cost, we have that $0 \geq \mathrm{cost}(\mathcal{V}) = \max_{\mathbf{x} \in \mathcal{X}^\star_{G[\mathcal{V}]}} \sum_{i \in [2]} V_i(\mathbf{x}) \geq V_1(s, c_j^{a_i})$. By combining these, we obtain that $U_1(s, c_j^{a_i}) \geq 2$. The only strategy for p_1 that satisfies this condition is $c_j^{a_i}$. Since $c_j^{a_i}$ dominates a_i, (ii) follows directly.

To prove (iii), assume that both $c_j^{a_i}$ and $c_l^{a_i}$ are undominated for p_2. By (i) and (ii), strategy $c_j^{a_i}$ dominates a_i for p_1 and $c_j^{a_i}$ is undominated. Thus $[U_1 + V_1](c_j^{a_i}, c_l^{a_i}) \geq U_1(a_i, c_l^{a_i}) = 2$. From $U_1(c_j^{a_i}, c_l^{a_i}) = 0$ it follows that $V_1(c_j^{a_i}, c_l^{a_i}) \geq 2$. But since $c_j^{a_i}$ is undominated for p_1 and $c_l^{a_i}$ for p_2, $(c_j^{a_i}, c_l^{a_i}) \in \mathcal{X}^\star$ and thus $cost(\mathcal{V}) =$

Player p_1 \ Player p_2	$c^{a_0}_{0,1}$	$c^{a_0}_{0,2}$	$c^{a_0}_{0,3}$	$\dots$	$c^{a_i}_{i,1}$	$c^{a_i}_{i,2}$	$c^{a_i}_{i,3}$	$\dots$	$c^{a_{i'}}_{i',2}$	$\dots$	$c^{a_{3\hat{n}-1}}_{3\hat{n}-1,3}$
$c^{a_0}_{0,1}$	2			$\dots$				$\dots$		$\dots$	
$c^{a_0}_{0,2}$		2		$\dots$				$\dots$		$\dots$	
$c^{a_0}_{0,3}$			2	$\dots$				$\dots$		$\dots$	
$\vdots$	$\vdots$	$\vdots$	$\vdots$		$\vdots$	$\vdots$	$\vdots$		$\vdots$		$\vdots$
$c^{a_i}_{i,1}$				$\dots$	2			$\dots$	1	$\dots$	
$c^{a_i}_{i,2}$				$\dots$		2		$\dots$		$\dots$	
$c^{a_i}_{i,3}$				$\dots$			2	$\dots$		$\dots$	
$\vdots$	$\vdots$	$\vdots$	$\vdots$		$\vdots$	$\vdots$	$\vdots$		$\vdots$		$\vdots$
$c^{a_{i'}}_{i',2}$				$\dots$	1			$\dots$	2	$\dots$	
$\vdots$	$\vdots$	$\vdots$	$\vdots$		$\vdots$	$\vdots$	$\vdots$		$\vdots$		$\vdots$
$c^{a_{3\hat{n}-1}}_{3\hat{n}-1,3}$				$\dots$				$\dots$		$\dots$	2
a_0	2	2	2	$\dots$				$\dots$		$\dots$	
$\vdots$	$\vdots$	$\vdots$	$\vdots$		$\vdots$	$\vdots$	$\vdots$		$\vdots$		$\vdots$
a_i				$\dots$	2	2	2	$\dots$	1	$\dots$	
$\vdots$	$\vdots$	$\vdots$	$\vdots$		$\vdots$	$\vdots$	$\vdots$		$\vdots$		$\vdots$
$a_{3\hat{n}-1}$				$\dots$				$\dots$		$\dots$	2

Player p_2 \ Player p_1	$c^{a_{i-1}}_{i-1,1}$	$c^{a_{i-1}}_{i-1,2}$	$c^{a_{i-1}}_{i-1,3}$	$c^{a_i}_{i,1}$	$c^{a_i}_{i,2}$	$c^{a_i}_{i,3}$
$c^{a_i}_{i,1}$	1	1	1			
$c^{a_i}_{i,2}$	1	1	1			
$c^{a_i}_{i,3}$	1	1	1			
$c^{a_{i+1}}_{i+1,1}$				1	1	1
$c^{a_{i+1}}_{i+1,2}$				1	1	1
$c^{a_{i+1}}_{i+1,3}$				1	1	1
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
a_i	1	1	1			
a_{i+1}				1	1	1

