Complexity and Algorithms of K-implementation

Yuan Deng
Dept. of Computer Science
Duke University
ericdy@cs.duke.edu

Pingzhong Tang
Institute of interdisciplinary information sciences
Tsinghua University, China
kenshinping@gmail.com

Shuran Zheng
Institute of interdisciplinary information sciences
Tsinghua University, China
sherry.zhengshuran@gmail.com

ABSTRACT

This paper settles the complexity of *K-implementation*, a ten-year open problem in AI. The problem is for a designer to modify an existing normal-form game, in a cost-optimal way, so as to ensure the solutions of the modified game fall into a given set of outcomes. We first prove that the problem is NP-COMPLETE for general games with respect to dominance by pure strategies, and then provide an alternative proof showing that the problem is NP-COMPLETE even for two-player games with respect to dominance by mixed strategies. We then consider a related but different objective, show its hardness and develop computationally efficient algorithms for a class of well-known games called *supermodular games*. For this objective, we are able to provide an optimal algorithm based on mixed-integer linear program. Interestingly, this algorithm also provides a lower-bound approximation guarantee for the original K-implementation problem and approximates the optimal solution well in experiments.

General Terms

Algorithms; Economics; Theory;

Keywords

K-implementation; Complexity; Algorithm;

1. INTRODUCTION

Using monetary payment to incentivize agents' behaviors has now become a standard approach in artificial intelligence, multi-agent systems and electronic commerces. The foundations of this approach date back to the pioneer works of *mechanism design theory* [2, 8, 11, 15, 20], where payments are used to design incentive compatible institutions and to circumvent impossibility results [7, 16].

Building upon these seminal works is an interesting line of research [4, 5, 6, 13, 14], rooted in the AI and MAS community, that looks at a different type of implementation problems, where there exists an incumbent game and the designer wants to subsidize the payoffs of the incumbent game, in a cost-minimum way, so as to ensure the solutions of the revised game fall into a pre-specified set of outcomes.

Appears in: *Proceedings of the 15th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2016), J. Thangarajah, K. Tuyls, C. Jonker, S. Marsella (eds.), May 9–13, 2016, Singapore.*

For example, consider taxation in the U.S. [10] where policy makers do not have full flexibility in designing a new taxation policy and can only modify the current system to some minimum extent. Similar is the case of redesign salary for government employees in China.

Monderer and Tennenholtz [13] initiate the first problem of this sort, called *K-implementation*, where the solution concept is *one-round dominance*, namely, whether one can distribute a positive[1] payment K to the incumbent game payoffs such that, by applying one round elimination of dominated strategies in the modified game, the remaining outcomes belong to a predefined sets. Clearly, the solution of *K-implementation* also provides a feasible solution if the solution concept is replaced by *iterated dominance*, since the outcomes survive iterated dominance can only be a subset of that of one-round dominance. They further show that the general problem is NP-HARD and study the cases where the predefined sets are restricted to singletons or the incumbent games are the VCG mechanisms.

This particular work generates fairly considerable interests, both within and outside the AI community, leads to a number of extensions. For instance, Ashlagi et al. [1] generalized to incomplete information setting and applied their new theories to position auctions and Zhang et al. [21] considered the case where an interested party aims to influence an agent's decisions by making limited changes to the agent's environment. Furthermore, the concept is applicable in dynamic mechanism design settings where, as time goes by, the designer needs to adaptively modify an existing game without flexibility to design a new games [9].

Unfortunately, the hardness proof from 3-SAT by Monderer and Tennenholtz is incorrect. As pointed out by [5], "while there indeed exists a 2-implementation for every satisfiable formula, it can be shown that 2-implementations also exist for non-satisfiable formulas", showing the incorrectness of the original proof. The complexity of K-IMPLEMENTATION has been open since [5, 6, 13].

In the first part of the paper, we pin down the correct proofs of the *K-implementation* problem. More specifically, we prove the following two results.

- We first show that K-IMPLEMENTATION is NP-COMPLETE, by reducing the 3-SAT problem to an instance of K-IMPLEMENTATION in a 6-player game.
- We then come up with a different proof that reduces the 3-SAT problem to a K-IMPLEMENTATION instance in a 2-player game only.

[1]There always exists a $-\infty$ solution to this problem.

The reason that we have two separate proofs is that the 6-player proof works under both dominance by pure strategies and dominance by mixed-strategies, while the 2-player works only under the notion of dominance by mix-strategies.

These results confirm that, even in the simplest game-theoretical setting with only two players[2], K-IMPLEMENTATION is computationally hard.

In the second part of the paper, we look for practical algorithms to solve a variation of K-IMPLEMENTATION problem with approximation guarantee, called INDIVIDUAL K-IMPLEMENTATION problem. Our results are two algorithms, one based on mixed-integer linear programming for the general INDIVIDUAL K-IMPLEMENTATION problem, the other is a polynomial time algorithm for a certain subclass of games. Finally, through experiments, we show that the optimal solution for INDIVIDUAL K-IMPLEMENTATION problem approximates K-IMPLEMENTATION well in practice.

2. PRELIMINARIES

Let $G = \langle N, S, U\rangle$ be a normal form game, where N is the set of n player, $S = S_1 \times \cdots \times S_n$ is the set of strategy profiles and $U = (U_1, \cdots, U_n)$ is the set of utility functions for each player. As usual, denote $s = (s_i, s_{-i}) \in S$ where $s_i \in S_i$ and $s_{-i} \in S_{-i}$ represents the strategies played by players other than player i.

A strategy $s_i \in S_i$ is weakly dominated by pure strategies if there exists $s'_i \in S_i$ such that for all $s_{-i} \in S_{-i}$, $U_i(s_i, s_{-i}) \leq U_i(s'_i, s_{-i})$. A strategy $s_i \in S_i$ is weakly dominated by mixed strategies if there exists $\sigma_i \in \Delta(S_i)$ such that for all $s_{-i} \in S_{-i}$, $U_i(s_i, s_{-i}) \leq U_i(\sigma_i, s_{-i})$, where $\Delta(S_i)$ is the set of mixed strategies on S_i.

The designer has a set of desirable strategy profiles $O \subseteq S$, and players are rational in the sense that they do not adopt a strategy that is weakly dominated[3] by another, *possibly mixed*, strategy.

A non-negative vector of payoff functions $V = (V_1, \cdots, V_n)$ implements O in game G if in the new game $\langle N, S, U + V\rangle$, the set of strategies $X = \{X_1 \times X_2 \times \cdots \times X_n\}$, satisfies

1. for all i, $S_i \setminus X_i$ is weakly dominated;
2. $\emptyset \subset X \subseteq O$.

Such a V is called a k-implementation of O in game G, if $\sum_{i=1}^{n} V_i(x) \leq k, \forall x \in X$. Denote $\langle G, O, k\rangle$ as the set of non-negative vector of payoff functions V such that V is a k-implementation of O in game G. Moreover, if k is the minimum cost such that $\langle G, O, k\rangle$ is not empty, we say the *implementation value (cost)* of G w.r.t O is k.

In order to prove the NP-COMPLETENESS of checking whether set $\langle G, O, k\rangle$ is empty or not, it suffices to consider only the case, where the desirable set O can be expressed as a Cartesian product of individual sets: $O = \prod_{i=1}^{n} O_i$ with $O_i \subseteq S_i$.

Moreover, to develop a practically efficient algorithm, we consider a variant, called INDIVIDUAL K-IMPLEMENTATION. V is an individual k-implementation of O in game G, if $V_i(x) \leq k, \forall x \in X, i \in N$. In other words, we measure the largest cost spent on an individual player[4].

While our investigation of individual K-implementation is of independent interests, we are also interested in the connections between the two problems: it turns out the INDIVIDUAL K-IMPLEMENTATION admits a compact mixed integer linear programming formation and its optimal solution also provides an approximately good solution to the original problem. Furthermore, the new problem can be solved efficiently on certain subclasses of games. We show this by developing an efficient algorithm for INDIVIDUAL K-IMPLEMENTATION in supermodular games [12, 18, 19].

