Computing Equilibria in Binary Networked Public Goods Games

Sixie Yu[1]∗ and **Kai Zhou**[1]∗ and **P. Jeffrey Brantingham**[2] and **Yevgeniy Vorobeychik**[1]
[1]Department of Computer Science and Engineering, Washington University in St. Louis, {sixie.yu,zhoukai,yvorobeychik}@wustl.edu
[2]Department of Anthropology, University of California, Los Angeles, branting@ucla.edu
∗ Equal Contribution

Abstract

Public goods games study the incentives of individuals to contribute to a public good and their behaviors in equilibria. In this paper, we examine a specific type of public goods game where players are networked and each has binary actions, and focus on the algorithmic aspects of such games. First, we show that checking the existence of a pure-strategy Nash equilibrium is NP-Complete. We then identify tractable instances based on restrictions of either utility functions or of the underlying graphical structure. In certain cases, we also show that we can efficiently compute a socially optimal Nash equilibrium. Finally, we propose a heuristic approach for computing approximate equilibria in general binary networked public goods games, and experimentally demonstrate its effectiveness.

1 Introduction

Public goods games have been a major subject of inquiry as a natural way to model the tension between individual interest and social good [14, 20]. In a canonical version of such games, individuals choose the amount to invest in a public good, and the subsequent value of the good, which is determined by total investment of the community, is then shared equally by all. A number of versions of this game have been studied, including variants where players are situated on a network, with individual payoffs determined solely by the actions of their network neighbors [3].

An important variation of networked public goods games treats investment decisions as *binary* [11]. One motivating example is crime reporting, known to vary widely, with the general observation that crime is considerably underreported [19]. In a game theoretic abstraction of crime reporting, individuals choose whether or not to report crimes, and benefits accrue to the broader community, for example, causing a reduction in overall crime rate. Another example of binary public goods games is vaccination. In this example, parents decide whether to vaccinate their children, with herd immunity becoming the public good. To keep terminology general, we will refer to the binary decision whether or not to *invest* in a public good (thus, reporting a crime and vaccinating are forms of such investment).

A special case of binary public goods games (BNPGs), commonly known as *best-shot games*, has received some prior attention [7, 11, 15, 17]. However, best-shot games make a strong assumption about the structure of the player utility functions. While Levit et al. [17] recently showed that equilibria for best-shot games can be computed in polynomial time by best response dynamics, these may fail to find social welfare-optimal equilibria, and nothing is known about BNPGs more generally.

We study the problem of computing pure strategy Nash equilibria in general binary public goods games on networks. We provide examples to show that a pure strategy Nash equilibria is not guaranteed to exist for general BNPGs. Furthermore, we show that even determining if a pure strategy Nash equilibrium exists is hard in general. However, while pure strategy equilibria need not in general exist, and are hard to compute, we exhibit a number of positive results for important special cases. One class of special cases pertains to binary public goods games for completely connected networks (communities), in which case an equilibrium can be found efficiently if it exists. Similarly, we can efficiently find a pure strategy equilibrium in general BNPGs when the graph is a tree. If we further restrict the externalities to have an identical impact on all players (we call this the *homogeneous* case), we can characterize *all* pure strategy equilibria, and efficiently compute a socially optimal equilibrium. Moreover, if both investment costs and externalities are identical (termed the *fully homogeneous* case), we can characterize *all* pure strategy equilibria for arbitrary networks. Finally, we present a heuristic approach for finding approximate equilibria to tackle general BNPGs, and experimentally demonstrate that it is highly effective. Our algorithmic results are summarized in Table 1.

Related Work Our work relates to the broad literature on graphical games, a succinct representation of games in which utilities exhibit local dependences captured by a graph [13]. [9, 8, 12] studied the complexity of computing (mixed-strategy) equilibria in graphical games with utilities defined in the matrix form, and proposed efficient algorithms to find equilibria on graphs with restricted topologies.

Networked public goods games can be regarded as a special class of graphical games where the utilities are functions of the accumulated efforts of individuals. A number of model variations have been proposed to study public goods in different fields such as economics, innovation diffusion, and

medical research [10, 22, 5]. Our model is closely related to that proposed by Bramoullé and Kranton [3]. The two qualitative distinctions are that (a) we consider binary investment decisions, in contrast to Bramoullé and Kranton, who focus on the more traditional continuous investment model, and (b) Bramoullé and Kranton assume homogeneous concave utilities, while we consider a more general setting.

Other related variations of graphical games include supermodular network games [18] and best-shot games [7, 11, 15, 17]. The latter are a special case of BNPGs, and Levit et al. [17] recently showed that these are potential games with better response dynamics converging to a pure strategy equilibrium in polynomial time. No other algorithmic results are known for either of these special classes of graphical games.

	general	complete graph	tree
heterogeneous	hard	poly	poly
homogeneous	hard	poly	poly
fully-homogeneous	hard	poly	poly
fully-homogeneous + convex g	poly	poly	poly

Table 1: An overview of our results.

2 Model

A Binary Networked Public Goods game (henceforth BNPG game) is characterized by a graph $\mathcal{G} = (V, E)$, where the node set $V = \{1, 2, \cdots, n\}$ denotes the players and $E = \{(i, j)|i, j \in V\}$ represents the interdependencies among the players' payoffs, a binary strategy space $x_i \in \{0, 1\}$ for each player i (we can think of these as decisions whether to *invest in a public good*), and a collection of utility functions $U_i(x_1, \ldots, x_n)$. These utility functions have special structure, which we now describe, and we use the running example of crime reporting to illustrate the intuition behind this structure.

Let $\mathcal{N}_i = \{j \neq i|(j, i) \in E\}$ be the set of neighbors of i, and define $n_i = \sum_{j\in\mathcal{N}_i} x_j$, or equivalently, n_i is the number of i's network neighbors who choose to invest (*note the implicit dependence of n_i on the strategies of i's neighbors*). Thus, in the crime reporting scenario, n_i represents the number of i's neighbors who report crimes. We assume that each player gains whenever more people invest (report crimes), for example, because as a result crime rate drops. On the other hand, reporting crime is costly (time consuming, or even dangerous). We capture the latter feature by introducing a cost c_i which is incurred whenever i invests in a public good. Player i's utility function is then defined as

$$U_i(x_i, n_i) = g_i(x_i + n_i) - c_i x_i, \tag{1}$$

where $g_i(x_i + n_i) \geq 0$ captures the positive externality of investment choices by i's neighbors, such as the benefits to an individual when their neighbors report crimes. The only restriction we impose on $g_i(x)$ at this point is that they are *non-decreasing* functions in x (e.g., higher reporting reduces crime rate). Our definition of $g_i(\cdot)$ generalizes the one given by Bramoullé and Kranton 2007, who assume that $g_i(\cdot)$ is a twice differentiable strictly concave function. There are many scenarios these assumptions are violated. For example, studies of incentives in P2P systems entail natural models in which $g_i(\cdot)$ is an S-shaped function [4]; commonly considered best-shot games model $g_i(\cdot)$ as a step function [11]; and $g_i(\cdot)$ may be convex as in models of social unrest [6]. However, our definition of $g_i(\cdot)$ is flexible enough to model the aforementioned scenarios. Note that the size of BNPG representation is $O(n^2)$ for an arbitrary $g_i(\cdot)$. A useful observation is that player i chooses to invest if and only if $U_i(1, n_i) \geq U_i(0, n_i)$, which can be rewritten as

$$c_i \leq g_i(n_i + 1) - g_i(n_i) \triangleq \Delta g_i(n_i). \tag{2}$$

As each player has a different mapping function $g_i(\cdot)$, we henceforth call the general version of this game a *heterogeneous* BNPG game. We will study several important special cases. The first is when the externality function $g(\cdot)$ is the same for all players, which we term a *homogeneous* BNPG game. The second is a further restriction of a homogeneous game where all players have identical costs $c_i = c$; we term this a *fully homogeneous* BNPG game. The best-shot games, which received some prior attention, are a special case of homogeneous BNPGs with $g_i(x_i + n_i) = \min\{K, x_i + n_i\}$ for some integer K.