Table 1: Left: The partial utility matrix of p_1 from the proof of Theorem 3.1. For each element $a_\ell \in \mathcal{A}$, let $C_{\ell,1}, C_{\ell,2}, C_{\ell,3}$ denote the sets containing it. For the sake of example, we assume that $a_i \in C_t$ such that $C_{i',2} = C_{i,1} = C_t$. Columns corresponding to strategies $a_i, \in [3\hat{n}]$ for p_2 are omitted: their values are 0. Right: The utility matrix of p_2 from the proof of Theorem 3.1. Columns corresponding to strategies $a_i, \in [3\hat{n}]$ for p_1 are omitted: their values are 0.

$\max_{\mathbf{x}\in\mathcal{X}^\star} \sum_{i\in[2]} V_i(\mathbf{x}) \geq V_1(c^{a_i}_j, c^{a_i}_l) = 2$ which a contradiction to $\text{cost}(V) = 0$. $\diamond$

Claim 3.4. *Let $C_j \in \mathcal{C}, a_i \in \mathcal{A}$. If $c^{a_i}_j \in X^\star_1$, then $c^{a_{i+1}}_l \in X^\star_2$ for some $C_l \in \mathcal{C}$ such that $a_{i+1} \in C_l$.*

Proof of Claim 3.4. Since $\mathcal{V}$ implements $\mathcal{O}$, by Observation 2.3, there is an undominated strategy $s \in X^\star_2$ that dominates a_{i+1} for p_2. For s to dominate a_{i+1} we need that $[U_2 + V_2](c^{a_i}_j, s) \geq U_2(c^{a_i}_j, a_{i+1}) = 1$.

Since s is undominated, $(c^{a_i}_j, s) \in \mathcal{X}^\star_{G[\mathcal{V}]}$. Therefore $0 \geq \text{cost}(\mathcal{V}) = \max_{\mathbf{x}\in\mathcal{X}^\star_{G[\mathcal{V}]}} \sum_{i\in[2]} V_i(\mathbf{x}) \geq V_2(c^{a_i}_j, s)$. Since we need $[U_2 + V_2](c^{a_i}_j, s) \geq 1$, it follows that $U_2(c^{a_i}_j, s) \geq 1$. The only strategies for p_2 that satisfy this condition are $c^{a_{i+1}}_l$ where $C_l \in \mathcal{C}$ such that $a_{i+1} \in C_l$. $\diamond$

We know from the definition of implementations that $X^\star_2 \neq \emptyset$. Therefore there are some $C_j \in \mathcal{C}, a_i \in C_j$ such that $c^{a_i}_j \in X^\star_2$. By Claim 3.3(i) we have that $c^{a_i}_j \in X^\star_1$. By Claim 3.4 we have that $c^{a_{i+1}}_p \in X^\star_2$ for some $C_p \in \mathcal{C}$ such that $a_{i+1} \in C_p$. By repeating this argumentation $3\hat{n}$ times, we obtain that for every $a_{i'} \in \mathcal{A}$, there is some $C_{j'} \in \mathcal{C}$ such that $a_{i'} \in C_{j'}$ and $c^{a_{i'}}_{j'} \in X^\star_2$.

This shows that $\mathcal{S} := \{C_j \mid C_j \in \mathcal{C}, \exists\, a_i \in \mathcal{A}, \text{ s.t. } c^{a_i}_j \in X^\star_2\}$ is a cover of $\mathcal{A}$, i.e., $\bigcup_{C\in\mathcal{S}} C = \mathcal{A}$. To show that $\mathcal{S}$ is an exact cover, we must show that if $C_j \in \mathcal{S}$ and $a_i \in C_j$ then for every $C_l \in \mathcal{C} \setminus \{C_j\}$ such that $a_i \in C_l$, we have that $C_l \notin \mathcal{S}$.

Assume, towards a contradiction, that some $a_i \in \mathcal{A}$ is covered twice, i.e., there are $C_j, C_l \in \mathcal{S}$ where $a_i \in C_j \cap C_l$. By Claim 3.3(iii) we cannot have both $c^{a_i}_j \in X^\star_2$ and $c^{a_i}_l \in X^\star_2$. Therefore, without loss of generality, assume that $c^{a_i}_j \in X^\star_2$ and $c^{a_z}_l \in X^\star_2$ for some $a_z \in C_l \setminus \{a_i\}$. Then by Claim 3.3(ii) $c^{a_i}_j$ dominates a_i for p_1. Moreover, $[U_1 + V_1](c^{a_i}_j, c^{a_z}_l) \geq U_1(a_i, c^{a_z}_l) = 1$ because $a_i \in C_l$.