3. THE COMPLEXITY OF K-IMPLEMENTATION

We develop two reductions, both from 3-SAT, to show NP-COMPLETENESS for K-IMPLEMENTATION, one for 6 players and the other for 2 players.

3.1 NP-COMPLETENESS for 6-player games

THEOREM 1. *It is* NP-COMPLETE *to decide whether* $\langle G, O, k\rangle$ *is empty in* 6*-player games.*

It is clear that the problem is in NP, since given a V vector, one can check whether $V \in \langle G, O, k\rangle$ in polynomial time w.r.t the input size [3].

To prove completeness, we reduce from 3-SAT. 3-SAT is the most famous NP-COMPLETE problem. An instance of 3-SAT is a Boolean formula in conjunctive normal form, which is "AND" operations over clauses, each of which is "OR" operations over exactly three ground literals (ground variables and their negations). The problem is to decide whether there exists an assignment of truth values to the variables such that the value of the formula is true.

Let $\phi = c_1 \wedge c_2 \wedge \cdots \wedge c_m$ be an instance of 3-SAT, where the ith clause $c_i = (l_i^1 \vee l_i^2 \vee l_i^3)$ and $l_i^j = +x_k$ or $-x_k$, where x_k is a ground variable from $\{x_1, \cdots, x_n\}$.

We now construct a K-IMPLEMENTATION instance $\langle G(\phi), O, \delta\rangle$, where δ is a real such that $0 < \delta \ll 1$, $G(\phi)$ is a 6-player game $\langle\{1, 2, 3, 4, 5, 6\}, S, U\rangle$. The strategy sets of the players are defined as follows:

- $S_1 = S_2 = S_3 = \bigcup_{i=1}^{n}\{+x_i\} \cup \bigcup_{i=1}^{n}\{-x_i\} \cup \{\epsilon\}$;
- $S_4 = \bigcup_{i=1}^{m}\{c_i\} \cup \{\epsilon\}$, $S_5 = S_6 = \bigcup_{i=1}^{n}\{+x_i\}$;

We now define the utility function for each player. In following definitions, we use $*_i$ to denote an arbitrary strategy of player i. The undefined utility entries are 0.

- Players 5 and 6 only care about each other, $\forall x_i$:
  - $U_5(*_1, *_2, *_3, *_4, +x_i, +x_i) = 1$
  - $U_6(*_1, *_2, *_3, *_4, +x_i, +x_{(i+1) \mod n}) = 1$
- Player 1 only cares about player 6, $\forall x_i$:
  - $U_1(+x_i, *_2, *_3, *_4, *_5, +x_i) = 1$
  - $U_1(-x_i, *_2, *_3, *_4, *_5, +x_i) = 1$

[2]1-agent K-IMPLEMENTATION is computationally easy.

[3]Strict and weak dominance are the same w.r.t K-implementation, since one can turn a solution of weak dominance into a strict one by adding an infinitely small cost to each outcome without changing the cost. For the same reason, our definition of *weak dominance* is also the same as the standard definition of weak dominance that requires at least one strict inequality.

[4]Our techniques can be extended to the case where the requirements are $V_i(x) \leq k_i, \forall x \in X, i \in N$ and the goal is to minimize $\sum_i k_i$. For simplicity, we only consider the special case with $k = k_i$ for all i.

- Player 2, 3 care about player 1, $\forall l$:
  - $U_2(l, l, *_3, *_4, *_5, *_6) = 1$
  - $U_3(l, *_2, l, *_4, *_5, *_6) = 1$
  - $U_2(*_1, \epsilon, *_3, *_4, *_5, *_6) = 0.1$
  - $U_3(*_1, *_2, \epsilon, *_4, *_5, *_6) = 0.1$
- Player 4 care about players 1, 2, 3, $\forall c_i = (l_i^1 \vee l_i^2 \vee l_i^3)$:
  - $U_4(l_i^1, l_i^2, l_i^3, c_i, *_5, *_6) = 1$
  - $U_4(*_1, *_2, *_3, \epsilon, *_5, *_6) = 0.1$

The set of desirable outcomes are defined by:

$$O_4 = \{\epsilon\} \text{ and for all } k \neq 4,\ O_k = S_k$$

We complete the reduction by showing that ϕ is satisfiable if and only if $\langle G(\phi), O, \delta\rangle \neq \emptyset$. The easier direction is proved in the following lemma.

Lemma 1. *If ϕ is satisfiable, then $\langle G(\phi), O, \delta\rangle \neq \emptyset$.*

Proof. Let literals $(l'_1, \cdots, l'_n)$ be a feasible assignment that satisfies ϕ. We show that the following V is in $\langle G(\phi), O, \delta\rangle$. (the undefined entries are 0)

- $V_1(-l'_i, *_2, *_3, *_4, *_5, *_6) = \delta, \forall 0 \leq i < n$;
- $V_2(l'_i, \epsilon, *_3, *_4, *_5, *_6) = 1, \forall 0 \leq i < n$;
- $V_3(l'_i, *_2, \epsilon, *_4, *_5, *_6) = 1, \forall 0 \leq i < n$;
- $V_4(l_j^1, l_j^2, l_j^3, \epsilon, *_5, *_6) = 1, \forall c_j = (l_j^1 \vee l_j^2 \vee l_j^3)$;

We can verify the following,

- By our construction of U_5 and U_6, no action of player 5 or 6 is weakly dominated, thus $X_5 = O_5$, $X_6 = O_6$.
- By our construction of V_1 and U_1, for player 1, l'_i is strictly dominated by $-l'_i$ since we now add δ to playing $-l'_i$. Therefore $X_1 = \bigcup_i \{-l'_i\}$.
- By our construction of V_2, V_3, for both players 2 and 3, l'_i is now strictly dominated by ϵ, therefore $X_2 = X_3 = \bigcup_i \{-l'_i\} \cup \{\epsilon\}$. In other words, all the literals that are *true* in the satisfying assignment are eleminated.
- For player 4, c_i is strictly dominated by ϵ, $X_4 = \{\epsilon\}$.

Clearly, for each i, we have $X_i \subseteq O_i$, therefore $X \subseteq O$, as required by K-implementation.

Finally, we prove that $k = \max_{x\in X} \sum_i V_i(x) = \delta$. Since $l'_i \notin X_1$, $(l'_i, *_2, *_3, *_4, *_5, *_6) \notin X$, i.e., V_2 or V_3 do not contribute to the implementation cost $\max_{x\in X} \sum_i V_i(x)$.

In addition, since $(l'_1, \cdots, l'_n)$ satisfies ϕ, we have for all j, at least one of l_j^1, l_j^2, l_j^3 is *true*. Notice that X_1, X_2 and X_3 only contain negations of l'_i's, then $(l_j^1, l_j^2, l_j^3, \epsilon, *_5, *_6) \notin X$. In other words, V_4 does not contribute to the implementation cost either.

It then follows that the implementation k is at most the maximum cost resulting from $V_1(\cdot)$, which is exactly δ. In other words, V is in $\langle G(\phi), O, \delta\rangle$. □

To establish the other direction, we have:

Lemma 2. *If $\langle G(\phi), O, \delta\rangle \neq \emptyset$, then ϕ is satisfiable.*

Proof. Let such a δ-implementation be V and set a literal l to be true if and only if $l \notin X_1$ or $l \notin X_2$ or $l \notin X_3$. We prove this is a satisfying assignment for ϕ.

Claim 1. *$X_5 = O_5$, $X_6 = O_6$.*

Proof. Notice that the payoffs of player 5 and player 6 is independent of the actions played by the other players, and thus, one can think of a 2-player game between player 5 and player 6 and compute X_5 and X_6 in this 2-player game.

Since S_5 and S_6 only contain positive variables and the fact that both X_5 and X_6 are nonempty, it is with loss of generality to assume that there exists $+x_k \in X_6$. We then have $+x_k \in X_5$ as well, since otherwise, in order to eliminate $+x_k$ from X_5 given $+x_k \in X_6$, there must exist a $j \neq k$, $+x_j \in X_5$ such that $V_5(*_1, *_2, *_3, *_4, +x_j, +x_k) \geq 1 > \delta$, which contradicts to our assumption that largest cost is less than or equal to δ. Similarly, suppose $+x_k \in X_5$, we must have $+x_{(k+1) \mod n} \in X_6$ as well by similar arguments.