We use $(\mathcal{G}, \mathcal{U})$ to parametrize a BNPG game with underlying graph $\mathcal{G}$ and a set of utility functions $\mathcal{U} = \{U_i\}_{i\in V}$. We seek *pure-strategy* Nash Equilibria (PSNE) $\boldsymbol{x}^* = (x_1^*, \ldots, x_n^*)$ of the game. Let $\boldsymbol{x}$ denote the action profile for all players, and let $\boldsymbol{x}_{-i}$ denote the joint actions of all the players other than player i.

Definition 2.1. *In a BNPG game $(\mathcal{G}, \mathcal{U})$, a pure-strategy Nash Equilibrium is a vector $\boldsymbol{x}^* \in \{0, 1\}^n$ satisfying $U_i(x_i^*, n_i^*) \geq U_i(x_i, n_i^*)$ for any x_i, where $n_i^* = \sum_{j\in\mathcal{N}_i} x_j^*$.*

3 Heterogeneous BNPG Games

Existence and Complexity

We begin by asking basic questions about BNPG games: is a PSNE guaranteed to exist, and if not, is it hard to check whether one exists?

Existence of PSNE We first show that a PSNE need not exist *even if the network is a complete graph*. Consider a game with two players A and B. Let $U_A(x_A, x_B) = g_A(x_A + x_B) - c_A x_A$ and $U_B(x_B, x_A) = g_B(x_B + x_A) - c_B x_B$ be the utilities of A and B, respectively. A sufficient condition under which PSNE does not exist is that $U_B(0, 0) < U_B(1, 0)$ **and** $U_A(0, 1) < U_A(1, 1)$ **and** $U_B(1, 1) < U_B(0, 1)$ **and** $U_A(1, 0) < U_A(0, 0)$. This condition can be achieved by choosing $g_A(\cdot)$ and $g_B(\cdot)$ such that $\Delta g_A(0) = c_A - \epsilon$, $\Delta g_A(1) = c_A + \epsilon$, $\Delta g_B(0) = c_B + \epsilon$, and $\Delta g_B(1) = c_B - \epsilon$, where ϵ is a positive constant. A possible configuration is that $g_A(0) = \epsilon$, $g_A(1) = c_A$, $g_A(2) = 2c_A + \epsilon$, and $g_B(0) = \epsilon$, $g_B(1) = c_B + 2\epsilon$, $g_B(2) = 2c_B + \epsilon$, where $\epsilon \leq c_A$ and $\epsilon \leq c_B$, ensuring that both g_A and g_B are non-decreasing.

Complexity Next, we show that determining the existence of PSNE is hard.

Theorem 1. *Determining the existence of a pure-strategy Nash equilibrium in a BNPG game is NP-Complete.*

Proof. Given an action profile, it takes polynomial time to check if it is a PSNE, so the problem is in NP. We construct a reduction from the INDEPENDENT SET (IS) problem. Given a graph $\mathcal{G}$, the IS problem is to decide whether there is an independent set S of size at least k in $\mathcal{G}$.

Given an instance of IS (i.e., a graph $\mathcal{G} = (V', E)$ and an integer k), we construct a new graph $\mathcal{H} = (V, E)$ by adding an additional node t and connecting t to every node in $\mathcal{G}$, that is $V = V' \cup \{t\}$. We define the best-response policy of the players as the following:

$$i \in V' : x_i = \begin{cases} 1, & n_i = 0 \\ 0, & n_i \neq 0 \end{cases} \text{ and } x_t = \begin{cases} 1, & 0 < n_t < k \\ 0, & n_t = 0 \text{ or } n_t \geq k. \end{cases} \tag{3}$$

Note that by choosing different functions g_i, the best-response policy above are always realizable. A heterogeneous BNPG is defined on the graph $\mathcal{H}$, with the best-response policies defined above. We now show that finding an independent set of size at least k in graph $\mathcal{G}$ is equivalent to finding a PSNE of the game.

First, suppose we have found an independent set S of size k. Then node t is not in S since t connects to every node in $\mathcal{G}$. We check the remaining nodes in $V \setminus S$ and iteratively add into S those nodes that have no connections to any node in S, which results in $\hat{S}$. Note that $\hat{S}$ is maximal and its size is at at least k. We argue that all nodes in $\hat{S}$ choosing 1 and the rest of the nodes choosing 0 is an equilibrium. For a node in $\hat{S}$, it will not deviate since none of its neighbors chooses 1. For node t, it will not deviate since it has at least k neighbors that choose 1. For a node $u \in V \setminus \hat{S}$, it must connect to at least one node in $\hat{S}$; otherwise $\hat{S}$ is not maximal. Thus u will also not deviate.

Second, suppose we have found an equilibrium. There must be some nodes choosing 1 in the equilibrium, otherwise for any node in $\mathcal{G}$ it has no investing neighbor and it would deviate to choose 1. If the equilibrium is that all nodes choose 1, we have $n_t < k$ since t chooses 1. Because t connects to all nodes in $\mathcal{G}$, which indicates that $n_t = |V| < k$. So the equilibrium cannot be that all nodes choose 1. Suppose in the equilibrium the nodes are divided into two sets S and T where nodes in S choose 1 and nodes in T choose 0. Further note that t cannot be in S. Because another node in S would prefer to choose 0 as it connects to t. Thus, the nodes in S form an independent set. For a node u in T, it must be true that u connects to at least one node in S since u chooses 0. Furthermore, t choosing 0 means that the size of S is at least k. Thus, the set S is an independent set of $\mathcal{G}$ with size at least k. □

Since determining existence of a PSNE is hard in general BPNG games, we next consider tractable special cases.

Games on Complete Graphs Consider the case where $\mathcal{G}$ is a complete graph. Remarkably, despite the fact that a PSNE need not exist even in this special case, we now show that we can find one (if it exists) or confirm non-existence in polynomial time. Define $\alpha(k)$ and $\beta(k)$ as follows:

$$\begin{aligned} \alpha(k) &:= \left|\left\{i | c_i < \Delta g_i(k-1)\right\}\right| \\ \beta(k) &:= \left|\left\{i | c_i > \Delta g_i(k-1)\right\}\right| \\ 0 &< k \leq n, \end{aligned} \tag{4}$$

where $\alpha(k)$ (resp. $\beta(k)$) is the number of players who will always invest (resp. not invest) if the total of k players invest. The following proposition is then immediate.

Proposition 2. *There exists a PSNE if and only if there exists $0 < k \leq n$ such that $\alpha(k) \leq k \leq n - \beta(k)$.*

The condition in Prop. 2 suggests the following algorithm to compute a PSNE, or prove that one doesn't exist. When $k = 0$, there is a PSNE if and only if the set $\left\{i | c_i < \Delta g_i(0)\right\}$ is empty, which and can be checked in $O(n)$ time. When $k \neq 0$, for each integer $k = 1, \ldots, n$, check if the condition in Prop. 2 holds, which takes $O(n^2)$ time. Thus, the full algorithm has running time of $O(n^2)$.