Because $U_1(c^{a_i}_j, c^{a_z}_l) = 0$, we have $V_1(c^{a_i}_j, c^{a_z}_l) \geq 1$. By Claim 3.3(i) we have $c^{a_i}_j \in X^\star_1$ and we assumed that $c^{a_z}_l \in X^\star_2$. Thus $\text{cost}(\mathcal{V}) \geq V_1(c^{a_i}_j, c^{a_z}_l) \geq 1 > 0$, a contradiction. Therefore each element $a_i \in \mathcal{A}$ is covered exactly once, and $\mathcal{S}$ is an exact cover. This finishes the proof of Theorem 3.1.

4 Game Implementation is NP-hard for Max. Degree Three and Two Strategies

In all earlier reductions, the number of strategies per player has been unbounded. In this section we show that in graphical games even bounding the number of strategies and the degree of players together does not help to lower the complexity.

Theorem 4.1 ($\star$). Game Implementation *on graphical games is NP-hard even for degree three and if each player has at most two strategies.*

Proof. To show NP-hardness we reduce from X3C. Let $(\mathcal{A} = \{a_0, \dots, a_{3\hat{n}-1}\}, \mathcal{C} = \{C_0, \dots, C_{3\hat{n}-1}\})$ be an instance of X3C. We construct an instance $(N, \mathcal{X}, \mathcal{U}, \mathcal{O}, \delta)$ of graphical Game Implementation in the following way:

Let $N := \mathcal{C} \cup \mathcal{A}$ be the set of players, let $X_p = \{T_p, F_p\}$ be the set of strategies for player $p \in N$.

Construct the underlying graph $H := (N, E)$, where $E := \{\{a_i, C_j\} \mid C_j \in \mathcal{C}, a_i \in C_j\}$. It is easy to see that H has degree 3. Throughout this proof, for an element $a_i \in \mathcal{A}$, let us denote the sets that include it as C_i^1, C_i^2, C_i^3 in the order of increasing indices in $\mathcal{C}$. When defining utility functions, if the utility of the player does not depend on the strategy played by some other player, the strategy of this player is omitted from the function arguments.

For each element $a_i \in \mathcal{A}$, we define

$$\begin{aligned} &U_{a_i}(T_{C_i^1}, T_{C_i^2}, T_{C_i^3}, F_{a_i}) = U_{a_i}(F_{C_i^1}, F_{C_i^2}, F_{C_i^3}, F_{a_i}) \\ &= U_{a_i}(T_{C_i^1}, T_{C_i^2}, F_{C_i^3}, F_{a_i}) = U_{a_i}(T_{C_i^1}, F_{C_i^2}, T_{C_i^3}, F_{a_i}) \\ &= U_{a_i}(F_{C_i^1}, T_{C_i^2}, T_{C_i^3}, F_{a_i}) = 1 \end{aligned}$$

The undefined combinations pay out 0. For every $C_j \in \mathcal{C}$, utility function U_{C_j} is 0 for every strategy profile.

For every $a_i \in \mathcal{A}$ we put $O_{a_i} = \{T_{a_i}\}$. For every $C_j \in \mathcal{C}$, we put $O_{C_j} = X_{C_j}$, concluding the construction. The correctness proof is in the full version. □

5 Exact Game Implementation is NP-hard

The complexity of Exact Game Implementation has so far been open. In this section we show that it is NP-hard even when we have only two identical players.

Theorem 5.1 ($\star$). Exact Game Implementation *is NP-hard, even for two players and rectangular desired strategy-profile sets.*

Proof. We give a reduction from the NP-hard 3-Coloring problemin which are a graph H and need to decide whether H can be *properly colored* with three colors. That is, whether we can assign each vertex exactly one color such that no two adjacent vertices receive the same color.