Therefore, if $+x_k \in X_6$, $+x_k \in X_5$, $+x_{(k+1) \mod n} \in X_6$ and $+x_{(k+1) \mod n} \in X_5$... By repeating this argument, we conclude that $X_5 = O_5$ and $X_6 = O_6$. □

The intuition of the lemma above should be clear: elimination of any actions of player 5 and player 6 is costly.

The next claim guarantees that the constructed assignment is well-defined in the sense that it never simultaneously assigns *true* to a variable and its negation.

Claim 2. *$\forall i$, either we have $+x_i \in X_1$ and $+x_i \in X_2$ and $+x_i \in X_3$ hold simultaneously; or we have $-x_i \in X_1$ and $-x_i \in X_2$ and $-x_i \in X_3$ hold simultaneously.*

Proof. First of all, by Claim 1, for player 1, $+x_i$ and $-x_i$ cannot be both strictly dominated, i.e., at least one of $+x_i$ and $-x_i$ is in X_1; since otherwise, there must exist $x' \in X_1$ such that $x' \neq +x_i, -x_i$ and $V_1(x', *_2, *_3, *_4, *_5, +x_i) \geq 1 > \delta$, which again contradicts to δ-implementation.

For player 2, we must have $l \in X_2$ if $l \in X_1$; since otherwise, there must exist $l' \in X_2$ such that $V_2(l, l', *_3, *_4, *_5, *_6) \geq 1 > \delta$, a contradiction; or $V_2(l, \varepsilon, *_3, *_4, *_5, *_6) \geq 1 - 0.1 = 0.9 > \delta$, also a contradiction to δ-implementation.

For player 3, we can apply the same argument to conclude that $l \in X_3$ if $l \in X_1$. □

To complete the proof of the lemma, we need to prove that for any clause $c_i = (l_i^1 \vee l_i^2 \vee l_i^3)$, at least one of l_i^1, l_i^2, l_i^3 is assigned true. According to our construction at the beginning of the proof, it is equivalent to that $\exists j, l_i^j \notin X_j$. Suppose otherwise that $l_i^1 \in X_1, l_i^2 \in X_2, l_i^3 \in X_3$, we must have $c_i \in X_4$; since otherwise, there exists $c' \in X_4$ such that $V_4(l_i^1, l_i^2, l_i^3, c', *_5, *_6) \geq 1 > \delta$ or $V_4(l_i^1, l_i^2, l_i^3, \epsilon, *_5, *_6) \geq 0.9 > \delta$. In other words, we need to cost at least 0.9 to eliminate c_i in S_4 in presence of $l_i^1 \in X_1, l_i^2 \in X_2, l_i^3 \in X_3$. Note that however, $c_i \in X_4$ is unacceptable since we need $X_4 = \{\epsilon\}$. We therefore conclude that $l_i^1 \in X_1, l_i^2 \in X_2, l_i^3 \in X_3$ cannot hold simultaneously, that is, at least one of l_i^1, l_i^2, l_i^3 is assigned true in any clause, i.e., the assignment we constructed is indeed a satisfying assignment. □

Combining Lemma 1 and Lemma 2, we show ϕ is satisfiable if and only if $\langle G(\phi), O, \delta\rangle \neq \emptyset$, completing the proof.

Remark 1. *The K-implementation defined by [13] is w.r.t dominance by pure strategies. The above proof is general in the sense that it applies to either dominance by pure strategies or mixed strategies.*

$U_1 \setminus U_2$	x_1	x_2	x_3	$+x_1$	$-x_1$	$+x_2$	$-x_2$	$+x_3$	$-x_3$	c_1	c_2
$(+x_1, c_1)$	$(0,\varepsilon_b)$	$(r_{x_2}^{+x_1,c_1},0)$	$(r_{x_3}^{+x_1,c_1},0)$	(0,1)	(0,0)	(0,0)	(0,0)	(0,0)	(0,0)	$(0,\varepsilon_a)$	(0,0)
$(+x_2, c_1)$	$(r_{x_1}^{+x_2,c_1},0)$	$(0,\varepsilon_b)$	$(r_{x_3}^{+x_2,c_1},0)$	(0,0)	(0,0)	(0,1)	(0,0)	(0,0)	(0,0)	$(0,\varepsilon_a)$	(0,0)
$(+x_3, c_1)$	$(r_{x_1}^{+x_3,c_1},0)$	$(r_{x_3}^{+x_3,c_1},0)$	$(0,\varepsilon_b)$	(0,0)	(0,0)	(0,0)	(0,0)	(0,1)	(0,0)	$(0,\varepsilon_a)$	(0,0)
$(-x_1, c_2)$	$(0,\varepsilon_b)$	$(r_{x_2}^{-x_1,c_2},0)$	$(r_{x_3}^{-x_1,c_2},0)$	(0,0)	(0,1)	(0,0)	(0,0)	(0,0)	(0,0)	(0,0)	$(0,\varepsilon_a)$
$(-x_2, c_2)$	$(r_{x_1}^{-x_2,c_2},0)$	$(0,\varepsilon_b)$	$(r_{x_3}^{-x_2,c_2},0)$	(0,0)	(0,0)	(0,0)	(0,1)	(0,0)	(0,0)	(0,0)	$(0,\varepsilon_a)$
$(+x_3, c_2)$	$(r_{x_1}^{+x_3,c_2},0)$	$(r_{x_2}^{+x_3,c_2},0)$	$(0,\varepsilon_b)$	(0,0)	(0,0)	(0,0)	(0,0)	(0,1)	(0,0)	(0,0)	$(0,\varepsilon_a)$
x_{1+}	(0,1)	(0,1)	(0,1)	(2,0)	(0,0)	(0,0)	(0,0)	(0,0)	(0,0)	(0,0)	(0,0)
x_{1-}	(0,1)	(0,1)	(0,1)	(0,0)	(2,0)	(0,0)	(0,0)	(0,0)	(0,0)	(0,0)	(0,0)
x_{1*}	(0,0)	(0,0)	(0,0)	(1.5,0)	(1.5,0)	(0,0)	(0,0)	(0,0)	(0,0)	(0,0)	(0,0)
x_{2+}	(0,1)	(0,1)	(0,1)	(0,0)	(0,0)	(2,0)	(0,0)	(0,0)	(0,0)	(0,0)	(0,0)
x_{2-}	(0,1)	(0,1)	(0,1)	(0,0)	(0,0)	(0,0)	(2,0)	(0,0)	(0,0)	(0,0)	(0,0)
x_{2*}	(0,0)	(0,0)	(0,0)	(0,0)	(0,0)	(1.5,0)	(1.5,0)	(0,0)	(0,0)	(0,0)	(0,0)
x_{3+}	(0,1)	(0,1)	(0,1)	(0,0)	(0,0)	(0,0)	(0,0)	(2,0)	(0,0)	(0,0)	(0,0)
x_{3-}	(0,1)	(0,1)	(0,1)	(0,0)	(0,0)	(0,0)	(0,0)	(0,0)	(2,0)	(0,0)	(0,0)
x_{3*}	(0,0)	(0,0)	(0,0)	(0,0)	(0,0)	(0,0)	(0,0)	(1.5,0)	(1.5,0)	(0,0)	(0,0)

Table 1: $G_{\varepsilon,\delta}(\phi)$ for $A = \{x_1, x_2, x_3\}$ and $c_1 = (+x_1 \vee +x_2 \vee +x_3)$, $c_2 = (-x_1 \vee -x_2 \vee +x_3)$.

3.2 NP-COMPLETENESS for 2-player games

THEOREM 2. *It is* NP-COMPLETE *to decide whether $\langle G, O, k\rangle$ is empty in 2-player games.*

Let ϕ be an instance of 3-SAT. Let A be the set of its variables $x_1, \cdots, x_n$, L be the corresponding literals and C be the set of clauses. Moreover, let L^+ be the set of positive literals and L^- be the set of negative ones. Define function $v : L \to A$ to return the corresponding variable of a literal, that is, $v(+x_i) = v(-x_i) = x_i$. We construct $G_{\varepsilon,\delta}(\phi)$ with $0 < \varepsilon \ll \delta \ll 1$ to be the following 2-player game, with an example shown in Table 1.