Games on Trees Another important special case is when $\mathcal{G}$ is a tree. We now present a polynomial-time algorithm to compute a PSNE on trees (or conclude one does not exist).

Suppose an action profile $\boldsymbol{x} = (x_1, \ldots, x_n)$ is given. Consider a sub-tree denoted by $\mathcal{G}_T = (V_T, E_T)$. Observe that $\boldsymbol{x}$ is a PSNE if and only if the actions restricted to the sub-tree, namely $\{x_i^* | i \in V_T\}$, form a (local) PSNE of the BNPG played on $\mathcal{G}_T$. This indicates that a PSNE on $\mathcal{G}$ can be constructed from those (local) PSNE on sub-trees. Our algorithm, `TreePSNE`, is inspired by `TreeNash` Kearns, Littman, and Singh 2013, but unlike the latter, it computes an exact PSNE (rather than an approximate mixed Nash equilibrium) in polynomial time, as formalized by the following theorem.

Theorem 3. *When $\mathcal{G}$ is a tree, it takes $O(d_{max} \cdot |V| + |E|)$ time to compute a PSNE or conclude that one does not exist, where d_{max} is the maximum degree of $\mathcal{G}$.*

Proof. We give a constructive proof of Theorem 3. Given a tree $\mathcal{G}$, we denote its root by R (the choice of the root is arbitrary). Suppose $\mathcal{G}$ is an inverted tree, where the root R is at the bottom and the leaves at the top. The nodes in $\mathcal{G}$ are categorized into three classes: leaf nodes, internal nodes, and the root. Our algorithm consists of two passes: a *downstream* pass and an *upstream* pass. In the downstream pass we traverse the nodes in a depth-first order (start with the leaves and end at the root). Each leaf or internal node passes a table to its parent. We call the table *conditional best-response table* since it contains the best responses of a node conditioned on its parent's actions and equilibrium actions of its children.

Before presenting the algorithm, we define $n'_{Y=1}$ (resp. $n'_{Y=0}$) as the number of node Y's children that invest in a PSNE of the associated subtree given that Y invests (resp. not invests). Let W and Y be a pair of nodes, where W is the parent of Y. Denote their action profile as (x_W, x_Y). The necessary condition that (x_W, x_Y) is part of a PSNE is in Eq. (5), where the "$\diamond$" is "$\geq$" if $x_Y = 1$ and "$\leq$" otherwise.

$$\Delta g_Y(\underbrace{n'_{Y=x_Y} + x_W}_{=n_Y}) \diamond c_Y, \tag{5}$$

Suppose Y is a leaf node and W is its parent. Since Y does not have any child, n'_Y is always zero no matter what Y's decision is. Thus, computing the table is equivalent to identify the action profiles (x_W, x_Y) such that they satisfy Eq. (5), which takes $O(1)$ time since there are at most four combinations of (x_W, x_Y). If no (x_W, x_Y) satisfies Eq. (5), we conclude a PSNE does not exist for the game. The procedure to compute the conditional best-response table for a leaf node Y is summarized in Algorithm 1.

Algorithm 1 Compute conditional best-response table T_Y for a leaf node

Input: $\Delta g_Y(\cdot), c_Y$
Initialize: T_Y
for (x_W, x_Y) in $(0,0),(0,1),(1,0),(1,1)$ **do**
  if $x_Y = 0$ **then**
    If $\Delta g_Y(x_W) \leq c_Y$ **then** put (x_W, x_Y) into T_Y **end if**
  else if $x_Y = 1$ **then**
    If $\Delta g_Y(x_W) \leq c_Y$ **then** put (x_W, x_Y) into T_Y **end if**
  end if
end for
if $T_Y = \emptyset$ **then**
  Return No PSNE exists
else
  Return T_Y
end if

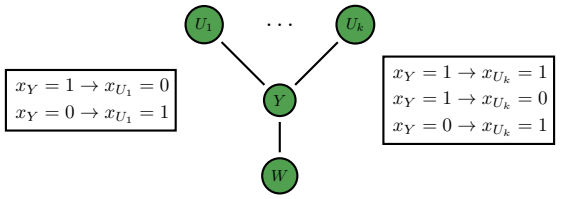

Figure 1: Conditional best-response table for internal nodes $U_1, \ldots, U_k$.

Now we consider an internal node Y. Suppose its parent is W and it has k children $U_1, \ldots, U_k$. For the purpose of induction, suppose the conditional best-response tables of $U_1, \ldots, U_k$ have been passed to Y. An example is showed in Figure 1, where the two tables beside U_1 and U_k indicate the best responses of the two nodes conditioned on their parent node Y's actions. We need to check if there is a an action profile (x_W, x_Y) satisfying Eq. (5), which requires us to compute n'_Y. Note that there may be ties, where a player is indifferent between investing and not investing given its parent's action. For example, in Figure 1 the right table indicates that node U_k is indifferent between $x_{U_k} = 1$ and $x_{U_k} = 0$, given $x_Y = 1$.

Let d_{max} be the maximum degree of G. When $x_Y = 0$, we denote the number of Y's children that always choose to invest (resp. not invest) as $n^1_{Y=0}$ (resp. $n^0_{Y=0}$). To compute $n^1_{Y=0}$ (resp. $n^0_{Y=0}$), we iterate through the conditional best-response tables of $U_1, \ldots, U_k$ and count the number of children that have unique best response $x_{U_i} = 1$ (resp. $x_{U_i} = 0$) given $x_Y = 0$. This step takes $O(d_{max})$. Note that $n^1_{Y=0} + n^0_{Y=0} \leq k$, since Y has k children. Thus, the possible values of $n'_{Y=0}$ are $\{n^1_{Y=0}, n^1_{Y=0}+1, \ldots, k - n^0_{Y=0}\}$. We obtain possible values of $n'_{Y=1}$ given $x_Y = 1$ using the similar procedure. Finally, we substitute the four combinations of (x_W, x_Y), as well as the corresponding values of n'_Y that condition on x_Y, into Eq.(5), which identifies the (x_W, x_Y) that can be part of a PSNE. If no (x_W, x_Y) satisfies Eq. (5), we conclude a PSNE does not exist for the game. The procedure to compute the conditional best-response table for an internal node Y is summarized in Algorithm 2.

Algorithm 2 Compute conditional best-response table T_Y for an internal node

Input: $\Delta g_Y(\cdot), c_Y, k$ and conditional best-response tables $T_{U_i}, \ldots, T_{U_k}$
Initialize: T_Y
Compute $n^1_{Y=0}, n^0_{Y=0}, n^1_{Y=1}, n^0_{Y=1}$
for (x_W, x_Y) in $(0,0),(0,1),(1,0),(1,1)$ **do**
  if $x_Y = 0$ **then**
    for $n'_{Y=0}$ in $\{n^1_{Y=0}, \ldots, k - n^0_{Y=0}\}$ **do**
      if $\Delta g_Y(n'_{Y=0} + x_W) \leq c_Y$ **then**
        Put (x_W, x_Y) into T_Y
      end if
    end for
  else if $x_Y = 1$ **then**
    for $n'_{Y=1}$ in $\{n^1_{Y=1}, \ldots, k - n^0_{Y=1}\}$ **do**
      if $\Delta g_Y(n'_{Y=1} + x_W) \geq c_Y$ **then**
        Put (x_W, x_Y) into T_Y
      end if
    end for
  end if
end for
if $T_Y = \emptyset$ **then**
  Return No PSNE exists
else
  Return T_Y
end if

We can compute the best response table of R in $O(d_{max})$ using a slightly modified version of Algorithm 2.; if the table is empty, we can conclude that no PSNE exists. Thus, computing the conditional best-response tables for all nodes takes $O(d_{max}|V|)$ time. In addition, it takes $O(|V| + |E|)$ to obtain the depth-first order. Thus, the downstream pass takes $O(d_{max}|V| + |E|)$ time.