Given an instance H of 3-Coloring we proceed as follows to construct an instance of Exact Game Implementation that consists of a game $G = (N, \mathcal{X}, \mathcal{U})$, a rectangular strategy profile set $\mathcal{O}$, and the real budget $\delta = 1$. There are two players, that is, $N = \{1, 2\}$. The sets of strategies of the two players are identical, that is, $\mathcal{X} = X_1 \times X_2$ and $X_1 = X_2$. Thus, we only describe X_1. For each vertex in $V(H)$ there is a corresponding strategy in X_1, that is, $V(H) \subseteq X_1$. We call these *vertex strategies*. Furthermore, for each combination of a color $c \in [3]$ and a vertex $v \in V(H)$ there is a strategy $(v, c) \in X_1$. We call these *color-choice strategies*, and we use C to denote the set of color-choice strategies, that is, $C = \{(v, c) \mid v \in V(H) \wedge c \in [3]\}$. Finally, we have a set D of $3n$ *dummy strategies*. Overall, $X_1 = X_2 = V(H) \cup C \cup D$.

The strategies to implement are the color-choice strategies, that is, $O_1 = O_2 = C$ and $\mathcal{O} = O_1 \times O_2$. Intuitively, the strategy (v, c) in O_1 that dominates strategy v after promises shall correspond to choosing color c for vertex v.

The utility functions are symmetric, that is, for each $x \in X_1 = X_2$ and $y \in X_2 = X_1$ we have $U_1(x, y) = U_2(y, x)$. Thus, we only describe U_1 explicitly. Moreover, we only give the non-zero values of U_1, all values not explicitly mentioned are 0. First, for each pair $(v, c_1), (v, c_2)$ of color-choice strategies that correspond to the same vertex $v \in V(H)$ we put

$$U_1((v, c_1), (v, c_2)) = \begin{cases} 3, c_1 = c_2 \\ 2, c_1 \neq c_2. \end{cases}$$

Second, for each pair $(u, c_1), (v, c_2)$ of color-choice strategies corresponding to adjacent vertices $u, v \in V(H)$ we put

$$U_1((u, c_1), (v, c_2)) = U_1((v, c_1), (u, c_2)) = \begin{cases} 1, c_1 = c_2 \\ 2, c_1 \neq c_2. \end{cases}$$

Third, for each vertex strategy v and each color-choice strategy (v, c) corresponding to v we put $U_1(v, (v, c)) = 3$. Fourth, for each vertex strategy u and each color-choice strategy (u, c) corresponding to a neighbor $u \in \mathrm{N}_H(v)$ of v we put $U_1(v, (u, c)) = 2$. Finally, for each color-choice strategy $x \in X_1$ we pick a distinct dummy strategy $y \in X_2$ and put $U_1(x, y) = 1$. This concludes the description of the Exact Game Implementation instance $\mathcal{I} = (G, \mathcal{O}, \delta)$ where $G = (N, \mathcal{X}, \mathcal{U})$ and $\delta = 1$.

Intuitively, the players are highly incentivized to play vertex strategies because of the values $U_1(v, (v, c)) = 3$. In order to dominate a vertex strategy v, we need to pick a color-choice strategy corresponding to that vertex v because in order to make a different strategy dominate v we would exceed the budget of 1. The values $U_1((v, c_1), (v, c_2))$ will enforce that both players select the same color for each vertex. Afterwards, if two adjacent vertices u, v would receive the same color c then the values $U_1((u, c), (v, c)) = 1 = U_2((u, c), (v, c))$ enforce that we would have to pay both players: These two color-choice strategies would have to dominate u (for player 1) and v (for player 2), respectively, and we have $U_1(u, (v, c)) = 2 = U_2((u, c), v)$.

The proof of the correctness is in the full version. □

6 Correction to Algorithm for Exact Game Implementation

Eidenbenz et al. (2011) gave an algorithm which on input of a game G and a desired strategy-profile region $\mathcal{O}$, finds the minimum δ such that $(G, \mathcal{O}, \delta)$ is a positive instance of Exact Game Implementation. This is Algorithm 1 in (Eidenbenz et al. 2011), which for completeness is contained in the full version.

The algorithm fails to give an exact implementation when for some player $i \in N$, a strategy in O_i dominates some other strategy in O_i. We show an example where it fails and provide a fix for a class of games which we refer to as equitable games.

To see that the algorithm does not always construct a correct payment promise $\mathcal{V}$, consider a 2-player instance where player 1 and player 2 both have two strategies $\{s_1, s_2\}$. Let us define the utility functions for both players $i \in [2]$ as

$$\begin{aligned} U_i(s_1, s_1) &= 2 \qquad & U_i(s_2, s_1) &= 1 \\ U_i(s_1, s_2) &= 1 \qquad & U_i(s_2, s_2) &= 0. \end{aligned}$$

Let $O_1 = \{s_1, s_2\}$ and $O_2 = \{s_1\}$.

