Network Design for Binary Networked Public Goods Games

December 3, 2021

Abstract

Networked Public Goods (NPG) games present the scenarios where the investment (effort or money) by a player depends on the amount of investment by its neighboring players, defined using individual utility functions. Binary Networked Public Goods (BNPG) models an NPG allowing only binary choice of investing or not investing to the players, i.e., whether a player invests or not (and not the amount of investment) depends on its neighbors. The problem essentially focuses on deciding whether the prevailing situation is sustainable or not, i.e., whether the BNPG game on a given network is in a Pure Strategy Nash Equilibrium (PSNE) or not. Furthermore, Authorities or policymakers may wish to promote a certain amount (or a certain set) of players to invest. To achieve this, one may also allow adding or removing connections (edges) between two players or modifying the utility functions, or bribing certain players to invest (or not invest). We study the former problem allowing Network Modifications in terms of edge editions to the network, ensuring a PSNE on the resulting network ($\mathcal{NDDS}$). In 2020, Kempe et al. [KYV20] studied the problem of $\mathcal{NDDS}$ w.r.t to polynomial time complexity. Taking inspiration from their work on $\mathsf{NP}-\mathsf{Hardness}$ of the problem, we present results for the same in Parameterized Complexity, presenting hardness results by non-trivial reductions to already W-hard problems and XP-algorithms w.r.t parameters including (not exhaustively) the budget, diameter, treewidth and maximum degree of the input graph. We introduce another viewpoint of the problem in terms of deficiency of vertices, and establish lower and upper bounds on the optimal solution size. Furthermore, we establish W-hardness and $\mathsf{para}-\mathsf{NP}$-hardness of the problem even under homogeneity constraint, followed by FPT w.r.t. (treewidth, lower bound on investment degree sets) and vertex cover number by a reduction to EDGE-K-CORE.

1 Introduction

We face numerous real world scenarios focused toward a public good such as contributing for a private colony road construction, vaccination, sharing a common property (a tool or a generic park) to be used by public in general. In such scenarios, the amount of investment by individuals largely depends on how much their neighbors are investing, but in turn, the overall profits are shared with the entire public in general. Thus it is apt to question how to figure out the balance between individual investment and the public good. This can be achieved by modeling the above problem as a game theoretical question of *Networked Public Goods (NPG)* games. It can be observed that this can be done by setting the utility functions dependent on the neighbors of the player in the graph representation of the network. as well as his individual investment. The book on Microeconomic Theory by Collel et al. [MCWG$^+$95] and paper by Bramoulle et al. [BK$^+$07] elaborate on the concept of public good games and NPG in a much more detailed manner. Apart from the mentioned practical implications, NPG can model many other real world scenarios, as discussed in the review article of economics and statistics by PA Samuelson in 1954 [Sam54].

Even if we ease the problem to just deciding whether a player invests or not, thereby totally neglecting the amount of investment by it, the resulting problem still poses as a significant algorithmic hurdle, offering innumerable practical implications such as estimating a turnout (or even influencing) a vaccination drive or voting-spree where essentially a major influence on the individual is his neighbors. Such a variant where players are allowed a binary choice between investing or not investing in an NPG is termed as *Binary Networked Public Goods (BNPG)* games as defined by Galeotti in 2010 [GGJ+10]. The problem has been studied well algorithmically, considering polynomial, as well as parameterized complexity. Apart from this, various practical studies employ modeling BNPG games on real world scenarios. One of the most recent of which is a series of two concurrent reports by Buchwald [Buc20, Won20] highlighting the impact of communities on mask-wearing practices by individuals.

Looking from the perspective of a policymaker or an administrator, BNPG offers very little or no power in their hands. It may happen that only a few diligent players invest, whereas the rest just abstain from investing, benefiting from the collective investment from the former group of diligent investors. This *Bystander Effect* might not be sustainable over the long term and as soon as the small subset of players investing realize that they are being exploited, they may as well withdraw their support. This motivates policymakers to devise a central mechanism introducing *modifications in the network* (typically constrained by a fixed budget) by introducing new (diligent) neighbors together or separating out the lethargic ones or even bribing a few to invest or not invest. In fact, game theoretical models [Rou07] such as auctions or mechanism designs often employ such incentivization to the manipulate the outputs. Through such manipulations, the central mechanism can force a particular number or set of individuals to invest for public good. The problem finds its practical application in modeling security applications similar to the study Hota et al. [HS16], which characterizes the expected risk of a node to be attacked by the amount of (1) its own security strength (investment), (2) and the strength (investment) of its neighbors. In fact, the notion of network modifications finds its applications in a wide range of practical implications such as maximizing the spread of cascades [SDE+12], manipulating opinion diffusion in social networks [BE17], election control in social networks [CFG20], and many more.

The aforementioned problem resulting from the confluence of finding a PSNE for BNPG and Network Modifications restricted to edge editions is formally termed as *Network Design for Degree Sets ($\mathcal{N}$DDS)*, first defined and studied algorithmically until very recently in 2020 by Kempe et al. [KYV20]. They essentially study the problem, considering variants w.r.t to the targeted investing set (discussed in detail in Section 6.1).

Our work considers the problem of $\mathcal{N}$DDS, as defined by Kempe, with the intending to study it w.r.t parameterized complexity. The motivation behind the same builds upon the fact that most of the variants considered by Kempe turn out to be NP-hard, thereby opening it to be studied w.r.t finer algorithmic approaches like approximation and parameterized algorithms. Intuitively, the input consists of an edge-weighted network, individual utility functions, and a budget for modifications with the goal to decide whether there exists a sequence of edge editions (additions or deletions); under the constraint of a given budget; such that the final modified network has a PSNE w.r.t to the required class (e.g. of classes include all players investing, or a certain input set S of players invests or at least r players invest). It should be pointed out that, although any player's utility is always non-decreasing w.r.t the increase in the number of neighbors, the functions capturing this non-decreasing behavior may vary. This motivates us to classify the problem into variants further depending on the type of this function (e.g., convex, concave, sigmoid, or general function).

We present Parameterized Hardness results by performing non-trivial reductions to already W-hard problems (such as *Edge Clique Cover*, EDGE-K-CORE) and design XP-algorithms w.r.t parameters including (not exhaustively) the budget, diameter, treewidth and maximum degree of the input graph. An overview of our results is depicted in Table 1.

2 Prior Work

The problems on network design have been discussed in various Graph Theory and Algorithm Design books [Kle07, IKMW07, Mor00] over time. *Networked Public Goods (NPG) games* have been defined in studies including [GGJ+10] [MCWG+95] [LKGM18]. The first algorithmic study on BNPG games was conducted by Yu et al. [YZBV20] in early 2020, focusing on polynomial complexity of the decision version of the problem and establishing NP-Completeness of the same. Building upon this, Maiti et al. [MD20] in late 2020 extended the results of deciding on PSNE of BNPG to parameterized complexity. Although, the works discussed so far did not consider any network modifications on the input graph. Moreover, the later models cannot generalize in the sense that they do not consider various PSNE classes putting an additional constraint to be satisfied ,such as all players invest or a certain input set S of players invests or at least r players invest.

Galeotti et al. [GGJ+10] studied the effects of network modifications on the welfare of NPG games from a much more economic perspective and is different from our focus purely algorithmic focus. Bramouelle [BK+07], [BKD14] presented one of the initial studies considering Network Modifications on NPG games, studying the variants with specific utility functions, i.e., concave, convex and a combination of both, i.e., sigmoid function. They also established a link between the PSNE and minimum eigenvalue of the input graph's graph-adjacency. However, not until very recently, in 2020, Kempe et al. [KYV20] defined and initiated a study on algorithmic effects of Network Modifications restricted to edge editions on BNPG games. Based on the class and type of utility functions, Kempe established polynomial time tractability of a few problem variants using a reduction of a perfect matching problem. The author also proved NP-Completeness for the rest of the other variants, thereby inspiring our work on the same in parameterized complexity.