In what follows we show that the downstream pass does not miss any PSNE if there is one, by an inductive argument. Suppose there is a PSNE that is missed by the downstream pass. Then we must wrongly conclude that no PSNE exists at some node Y as we move downward. First, the node Y cannot be a leaf node, since we exhaustively check the four possible action profiles of Y and its parent. Next, suppose the PSNE is not missed until an internal node Y. Since the PSNE is not missed by any child of Y, we must miss an action profile (x_Y, x_W) of Y and its parent that satisfies Eq. (5). However, we check all combinations of n'_Y, x_Y, and x_W, so we cannot miss such an action profile, which leads to a contradiction. The same argument applies for the root R.

In the upstream pass we traverse $\mathcal{G}$ in a reversed depth-first order (start at the root and end the leaves). We first select the action of the root. If both $x_R = 0$ and $x_R = 1$ are R's best responses, we arbitrarily pick one. Next, we sequentially determine each node's action based on its parent's action and the conditional best-response table passed to the parent. This pass takes $O(|V|)$ time. In conclusion, the total running time of TreePSNE is $O(d_{max}|V| + |E|)$ (see the Appendix for the detailed algorithm). □

Games on Arbitrary Graphs Finally, we present a heuristic algorithm to find approximate PSNE for general BNPG games. An approximate PSNE is defined as follow:

Definition 3.1. *In a BNPG game an ϵ-PSNE is a vector $\boldsymbol{x}^* \in \{0,1\}^n$ such that:*

$$U_i(x_i^*, n_{-i}^*) + \epsilon \geq U_i(x_i, n_{-i}^*),$$

$\forall i \in \{1, 2, \cdots, n\}$, *where* $n_i^* = \sum_{j \in \mathcal{N}_i} x_j^*$ *and* $0 \leq \epsilon \leq 1$.

The heuristic algorithm is composed of two subroutines. The first one is termed as `Asynchronous-BR`, which is based on the idea of best-response dynamics. In one execution of `Asynchronous-BR`, each player will sequentially update her action with the best-response to all other players, in an asynchronous manner. The second subroutine is termed `Evolve`, and will execute `Asynchronous-BR` k times and pick the minimum ϵ such that in the associated action profile $\boldsymbol{x}$, every player has achieved an ϵ-PSNE. In this way, `Evolve` finds the approximate PSNE with the best quality through best-response dynamics. Due to limited space, the `Asynchronous-BR` and `Evolve` are provided in the supplement. The heuristic algorithm to find an approximate PSNE is given in Algorithm 3. The input δ is the stopping criterion. When the distance d of two consecutive profiles is small enough ($< \delta$) the algorithm converges.

Algorithm 3 Heuristic to find approximate PSNE

```
Input: k, δ
Initialize: d = M                ▷ M: a large positive number
x ← Random initializatioin
if x is a PSNE then return x
while d > δ do
    x' ← Evolve(x, k)
    if x' is a PSNE then
        return x'
    else
        d ← ||x' − x||_p
    end if
end while
return x'
```

4 Homogeneous BNPG Games

Next, we consider homogeneous BNPG games where all the players share the same function g. We begin by showing that even in homogeneous BNPG games, a PSNE need not exit.

Existence of PSNE

We present a homogeneous game for which a PSNE does not exist. This game has three players $\{1, 2, 3\}$ and nodes corresponding to the players form a simple path with player 2 in the middle. Specifically, $c_1 = 1$, $c_2 = 2$, and $c_3 = 3$. We set $\Delta g(0) = 1.5$, $\Delta g(1) = 3.5$, and $\Delta g(2) = 0.5$, corresponding to a certain function g that we can select arbitrarily. There are a total of 8 possible action profiles of the game and an action profile is a PSNE if and only if a specific set of inequality constraints are satisfied. For example, the action profile $(1, 0, 1)$ is a PSNE if and only if $c_1 \leq \Delta g(0)$ **and** $c_2 > \Delta g(2)$ **and** $c_3 \leq \Delta g(0)$. By exhaustively checking all 8 profiles, it is not difficult to verify that none of the 8 sets of inequality constraints is satisfied, meaning that a PSNE does not exist in this game.

Complexity Next, we show that checking if a PSNE exists for a homogeneous BNPG is hard. The hardness result follows from Theorem 10, where we show that determining the existence of a PSNE in a special case of the homogeneous BNPG is NP-complete.

Theorem 4. *Determining the existence of a PSNE in a homogeneous BNPG game is NP-Complete.*

Next, we consider further restrictions that enable strong positive algorithmic results—far stronger than for the general BNPG games.

Games on Complete Graphs

Computing a PSNE We now show that there always exists a PSNE in homogeneous BNPG games on complete graphs, and it can be computed in time $O(n \log n)$, in contrast to the $O(n^2)$ algorithm for the general case.

Theorem 5. *Every homogeneous BNPG game on a complete graph has a PSNE, which can be computed in time $O(n \log n)$.*

Proof. Our proof is constructive. Without loss of generality, suppose $c_1 \leq c_2 \leq \cdots \leq c_n$ (we can start by sorting players in this order). If $\Delta g(0) < c_1$, we immediately obtain an equilibrium where no player invests, since $c_i > \Delta g(0)$ for every i. If $c_n \leq g(n-1)$, every player invests is an equilibrium. If neither of the two cases holds, we initialize $\boldsymbol{x}$ to all-zeros. Since $c_1 \leq \Delta g(0)$, we set $x_1 = 1$. For a subsequent player i, we set $x_i = 1$ if $c_i \leq \Delta g(n_i)$. Note that n_i in this case is the number of players (who choose to invest) with index smaller than i. We repeat this process for each player in ascending order of c_i until we found a certain player m such that $c_m > \Delta g(n_m)$. We then set the decisions of player m and her subsequent players to 0. We argue that the resulting action profile is a PSNE.

Note that in the above process, we have for player $(m-1)$, $c_{m-1} \leq \Delta g(n_{m-1}) = \Delta g(m-2)$, since n_{m-1} is the number of investing players preceding player $(m-1)$. For player m, we have $c_m > \Delta g(n_m) = \Delta g(m-1)$. Now, for any player i with $i \leq m-1$, we have $c_i \leq \Delta g(m-2)$ since $c_i \leq c_{m-1}$. Thus, player i will not deviate from her current decision, which is $x_i = 1$. For any player j with $j \geq m$, we have $c_j > \Delta g(m-1)$. That is, player j would not deviate from her current decision $x_j = 0$. The resulting action

profile is then a pure-strategy Nash equilibrium. The proof implies that the complexity of finding a PSNE is $O(n \log n)$, dominated by the time to sort players in order of c_i. □

The proof of Theorem 5 is constructive and gives an algorithm to find a PSNE. We term this algorithm `SimpleSort`. In fact, `SimpleSort` outputs an equilibrium with a special structure, where the first k players (in ascending order of c_i) will invest and the rest will not, for some k determined by the algorithm. As this equilibrium is not unique, we next characterize *all* PSNE of this game.

Characterizing all PSNE In order to characterize all equilibria, we classify these based on the number of players who choose to invest in an equilibrium. For an equilibrium $\boldsymbol{x}^*$, let k be the number of player who choose to invest. We term such an equilibrium a k-PSNE.