Algorithm 1: Minimum cost exact implementation

```
input  : A game $G = (N, \mathcal{X}, \mathcal{U})$ and a rectangular
         strategy profile region $\mathcal{O} = O_1 \times \cdots \times O_n$.
output : A payment promise $\mathcal{V}$ and $\delta \geq 0$ such that $\mathcal{V}$
         implements $\mathcal{O}$ and $\max_{\mathbf{o} \in \mathcal{O}} \sum_{i \in N} V_i(\mathbf{o}) = \delta$ is
         smallest possible.
1  foreach $i \in N$ do
2      foreach mapping $F_i \colon X_i \setminus O_i \to O_i$ do
3          $V^{F_i} \leftarrow$ ComputeV($F_i, X_i, O_i$)
4  $\delta \leftarrow \infty$; $\mathcal{V} \leftarrow (0, \ldots, 0)$;
5  foreach $F = (F_1, \ldots, F_n) \in \mathcal{F}$ do
6      $\delta^F \leftarrow \max_{\mathbf{o} \in \mathcal{O}} \sum_{i \in N} V^{F_i}(o)$;
7      if $\delta^F < \delta$ then $\delta \leftarrow \delta^F$; $\mathcal{V} \leftarrow F$;
8  foreach $i \in N$ do
9      foreach $o_i \in O_i$ do
10         foreach $\mathbf{x}_{-i} \in \mathcal{X}{-}i \setminus \mathcal{O}_{-i}$ do $V_i(o_i, \mathbf{x}_{-i}) \leftarrow \infty$;
11 return $\delta, \mathcal{V}$
12 def ComputeV($F_i, X_i, O_i$ ):
13     foreach $o_i \in O_i$ do
14         foreach $\mathbf{o}_{-i} \in \mathcal{O}_{-i}$ do
15             if $F_i^{-1}(o_i) \neq \emptyset$ then
16             $V_i(o_i, \mathbf{o}_{-i}) \leftarrow \max\{0,$
17              $\max_{x_i \in F_i^{-1}(o_i)} U_i(x_i, \mathbf{o}_{-i}) - U_i(o_i, \mathbf{o}_{-i})\}$;
18             else $V_i(o_i, \mathbf{o}_{-i}) \leftarrow 0$;
19 return $V_i$
```

We can see that for both players s_1 dominates s_2, so $X_i^\star = \{s_1\}$ for all $i \in [2]$. Because $|X_i^\star \setminus O_i| = 0$ for all $i \in [2]$, the check on line (1) of Algorithm 1 from (Eidenbenz et al. 2011) is always false and the algorithm returns that $\mathcal{O}$ can be implemented exactly with cost 0. However, V_1 constructed by the Algorithm is 0 everywhere, and thus $O_1 \neq X_1^\star$, meaning this is not an exact implementation of $\mathcal{O}$.

Towards a correction, if we can for every player $i \in N$, pick for each of its desired strategies $o_i \in O_i$ an undesired strategy profile $\mathbf{x}^{o_i} \in \mathcal{X} \setminus \mathcal{O}$, such that $\mathbf{x}_i^{o_i} = o_i$, and for no desired strategy $o_i' \in O_i \setminus \{o_i\}$ do we have that $\mathbf{x}_{-i}^{o_i} = \mathbf{x}_{-i}^{o_i'}$. In other words, we require

$$|O_i| \leq |\mathcal{X}_{-i} \setminus \mathcal{O}_{-i}|. \tag{1}$$

Then we can ensure at no extra cost that no strategy in O_i dominates another strategy in O_i. Let us call a game for which, for every $i \in N$, Equation (1) holds *equitable*.

We start by showing that, if a game is equitable then we can translate every non-exact implementation to an exact implementation, where the cost is bounded by the worst-case payment over $\mathcal{O}$ in the initial implementation.

Throughout this section, let $\mathcal{F}$ denote the Cartesian product of the possible functions from $X_i \setminus O_i$ to O_i for every player, i.e., $\mathcal{F} = (X_1 \setminus O_1 \to O_1) \times \cdots \times (X_n \setminus O_n \to O_n)$.