Deviating from the problem of finding PSNE, there are quite a few possible variants of network modifications on networks. Whereas our approach to network design focuses more on equilibria in games played over the network, other problems in network design more eccentric around optimizing path, flow, or diffusion properties in the network have been worked upon in the past. Notable works are on Maximizing the Spread of Cascades by Sheldon et al. [SDE+12], Manipulating Opinion Diffusion in social networks by Bredereck et al. [BE17], Election Control in social networks [CFG20]. Another problem along a similar line is by Sless et al. [SHKW14] ,which works around forming coalitions and facilitating relationships for completing tasks in social networks.

3 Objective

Network modification in networks is an essential component of the analysis of graph theoretic problems. Inducing Nash Equilibrium in games using modifications in networks is an indispensable part of mechanism design in algorithmic game theory as well as economics. Analyzing the problem of network modification on finding Pure Strategy Nash Equilibrium (PSNE) of Binary Networked Public Goods (BNPG) games, succors the hold of a policymaker on manipulating individuals. The problem of finding PSNE on BNPG without any modifications has already been studied in [MD20]. Encouraged by polynomial hardness results from [KYV20], we study the plausibility of network modifications on finding PSNE of BNPG games adopting the lens of Parameterized Complexity.

4 Significance of Our Work

Appropriateness of our work is backed up with the following supporting arguments:

1. The tool parameterized complexity is a relative neonate in the field of design and analysis of algorithms and has turned out to be a triumphant approach while solving NP-Hard problems. It aids in understanding the problem structure and identifying the hindrances using considered parameters, rendering significant results theoretically as well as in real-world scenarios.

2. We study Network modification for inducing PSNE in BNPG games to understand the competence of a central mechanism to affect individual strategies of players inducing a required joint strategy for the public good. Over the problem of finding PSNE of BNPG games (without network modifications), this problem introduces more degrees of freedom in terms of edge editions, thereby increasing the complexity of the problem and at the same time allowing a more generalized perspective of BNPG games. In terms of practical significance, the problem helps us analyze the potency of manipulation in the hands of policymakers over a particular network (such as electoral network or a vaccination drive, etc.)

3. Polynomial hardness results on Network Modifications [KYV20] and parameterized hardness results on the problem of finding PSNE without any modifications [MD20], lays ground to a problem lying in confluence of the above two, i.e., a parameterized analysis of finding PSNE on BNPG games allowing network modifications.

5 Our Contribution

We study the parameterized complexity of the $\mathcal{NDDS}$problem (defined in detail in Section 6.1) with respect to various natural parameters such as the budget, diameter, treewidth and maximum degree of the input graph. In particular, we first consider the natural input parameter, i.e., the budget (k) of the problem and establish $W[1]-Hardness$ for the general, concave, convex and sigmoid variants (as defined later in Section 6.1) on the class $[\geqslant r]$ of PSNE. With the further aim to ease the problem, we introduce an additional parameter as the input value r, resulting in the parameter of $k+r$. With respect to this parameter, we again establish $W[1]-Hardness$ for the considered variants. We then define a structural parameter of α (defined Section 6.2) intuitively dependent on the greed or deficit in a player's neighbors to make him/her invest. With an (FPT) reduction from already $W[1]-Hard$ problem of EDGE-K-CORE to our problem, we establish that the problem is hard even w.r.t $k+r+\alpha$. For the sigmoid variant, we further make the problem intractable in parameterized complexity establishing para-NP-hard w.r.t parameter r. By a reduction from results of Maiti et al. for BNPG (without modifications), we establish W-Hardness considering the number of players investing $(|I|)$, $n-|I|$, treewidth, maximum degree (Δ) and diameter of the input graph, number of distinct utility functions $(\delta,\ n_U)$.

Following from the non-existence of an FPT algorithm because of W[1]-Hardness of most variants, we devise XP algorithms for w.r.t k and r respectively for all and convex variants. Following this, we consider the *homogeneous* variant of the problem. We first establish the notion of deficiency in the input graph or its subgraphs, formulating the problem statement of

$\mathcal{N}$DDS$^\alpha$(convex, $\geqslant r^*$) in a second perspective. Then we establish the bounds on the number of edges to be added corresponding to the optimal solution for general graphs. We then establish a reduction to EDGE-K-CORE. This essentially, results in strictly bounding the number of edges in the optimal solution for the case of forests. Morever, using the same reduction, we extend the results of EDGE-K-CORE to obtain an FPT for $\mathcal{N}$DDS$^\alpha$(convex, $\geqslant r^*$) w.r.t parameters $(tw + \alpha)$. We also obtain an FPT for $\mathcal{N}$DDS$^\alpha$(convex, $\geqslant r^*$) w.r.t. VC as parameter, and complement this result by ruling out the existence of a polynomial kernel using the reduction to EDGE-K-CORE.

We summarize our results in Table 1.

Problem Variant	**Parameter**	**Result**
all, general	k (budget)	W[1]-Complete Theorem 3
$\{= S, \supseteq S, \geqslant r\}$, general	k	W[1]-Complete Corollary 1
$\{\supseteq S, \geqslant r\}$, concave	k	W[1]-Complete Theorem 4
$\{\supseteq S, \geqslant r\}$, sigmoid	k	W[1]-Complete Corollary 2
$\geqslant r$, {concave, convex, sigmoid}	$r + k$	W[1]-Complete Theorem 5
$\geqslant r$, convex	$k + r + \alpha$	W[1]-Hard Theorem 8
$\geqslant r$, sigmoid	$r + k$	para-NP-hard Section 7
$\{\geqslant r, \supseteq S\}$, general	$\|I\|$	W[2]-Hard Observation 2
$\{\geqslant r, \supseteq S\}$, general	$n - \|I\|$	W[2]-Hard Observation 2
$\{\geqslant r, \supseteq S\}$, general	treewidth	W[1]-Hard Observation 3
$\{\geqslant r, \supseteq S\}$, general	Δ	para-NP-hard Observation 4
$\{\geqslant r, \supseteq S\}$, general	(δ, n_u)	para-NP-hard Observation 6, 5
$\{-any-, -any-\}$, -any-	k	$n^{O(k)}$ XP Theorem 10
Homogeneous Variant: $\mathcal{N}$DDS$^\alpha$		
$\geqslant r$, {convex, sigmoid, general}	$k + r + \alpha$	W[1]-Hard Corollary 3
$\geqslant r$, {convex, sigmoid, general}	$r + k$	para-NP-hard Corollary 4
$c = \left\lceil \frac{1}{2} \sum_{v \in V(H)} df(v) \right\rceil$		$k \in [c, 2c]$ Theorem 11
$\mathcal{N}$DDS$^\alpha$(convex, $\geqslant r$) $\leqslant_{FPT}$ EDGE-K-CORE		Theorem 12
Forests: $\geqslant r$, convex	α	$O(\alpha n^2)$ Observation 8
$\geqslant r$, convex	VC	FPT Observation 9
$\geqslant r$, convex	$tw + \alpha$	FPT Observation 10

Table 1: Summary of results.

6 Preliminaries

Our work explores the considered problem using a set of tools in Algorithms called Parameterized Complexity. It is a relatively new field in the Analysis of Algorithms and has already rendered FPT algorithms for most of the hard problems including NP-Hard as well as APX-Hard problems. Thus we define the necessary terminology required for our work.

Definition 1 (Parameterized Problem). *[CFK+15] A Parameterized problem is a language $L \subseteq \Sigma^* \times \mathbb{N}$, where Σ is a fixed, finite alphabet. For instance, $(x, k) \in \Sigma^* \times \mathbb{N}$, k is called the parameter.*

This prompts us to think of possible algorithms and running times for the Parameterized Problems. We first define algorithms with running time $f(k)n^{O(1)}$, termed as fixed-parameter algorithms, or FPT algorithms. Formally:

Definition 2 (Fixed Parameter Tractable(FPT) algorithms). *[CFK+15] A Parameterized problem* $L \subset \Sigma^* \times N$ *is called fixed parameter tractable* (FPT) *if there exists an algorithm* A *(called a fixed parameter algorithm), a computable function* $f : N \to N$*, and a constant* c *such that, given* $(x, k) \in \Sigma^* \times N$*, the algorithm* A *correctly decides whether* $(x, k) \in L$ *in time bounded by* $f(k).|(x, k)|^c$*. The complexity class containing all fixed-parameter tractable problems is called* FPT.