Let $I(\mathsf{cond}(c_i))$ be the index set of the players whose associated c_i satisfies a certain condition $\mathsf{cond}(c_i)$. Given a k-PSNE, for a player i who has chosen 1, we have $c_i \leq \Delta g(k-1)$ based on the condition in Eqn. (2). For a player i who has chosen 0, we have $c_i > \Delta g(k)$. Equivalently, if $c_i > \Delta g(k-1)$ (respectively, $c_i \leq \Delta g(k)$), we know that player i has chosen 0 (respectively, 1). Based on this, given any k, we can characterize all possible k-PSNE.

Proposition 6. *When $\Delta g(k) > \Delta g(k-1)$, a k-PSNE exists if and only if: (1) there is no c_i such that $\Delta g(k-1) < c_i \leq \Delta g(k)$ and (2) $|I(c_i \leq \Delta g(k-1))| = k$. The resulting unique k-PSNE is that all players in the set $I(c_i \leq \Delta g(k-1))$ invest.*

We introduce some notation. Let $I_+ = I(c_i \leq \Delta g(k))$, $I_\times = I(\Delta g(k) < c_i \leq \Delta g(k-1))$, and $I_- = I(c_i > \Delta g(k-1))$. Let $k_+ = |I_+|$, $k_\times = |I_\times|$, and $k_- = |I_-|$.

Proposition 7. *When $\Delta g(k) \leq \Delta g(k-1)$, k-PSNE exists if and only if $k_\times + k_+ \geq k$ and $k_+ \leq k$. Specifically, let $I_{\times+}$ be an arbitrary subset of $I_\times$ with cardinality $k - k_+$. Then a k-PSNE is that all players in the set $I_{\times+} \cup I_+$ invest.*

Proof. Consider a k-PSNE for a fixed integer k, where there are k players decided to invest. Recall from Eqn. (2) that a player i decides to report if and only if $c_i \leq \Delta g_i(n_i)$, where n_i is the number of i's neighbors who decide to invest. As the underlying graph is complete, n_i is the total number of investing players in the game excluding i. Note that for a player i who decides to invest, $n_i = k-1$ while for a player i who decides not to invest, $n_i = k$. Then it is sufficient to compare c_i associated with each player to $\Delta g(k)$ and $\Delta g(k-1)$.

Based on the investing condition in Eqn (2), we have $c_i \leq \Delta g(k-1)$ for the player i who decided to invest in a k-PSNE. For a player i who decided not to invest, we have $c_i > \Delta g(k)$. Equivalently, for an arbitrary player i, we have two observations:

- if $c_i > \Delta g(k-1)$, player i has decided to not to invest;
- if $c_i \leq \Delta g(k)$, player i has decided to invest.

Then to check the existence of k-PSNE, it is sufficient to examine the above two conditions and check whether there are exactly k players chosen to invest. We divide the discussion into two cases: $\Delta g(k) > \Delta g(k-1)$ and $\Delta g(k) \leq \Delta g(k-1)$, which corresponds to Proposition 5 and Proposition 6, respectively.

Case 1: When $\Delta g(k) > \Delta g(k-1)$, if there is a player i such that $\Delta g(k-1) < c_i \leq \Delta g(k)$, a k-PSNE does not exist. Moreover, a k-PSNE exists only if there are exactly k players whose associated c_i satisfies $c_i \leq \Delta g(k-1)$. And that k players reporting is a k-PSNE. This proves Proposition 5.

Case 2: When $\Delta g(k) \leq \Delta g(k-1)$, the two values $\Delta g(k)$ and $\Delta g(k-1)$ divide the players into three sets based on c_i: $I_+ = I(c_i \leq \Delta g(k))$, $I_\times = I(\Delta g(k) < c_i \leq \Delta g(k-1))$, and $I_- = I(c_i > \Delta g(k-1))$. Let $k_+ = |I_+|$, $k_\times = |I_\times|$, and $k_- = |I_-|$. Then if a k-PSNE exists, all players in I_+ will invest and all players in I_- will not invest. Furthermore, to ensure that there are exactly k players investing, there should be exactly $(k - k_+)$ players in the set $I_\times$ choosing to invest. Thus, a k-PSNE exists if and only if $k_\times + k_+ \geq k$ and $k_+ \leq k$. This proves Proposition 6. □

By considering all k from 0 to n, Proposition 6 and Proposition 7 can fully characterize all possible PSNE in polynomial time. We note that Proposition 7 shows that for a given k, the number of possible k-PNSE could be exponential in $k_\times$. However, we can pick one k-PNSE by arbitrarily (e.g., randomly) selecting a subset $I_{\times+}$ from I_+.

The above two propositions also characterize the structure of the equilibrium action profiles $\boldsymbol{x}^*$, which is illustrated in Fig. 2. Specifically, if we sort the players in ascending order of c_i, an equilibrium $\boldsymbol{x}^*$ is divided into three regions. The first (respectively, third) region corresponds to players who choose 1 (respectively, 0) and the second region is a mixed zone where some of the players choose 1 and others 0.

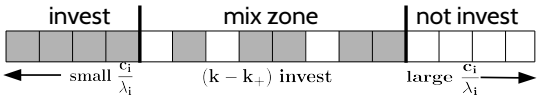

Figure 2: Structure of pure-strategy equilibria. Dark squares denote investing players.

Social Welfare Social welfare is defined as the sum of utilities of all players:

$$\mathsf{SW}(\boldsymbol{x}) \triangleq \sum_{i=1}^{n} U_i(x_i, n_i). \tag{6}$$

A common goal is to seek the PSNE with maximum social welfare—the *socially optimal* PSNE. Although the `SimpleSort` algorithm can always find a PSNE, it is not necessarily socially optimal. We now use the results in Propositions 6 and 7 to efficiently find a socially optimal PSNE.

For a specific k-PSNE, we have $x_i^* + n_i^* = k$, the same for all players. Thus, the social welfare in a k-PSNE, denoted

as $\mathsf{SW}(\boldsymbol{x}^*; k)$, can be rewritten as a linear function of $\boldsymbol{x}$:

$$\mathsf{SW}(\boldsymbol{x}^*; k) = n \cdot g(k) - \sum_{i=1}^{n} c_i x_i^*. \tag{7}$$

For a fixed k, there are two possible cases, i.e., $\Delta g(k) > \Delta g(k-1)$ or $\Delta g(k) \leq \Delta g(k-1)$. For the first case, we can efficiently compute a unique PSNE based on Proposition 6 if it exists. For the second case, Proposition 7 states that in an equilibrium, there are exactly $(k - k_+)$ players in the index set $I_\times$ choosing to invest. As $\mathsf{SW}(\boldsymbol{x}^*; k)$ is linear in $\boldsymbol{x}$, it is maximized when we pick the top-$(k - k_+)$ players in the set $I_\times$ in ascending order of c_i and set the corresponding decision variables x_i to 1. That is, for any k, we can either confirm that a k-PSNE does not exist or efficiently compute the socially optimal k-PSNE for that fixed k. Thus, by considering k from 0 to n, we can efficiently compute the socially optimal PSNE.

Fully Homogeneous Games on Arbitrary Graphs

In this section, we discuss equilibrium computation in BNPG games with arbitrary graph structure. As demonstrated in the counterexample above, a PSNE need not exist even if g is homogeneous. We now consider the question for a more restricted class of fully homogeneous BNPG games, where all players share the parameters c. Further, we restrict g to be strictly convex. In this case, we show that equilibria always exist, and all equilibria can be computed efficiently.[1] For this discussion, let D be the maximum degree in $\mathcal{G}$.