Theorem 6.1 ($\star$). *Let $G = (N, \mathcal{X}, \mathcal{U})$ be an equitable game and $\mathcal{O} \subset \mathcal{X}$ a rectangular strategy profile region. Let $\mathcal{V}$ implement $\mathcal{O}$ (not necessarily exactly) and let $\delta = \max_{\mathbf{o} \in \mathcal{O}} \sum_{i \in N} V_i(\mathbf{o})$. Then the payment promise $\mathcal{V}^*$ below implements $\mathcal{O}$ exactly with* $\text{cost}(\mathcal{V}^*) = \delta$. *Let $M = U_{max} + \delta + 1$ with $U_{max} = \max_{i \in N, x \in \mathcal{X}} U_i(\mathbf{x})$, (i.e., U_{max} is the maximum amount any player may receive in utility). For every player $i \in N$, for each of its desired strategies $o_i \in O_i$, we choose an undesired strategy profile $\mathbf{x}^{o_i} \in \mathcal{X} \setminus \mathcal{O}$, such that $\mathbf{x}_i^{o_i} = o_i$, and for no desired strategy $o_i' \in O_i \setminus \{o_i'\}$ do we have that $\mathbf{x}_{-i}^{o_i} = \mathbf{x}_{-i}^{o_i'}$. Since G is equitable, we can choose such strategy profiles for every $i \in N, o_i \in O_i$.*

To define $\mathcal{V}^*$, for every $i \in N$, $\mathbf{x} \in \mathcal{X}$, set

$$V_i^*(\mathbf{x}) = \begin{cases} 0, & \text{if } \mathbf{x}_i \in X_i \setminus O_i \\ V_i(\mathbf{x}), & \text{if } \mathbf{x}_i \in O_i, \mathbf{x}_{-i} \in \mathcal{O}_{-i} \\ M + 1 - U_i(\mathbf{x}), & \text{if } \mathbf{x}_i \in O_i, \mathbf{x}_{-i} = \mathbf{x}_{-i}^{o_i} \\ M - U_i(\mathbf{x}), & \text{otherwise.} \end{cases} \tag{2}$$

The idea behind the payments is to enforce that every desired strategy has one undesired strategy profile where it is the best possible option. This prevents any other strategy from dominating it.

Next we show that Algorithm 1 by Eidenbenz et al. (2011) identifies a payment promise $\mathcal{V}$ minimizing $\max_{\mathbf{o} \in \mathcal{O}} \sum_{i \in N} V_i(\mathbf{o})$. To make the analysis of the algorithm easier, we give a simplified version of their algorithm, which is shown in Algorithm 1. We obtain the following:

Theorem 6.2 ($\star$). *For a given equitable game $G = (N, \mathcal{X}, \mathcal{U})$ and a set of desired strategy profiles $\mathcal{O} \subseteq \mathcal{X}$, the smallest $\delta \geq 0$ such that $(G, \mathcal{O}, \delta)$ is a positive instance of* Exact Game Implementation *can be identified in time* $O(|\mathcal{O}| \max_{i \in N}(n|O_i|^{|X_i \setminus O_i|}|\mathcal{O}||X_i \setminus O_i| + |O_i|^{n|X_i \setminus O_i|}))$.

7 Characterization of Zero-Budget Implementation

In this section we characterize rectangular strategy profiles $\mathcal{P} = P_1 \times \ldots \times P_n$ that can be implemented at zero cost. We call such profiles promise-Nash equilibrium (PNE). The naming comes from two considerations. First, if each player i has only one strategy in P_i, then a PNE is equivalent to a Nash equilibrium. Second, a PNE encapsulates the notion that no player i has an incentive to switch towards a strategy outside of P_i *provided* that each other player j plays only strategies in P_j. We believe this notion to be of independent interest because it models situations in which certain types of strategies may be off limits for, e.g., moral reasons. Using PNE may thus enable studying the price (or value) of morality and similar ideas. Formally:

Definition 7.1. Let $G = (N, \mathcal{X}, \mathcal{U})$ be a game. A rectangular strategy profile region $P_1 \times \cdots \times P_n \subset X_1 \times \cdots \times X_n$ is a *promise-Nash equilibrium (PNE)* if

$$\forall\, i \in N, \forall\, x_i \in X_i \setminus P_i \;\exists\, p_i \in P_i \colon$$
$$\forall\, \mathbf{p}_{-i} \in P_{-i} \colon \quad U_i(p_i, \mathbf{p}_{-i}) \geq U_i(x_i, \mathbf{p}_{-i}).$$

We observe the following.