Typically the goal in Parameterized algorithmics is to design FPT algorithms, trying to make both the $f(k)$ factor and the constant c, which is the power of n in running time; in the bound on the running time as small as possible. We further define another complexity with the power of n as a function of the input parameter as well as follows:

Definition 3 (Slice-wise polynomial (XP) algorithms). *[CFK+15] A Parameterized problem* $L \subset \Sigma^* \times N$ *is called slice-wise polynomial (XP) if there exists an algorithm* A *and two computable function* $f, g : N \to N$ *and given* $(x, k) \in \Sigma^* \times N$*, the algorithm* A *correctly decides whether* $(x, k) \in L$ *in time bounded by* $f(k).|(x, k)|^{g(k)}$*. The complexity class containing all slice-wise polynomial problems is called XP.*

FPT algorithms can be put in contrast with less efficient XP algorithms (for slice-wise polynomial), where the running time is of the form $f(k)n^{g(k)}$, for some computable functions f, g. It should be noted that there is a tremendous difference in the running times $f(k)n^{g(k)}$ and $f(k)n^{O(1)}$ $(f(k)n^c)$. In Parameterized algorithms, k is simply a relevant secondary measurement that encapsulates some aspect of the input instance, be it the size of the solution sought after or a number describing how "structured" the input instance is.

Another tool employed for designing FPT algorithms is *Bounded search trees* or simply *branching*. The technique of branching is well known in the other fields of design and analysis of algorithms. It provides us with one of the simplest and most commonly used techniques in Parameterized complexity that is widely used and takes its origin from the idea of backtracking. A Bounded Search Tree tries to build a feasible solution to the problem by making a sequence of decisions on its branching at the considered node, deciding whether to include some vertex or edge into the solution or not. Thus it can be considered as a search tree, which is traversed by the algorithm until we reach an optimal solution (maybe a feasible solution) at least one of the leaf nodes of the bounded search tree. In fact backtracking the tree in a bottom-up manner from the concerned leaf to the root can give us the solution set corresponding to the path. The running time is limited by (1) limiting the individual running time of each node of the tree (2) by bounding the number of branches at the nodes (3) by limiting the depth of the tree.

Now we define another set of key tools in parameterized complexity called Kernelization. We used a preprocessing subroutine or simply a reduction rule to reduce the size or clean the input. The goal of such a reduction rule is to act as a preprocessing subroutine to be able to solve the "easy parts" of a problem instance efficiently and reduce it (shrink it) to its computationally difficult "core" structure. We now formally define the terms introduced in these terms:

Definition 4 (Reduction Rule). *[CFK+15] A data reduction rule, or simply, reduction rule, for a parameterized problem* Q *is a function* $\psi : \Sigma^* \times N \to \Sigma^* \times N$ *that maps an instance* (I, k) *of* Q *to an equivalent instance* (I', k') *of* Q *such that* ψ *is computable in time polynomial in* $|I|$ *and* k*.*

Definition 5 (Kernelization, kernel). *[CFK+15] Given parameterized problem* Q*, an algorithm A is called a kernelization algorithm (or a kernel) if given an instance* (I, k) *of Q returns an equivalent instance* (I, k) *of* Q *of a size upper bounded by* $A(k) \leqslant g(k)$ *for some computable function* $g : N \to N$ *while running in a polynomial time.*

Now we define the notion of $\mathrm{para}-\mathsf{NP}$-$\mathrm{hardness}$, which we will be using for our proofs. The notion of $\mathrm{para}-\mathsf{NP}$-$\mathrm{hardness}$ states that the problem is NP-hard for a given constant value or "piece" of parameter. For instance, graph coloring is $\mathrm{para}-\mathsf{NP}$-hard; considering the parameter as the number of colors allowed; as it is NP-hard for three colors (3-Colorabity of graphs is a famous result itself).

Definition 6 (para-NP)**.** *[FG06] Para-NP is the class of parameterized problems that a non-deterministic algorithm can solve in time* $f(k).|x|^{O(1)}$ *where* f *is a computable function.*

Henceforth, given an input parameter, if the considered problem is NP-Hard for a constant assignment to the parameter, then it is *para-NP-Hard*. Another equivalent of $\mathsf{P} \neq \mathsf{NP}$ conjecture in parameterized complexity is $\mathsf{FPT} \neq \mathrm{para}{-}\mathsf{NP}$, and it has been proven that $\mathsf{FPT} = \mathrm{para}{-}\mathsf{NP}$ iff $\mathsf{P} = \mathsf{NP}$. This again extends to the fact that para-NP-hard problems cannot belong to XP unless until the conjecture $\mathsf{P} \neq \mathsf{NP}$ fails.

For proving the hardness of problems in parameterized complexity, we first need to explain the notion of reduction in the same. It is given as follows:

Definition 7 (Parameterized reduction)**.** *[CFK+15] Given two parameterized problems* $A, B \subseteq \Sigma^* \times \mathbb{N}$. *A reduction from* A *to* B*, denoted by* $A \leqslant_{\mathrm{param}} B$ *or simply as* $A \leqslant B$ *if the context is clear is a parameterized reduction, defined as an algorithm/oracle which takes in an instance* $I(x, k)$ *of* A*, returns an instance* $I'(x', k')$ *of* B*such that the following conditions are satisfied:*

1. $I(x, k)$ *is a Yes-instance of* A *iff* $I'(x', k')$ *is a Yes-instance of* B*,*
2. $k' \leqslant g(k)$ *where* $g(.)$ *is a computable function*
3. *the algorithm/oracle runs in* FPT *time, i.e., it runs in time* $f(k).|x|^{O(1)}$*, where* $f(.)$ *is a computable function.*

It is followed by the fact that :

Theorem 1. *[CFK+15, Theorem 13.2, 13.3] If there is a parameterized reduction from* A *to* B *and* B *is* FPT*, then* A *is* FPT *as well. Also, it is transitive in nature, i.e., if there are parameterized reductions from* A *to* B *and* B *to* C*, then there is a parameterized reduction from* A *to* C*.*

We now borrow the notion of $W-\mathrm{Hierarchy}$ from [CFK+15]. Not to extend the explanations more, we refer the reader to [CFK+15] for the basic notion of *Boolean Circuits*, the *weft of a circuit* and the definition of the *Weighted Circuit Satisfiability (WCS) problem*. $\mathsf{WCS}[\mathcal{C}]$ is defined as the restriction of the problem where the input circuit C of WCS problem belongs to the given class of circuits $\mathcal{C}$. The maximum number of large nodes on a path from an input node to the output node of the circuit is defined as *weft of the circuit*. The class of circuits with weft at most t and depth at most d is denoted as $\mathcal{C}_{t,d}$.

Definition 8 (W-hierarchy)**.** *[CFK+15, Definition 13.16] For* $t \geqslant 1$*, given a parameterized problem* P*, it is said to belong to class* $W[t]$ *if there is a parameterized reduction from* P *to* $\mathsf{WCS}[\mathcal{C}_{t,d}]$ *for some* $d \geqslant 1$*. Furthermore, if every problem in* $W[t]$ *can be reduced to* P *implies that* P *is* $W[t]-\mathrm{hard}$*.*

For instance, the problems like Weighted t-normalized Satisfiability, Weighted Monotone t-normalized Satisfiability, Weighted Monotone (t + 1)-normalized Satisfiability are $W[t]-\mathrm{Complete}$, for every even $t \geqslant 2$.