Since g is strictly convex, we have $\Delta g(0) < \Delta g(1) < \cdots < \Delta g(D)$. When $c \leq \Delta g(0)$, all players investing is a *unique* PSNE. When $c > \Delta g(0)$, there is a PSNE with no one investing; however, other equilibria might exist. To distinguish, we term the equilibria $\boldsymbol{x}^* = \mathbf{0}$ and $\boldsymbol{x}^* = \mathbf{1}$ as two *trivial* equilibria, and others where at least one player (but not all) invest are termed *non-trivial*. The non-trivial equilibria are closely-related to a k-core of a graph. Specifically, a k-core of a graph $\mathcal{G}$ is a *maximal* induced subgraph $\mathcal{H}$ of $\mathcal{G}$, where each node in $\mathcal{H}$ has degree at least k. We have the following lemma from Bickle [2] charactering the k-core of a graph.

Lemma 4.1 (Bickle [2])**.** *Given an arbitrary graph G and an integer k, if a k-core of $\mathcal{G}$ exists, it is unique.*

The following theorem characterizes the relationship between a non-trivial PSNE and a k-core of the graph $\mathcal{G}$.

Theorem 8. *In a fully-homogeneous BNPG game with strictly convex function g, let $\boldsymbol{x}$ be an action profile and $\mathcal{H}$ be the subgraph of $\mathcal{G}$ induced from the nodes corresponding to the investing players in $\boldsymbol{x}$. If $\boldsymbol{x}$ is a non-trivial PSNE, then $\mathcal{H}$ is a k-core of $\mathcal{G}$.*

Proof. If $\boldsymbol{x}^*$ is a non-trivial PSNE, then we have $\Delta g(k-1) < c \leq \Delta g(k)$ for some integer $k \in \{1, 2, \cdots, D\}$. Thus, for an arbitrary player who chooses to invest in the equilibrium $\boldsymbol{x}^*$, she must be connected to k other players who also choose to invest. This holds for any player who invests in the equilibrium. As a result, the induced subgraph $\mathcal{H}$ has minimum node degree at least k. Now, for any other node v not in $\mathcal{H}$, it represents a player who decides not to invest. By the definition of equilibrium, the v must be connected to fewer than k nodes in $\mathcal{H}$; otherwise, the corresponding player will change her action to *invest*. That is, adding v to $\mathcal{H}$ will violate the condition that each node in $\mathcal{H}$ has at least degree k, meaning that $\mathcal{H}$ is maximal. Thus, $\mathcal{H}$ is a k-core of $\mathcal{G}$. □

Note that Theorem 8 does not guarantee the existence of a non-trivial PSNE. It only states that if a non-trivial PSNE exists, the corresponding induced subgraph is a k-core of $\mathcal{G}$. Thus, it provides a guideline to find a non-trivial PSNE for a specific game – finding the k-core of the underlying graph.

Fortunately, a simple pruning algorithm can efficiently find the k-core of a given graph provided that it exists. Specifically, given a graph $\mathcal{G}$ and an integer k, the pruning algorithm will remove all the nodes that have degree less than k iteratively until no more nodes can be removed. If the remaining set of nodes is not empty, the resulting subgraph is a k-core; otherwise, a k-core does not exist. Moreover, if a k-core exists, the corresponding non-trivial PSNE is unique from Lemma 4.1. Now we are able to fully characterize the equilibria in a fully homogeneous BNPG game.

Corollary 9. *For a fully homogeneous BNPG game parameterized by (c, g), where g is strictly convex, we have:*

- *when $c \leq \Delta g(0)$, $\boldsymbol{x}^* = \mathbf{1}$ is a unique trivial PSNE;*
- *when $c > \Delta g(D)$, $\boldsymbol{x}^* = \mathbf{0}$ is a unique trivial PSNE;*
- *otherwise, there is always one trivial PSNE $\boldsymbol{x}^* = \mathbf{0}$ and possibly one non-trivial PSNE corresponding to the unique k-core of graph $\mathcal{G}$, where k satisfies $\Delta g(k-1) < c \leq \Delta g(k)$.*

The algorithm to find all possible equilibria is then straightforward. Given a fully homogeneous game, we can check if there is an integer k such that $\Delta g(k-1) < c \leq \Delta g(k)$. If it is the case, we can use the pruning algorithm to find the k-core of graph $\mathcal{G}$ (if it exists), corresponding to a non-trivial PSNE. For the other two special cases, we can find the two trivial PSNE. As there are at most two PSNE of the game, we can easily identify the socially optimal equilibrium. Finally, we show that convexity of $g(\cdot)$ is necessary to find all equilibria of this game in polynomial time.

Theorem 10. *In a fully-homogeneous BNPG game with general function g, determining the existence of a PSNE is NP-Complete.*

Proof. Given an action profile $\boldsymbol{x}$ it takes polynomial time to certify if it is a PSNE, so the problem is in NP. When $\boldsymbol{x} = \mathbf{0}$ or $\mathbf{1}$ it takes polynomial time to check if it is a PSNE. Thus, we consider the problem to determine if there is a *non-trivial* $\boldsymbol{x}$ ($\boldsymbol{x} \neq \mathbf{0}$ and $\mathbf{1}$) such that it is a PSNE. We construct a reduction from the K-CLIQUE (KC) problem. Given a graph $\mathcal{G} = (V', E')$, the KC problem asks if there is a clique of size at least k in $\mathcal{G}$. For an arbitrary instance $KC(\mathcal{G}, k)$, we construct a graph $\mathcal{H} = (V, E)$. First, we add a clique of size M to $\mathcal{H}$, where M is a large integer satisfying $M \gg |V'|$. The M nodes are denoted by a set T. For every node $i \in T$ we connect it to every node in V'.

[1] In this case, the game is a supermodular game [21, 18]. While supermodular games always have a PSNE, computing one is not necessarily efficient.

Next, we specify a homogeneous BNPG on $\mathcal{H}$. Let the players in $T \cup V'$ share the same cost c. For each player $i \in T \cup V'$ the best-response policy is

$$x_i = \begin{cases} 1, & n_i = k - 1 + M \\ 0, & n_i = k - 1 + M \\ 0, & otherwise, \end{cases} \quad (8)$$

where the player is indifferent between investing and not investing when $n_i = k - 1 + M$. The best-response policy above is realizable, for example, by a function $g(x)$ satisfying $\Delta g(x) = c$ when $x = k - 1 + M$ and $\Delta g(x) = \alpha$ when $x \neq k - 1 + M$, where $0 < \alpha < c$.

Suppose a k-clique in $\mathcal{G}$ is given. Denote the players in this k-clique by S. We claim that letting the players in $S \cup T$ invest and others not invest is a non-trivial PSNE. For a player $i \in S$, she is connected to the M investing players in T, as well as the other $k - 1$ investing players in S, which means $n_i = k - 1 + M$ and she will not deviate. Similar argument shows that for any player $t \in T$ she will not deviate because $n_t = k - 1 + M$. For a player $j \in V' \setminus S$, the number n_j of her investing neighbors falls into one of the three cases: $n_j > k - 1 + M$, $n_j = k - 1 + M$, and $n_j < k - 1 + M$, so she will not deviate.