In $G_{\varepsilon,\delta}(\phi)$, the action set of the column player is $S_2 = A \cup L \cup C$. As for the row player, define a product set $B = \{(l, c) \in L \times C | l \in c\}$, and the action set of the row player is $S_1 = B \cup (A \times \{-, +, *\})$. For convenience, denote by $s_{l,c}$ a strategy in B, and v_-, v_+ v_* for elements in $(A \times \{-, +, *\})$. Let the desirable sets be $O_1 = B \cup (A \times \{-, +\}) \subseteq S_1$ and $O_2 = A \cup L \subseteq S_2$. Payoff functions U is defined as follows,

- $U_2(s_{l,c}, l) = 1$, $U_2(s_{l,c}, c) = \varepsilon_a$, $\forall (l, c) \in B$;
- $U_2(s_{l,c}, v) = \varepsilon_b$, $\forall (l, c) \in B, v \in A$ with $v(l) = v$;
- $U_1(v_-, l) = 2$, $\forall l \in L^-, v \in A$ with $v(l) = v$;
- $U_1(v_+, l) = 2$, $\forall l \in L^+, v \in A$ with $v(l) = v$;
- $U_1(v_*, l) = 1.5$ $\forall l \in L, v \in A$ with $v(l) = v$;
- $U_2(v_+, v') = 1, U_2(v_-, v') = 1$ $\forall v, v' \in A$;
- For each $s_{l,c} \in B$, set $U_1(s_{l,c}, v) = r_v^{l,c}$ for $v \in A$. The vector $r^{l,c}$ satisfies that
  - $r_{v(l)}^{l,c} = 0$;
  - Let $r'^{l,c}$ as a $(|A| - 1)$-dimension vector by removing the $v(l)$-th element from $r^{l,c}$, we require $\|r'^{l,c}\|_2 = 1$ and $\|r'^{l,c} - \frac{1}{\sqrt{|A|-1}}\mathbf{1}\|_2 = \delta$,

    (where $\mathbf{1}$ is a vector with 1s);
  - [5]For all possible pairs $(l', c'), (l, c)$ with $l' \neq l$ but $v(l') = v(l)$, $r^{l',c'} \neq r^{l,c}$;

where $\varepsilon_b = 1 - \frac{1}{2}\varepsilon$, ε_a is a number that satisfies $\frac{1}{3}\varepsilon_b + \varepsilon < \varepsilon_a < \frac{\varepsilon_b}{2+\varepsilon_b} + \varepsilon$,[6] and undefined payoff entries are 0. Choose ε according to δ by the following lemma.

LEMMA 3. *Given $0 < \delta \ll 1$, there exists a sufficiently small ε with $0 < \varepsilon \ll \delta$ such that given any non-negative vector of payoff functions V, if a row strategy $b \in B$ but $b \notin X_1$, then there exists $v \in A, d \in X_1$ such that $V_1(d, v) > \varepsilon$.*[7]

We complete the reduction by showing that ϕ is satisfiable if and only if $\langle G_{\varepsilon,\delta}(\phi), O, \varepsilon\rangle \neq \emptyset$. The "only if" direction is relatively straightforward (Lemma 4): if ϕ is satisfiable, one can directly construct V matrices that guarantee a ε-implementation. The "if" direction is more involved.

To prove the "if" direction, we show that, given any $V \in \langle G_{\varepsilon,\delta}(\phi), O, \varepsilon\rangle$ and its X_1, X_2, assigning truth to literals in $X_2 \cap L$ forms a satisfying assignment. To verify this claim, we show that our construction enjoys two properties:

1. For all $v \in A$, $+v$ and $-v$ cannot be in X_2 simultaneously, which guarantees that the assignment induced by $X_2 \cap L$ is well-defined (Claim 3).

2. For each clause $c = (l_1 \vee l_2 \vee l_3) \in C$, at least one $l_j \in X_2$ for $j \in \{1, 2, 3\}$. This property is achieved by first showing that $B \subseteq X_1$ (Claim 4). Together with property 1 above, we are able to conclude if none of l_j is in X_2, we must have c is in X_2, contradicting to our definition of O_2.

LEMMA 4. *If ϕ is satisfiable, then $\langle G_{\varepsilon,\delta}(\phi), O, \varepsilon\rangle \neq \emptyset$.*

PROOF. Suppose the literals $(l'_1, \cdots, l'_n)$ is a satisfying assignment for ϕ. We show that the following V is in $\langle G_{\varepsilon,\delta}(\phi), O, \varepsilon\rangle$ (the undefined entries are simply 0)

- $V_2(s_{l,c}, l'_i) = \varepsilon$, $\forall i, (l, c) \in B$;
- $V_2(s_{l,c}, v) = \varepsilon$, $\forall (l, c) \in B, v \in A$;
- $V_1(v_b, -l'_i) = 2$, $\forall i, v(l'_i) = v, l'_i \notin L^b$;

First, we prove that the constructed V implements O.

[5]This requirement ensures that for all pairs (l, c), their corresponding $r^{l,c}$ are distinct.

[6]Such an ε_a always exists cause $\frac{1}{3}\varepsilon_b < \frac{\varepsilon_b}{2+\varepsilon_b}$ when $0 < \varepsilon_b < 1$

[7]Missing proofs are deferred to the full version.

- $C \cap X_2 = \emptyset$: For each clause $c = (l_1 \vee l_2 \vee l_3) \in C$, since $(l'_1, \cdots, l'_n)$ satisfies ϕ, at least one $l'_j \in c_i$. Without loss of generality, suppose l_1 is assigned true. We claim that the column strategy c is dominated by the convex combination of the column strategy $l_1, v(l_2), v(l_3)$ with probability $(p, \frac{1-p}{2}, \frac{1-p}{2})$, where $p = \frac{\varepsilon_b}{\varepsilon_b+2}$. Notice that in utility matrix $U_2 + V_2$, column c only has three non-zero entries with value ε_a in rows $s_{l_1,c}, s_{l_2,c}, s_{l_3,c}$ and the utilities of $l_1, v(l_2), v(l_3)$ in these three rows are $(1+\varepsilon, \varepsilon, \varepsilon), (\varepsilon, \varepsilon+\varepsilon_b, \varepsilon), (\varepsilon, \varepsilon, \varepsilon+\varepsilon_b)$.

$$p(1+\varepsilon,\varepsilon,\varepsilon) + \frac{1-p}{2}(\varepsilon,\varepsilon+\varepsilon_b,\varepsilon) + \frac{1-p}{2}(\varepsilon,\varepsilon,\varepsilon+\varepsilon_b)$$
$$= (p+\varepsilon, p+\varepsilon, p+\varepsilon) \geq (\varepsilon_a, \varepsilon_a, \varepsilon_a),$$

since $\frac{1-p}{2} \cdot \varepsilon_b = p$ and $\varepsilon_a < \varepsilon + p$.

- $A \times \{*\} \cap X_1 = \emptyset$: Either v_+ or v_- can dominate v_*.

Next, we show that V has implementation cost ε. Note that each column $-l'_i$ is dominated by column $v(-l'_i)$, since $\varepsilon_b + \varepsilon > 1$, implying V_1 does not contribute to the implementation cost. Thus, the implementation cost is ε. □

LEMMA 5. *If $\langle G_{\varepsilon,\delta}(\phi), O, \varepsilon\rangle \neq \emptyset$, then ϕ is satisfiable.*

PROOF. Suppose $V \in \langle G_{\varepsilon,\delta}(\phi), O, \varepsilon\rangle$. We prove that giving truth to literals in $X_2 \cap L$ forms a satisfying assignment for ϕ. First, we show that such assignment is well-defined.