We greatly exploit the graph theoretical interpretation of the problem for proving the results. Thus we borrow basic graph theory terminologies as is. Formally, A directed graph $\mathcal{G}$ is a tuple

$(\mathcal{V}, \mathcal{E})$ where $\mathcal{E} \subseteq \{(x, y) : x, y \in \mathcal{V}, x \neq y\}$. For a graph $\mathcal{G}$, we denote its set of vertices by $\mathcal{V}[\mathcal{G}]$, its set of edges by $\mathcal{E}[\mathcal{G}]$, the number of vertices by n, and the number of edges by m. Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, a sub-graph $\mathcal{H} = (\mathcal{V}', \mathcal{E}')$ is a graph such that (i) $\mathcal{V}' \subseteq \mathcal{V}$, (ii) $\mathcal{E}' \subseteq \mathcal{E}$, and (iii) for every $(x, y) \in \mathcal{E}'$, we have $x, y \in \mathcal{V}'$. A sub-graph $\mathcal{H}$ of a graph $\mathcal{G}$ is called a *spanning sub-graph* if $\mathcal{V}[\mathcal{H}] = \mathcal{V}[\mathcal{G}]$ and *induced subgraph* if $\mathcal{E}[\mathcal{H}] = \{(x, y) \in \mathcal{E}[\mathcal{G}] : x, y \in \mathcal{V}[\mathcal{H}]\}$. Given an induced path P of a graph, we define an *end vertex* as a vertex with 0 outdegree in P and *start vertex* as a vertex with 0 indegree in P.

Almost all the parameters considered for solving the problem are self-explanatory, except the parameter of treewidth. The *treewidth* of a graph is one of the most immensely employed tools in parameterized algorithms nowadays. Intuitively, treewidth measures how close the given graph is to a tree. Smaller treewidth suggests the existence of a structural decomposition of the graph into pieces of bounded size connected in a tree-like fashion, thereby allowing one to analyze the problem with typical tree algorithms such as Dynamic Programming. A tree has a treewidth of 1, whereas a clique or a complete graph has treewidth of $n - 1$ and for a complete bipartite graph $K_{m,n}$ treewidth is $\min\{m, n\} - 1$. Formally we define a tree decomposition and subsequently the treewidth of a graph as follows:

Definition 9 (Tree-Decomposition, Treewidth). *[CFK+15] Tree decomposition (may not be unique) of a graph G is a pair $\mathcal{T} = (T, \{X_t\}_{t \in V(T)})$, where T is a tree whose every node t is assigned a vertex subset $X_t \subseteq V(G)$, called a bag, such that the following three conditions hold:*

1. $\cup_{t \in V(T)} X_t = V(G)$. *Ensuring that each vertex of G is in some bag.*

2. *For every $uv \in E(G)$, there exists a node t of T such that bag X_t contains both u and v.*

3. *For every $u \in V(G)$, the set $T_u = \{t \in V(T) : u \in X_t\}$, i.e., the set of nodes whose corresponding bags contain u, induces a connected subtree of T.*

Treewidth is of G is defined as the minimum of all the widths of all possible tree decompositions $\mathcal{T} = (T, \{X_t\}_{t \in V(T)})$ where the width of $\mathcal{T}$ refers to the maximum size bag from all the bags minus one, i.e., $\max_{t \in V(T)} |X_t| - 1$.

Using the concept of treewidth, FPT algorithms have been developed for otherwise hard problems, including Weighted Independent Set, Dominating Set, Steiner Tree, Subgraph Isomorphism and Minimum Bisection.

6.1 Problem Definition

In this section, we formally define our problem, which is adapted from Kempe et al. [KYV20]. We begin with defining a BNPG game.

In a binary networked public goods (BNPG) game, we are given the following as a part of the input instance:

1. An undirected, simple graph $G(V, E)$, where $V[G]$ represents n players and $E[G]$ represents m dependencies between pairs of players;

2. A binary strategy space $\{0, 1\}$ for each player i where an individual strategy of 1 means investing by the corresponding player, whereas a strategy of 0 implies that the corresponding player does not invest. Let us denote by x_i the strategy played by i^{th} player and the joint pure (we do not employ the game theoretic concept of mixed strategies for this variant) profile of all players as $x = (x_1, ..., x_n)$.

3. Utility function $U_i(x)$ of each player is defined as follows:

$$
\begin{aligned}
&U_i(x) = U_i(x_i, n_i^x) = g_i(x_i + n_i^x) - c_i x_i \\
&where: \\
&\quad n_i^x = \{j \mid (j,i) \in E[G] \text{ and } x_j = 1\} \\
&\quad g_i() := non-negative, non-decreasing\ function
\end{aligned}
$$

We may interchangeable use the terms strategy and action at times.

Definition 10 (PSNE of BNPG). *[KYV20] A Pure Strategy Nash Equilibrium (PSNE) of a given BNPG game is defined a joint pure strategy profile* $x \in \{0,1\}^n$ *such that* $U_i(x_i, n_i^x) > U_i(1-x_i, n_i^x)$, *or* $U_i(x_i, n_i^x) = U_i(1-x_i, n_i^x)$ *and* $x_i = 1$, *for every player* i. *Thereby breaking ties in favor of investing whenever applicable.*

The uniqueness of PSNE for a game may not be there, implying that a BNPG game may not have a single unique joint pure strategy acting as a PSNE. We now introduce the concept of classes of these multiple PSNE profiles, depending on the players investing in a given profile. Particularly, the aim to perform edge editions to the input graph such that out of all possible PSNE profiles, there is at least one profile that lies in the given class X (the class is given as a part of input instance). For notational convenience, we define X as a set of strategy vectors $x \in \{0,1\}^n$ and $X_b = \{i \mid x_i = b\} \quad \forall b \in \{0,1\}$ and use them interchangeably depending on the context. We classify a PSNE into the following classes (note that these classes need not be disjoint of each other):

▷ all: Every player invests, i.e., $X = \{\{1,2,\ldots,n\}\}$.

▷ $= S$: Exactly a given set S of players invests (and the other players do not), i.e., $X_1 = \{S\}$. All players investing is the special case $S = \{1,\ldots,n\}$.

▷ $\supseteq S$: At least the set S of players invests; other players may also invest. Here, $X_1 = \{T \mid T \supseteq S\}$.

▷ $\geqslant r$: At least r players invest. Here, $X_1 = \{T \mid |T| \geqslant r\}$

Definition 11 (Network Design for BNPG). *[KYV20] Given a BNPG instance* $G(V,E)$ *, edge costs* $\{\gamma_{e\in\binom{n}{2}}\}$ *, desired PSNE class* X*, and budget* k*, find an edge set* S *with* $\sum_{e\in S\ominus E}\gamma_e \leqslant k$ *such that the BNPG game on* $G'(V, E' = S \ominus E)$ *has at least one PSNE in* X*.*

Now, we propose another classification of the problem of finding PSNE of BNPG w.r.t the variations in properties of g_i (which is a part of U_i). We partition it into four types (1)concave, (2)convex, (3)sigmoid, or (4) General, for all players $i \in [n]$, where the names of classes are self-explanatory for the types of functions contained in them.

We adopt a crucial observation from Kempe's paper, which helps characterize the information carried by the utility function of i^{th} player. We defined *Investment Degree Set* for player i, denoted by D_i, to be the set of numbers of neighbors of player i that are investing and that would force player i to invest as well. From a result by Kempe, the structure of D_i links directly to the type of function $g_i(.)$ as follows:

Theorem 2. *[KYV20, Lemma 2.3] For every non-decreasing function* $g_i : [0, n-1] \to R_+$ *and cost* c_i *, there exists a unique set* $D_i \subseteq \{0,1,\ldots,n-1\}$ *such that* $x_i = 1$ *is a best response to* n_i^x *(or simply* n_i*) if and only if* $n_i \in D_i$ *. Furthermore,*

▷ *If* g_i *is concave, then* D_i *is a downward-closed interval.*

▷ *If g_i is convex, then D_i is an upward-closed interval.*

▷ *If g_i is sigmoid, then D_i is an interval.*

The converse of these statements also holds.