Another direction is to show that given a non-trivial PSNE $\boldsymbol{x}$, there is a clique of size at least k in $\mathcal{G}$. First, there must be at least one investing player since the PSNE is non-trivial. Next, we argue there must be at least one investing player in T. If all the players in T were not investing, then for any player $i \in V'$ we have $n_i < k - 1 + M$, so all players in $V' \cup T$ would not invest, which contradicts the fact that $\boldsymbol{x}$ is a non-trivial PSNE. Furthermore, We claim that all players in T are investing players. Suppose the set T is partitioned into two disjoint sets $T = S_1 \cup S_0$, where players in S_1 invest while players in S_0 not. Then for any player $i \in S_1$ we have $n_i = k - 1 + M$. Since M is a large integer, S_0 must be an empty set, otherwise the player i cannot have enough investing neighbors. Due to the best-response policy in Eq. (8), there are exactly k investing players in V', which are denoted by S. For any player $i \in S$ we know $n_i = k - 1 + M$, so the player i must be connected to the other $k - 1$ players in S, which shows that S is a k-clique. □

5 Experiments

In this section we conducted experiments to show the effectiveness of Algorithm 3 for finding approximate PSNE in general BNPG games. The parameters c_i are sampled from the uniform distribution on $[0, 1]$. In our experiments we assume $g_i = \lambda_i h_i$ where λ_i is sampled uniformly at random on $[0, 1]$, and h_i are either convex or concave, corresponding to strategic substitutes and complements, respectively [11]. If $g_i(x)$ is convex, it is sampled from a set of convex functions $\{-\alpha \log(x + 1)^\beta\}$ by uniformly at random sampling α and β from $\{0.1, 0.3, 0.5, 0.7, 0.9\}$ and $\{1.2, 1.5, 2.0\}$, respectively. Similarly, if $g_i(x)$ is concave it is sampled from a set of concave functions $\{-\alpha x^\beta\}$ by sampling α and β from the same sets. We normalize the values of the utility functions U_i to $[0, 1]$ when evaluating the approximation quality of an ϵ-PSNE. In each simulated game, we feature a mix of players with concave and convex g_i. This mix is determined using a parameter $\gamma \in [0, 1]$, which is the probability that a particular player i's g_i is concave; consequently, higher γ implies a larger fraction of the population with convex utility functions. For each value of γ we simulated 1000 BNPG games. We first conduct experiments on a Facebook network [16], which is a connected graph with 4093 nodes and 88234 edges. Then, we experiment on synthetic networks to study the impact of network topologies on the fraction of players reporting in PSNE.

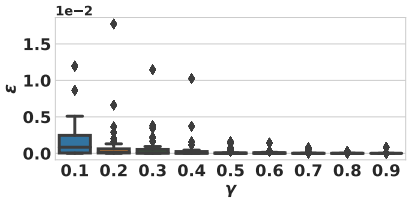

Figure 3: Averaged ϵ of the approximate PSNE (in % units).

Facebook network The experimental results are shown in Figure 3, where we vary γ. Each bar is the averaged ϵ of the approximate PSNE computed using Algorithm 3. We omit the two corner cases of $\gamma = 0$ and $\gamma = 1$ because in all of these instances our algorithm returned an *exact* PSNE. The main takeaway from these results is two-fold: first, that even when populations are mixed so that a PSNE is not guaranteed to exist, there is usually an approximate PSNE with a small ϵ (maximum benefit from deviation), and second, that our heuristic algorithm finds good approximate PSNE (observe that even outliers have $\epsilon < 0.02$).

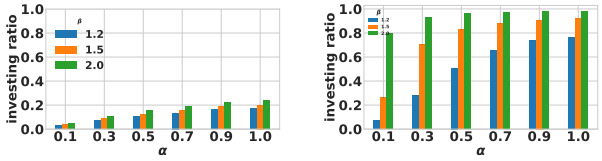

Figure 4: Ratio of investing players when $\gamma = 0$ (left) and $\gamma = 1$ (right).

Next, we study the impact of our three parameters, α, β, and γ, on the fraction of players reporting in PSNE. Figure 4 presents the results for $\gamma = 0$ (left) and $\gamma = 1$ (right) (these are the two cases where we can compute the exact PSNE, as mentioned above). We can note that in both cases, increasing α or β leads to increasing fraction of investing players. This is because higher values of either increase the rate of change of the utility function as more players invest, and consequently network effects lead to higher overall investment. The most dramatic difference, however, is between $\gamma = 0$ and $\gamma = 1$. In the former case, a relatively small fraction ultimately invest, whereas $\gamma = 1$ leads to a great deal of equilibrium investment, particularly for sufficiently large α and β. This is because convex g_i imply that as more players invest, the marginal benefits to investment increase, and therefore

equilibrium may be seen as a cascade of investment decisions that ultimately captures most of the community.

Synthetic networks We conduct experiments on two types of synthetic networks. The first type is the Barabasi-Albert network (BA) [1]. BA is characterized by its power-law degree distribution, where connectivity is heavily skewed towards high-degree nodes. The power-law degree distribution, $P(k) \sim k^{-r}$, gives the probability that a randomly selected node has k neighbors. We consider three variants of the BA network with different exponents r. We also experiment on Small-World (SW) network [23]. The Small-World network is well-known for balancing shortest path distance between pairs of nodes and local clustering in a way as to qualitatively resemble real networks. We consider three variants of the SW network, where they differ in the local clustering coefficients. The details about BA and SW networks are listed in Table 2. We use the same experimental setup as the one for Facebook network. Note that we only consider $\gamma = 0$ and $\gamma = 1$, where a PSNE can always be found by Algorithm 3. When $\gamma = 1$, a player has higher incentive to invest as more neighbors invest, while the incentive structure is the opposite when $\gamma = 0$.

	BA-1	BA-2	BA-3
exponent r	2.7167	2.2789	2.0374
	SW-1	SW-2	SW-3
local clustering coeff.	0.3667	0.3893	0.4070

Table 2: Details about the BA and SW networks.

The experimental results on BA networks are showed in Figure 5. The columns from left to right correspond to results on BA-1, BA-2, and BA-3. Note that as r decreases it is more likely to have high-degree nodes. The top row (resp. bottom row) is the case where $\gamma = 0$ (resp. $\gamma = 1$). At the bottom row, when α and β are fixed, there are more high-degree nodes as r decreases, thus the investing decisions of these high-degree nodes can encourage more nodes to invest, which results in an overall increasing trend to invest. The results at the top row have the opposite trend, where more free riders occur due to the increase of high-degree nodes, which leads to an overall decreasing trend to invest. The experimental results for SW networks are showed in Figure 6. The columns from left to right correspond to results on SW-1, SW-2, and SW-3. Note that a larger local clustering coefficient leads to denser local structures. Intuitively, when $\gamma = 1$, an investing player's decision can encourage more neighbors to invest as the local structure becomes denser. On the other hand, when $\gamma = 0$, a denser local structure can also lead to more free riders. This intuition is supported by the overall trend exhibited in Figure 6.

6 Conclusion

We study binary networked public goods games from an algorithmic perspective. We show that pure strategy equilibria may not exist even in highly restricted cases, and checking equilibrium existence is computationally hard in general. However, we exhibit a number of important special cases in which such equilibria exist, can be efficiently computed, and even where socially optimal equilibria can also be efficiently found. Finally, we presented a heuristic approach for approximately solving such games, shown to be highly effective in our experiments.