CLAIM 3. *For all $v \in A$, $\{+v, -v\} \not\subseteq X_2$.*

PROOF. Suppose not, then the row player's utility on rows v_+, v_-, v_* on column $+v$ and $-v$ forms a gadget $v_+ : (2, 0)$, $v_- : (0, 2)$, $v_* : (1.5, 1.5)$. In order to dominate v_*, the implementation cost is at least $0.5 > \varepsilon$ such that v_+ changes to $(2.5, 0.5)$, v_- changes to $(0.5, 2.5)$ and thus, $(\frac{1}{2}v_+ + \frac{1}{2}v_-)$ weakly dominates v_*. □

CLAIM 4. *$B \subseteq X_1$.*

Finally, we prove that at least one literal is assigned true in each clause. Suppose not, and there exists $c = (l_1 \vee l_2 \vee l_3)$ such that $l_j \notin X_2$ for all $j \in \{1, 2, 3\}$. Since V implements O, we have $c \notin X_2$, which means that there exists a convex combination of strategies in X_2 that can weakly dominate c. Notice that in utility matrix U_2, column c only has three non-zero entries $(\varepsilon_a, \varepsilon_a, \varepsilon_a)$ in rows $s_{l_1,c}, s_{l_2,c}, s_{l_3,c}$. Given that $l_j \notin X_2$ for all j, the only strategies in X_2 that have non-zero entries in these three rows are $v(l_1), v(l_2), v(l_3)$, with utilities $(\varepsilon_b, 0, 0), (0, \varepsilon_b, 0), (0, 0, \varepsilon_b)$ respectively. Based on $\{s_{l_1,c}, s_{l_2,c}, s_{l_3,c}\} \subseteq B \subseteq X_1$ and $\frac{\varepsilon_b}{3} + \varepsilon < \varepsilon_a$, it is impossible to dominate c with implementation cost ε. □

Based on a similar construction (let $U_1(v_*, c_i) = 1$, for all $c_i \in C$ to enforce $C \cap X_2 = \emptyset$), we conclude

COROLLARY 1. *It is* NP-COMPLETE *to decide whether $\langle G, O, k\rangle$ is empty in 2-player case even if $O_1 = S_1$.*

Moreover, the construction still holds if we apply iterative elimination of dominated strategies. In fact, we remark without proof that the two notions of dominance are equivalent with respect to K-implementation.

COROLLARY 2. *It is* NP-COMPLETE *to decide whether $\langle G, O, k\rangle$ is empty even in the 2-player case, under iteratively elimination of strictly dominated strategies.*

4. INDIVIDUAL K-IMPLEMENTATION

We now turn to the INDIVIDUAL K-IMPLEMENTATION problem. As mentioned, while this problem is of independent interest, it turns out that one can also connect the two problems, by using an exact algorithm of INDIVIDUAL K-IMPLEMENTATION to return approximately optimal solutions to the original problem. A polynomial time algorithm is further developed for a special subclass of games. With the same proof as that of Theorem 1, we have,

THEOREM 3. *It is* NP-COMPLETE *to decide whether $\langle G, O, k\rangle$ is empty for* INDIVIDUAL K-IMPLEMENTATION.

The following proposition establishes the approximation guarantee for K-IMPLEMENTATION problem via solving INDIVIDUAL K-IMPLEMENTATION.

PROPOSITION 1. *Given n-player games, the optimal* INDIVIDUAL K-IMPLEMENTATION *solution is an n-approximation solution of* K-IMPLEMENTATION.

4.1 A mixed-integer linear program

We propose an exact algorithm via mixed-integer linear program (MILP), starting with the following observation.

OBSERVATION 1. *For* INDIVIDUAL K-IMPLEMENTATION, *if $V \in \langle G, O, \varepsilon\rangle$, then let $V' = V$ except $V'_i(x) = \varepsilon$ for all $1 \leq i \leq n$ and $x \in X$, we still have $V' \in \langle G, O, \varepsilon\rangle$.*

The above observation states that for every $x \in X$, we can increase $V_i(x)$ to ε without violating implementability.

4.1.1 Two-player, product desirable set

For ease of presentation, we first consider two player normal-form game with product desirable set $O = O_1 \times O_2$. Let the strategy sets be $S_1 = [n]$ and $S_2 = [m]$ while $O_1 \subseteq S_1$, $O_2 \subseteq S_2$, respectively. Moreover, let the payoff matrix be A for row player and B for column player. That is, $U_1(i, j) = A_{i,j}$ and $U_2(i, j) = B_{i,j}$.

Without loss of generality, suppose all utilities are positive and let M be the largest utility. Divide each utility by M to normalize the utility so that no utility is greater than 1.

To formulate MILP, we first introduce an indicator vector $\mathbf{r} \in \{0, 1\}^n$ to indicate whether row player actions are in X or not. Formally, $\mathbf{r}_i = 1$ if and only if action $i \in X_1$. Similarly, we define such a indicator vector $\mathbf{c} \in \{0, 1\}^n$ for column actions and $\mathbf{c}_j = 1$ if and only if action $j \in X_2$.

We must ensure that, row player action i is indeed weakly dominated if and only if $\mathbf{r}_i = 0$. This step is done by a well-known linear program by Conitzer and Sandholm [3] and restated in [17]. For each row player action k, define a vector of probability variables $\mathbf{p}^k \in [0, 1]^n$, where $\mathbf{p}^k_i$ represents the probability assigned to row strategy i. One can check weak dominance for row strategy k by following linear program,

$$\begin{array}{ll} \min & 0 \\ s.t. & A^T \mathbf{p}^k \geq A^T_{:,k} \\ & \mathbf{p}^k \geq 0, |\mathbf{p}^k|_1 = 1 \end{array}$$

Here, $A^T_{:,k}$ denotes the k-th column of matrix A^T. To extend this linear programming to INDIVIDUAL K-IMPLEMENTATION setting, we first need to ensure that if $\mathbf{r}_i = 0$, then $\mathbf{p}^k_i = 0$ and if $\mathbf{r}_i = 1$, $\mathbf{p}^k_i \in [0, 1]$. In other words, if an action is dominated, it cannot appear as support in any mixed strategy for the purpose of checking dominance. The constraints can be conveniently presented as follows:

$$0 \le \mathbf{p}_i^k \le \mathbf{r}_i \quad 1 \le i, k \le n$$

Moreover, for indicator $\mathbf{r}_i$, if $i \notin O_1$, $\mathbf{r}_i$ must be 0. Thus, the integer constraints are

$$\begin{array}{ll} \mathbf{r}_i \in \{0,1\} & 1 \le i \le n \\ \mathbf{r}_i = 0 & \forall i \notin X_1 \end{array}$$

To put the Conizter-Sandholm LP framework under the context of ε-implementation, and further incorporate Observation 1, the constraints for the row players become:

$$(A^T + V_\varepsilon)\mathbf{p}^k \ge A^T_{:,k}$$

where V_ε is a matrix with ε in all entries, equivalent to

$$A^T\mathbf{p}^k \ge A^T_{:,k} - \varepsilon\mathbf{1}$$

where $\mathbf{1}$ is a vector with all 1's.

The constraint above is otherwise correct with one important exception. If a column strategy $j \notin X_2$, since we only count the implementation cost for action profiles in X in individual implementation, there is no implementation cost to dominate row player action k on the j-th column, since one can set sufficient large value of V_1 on column j without worrying about the objective value. Due to normalization, 1 is sufficiently large for this purpose and can be conveniently represented as $1-c_j$ if this case happens (i.e., $c_j = 0$). Thus, the correct constraints for the row players are

$$A^T\mathbf{p}^k + (\mathbf{1} - \mathbf{c}) \ge A^T_{:,k} - \varepsilon\mathbf{1}$$

Similarly, the linear constraints on columns are

$$B\mathbf{q}^l + (\mathbf{1} - \mathbf{r}) \ge B_{:,l} - \varepsilon\mathbf{1}$$

To sum up, the MILP is described as follows:

$$\begin{array}{lll} \min & \varepsilon & \\ s.t. & A^T\mathbf{p}^k + (\mathbf{1}-\mathbf{c}) \ge A^T_{:,k} - \varepsilon\mathbf{1} & 1 \le k \le n \\ & B\mathbf{q}^l + (\mathbf{1}-\mathbf{r}) \ge B_{:,l} - \varepsilon\mathbf{1} & 1 \le l \le m \\ & & \\ & 0 \le \mathbf{p}_i^k \le \mathbf{r}_i, 0 \le \mathbf{q}_j^l \le \mathbf{c}_j & 1 \le i,k \le n, 1 \le j,l \le m \\ & |\mathbf{p}^k|_1 = 1, |\mathbf{q}^l|_1 = 1 & 1 \le k \le n, 1 \le l \le m \\ & \mathbf{r}_i, \mathbf{c}_j \in \{0,1\} & 1 \le i \le n, 1 \le j \le m \\ & \mathbf{r}_i = 0, \mathbf{c}_j = 0 & \forall\, i \notin X_1, \forall\, j \notin X_2 \end{array} \tag{1}$$

The implementation cost to Individual K-implementation is $\varepsilon^* M$, where ε^* is the optimal value of the above program and M is largest utility in the original payoff matrices.