We now define the problem of finding PSNE belonging to a particular class X in terms of investment degree sets as follows:

Definition 12 (Network Design for Degree Sets (NDDS))**.** *[KYV20, Definition 2.4] The problem* $\mathrm{NDDS}(P, X)$ *is defined as follows:*
Given a graph $G(V, E)$, investment degree sets D_i for all players i consistent with a function property P (such as convexity, concavity, sigmoid, or general), edge costs $\{\gamma_{e \in \binom{n}{x}}\}$, desired PSNE class X, and budget k, decide whether there exists an edge set S with $\sum_{e \in E \ominus S} \gamma_e \leqslant k$ such that there exists a set $I \in X$ of investing players such that in the modified graph $G'(V, E' = E \ominus S)$

$$\begin{aligned} |\mathcal{N}_i^{G'} \cap I| &\in D_i && \forall i \in I; \\ |\mathcal{N}_i^{G'} \cap I| &\notin D_i && \forall i \notin I. \end{aligned}$$

Morever, getting motivated from the inherent hardness for the general case of the problem, we define the *homogeneous* variant of the problem. In the general variant, every player $i \in V[G]$ has a different investment degree set D_i and hence we call this version of the game a *heterogeneous* BNPG game. If not mentioned otherwise, by BNPG game, we refer to a *heterogeneous* BNPG game. On contrary, the *homogeneous* variant imposes an additional condition that the maximum element of each of the investment degree sets for all players should be same, i.e., $\forall i \in V[G],\ \alpha = \alpha_i = \min\{z \mid \text{s.t. } z \in D_i\}$. We will denote the homogeneous variant of the problem as $\mathcal{N}\mathrm{DDS}^{\alpha}$.

6.2 Parameters Used

We study numerous natural as well as structural parameters for analyzing the problem. Most of them are self-explanatory and follow directly from basic graph theoretic definition, whereas we did define a few new parameters to analyze the input instance more closely. Following is the exhaustive list of parameters considered for our analysis:

1. $k :=$ natural parameter of input budget;
2. $r := r$ from problems $\mathcal{N}\mathrm{DDS}(P, \geqslant r)$;
3. $\alpha :=$ Minimum of lower bounds from all $D_{v \in V[G]}$. $\forall v \in V[G],\ D_v = \{\alpha_v,\ \ldots, n-1\}, \alpha_v \in [n-1]$. We consider the parameter $\alpha = \min_{v \in V[G]}\{\min(z \mid z \in D_v)\}$.
4. $\delta :=$ diameter of input graph
5. $n_{\mathcal{U}} :=$ number of distinct utility functions.
6. treewidth
7. $\mathcal{D} = \max_{v \in V[G]} |D_v|$.
8. $\Delta :=$ maximum degree of input graph
9. $\mathrm{vc} :=$ vertex cover number of the input graph

7 Hardness Results

In this section, we begin with the natural input parameter, i.e., the budget (k) of the problem and establish $W[1] - \mathtt{Hardness}$ for the general, concave, convex and sigmoid variants on the class $[\geqslant r]$ of PSNE. With the further aim to ease the problem, we introduce an additional parameter as the input value r, resulting in the parameter of $k+r$. With respect to this parameter, we again establish $W[1]-\mathtt{Hardness}$ for the same considered mentioned variants. With an (FPT) reduction from already $W[1]-\mathtt{Hard}$ problem of EDGE-K-CORE to our problem, we establish that the problem is hard even w.r.t $k+r+\alpha$. For the sigmoid variant, we further make the problem intractable in parameterized complexity by establishing para-NP-hard w.r.t parameter r. By a reduction from results of Maiti et al. for BNPG (without modifications), we establish W-Hardness considering the number of players investing $(|I|)$, $n-|I|$, treewidth, maximum degree (Δ) and diameter of the input graph, the number of distinct utility functions $(\delta,\ n_U)$.

We prove the hardness of $\mathcal{N}\mathrm{DDS}$(general, all) by a reduction from *r-regular Clique* problem, which is already $W[1] - \mathtt{Complete}$.

Theorem 3. *The problem of* $\mathcal{N}\mathrm{DDS}$*(general, all) is* $W[1] - \mathtt{Complete}$ *w.r.t the parameter* k*, i.e., the budget. In fact, the* $W[1] - \mathtt{Completeness}$ *is applicable even when the input graph is unweighted.*

Proof. Consider an instance $I = (G(V, E), k)$ of r-regular Clique problem where G is r-regular graph and the goal is to decide whether there exists a k-clique as a sub-graph of G. We construct an instance $I' = (G'(V', E'), k')$ of NDDS(general, all) as follows:

- $\triangleright$ $V'[G'] = V[G] \cup Z$, where $Z = \{z_1, ..., z_k\}$;
- $\triangleright$ $E'[G'] = E[G] \cup \{(v_i, z_j) \mid \forall v_i \in V[G],\ , j \in [k]\}$;
- $\triangleright$ $\gamma_e = 1,\ \ \forall e \in E'[G']$;
- $\triangleright$ $D_{v_i} = \{r-k-1,\ r+k\},\ \ \forall v_i \in V[G]$;
- $\triangleright$ $D_{z_j} = \{n-k\},\ \forall j \in [k]$;
- $\triangleright$ $k' = k^2 + \binom{k}{2}$.

This completes the construction of reduced instance. We establish that I is a Yes instance of r-regular Clique iff I' is a Yes instance of $\mathcal{N}\mathrm{DDS}$(general, all). In other words, there exists a k-clique as sub-graph of G iff there exists a network modification of budget k' such that the modified graph G'' has a PSNE where all players invest i.e. $d^{G''}(u) \in D_u,\ \forall u \in V'[G']$.

For forward direction the proof is relatively easier. Let $K_k = \{u_1, ..., u_k\}$ be the vertices of k-clique in G. Consider the instance $G'(V', E')$ constructed by reduction from G. We specify the set of edges to be edited (in this case considering only delete operation will suffice) as $E_{mod} = E_{del} = \{(u_i, u_j) \mid \forall i,\ j \in [k]\} \cup \{(u_i, z_j) \mid \forall i,\ j \in [k]\}$. The cost of edges modified (deleted) $c(E_{del}) = |E_{del}| = \binom{k}{2} + k^2 = k'$ (as all edge costs are 1 i.e. graph is unweighted). Degrees in the modified graph G'' are as follows :

- $\triangleright$ $d^{G''}(v) = r + k,\ \forall v \in V'[G''] \setminus K_k$;
- $\triangleright$ $d^{G''}(u_i) = r - k + 1,\ \forall i \in [k]$;
- $\triangleright$ $d^{G''}(z_j) = n - k,\ \forall j \in [k]$.

Since degrees of all vertices in modified graph in their respective investment degree sets, we have a PSNE in where all the players invest. Thus the reduced instance is a Yes instance to NDDS(general, all). This completes the proof in forward direction.

For reversed direction, consider $I' = (G'(V', E'), k')$ as a Yes instance to $\mathcal{N}$DDS(general, all). Let E_{mod} be the set of edges to be modified to obtain solution graph G''. Consider $E^{bipart}_{del} \subseteq E_{mod}$ as the set of edges deleted with one end incident in Z and other end incident on vertex in $V[G]$. Clearly $|E^{bipart}_{del}| \geqslant k^2|$ as the initial degree of each vertex of Z in G' is $d^{G'}(z_i) = n\ \forall i \in [k]$, whereas the final degree of each vertex of Z in modified graph G'' is $d^{G''}(z_i) = n - k,\ \forall i \in [k]$, which accounts for at least $nk - (n-k)k = k^2$ deletions. Based on the final degree; $d^{G''}()$ of vertices from $V[G]$, we partition it into two parts $V_{c \in \{r-k+1,\ r+k\}}$. The other incident points of these edges in E^{bipart}_{del} in $V[G]$ can be in one of $V_{c \in \{r-k+1,\ r+k\}}$. This partitions E^{bipart}_{del} into $E_i,\ \forall i \in \{r-k-1, r+k\}$ depending on the degree of vertex from $V[G]$ in G''. For every 2 edges from E_{r+k}, we need at to add at least one edge within points in V_{r+k} i.e. on average basis every edge from E_{r+k} requires compensation with at least $\frac{1}{2}$ edges additions. In case of E_{r-k+1}, minimum average compensation occurs only when $E_{r+k} = \phi$ and E_{r-k+1} contains k^2 edges incident on exactly k vertices from $V[G]$; say $K_k = \{u_1, \ldots, u_k\}$; and K_k forms a clique. Number of edge deletions required in this case are $\binom{k}{2}$. This implies the minimum possible edge edition budget sums up to $\binom{k}{2} + k^2 = k'$. This makes it the only possible case for I' to be a Yes instance. Thus, we get that given I' as a Yes instance, the original graph G contains a k-clique. This concludes the proof. □