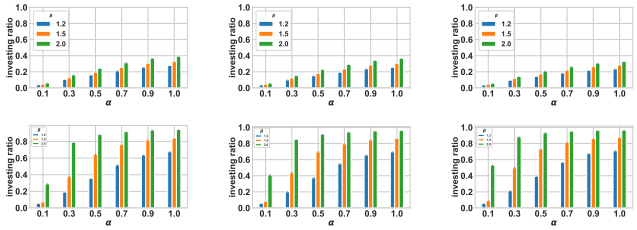

Figure 5: Ratio of investing players when $\gamma = 0$ (top row) and $\gamma = 1$ (bottom row) on BA networks. From left to right: BA-1, BA-2, and BA-3.

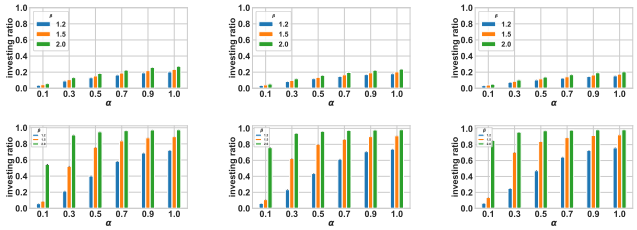

Figure 6: Ratio of investing players when $\gamma = 0$ (top row) and $\gamma = 1$ (bottom row) on Small-World networks. From left to right: SW-1, SW-2, and SW-3.

Acknowledgments

This work was partially supported by the National Science Foundation (grant IIS-1903207) and Army Research Office (MURI grant W911NF1810208).

References

[1] Barabási, A.-L. 2009. Scale-free networks: a decade and beyond. *science* 325(5939):412–413.

[2] Bickle, A. 2010. The k-cores of a graph.

[3] Bramoullé, Y., and Kranton, R. 2007. Public goods in networks. *Journal of Economic Theory* 135(1):478–494.

[4] Buragohain, C.; Agrawal, D.; and Suri, S. 2003. A game theoretic framework for incentives in p2p systems. In *Proceedings Third International Conference on Peer-to-Peer Computing (P2P2003)*, 48–56. IEEE.

[5] Burt, R. S. 1987. Social contagion and innovation: Cohesion versus structural equivalence. *American journal of Sociology* 92(6):1287–1335.

[6] Chwe, M. S.-Y. 2000. Communication and coordination in social networks. *The Review of Economic Studies* 67(1):1–16.

[7] Dallasta, L.; Pin, P.; and Ramezanpour, A. 2011. Optimal equilibria of the best shot game. *Journal of Public Economic Theory* 13(6):885–901.

[8] Daskalakis, C.; Goldberg, P. W.; and Papadimitriou, C. H. 2009. The complexity of computing a nash equilibrium. *SIAM Journal on Computing* 39(1):195–259.

[9] Elkind, E.; Goldberg, L. A.; and Goldberg, P. 2006. Nash equilibria in graphical games on trees revisited. In *Proceedings of the 7th ACM Conference on Electronic Commerce*, 100–109. ACM.

[10] Feick, L. F., and Price, L. L. 1987. The market maven: A diffuser of marketplace information. *The Journal of Marketing* 83–97.

[11] Galeotti, A.; Goyal, S.; Jackson, M. O.; Vega-Redondo, F.; and Yariv, L. 2010. Network games. *The review of economic studies* 77(1):218–244.

[12] Gottlob, G.; Greco, G.; and Scarcello, F. 2005. Pure nash equilibria: Hard and easy games. *Journal of Artificial Intelligence Research* 24:357–406.

[13] Kearns, M.; Littman, M. L.; and Singh, S. 2013. Graphical models for game theory. *arXiv preprint arXiv:1301.2281*.

[14] Kollock, P. 1998. Social dilemmas: The anatomy of cooperation. *Annual review of sociology* 24(1):183–214.

[15] Komarovsky, Z.; Levit, V.; Grinshpoun, T.; and Meisels, A. 2015. Efficient equilibria in a public goods game. In *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, 214–219.

[16] Leskovec, J., and Mcauley, J. J. 2012. Learning to discover social circles in ego networks. In *Advances in neural information processing systems*, 539–547.

[17] Levit, V.; Komarovsky, Z.; Grinshpoun, T.; and Meisels, A. 2018. Incentive-based search for efficient equilibria of the public goods game. *Artificial Intelligence* 262:142–162.

[18] Manshadi, V. H., and Johari, R. 2009. Supermodular network games. In *Communication, Control, and Computing, 2009. Allerton 2009. 47th Annual Allerton Conference on*, 1369–1376. IEEE.

[19] Morgan, R. E., and Truman, J. L. 2017. Criminal victimization. Technical report, Bureau of Justice Statistics.

[20] Santos, F. C.; Santos, M. D.; and Pacheco, J. M. 2008. Social diversity promotes the emergence of cooperation in public goods games. *Nature* 454(7201):213.

[21] Sundararajan, A. 2007. Local network effects and complex network structure. *The BE Journal of Theoretical Economics* 7(1).

[22] Valente, T. W. 1996. Network models of the diffusion of innovations. *Computational & Mathematical Organization Theory* 2(2):163–164.

[23] Watts, D. J., and Strogatz, S. H. 1998. Collective dynamics of 'small-world'networks. *nature* 393(6684):440.

Appendix

Algorithms for `Asynchronous-BR` and `Evolve`

Algorithm 4 Asynchronous-BR

Input: $\boldsymbol{x}$
for $i = 1, \cdots, n$ **do**
  if $\Delta g_i(n_i) \geq c_i$ **then**
    $\boldsymbol{x}_i = 1$
  else
    $\boldsymbol{x}_i = 0$
  end if
end for

Algorithm 5 Evolve

Input: $\boldsymbol{x}, k$
Initialize: $\epsilon' = M, \boldsymbol{x}' = \mathbf{0}$ ▷ M: a large positive number
for $i = 1, \cdots, k$ **do**
  $\epsilon \leftarrow$ MAXEPSILON($\boldsymbol{x}$) ▷ MAXEPSILON($\boldsymbol{x}$): maximum deviation of $\boldsymbol{x}$ from being a PSNE
  if $\epsilon = 0$ **then** return $\boldsymbol{x}$ **end if** ▷ Found a PSNE
  if $\epsilon < \epsilon'$ **then** $\epsilon' = \epsilon$ and $\boldsymbol{x}' = \boldsymbol{x}$ **end if**
  $\boldsymbol{x} \leftarrow$ `Asynchronous-BR`($\boldsymbol{x}$)
end for
return $\boldsymbol{x}'$

Algorithm `TreePSNE`

Algorithm 6 TreePSNE

Input: a BNPG game $(\mathcal{G}, \mathcal{U})$.
Initialize: T_R
Compute a Depth-first order $\mathcal{O}$ (start with leaves and end with the root)
for Y in $\mathcal{O}$ **do**
  if Y is the root **then**
    Compute n'_R conditioned on x_R.
    Compute the best responses of the root by a modified version of Algorithm 2.
  else if Y is a leaf node **then**
    Compute the conditonal best-response table T_Y by Algorithm 1.
    Pass T_Y to its parent
  else Y is an internal node
    Compute the conditonal best-response table T_Y by Algorithm 2.
    Pass T_Y to its parent
  end if
end for
Let $\hat{\mathcal{O}}$ be the reversed Depth-first order.
for Y in $\hat{\mathcal{O}}$ **do**
  if Y is the root **then**
    Choose an action $x_R \in T_R$.
    Determine the actions of its children by the best-responses tables passed from them.
  else if Y is a leaf node **then**
    Pass
  else Y is an internal node
    Determine the actions of its children by the best-responses tables passed from them.
  end if
end for
Return: A PSNE consists of the actions of the nodes.