Theorem 4. *The MILP(1) correctly computes the optimal solution of* Individual K-implementation *in two-player, product desirable set setting.*

4.1.2 Multi-player, general desirable set

In a n-player normal-form game, player k's strategy set is $S_k = [m_k]$ and his utility in strategy profile $s = (s_k, s_{-k})$ is $A^k(s_{-k}, s_k)$, where A^k is a matrix with size $(\prod_{i \ne k} m_i) \times m_k$, and the desirable set is $O \subseteq S = \prod_1^n S_i$. Without loss of generality, suppose all utilities are positive and let M be the largest utility. Divide each utility by M to normalize the utility so that no utility is greater than 1.

- $\mathbf{g}^k \in \{0,1\}^{m_k}$ is a indicator vector that indicates whether player k's strategies are in X or not. Formally, $g_i^k = 1$ if and only if action $i \in X_k$;
- $\mathbf{p}^{k,l}$ is the probability assigned to player k's strategies for the purpose of checking dominance of player k's l-th strategy;

Similar to two-player games, constraints on $\mathbf{p}_i^{k,l}$ are,

$$0 \le \mathbf{p}_i^{k,l} \le \mathbf{g}_i^k \quad \forall k, 1 \le i, l \le m_k$$

Moreover, note that a strategy profile $s \notin X$ if and only if $\neg(\forall k, s_k \in X_k)$. Thus, the desirable set requirements can be presented by indicator vectors $\mathbf{g}^k$ as follows:

$$\sum_k \mathbf{g}^k_{s_k} \le n - 1 \quad \forall s \notin O$$

Finally, when consider player k, if $s_{-k} \notin X_{-k}$, there is no implementation value and one can set a sufficiently large value without worrying about the objective value. Due to normalization, 1 is enough and

$$\mathbf{z}^k_{s_{-k}} := \sum_{j \ne k}(1 - \mathbf{g}^j_{s_{-k_j}}) \ge 1 \quad \forall k, s_{-k} \in S_{-k}$$

if and only if $s_{-k} \notin X_{-k}$ holds[8]. To sum up, the MILP for multi-player general cases can be described as follows:

$$\begin{array}{lll} \min & \varepsilon & \\ s.t. & A^k\mathbf{p}^{k,l} + \mathbf{z}^k \ge A^k_{:,l} - \varepsilon\mathbf{1} & \forall k, 1 \le l \le m_k \\ & \mathbf{z}^k_{s_{-k}} = \sum_{j \ne k}(1 - \mathbf{g}^j_{s_{-k_j}}) & \forall k, s_{-k} \in S_{-k} \\ & 0 \le \mathbf{p}_i^{k,l} \le \mathbf{g}_i^k, \quad |\mathbf{p}^{k,l}|_1 = 1 & \forall k, 1 \le i, l \le m_k \\ & \sum_k \mathbf{g}^k_{s_k} \le n - 1 & \forall s \notin O \\ & \mathbf{g}_i^k \in \{0,1\} & \forall k, 1 \le i \le m_k \end{array} \tag{2}$$

Theorem 5. *The MILP(2) correctly computes the optimal solution of* Individual K-implementation *in multi-player, general desirable set setting.*

4.2 Supermodular games

In this subsection, we show that in finite *supermodular games* [12, 18, 19], Individual K-implementation with a consecutive product desirable set can be solved efficiently.

Definition 1. *A n-player finite game with player i's strategy set $S_i = [m]$ is supermodular if for all i, player i's utility function A^i has increasing difference, i.e., $\forall s, s' \in S$ and $s \ge s'$ (that is, $\forall j$, $s_j \ge s'_j$),*

$$A^i(s_{-i}, s_i) - A^i(s_{-i}, s'_i) \ge A^i(s'_{-i}, s_i) - A^i(s'_{-i}, s'_i).$$

We say a product outcome set $O = \prod O_i$ is consecutive if it is in the form of $O_i = \{l_i, l_i + 1, \cdots, r_i\} \subseteq S_i$. Without loss of generality, assume for all $s \in S$, $0 \le A^i(s) \le 1$.

Note that, we can compute the cost of Individual K-implementation by enumerating all $X \subseteq O$ and solving the following variation of MILP(2) to exactly implement X (that

[8]In the formula, we abuse the notation and use s_{-k_j} to represent player j's strategy in strategy profile s_{-k}.

is, for all i, any strategy in $S_i \setminus X_i$ is weakly dominated):

$$\begin{array}{lll} \min & \varepsilon & \\ s.t. & A^k \mathbf{p}^{k,l} + \mathbf{z}^k \geq A^k_{:,l} - \varepsilon \mathbf{1} & \forall k, 1 \leq l \leq m_k \\ & \mathbf{z}^k_{s_{-k}} = \sum_{j \neq k} (1 - \mathbf{g}^j_{s_{-k_j}}) & \forall k, s_{-k} \in S_{-k} \\ & 0 \leq \mathbf{p}^{k,l}_i \leq \mathbf{g}^k_i, \quad |\mathbf{p}^{k,l}|_1 = 1 & \forall k, 1 \leq i, l \leq m_k \\ & \mathbf{g}^k_i = 0 & \forall k, i \notin X_k \\ & \mathbf{g}^k_i = 1 & \forall k, i \in X_k \end{array} \tag{3}$$

Suppose the optimal solution of MILP(3) w.r.t $X \subseteq O$ is $OPT(X)$. Then, the optimal implementation value ε_* for O can be computed by $\varepsilon_* = \min_{X \subseteq O} OPT(X)$.

The next lemma states that we can compute the cost of INDIVIDUAL K-IMPLEMENTATION in finite supermodular games with a consecutive product desirable set by *only* enumerating all possible consecutive product outcome sets $X \subseteq O$.

LEMMA 6. *If $X = \prod X_i \subseteq O$ and $X' = \prod X'_i$ where $X'_i = \{\min X_i, \min X_i + 1, \cdots, \max X_i\}$, $OPT(X') \leq OPT(X)$.*

PROOF. We prove this lemma via MILP(3). Our idea is to show that if ε^* is the optimal solution for MILP(3) w.r.t X and its corresponding variables are $(\varepsilon^*, \mathbf{p}, \mathbf{z}, \mathbf{g})$, then one can construct a feasible solution $(\varepsilon^*, \mathbf{p}', \mathbf{z}', \mathbf{g}')$ of MILP(3) w.r.t X'. It immediately follows that $OPT(X') \leq OPT(X)$. The variables $\mathbf{p}'$ are constructed from $\mathbf{p}$ as follows:

- $\mathbf{p}' = \mathbf{p}$, except when $l \in X'_k$:
  ${\mathbf{p}'}^{k,l}_l = 1$ and $\forall i \neq l$, ${\mathbf{p}'}^{k,l}_i = 0$;

Notice that the indicator vectors $\mathbf{g}$ and $\mathbf{g}'$ are determined by X and X', respectively. We need to show that in our construction of $\mathbf{p}'$, the constraints $0 \leq {\mathbf{p}'}^{k,l}_i \leq {\mathbf{g}'}^k_i$ still hold. Notice that, according to our construction, $X \subseteq X'$, so we have ${\mathbf{g}'}^k_i \neq \mathbf{g}^k_i$ if and only if strategy $i \in X'_k \setminus X_k$ and thus, when ${\mathbf{g}'}^k_i \neq \mathbf{g}^k_i$, it is certain that ${\mathbf{g}'}^k_i = 1$ and $\mathbf{g}^k_i = 0$. Thus, for all k and $l \notin X'_k$, $0 \leq {\mathbf{p}'}^{k,l}_i \leq \mathbf{g}^k_i \leq {\mathbf{g}'}^k_i$. As for the case $l \in X'_k$, since ${\mathbf{g}'}^k_l = 1$, we conclude $1 = {\mathbf{p}'}^{k,l}_l \leq {\mathbf{g}'}^k_l = 1$ and $\forall i \neq l$, $\mathbf{0} = {p'}^{k,l}_i \leq {\mathbf{g}'}^k_i$.