By setting $S = V[G]$ (resp. $r = n$), the Theorem 3 can be extended to establish $W[1] -$ Completeness of $\mathcal{N}$DDS(general, =S), $\mathcal{N}$DDS(general, $\supseteq S$) (resp. $\mathcal{N}$DDS(general, $\geqslant r$) with respect to the parameter k. This can be suummarized as the following corollary:

Corollary 1. *For all $X \in \{\text{all},\ = S,\ \supseteq S,\ \geqslant r\}$, the problem of $\mathcal{N}$DDS(general, X) is $W[1] -$ Complete w.r.t the parameter k i.e. the budget even when the input graph is unweighted.*

Now we use reductions from [KYV20] to obtain the following results:

Theorem 4. *$\mathcal{N}$DDS(concave, X) for $X \in \{\supseteq S,\ \geqslant r\}$ is $W[1] -$ Complete w.r.t the parameter k.*

Proof. The polynomial reduction in [KYV20] from Independent Set with the natural parameter k, preserves the parameter as the parameter for the reduced $\mathcal{N}$DDS instance is $k' = k$. Since Independent Set is $W[1] -$ Hard w.r.t to the k, we get that the problem is $W[1] -$ Complete (verifying a certificate to the input instance takes at most $poly(n)$ time. □

Trivially as per our definition, every set of all concave functions is a subset of sigmoid functions. Thereby giving us an extension of Theorem 4 over sigmoid functions.

Corollary 2. *$\mathcal{N}$DDS(sigmoid, X) for $X \in \{\supseteq S,\ \geqslant r\}$ is $W[1] -$ Complete w.r.t the parameter k.*

Another direct result that can be inferred from Polynomial reductions in [KYV20] is of $W[1]-$ Complete w.r.t parameter r for the following problems:

Theorem 5. *$\mathcal{N}$DDS(P, $\geqslant r$) for $P \in \{\text{concave},\ \text{convex},\ \text{sigmoid}\}$ is $W[1] -$ Complete w.r.t the parameter r. Even when $k = 0$.*

Proof. The polynomial reduction in [KYV20] from Independent Set with the natural parameter k, preserves the parameter as the parameter for the reduced $\mathcal{N}$DDS instance is $r = k$. Since Independent Set is $W[1] -$ Hard w.r.t to the r, we get that the problem is $W[1] -$ Complete (verifying a certificate to the input instance takes at most $poly(n)$ time. □

Since for $\mathcal{N}\mathrm{DDS}(\text{convex}, \geqslant r)$ is W[1]-Hard by Theorem 5, one may think of involving more parameters into the equation to obtain an FPT. From Theorem 2, we know that investment degree for a convex function is downward closed, i.e. $\forall v \in V[G],\ D_v = \{\alpha_v,\ \ldots, n-1\}, \alpha_v \in [n-1]$. We prove that the problem when paramterized by $\alpha = \min_{v \in V[G]} \alpha_v$ is $W[1]-\mathrm{Hard}$. To be specific we prove that the problem is $W[1]-\mathrm{Hard}$ parameterized by $k+r+\alpha$ by a FPT-reduction from an already $W[1]-\mathrm{Hard}$ problem of EDGE-K-CORE. The problem EDGE-K-CORE is defined as below.

Definition 13 (EDGE-K-CORE)**.** *Given a simple, undirected graph* $G = (V, E)$ *and integers* k*,* α*, and* r*, decide if there exists a set of vertices* $H \subseteq V[G]$ *such that adding at most* k *edge additions to* G*, we obtain a graph* G' *and every* $v \in H$ *has* $\deg_{G'[H]}[v] \geqslant \alpha$*.*

Theorem 6. *[CT18, Corollary 1]* EDGE-K-CORE *is* $\mathrm{NP}-\mathrm{hard}$ *for* $\alpha = 3$*, even on planar graphs of max degree* 5*.*

Theorem 7. *[CT18, Theorem 4]* EDGE-K-CORE *is W[1]-hard parameterized by* $r+k$*, for* $\alpha = 3$*.*

For the reduction, the key observation for $\mathcal{N}\mathrm{DDS}(\text{convex}, \geqslant r)$ is that any optimal algorithm would have no incentive in removing any edges even if the edge deletion cost is set to 0 (note that the problem considers only non-negative weights on for the edges, if the weights are negative then this may no longer hold true).

Observation 1. *For* $\mathcal{N}\mathrm{DDS}(\text{convex}, \geqslant r)$*, any optimal solution set minimal solution set* S*, is a subset of* $\binom{V}{2} \setminus E[G]$ *i.e. only modifications in the graph are edge additions. Furthermore, each edge from* S*, should be incident to at least one vertex from final set of players investing i.e.* I*.*

Proof. Let us assume that there is a minimal feasible solution S with a non-empty intersection with $E[G]$, which would mean that we are deleting a non-zero number of edges to reach to the solution. Lets call this set as E_{del}. We can create another solution $S' = S \setminus E_{del}$. This is equivalent to saying that we undo the deletion of edges, since adding back the edges to the solution graph corresponding to S, would not drop the degree of any vertex, it would still be a feasible solution. Thus, we conclude the proof by contradiction that S is not a minimal feasible solution. Similarly we can establish that S would have each edge incident to at least one vertex from set of players investing finally. □

Observation 1 eases down the the reduction by laying down that both the problems involve modifications only in the form of edge additions. We now present the main reduction result:

Theorem 8. $\mathcal{N}\mathrm{DDS}$*(convex,* $\geqslant r$*) is* $\mathrm{W}[1]$*-hard with respect to the parameter* $k+r+\alpha$*, in particular the problem is* $\mathrm{W}[1]$*-hard w.r.t parameter* $k+r$ *even when* $\alpha = 3$*. This applies even when the graph is unweighted.*

Proof. We reduce EDGE-K-CORE to our problem preserving the parameter (FPT Reduction). Consider the input instance of EDGE-K-CORE: Simple undirected unweigthed graph $G(V,\ E)$, limit on edge additions k, mimimum required degree of subgraph H say α and minimum size of H say r. We create an instance of $\mathcal{N}\mathrm{DDS}(\text{convex}, \geqslant r^*)$ $[G^*(V^*,\ E^*), k^*]$ as follows:

1. $G^* = G$ i.e. $V^* = V$ and $E^* = E$;
2. $D_v = \{\alpha,\ \ldots,\ n-1\}\ \forall v \in V^*$;
3. $r^* = r$
4. $k^* = k$

This completes the construction of the reduction. Clearly the reduction is FPT Reduction as it preserves the parameters. We claim that a there exists a solution S to EDGE-K-CORE instance is a solution to if and only if there is a solution $S^* = S$ for the corresponding reduced $\mathcal{NDDS}$ instance.

For forward direction, assume, S is the minimal feasible solution to EDGE-K-CORE instance. The final graph after edge modifications be G'. Since S is feasible solution, we know that there is a set $H \subseteq V[G']$ such that adding at S to G, we obtain a graph G' and every $v \in H$ has $deg_{G[H]} \geqslant \alpha$ and $|H| \geqslant r$. For the $\mathcal{NDDS}$ instance, set solution edge set to be $S^* = S$. Now the final set of players investing, say I^*, is a superset of H i.e. $I \supseteq H$. Since $v \in H$ has $deg_{G[H]} \geqslant \alpha \geqslant \alpha_v$. We can claim that all vertices from H are investing $(x_v = 1)$. Thus we get a $|I^*| \geqslant |H| \geqslant r$.