Theorem 7.2 ($\star$). *Let $G = (N, \mathcal{X}, \mathcal{U})$ be a game. A rectangular strategy profile region $\mathcal{P} = P_1 \times \cdots \times P_n \subset X_1 \times \cdots \times X_n$ can be implemented with cost 0 if and only if $\mathcal{P}$ is a promise-Nash equilibrium.*

In fact, this theorem has an interpretation in the morality setting above: Let a profile $\mathcal{P}$ consist only of moral strategies and no amoral ones. Then $\mathcal{P}$ is a PNE if and only if morality is incentivized without any incentives having to actually be realized. In other words, morality is self-enforcing if and only if it constitutes a PNE.

Acknowledgments

Jiehua Chen and Sofia Simola are supported by the Vienna Science and Technology Fund (WWTF), grant number VRG18-012. Manuel Sorge acknowledges funding by the Alexander von Humboldt Foundation.

References

Brill, M.; Freeman, R.; and Conitzer, V. 2015. Computing the Optimal Game. In *Proceedings of the 2nd Workshop on Exploring Beyond the Worst Case in Computational Social Choice*, 1–8.

Buterin, V.; Reijsbergen, D.; Leonardos, S.; and Piliouras, G. 2020. Incentives in Ethereum's hybrid Casper protocol. *International Journal of Network Management*, 30(5).

Chen, J.; Khavidaki, N. L.; Haydn, S. V.; Simola, S.; and Sorge, M. 2022. Game Implementation: What Are the Obstructions? Technical report, arXiv:2212.00699 [cs.GT].

Conitzer, V.; and Sandholm, T. W. 2014. Complexity of Mechanism Design. Technical report, arXiv:cs/0205075 [cs.GT].

Deng, Y.; and Conitzer, V. 2017. Disarmament Games. In Singh, S.; and Markovitch, S., eds., *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, 473–479.

Deng, Y.; and Conitzer, V. 2018. Disarmament Games With Resource. In McIlraith, S. A.; and Weinberger, K. Q., eds., *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI 2018)*, 981–988. AAAI Press.

Deng, Y.; Tang, P.; and Zheng, S. 2016. Complexity and Algorithms of K-implementation. In *Proceedings of the 15th International Conference on Autonomous Agents and Multiagent Systems (AAMAS '16)*, 9.

Eidenbenz, R.; Pignolet, Y. A.; Schmid, S.; and Wattenhofer, R. 2011. Cost and complexity of harnessing games with payments. *International Game Theory Review*, 13(01): 13–44.

Hamari, J. 2019. *Gamification*, 1–3. John Wiley & Sons, Ltd.

Kearns, M. J.; Littman, M. L.; and Singh, S. 2001. Graphical Models for Game Theory. In Breese, J. S.; and Koller, D., eds., *Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence (UAI '01)*, 253–260. Morgan Kaufmann.

Letchford, J.; and Conitzer, V. 2010. Computing optimal strategies to commit to in extensive-form games. In *Proceedings of the 11th ACM conference on Electronic commerce (EC '10)*, 83–92. Association for Computing Machinery.

Maskin, E. 1999. Nash Equilibrium and Welfare Optimality. *Review of Economic Studies*, 66(1): 23–38.

Maskin, E.; and Sjöström, T. 2002. *Implementation Theory*, volume 1 of *Handbook of Social Choice and Welfare*, 237–288. Elsevier.

Monderer, D.; and Tennenholtz, M. 2004. K-Implementation. *Journal of Artificial Intelligence Research*, 21: 37–62.

Moscibroda, T.; and Schmid, S. 2009. On Mechanism Design without Payments for Throughput Maximization. In *Proceedings of the 28th IEEE International Conference on Computer Communications (INFOCOM 2009)*, 972–980.

Schaefer, T. J. 1978. The Complexity of Satisfiability Problems. In *Proceedings of the Tenth Annual ACM Symposium on Theory of Computing (STOC '78)*, 216–226.

Thaler, R. H.; and Sunstein, C. R. 2008. *Nudge: Improving Decisions about Health, Wealth, and Happiness*. Yale University Press.

Wooldridge, M.; Endriss, U.; Kraus, S.; and Lang, J. 2013. Incentive engineering for Boolean games. *Artificial Intelligence*, 195: 418–439.