Moreover, the vector $\mathbf{z}'$ are determined by $\mathbf{g}'$. Thus, the only remaining constraints needed to check are

$$A^k {\mathbf{p}'}^{k,l} + {\mathbf{z}'}^k \geq A^k_{:,l} - \varepsilon' \mathbf{1} \quad \forall k, 1 \leq l \leq m_k$$

Note that when $l \notin X'_k$, it does not hold only if for some s_{-k}, $0 = {\mathbf{z}'}^k_{s_{-k}} < \mathbf{z}^k_{s_{-k}}$, that is $s_{-k} \in X'_{-k}$ while $s_{-k} \notin X_{-k}$.

Suppose there exists player $k = k^*$ and strategy $l = l^*$ in which the assignments $\varepsilon' = \varepsilon^*$ and our construction of $(\mathbf{p}', \mathbf{g}', \mathbf{z}')$ violate the above constraint.

- *Case 1*: $\min X'_k \leq l^* \leq \max X'_k$. In this case, according to our construction of X', strategy $l^* \in X'_{k^*}$. Thus, strategy l^* can weakly dominate itself on its own and actually, in our construction, ${p'}^{k^*,l^*}_{l^*} = 1$ if $l^* \in X'_{k^*}$;

Intuitively, in Case 2 and 3, we argue that for player k^*, the optimal cost only depends on the weak dominance of $s^{\min}_{-k^*} = \min X_{-k^*}$ or $s^{\max}_{-k^*} = \max X_{-k^*}$.

- *Case 2*: $l^* < \min X'_{k^*}$. In this case, $l^* \notin X'_{k^*}$ and thus, according to our construction, ${\mathbf{p}'}^{k^*,l^*} = \mathbf{p}^{k^*,l^*}$. Consider $s^{\min}_{-k^*} = \min X'_{-k^*}$, i.e. ${s_{-k^*}}_j = \min X_j$. Since $(\varepsilon^*, \mathbf{p}, \mathbf{z}, \mathbf{g})$ is a feasible solution for MILP(3) w.r.t X and $s^{\min}_{-k^*} \in X_{-k^*}$, it is true that,

$$A^{k^*} \mathbf{p}^{k^*,l^*} + \mathbf{z}^{k^*} \geq A^{k^*}_{:,l^*} - \varepsilon^* \mathbf{1}.$$

Henceforth, for a specific $s^{\min}_{-k^*}$, we have

$$A^{k^*}_{s^{\min}_{-k^*},:} \mathbf{p}^{k^*,l^*} + \mathbf{z}^{k^*}_{s^{\min}_{-k^*}} \geq A^{k^*}_{s^{\min}_{-k^*},l^*} - \varepsilon^*$$

which is equivalent to

$$\sum_{i \in X_{k^*}} (A^{k^*}_{s^{\min}_{-k^*},i} - A^{k^*}_{s^{\min}_{-k^*},l^*}) \mathbf{p}^{k^*,l^*}_i \geq -\varepsilon^*$$

since $\sum_{i \in X_{k^*}} \mathbf{p}^{k^*,l^*}_i = 1$ and because of $s^{\min}_{-k^*} \in X_{-k^*}$, $\mathbf{z}^{k^*}_{s^{\min}_{-k^*}} = 0$. Notice that, what we want to prove is, for all s_{-k^*}, the following inequality holds

$$A^{k^*}_{s_{-k^*},:} \mathbf{p}^{k^*,l^*} + {\mathbf{z}'}^{k^*}_{s_{-k^*}} \geq A^{k^*}_{s_{-k^*},l^*} - \varepsilon^*$$

Recall that the constraint is violated only if ${\mathbf{z}'}^{k^*}_{s_{-k^*}} = 0$. Therefore, the above constraint is equivalent to,

$$\sum_{i \in X_{k^*}} (A^{k}_{s_{-k^*},i} - A^{k^*}_{s_{-k^*},l^*}) \mathbf{p}^{k^*,l^*}_i \geq -\varepsilon^*$$

and in additional, we have $s_{-k^*} \in X'_{-k^*}$ and thus, $s^{\min}_{-k^*} \leq s_{-k^*}$. Applying the definition of supermodularity, we can conclude this case by

$$\begin{aligned} & \sum_{i \in X_{k^*}} (A^{k^*}_{s_{-k^*},i} - A^{k^*}_{s_{-k^*},l^*}) \mathbf{p}^{k^*,l^*}_i \\ \geq & \sum_{i \in X_{k^*}} (A^{k^*}_{s^{\min}_{-k^*},i} - A^{k^*}_{s^{\min}_{-k^*},l^*}) \mathbf{p}^{k^*,l^*}_i \geq -\varepsilon^* \end{aligned}$$

- *Case 3*: $l^* > \max X'_{k^*}$. Consider $s^{\max}_{-k^*} = \max X'_{-k^*} \in X_{-k^*}$ and the proof is similar to the one in Case 2.

Therefore, our construction of $(\mathbf{p}', \mathbf{z}', \mathbf{g}')$ along with ε^* is a feasible solution to MILP(2) w.r.t X'. □

With Lemma 6, one can reduce computing the optimal implementation cost of O to computing $OPT(X)$ for all consecutive outcome sets $X \subseteq O$, which can be done in polynomial time. Note $\{X \mid X \text{ is a consecutive outcome set}, X \subseteq O\} = O(m^{2n})$, a polynomial w.r.t. the size of game $O(m^n)$.

THEOREM 6. *In supermodular games with consecutive product desirable sets, the cost of* INDIVIDUAL K-IMPLEMENTATION *can be computed in polynomial time, precisely, $O(m^{2n}n)$.*

5. EXPERIMENTS

We implement our MILP algorithm proposed in the previous section. We measure the time efficiency and optimality of our MILP algorithms, by comparing it to the algorithm implemented in [5, 6], called BRUTE-FORCE-PURE.

The BRUTE-FORCE-PURE is a double-exponential algorithm that computes general K-IMPLEMENTATION under dominance by pure-strategies, by exhaustively enumerating all possible $X \subseteq O$ (which could be exponentially many) and using an exponential algorithm to compute the cost of exact K-implementation w.r.t each X.

5.1 Experiments Setup

All our experiments are performed on a PC with 2.6 GHz Intel Core i5 processor, and 8 GB 1600 MHz DDR3 memory. We use MATLAB 2014b to serve as MILP solver. We focus on 2-player normal-form games with product desirable sets.