For reverse direction, assume, S^* is the minimal feasible solution to $\mathcal{NDDS}$ instance. From Observation 1, we know that the S^* is disjoint from set of edges E^*. Or in simple words, S^* corresponds to only edge additions to the graph G^*. The final graph after edge modifications be $G^{*'}$. Since S^* is feasible solution, we know that there is a set $I^* \subseteq V^*$, such that $\forall v \in I^* \quad deg_{G[I^*]}[v] \subseteq D_v^{G^*}$. This further implies that $\forall v \in I^* \quad deg_{G[I^*]}[v] \geqslant \alpha_v \geqslant \alpha$ and $|I^*| \geqslant r$. We can construct feasible solution $S = S^*$ of EDGE-K-CORE. The required subgraph satisfying the min degree constraint is $H = I^*$. This completes the reduction. Since EDGE-K-CORE is W[1]-hard w.r.t $k + r$ even when $\alpha = 3$ and the reduction directly maps the parameters $(k,\ r,\ \alpha)$ to $(k^*,\ r^*,\ \alpha^*)$. This gives us $W[1] - hardness$ to $\mathcal{NDDS}$(convex, $\geqslant r$). □

Corollary 3. *$\mathcal{NDDS}^{\alpha}$(convex, $\geqslant r$) is W[1]-hard with respect to the parameter $k + r + \alpha$, in particular the problem is W[1]-hard $ness$ w.r.t parameter $k + r$ even when $\alpha = 3$. This applies even when the graph is unweighted.*

Proof. The reduction from Theorem 8, can be clearly seen to be done to the homogeneous instances of $\mathcal{NDDS}$ only. This extends the W[1]-hard to $\mathcal{NDDS}$(convex, $\geqslant r$). □

We now try and look towards a more general class of functions, i.e., sigmoid functions. We prove that the problem is W[1]-hard w.r.t the parameter r. For this, we reduce from the problem of R-REGULAR SUBGRAPH defined as follows:

Definition 14 (R-REGULAR SUBGRAPH). *Given a simple, undirected Graph $G(V, E)$, decide whether there exists a $H \subseteq V[G]$, such that subgraph on H i.e. $G[H]$, is r-regular.*

The problem of R-REGULAR SUBGRAPH is para-NP-hard w.r.t parameter r, even with maximum degree $\Delta = 7$. It $para -$ NP-$hardness$ further extends to planar graphs (even with $\Delta = 4$) and bipartite graphs.

We propose a reduction from R-REGULAR SUBGRAPH to $\mathcal{NDDS}$(sigmoid, $\geqslant r$). We also consider a new parameter $\mathcal{D} = max_{v \in V[G]} |D_v|$. Moreover, the problem is also W[1]-hard w.r.t r, k=0.

Theorem 9. *$\mathcal{NDDS}$(sigmoid, $\geqslant r$) is para-NP-hard w.r.t parameter $r + k$ even when the maximum size of investment degree set i.e. $\mathcal{D}$ is 1, k=0, and the graph is unweighted.*

• Moreover, with respect to the it is para-NP-hard even w.r.t all $\alpha_v \ \forall v \in V[G]$.

Proof. We reduce R-REGULAR SUBGRAPH to our problem. Consider the input instance of R-REGULAR SUBGRAPH: Simple undirected unweigthed graph $G(V,\ E)$, parameter r (required degree of regular subgraph H). We create an instance of $\mathcal{NDDS}$(sigmoid, $\geqslant r^*$) $[G^*(V^*,\ E^*), k^*]$ as follows:

1. $G^* = G$ i.e. $V^* = V$ and $E^* = E$;

2. $D_v = \{r\} \ \forall v \in V^*$;

3. $r^* = r$

4. $k^* = 0$

5. weight of each edge = 1.

This completes the construction of the reduction. Clearly, the reduction runs in polynomial time and preserves the parameters. We claim that there exists a solution subgraph on H of R-REGULAR SUBGRAPH instance if and only if there is a solution with for the corresponding reduced $\mathcal{N}$DDS instance.

For forward direction, assume, H be a maximal solution to R-REGULAR SUBGRAPH instance, i.e., $G[H]$ is r-regular and cannot any more vertices from the rest of the graph to H to obtain larger r-regular graph. The set H forms the set of players investing in $\mathcal{N}$DDS, i.e. $I^* = H$. This follows from the observation that every $v \in I^*$ has $\deg_{G[I^*]} = r \in D_v$ and the set is maximal.

For the reverse direction, assume, I^* is the set of players investing in the final graph of $\mathcal{N}$DDS instance (here we can just use final graph and input graph interchangeably as no modifications are done since $k = 0$ and edges have weight 1). Following the fact that every $v \in I^*$ has $\deg_{G[I^*]} = r$, we can directly consider I^* as a solution to R-REGULAR SUBGRAPH. This completes the reduction. Since R-REGULAR SUBGRAPH is para-NP-hard w.r.t $k + r$ even when $\alpha = 3$ and the reduction directly maps the parameters $(k,\ r,\ \alpha)$ to $(k^*,\ r^*,\ \alpha^*)$. We are able to establish that $\mathcal{N}$DDS(sigmoid, $\geqslant r$) is para-NP-hard w.r.t parameter r even when the maximum size of investment degree set i.e. $\mathcal{D}$ is 1, k=0, and the graph is unweighted. □

We can further observe from the reduction that the result applies to homogeneous variant as well.

Corollary 4. *$\mathcal{N}$DDS$^\alpha$(sigmoid, $\geqslant r$) is para-NP-hard w.r.t parameter $r + k$ even when the maximum size of investment degree set i.e. $\mathcal{D}$ is 1, k=0, and the graph is unweighted. Moreover, with respect to the it is para-NP-hard even w.r.t α.*

Proof. It trivial to notice that the reduced instance in all correspond to $\mathcal{N}$DDS$^\alpha$(sigmoid, $\geqslant r$). □

[MD20] deals with the problem of deciding the existence of PSNE in BNPG games. The problem can be reduced to $\mathcal{N}$DDS(general, $\geqslant r/ \supseteq S$), by setting budget $k = 0$ and some arbitrary non-zero weight to all the vertex pairs (which is basically the cost of addition or deletion of an edge) and $r = 0$ or $S = \phi$. This constraints the problem to not edit any edge. With some more observation, we can claim that there is a PSNE in the BNPG game (editions not allowed) if and only if there is a PSNE with $r = 0$ (resp. $S = \phi$), for $\mathcal{N}$DDS(general, $\geqslant r$ or $\supseteq S$). Thus we inherit the following results from [MD20]:

Observation 2. *For $X \in \{\geqslant r,\ \supseteq S\}$, $\mathcal{N}$DDS(general, X) is $W[2]-\mathrm{Hard}$ w.r.t $|I|$ where I is the set of players investing in the final solution. In fact the problem is $W[2]-\mathrm{Hard}$ w.r.t $n - |I|$ as well.*

Observation 3. *For $X \in \{\geqslant r,\ \supseteq S\}$, $\mathcal{N}$DDS(general, X) is $W[1]-\mathrm{Hard}$ w.r.t parameter $\mathrm{treewidth}$ even when all the players have identical utility functions.*

Observation 4. *For $X \in \{\geqslant r,\ \supseteq S\}$, $\mathcal{N}$DDS(general, X) is para-NP-hard w.r.t maximum degree of input graph (Δ) as parameter even when all the players have identical utility functions.*

Observation 5. *For $X \in \{\geqslant r,\ \supseteq S\}$, $\mathcal{N}$DDS(general, X) is para-NP-hard w.r.t the diameter (δ) of input graph as parameter even when all the players have identical utility functions.*

Observation 6. *For $X \in \{\geqslant r,\ \supseteq S\}$, $\mathcal{N}$DDS(general, X) is para-NP-hard w.r.t the (diameter (δ), number of distinct utility functions (n_U) of input graph as parameter even when all the players have identical utility functions.*

8 Algorithms

Following numerous hardness results in FPT complexity from section 7, we now aim to explore algorithms within XP complexity, which is the only solution plausible for the problems which are proven W[1]-hard.

We first present a trivial XP algorithm for $\mathcal{N}$DDS w.r.t the natural parameter k.