We implemented two versions of MILP(1) that corresponds to INDIVIDUAL K-IMPLEMENTATION problem w.r.t dominance

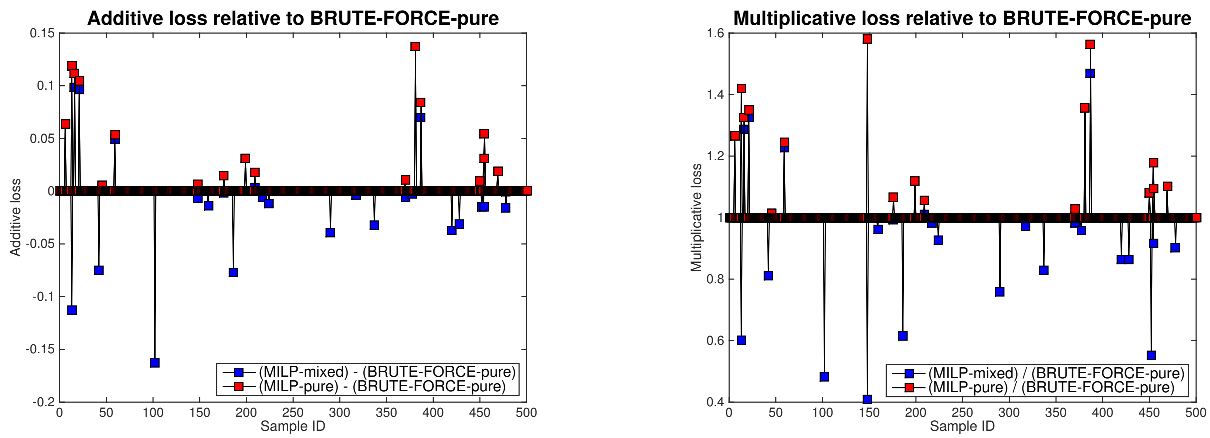

Figure 1: Additive/multiplicative losses of MILP-MIXED and MILP-PURE compared to BRUTE-FORCE-PURE

by pure strategies and mixed strategies respectively. To better approximate the original K-IMPLEMENTATION, the INDIVIDUAL K-IMPLEMENTATION we implement here is a more general, player-specific version in the sense that we impose individual implementation cost upper bound $V_i(x) \le k_i, \forall x \in X, i \in \{1,2\}$ and measure the sum of individual cost $k_1 + k_2$ as objective. The MILP for this more general problem can also be straightforwardly adjusted from MILP(1).

To measure the time efficiency of our MILP, we randomly generate 20 cases of $N \times N$ 2-player normal-form games with every utility uniformly drawn from $[0,1]$, while the probability that a strategy j is in the desirable set is $1/2$. We measure the running time as a function of N.

To measure the optimality of our MILP, due to the running time of BRUTE-FORCE-PURE, we can only repeat the experiments w.r.t a relatively small game size. In particular, we general 500 cases of 2-player normal-form games with $N = 10$ with same utility distribution as before. Moreover, we randomly generate O satisfying $1 \le |O_1|, |O_2| \le 5$. To sum up, we compare the following:

1. MILP algorithm with respect to dominance by mixed strategies, called MILP-MIXED,
2. MILP algorithm with respect to dominance by pure strategies, called MILP-PURE,
3. BRUTE-FORCE-PURE. This algorithm serves as baseline for the following loss calibrations.

5.2 Experimental Results

The average running time of our MILP according to N is shown in Table 2. For time efficiency, our MILP runs considerably fast when N is small and reasonably fast as N increases up to 70, while the base-line algorithm finished in 580s on average for $N = 9$.

Size	10	20	30	40
Time(s)	0.0879	0.3391	1.7455	9.6937
Size	50	60	70	···
Time(s)	33.4201	89.4129	216.3254	···

Table 2: Running time of MILP

As for the comparison of implementation cost, the results are shown in Figure 1 with two calibrations of loss,

- Additive: (Algorithm) - (Baseline);
- Multiplicative: (Algorithm) / (Baseline);

Moreover, the average additive loss and multiplicative loss between MILP-MIXED and BRUTE-FORCE-PURE are -5.1397×10^{-5} and 0.9975, while between MILP-PURE and BRUTE-FORCE-PURE are 0.0017 and 1.0077. One can see that, in randomly generated cases under the notion of dominance by pure strategies, optimal solution INDIVIDUAL K-IMPLEMENTATION approximates K-IMPLEMENTATION pretty well while MILP-MIXED can sometimes perform better than BRUTE-FORCE-PURE due to that dominance by mixed strategies is allowed in the MILP. By the experimental results between MILP-PURE and BRUTE-FORCE-PURE, we expect that MILP-MIXED approximates K-IMPLEMENTATION reasonably well.

6. CONCLUSION

In this paper, we pin down the correct hardness proofs to show that K-IMPLEMENTATION problem is indeed NP-complete. In addition, we study a variation of K-IMPLEMENTION problem called INDIVIDUAL K-IMPLEMENTATION problem, which provides an approximation guarantee to K-IMPLEMENTION problem and enjoys a mixed-integer linear programming for general INDIVIDUAL K-IMPLEMENTATION problem and a polynomial time runnable algorithm for supermodular games with a consecutive product desirable set. Finally, our experiments confirm that the optimal solution with respect to INDIVIDUAL K-IMPLEMENTATION problem approximates K-IMPLEMENTATION problem well in practice.

Acknowledgments

This work was supported by the National Basic Research Program of China Grant 2011CBA00300, 2011CBA00301, the Natural Science Foundation of China Grant 61033001, 61361136003, 61303077, 61561146398, a Tsinghua Initiative Scientific Research Grant and a China Youth 1000-talent program.

REFERENCES

[1] I. Ashlagi, D. Monderer, and M. Tennenholtz. Mediators in position auctions. In *Proceedings of the 8th ACM conference on Electronic commerce*, pages 279–287. ACM, 2007.

[2] E. H. Clarke. Multipart pricing of public goods. *Public Choice*, 2:19–33, 1971.

[3] V. Conitzer and T. Sandholm. Complexity of (iterated) dominance. In *Proceedings 6th ACM Conference on Electronic Commerce (EC-2005), Vancouver, BC, Canada, June 5-8, 2005*, pages 88–97, 2005.

[4] R. Eidenbenz, Y. A. Oswald, S. Schmid, and R. Wattenhofer. Manipulation in games. In *Algorithms and Computation*, pages 365–376. Springer, 2007.

[5] R. Eidenbenz, Y. A. Oswald, S. Schmid, and R. Wattenhofer. Mechanism design by creditability. In *Combinatorial Optimization and Applications*, pages 208–219. Springer, 2007.

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

[7] A. Gibbard. Manipulation of voting schemes: A general result. *Econometrica*, 41(4):587–601, July 1973.

[8] T. Groves. Incentives in Teams. *Econometrica*, 41:617–631, 1973.

[9] J. M. Kleinberg and S. Oren. Time-inconsistent planning: a computational problem in behavioral economics. In *ACM Conference on Economics and Computation, EC '14, Stanford , CA, USA, June 8-12, 2014*, pages 547–564, 2014.

[10] N. G. Mankiw, M. Weinzierl, and D. Yagan. Optimal taxation in theory and practice. Technical report, National Bureau of Economic Research, 2009.

[11] E. Maskin and T. Sjostrom. *Implementation theory*, volume 1 of *Handbook of Social Choice and Welfare*. Elsevier, March 2002.

[12] P. Milgrom and J. Roberts. Rationalizability, Learning, and Equilibrium in Games with Strategic Complementarities. *Econometrica*, 58(6):1255–77, November 1990.

[13] D. Monderer and M. Tennenholtz. K-implementation. *J. Artif. Intell. Res. (JAIR)*, 21:37–62, 2004.

[14] T. Moscibroda and S. Schmid. On mechanism design without payments for throughput maximization. In *INFOCOM 2009, IEEE*, pages 972–980. IEEE, 2009.

[15] R. B. Myerson. Optimal auction design. *Mathematics of Operations Research*, 6(1):58–73, 1981.

[16] M. A. Satterthwaite. Strategy-proofness and arrow's conditions: Existence and correspondence theorems for voting procedures and social welfare functions. *Journal of Economic Theory*, 10(2):187–217, April 1975.

[17] Y. Shoham and K. Leyton-Brown. *Multiagent Systems: Algorithmic, Game theoretic and Logical Fundations.* Cambridge Uni. Press, 2009.

[18] P. Tang and F. Lin. Two equivalence results for two-person strict games. *Games and Economic Behavior*, 71(2):479–486, 2011.

[19] D. Topkis. *Supermodularity and Complementarity.* Frontiers of Economic Research. Princeton University Press, 2011.

[20] W. Vickrey. Counterspeculation, Auctions and Competitive Sealed Tenders. *Journal of Finance*, pages 8–37, 1961.

[21] H. Zhang, Y. Chen, and D. C. Parkes. A general approach to environment design with one agent. In *IJCAI 2009*, pages 2002–2014, 2009.