Theorem 10. *All versions of* $\mathcal{N}$DDS *can be solved in XP time* $n^{O(k)}$ *where* k *is the input parameter (budget) and* n *is the total number of players.*

Proof. Forall $k' \leqslant k$, we can enumerate all possible combinations of edges from $\binom{n}{2}$ possibilities and guess the solution set by a brute force in time at most $k\binom{\binom{n}{2}}{k}$, which is at most $n^{O(k)}$. □

8.1 The Homogeneous Variant: $\mathcal{N}$DDS$^{\alpha}$

Following the inherent hardness of the problem, we explore the $\mathcal{N}$DDS$^{\alpha}$ variant with the aim to resolve the problem with a relaxed set of constraints. Recall that in $\mathcal{N}$DDS$^{\alpha}$, we have that the corresponding degree sets have the same minimum value, i.e., $\forall i \in V[G]\ \alpha_i = \alpha$. We consider the $\mathcal{N}$DDS$^{\alpha}$(convex, $\geqslant r$) problem, which has already been proven to be W[1]-hard with respect to the parameter $k + r + \alpha$ (Corollary 3) as well as para-NP-hard w.r.t parameter $r + k$ (Crefcorr:sigrdd).

We first define the notion of *deficiency* for a player as follows:

Definition 15 (Deficiency)**.** *For a graph* G*, and its vertex* $v \in V(G)$*, let* $\mathrm{df}_G(v) = \max\{0, k - \deg_G(v)\}$ *denote the* deficiency *of* v *in* G*. By* $\mathrm{df}(G) = \sum_{v \in V(G)} \mathrm{df}_G(v)$ *we denote the* total deficiency *in* G.

Note that the problem $\mathcal{N}$DDS$^{\alpha}$(convex, $\geqslant r$) can now be stated as follows: "Decide whether there exists an induced subgraph H of input graph G, such that there exists a set of edges of size at most k, that can be added to H, to satisfy the deficiencies of all the vertices in H". In other words, H after edge edition, we have that $\mathrm{df}_H(v) = 0\Big|_{\forall v \in V[H]}$. Moreover, addition of an edge between two vertices of G can decrease $\mathrm{df}(G)$ by at most two. It also does not make any sense to add edges that do not decrease deficiency if we aim to complete G to a graph of minimum degree k. We distinguish added edges by whether they decrease deficiency by two or one. That is, adding an edge between two vertices $u, v \in V(G)$ (we can do that only if $uv \notin E(G)$), decreases the total deficiency by two only if both $\mathrm{df}_G(u) > 0$ and $\mathrm{df}_G(v) > 0$. We call such edge addition *good*. Otherwise, an edge addition makes sense only if at least one of $\mathrm{df}_G(u)$ and $\mathrm{df}_G(v)$ is greater than zero. In that case, it decreases the total deficiency in G by one, and we call such edge addition *bad*.

Thus adding a good edge decreases the total deficiency by 2 and adding a bad one by 1. This gives us the following lemma for $\mathcal{N}$DDS$^{\alpha}$(convex, $\geqslant r$) on general graphs:

Theorem 11. *Given* α *and input graph* H *on at least* $\alpha + 1$ *vertices as an instance of* $\mathcal{N}$DDS$^{\alpha}$*(convex,* $\geqslant r$*), all the players in* H *can be made to invest by adding* k *edges where:*

$$\left\lceil \frac{1}{2} \sum_{v \in V(H)} \mathrm{df}(v) \right\rceil \leqslant k \leqslant \sum_{v \in V(H)} \mathrm{df}(v)$$

, and this cannot be done with lesser edge additions.

Proof. From the modified problem definition, we know that – in order to bound the cardinality (k) of the set of edges, that can be added to H, we need satisfy the deficiencies of all the vertices in H. In other words, H after edge edition, we have that $\mathrm{df}_H(\nu) = 0\Big|_{\forall \nu \in V[H]}$. Note that an addition of an edge between two vertices of H can decrease $\mathrm{df}(H)$ by at most two. It also does not make any sense to add edges that do not decrease deficiency if we aim to complete H to a graph of minimum degree k. The optimal number of such edges would depend on the number of added edges by whether they decrease deficiency by two or one, i.e., whether they are good or bad. Thus adding a good edge decreases the total deficiency by 2 and adding a bad one by 1. In the worst case, each edge can be a bad edge, giving us the claimed upper bound. Whereas, considering the best case, when all the edges in the optimal solution set are good, we establish the lower bound. □

Studying the problem of $\mathcal{N}\mathrm{DDS}^{\alpha}$(convex, $\geqslant r$) in terms of deficiencies of the vertices, we get the notion about similarity of the the problem with EDGE-K-CORE. We provide a parameter preserving reduction from $\mathcal{N}\mathrm{DDS}^{\alpha}$(convex, $\geqslant r$) to EDGE-K-CORE. This would apply any algorithmic results available for EDGE-K-CORE to our problem.

Theorem 12.

$$\mathcal{N}\mathrm{DDS}^{\alpha}(convex, \geqslant r) \leqslant_{FPT} \text{EDGE-K-CORE}$$

Proof. We reduce to EDGE-K-CORE from our problem preserving the parameter (FPT Reduction). Consider the input instance of $\mathcal{N}\mathrm{DDS}^{\alpha}$(convex, $\geqslant r^*$) $[G^*(V^*, E^*), k^*, \alpha]$. We create an instance of EDGE-K-CORE: A simple undirected unweigthed graph $G(V, E)$, limit on edge additions k, mimimum required degree of subgraph $H = \alpha$ and minimum size of $H = r$; as follows:

1. $G = G^*$ i.e. $V = V^*$ and $E = E^*$;
2. $r = r^*$
3. $k = k^*$
4. $\alpha = \alpha^*$

This completes the construction of the reduction. Clearly the reduction is FPT Reduction as it preserves the parameters. We claim that a there exists a solution S to EDGE-K-CORE instance is a solution to if and only if there is a solution $S^* = S$ for the corresponding reduced $\mathcal{N}\mathrm{DDS}^{\alpha}$ instance.

For forward direction, assume, S is the minimal feasible solution to EDGE-K-CORE instance. The final graph after edge modifications be G'. Since S is feasible solution, we know that there is a set $H \subseteq V[G']$ such that adding at S to G, we obtain a graph G' and every $\nu \in H$ has $\deg_{G[H]} \geqslant \alpha$ and $|H| \geqslant r$. For the $\mathcal{N}\mathrm{DDS}^{\alpha}$(convex, $\geqslant r^*$) instance, set solution edge set to be $S^* = S$. Now the final set of players investing, say I^*, is a superset of H i.e. $I \supseteq H$. Since $\nu \in H$ has $\deg_{G[H]} \geqslant \alpha \geqslant \alpha_\nu$. We can claim that all vertices from H are investing $(x_\nu = 1)$. Thus we get a $|I^*| \geqslant |H| \geqslant r$.

For reverse direction, assume, S^* is the minimal feasible solution to $\mathcal{N}\mathrm{DDS}^{\alpha}$(convex, $\geqslant r^*$) instance. From Observation 1, we know that the S^* is disjoint from set of edges E^*. Or in simple words, S^* corresponds to only edge additions to the graph G^*. The final graph after edge modifications be $G^{*'}$. Since S^* is feasible solution, we know that there is a set $I^* \subseteq V^*$, such that $\forall \nu \in I^* \quad \deg_{G[I^*]}[\nu] \subseteq D_\nu^{G^*}$. This further implies that $\forall \nu \in I^* \quad \deg_{G[I^*]}[\nu] \geqslant \alpha_\nu \geqslant \alpha$ and $|I^*| \geqslant r$. We can construct feasible solution $S = S^*$ of EDGE-K-CORE. The required subgraph satisfying the min degree constraint is $H = I^*$. This completes the reduction. □

In this section we present our polynomial time algorithm for $\mathcal{N}$DDS$^\alpha$(convex, $\geqslant r$) on forests and the underlying graph-theoretical result, which we find interesting on its own. The algorithm itself is a dynamic programming over subtrees. Normally, an algorithm like this would go from leaves to larger and larger subtrees, storing for every subtree a list of possible configurations a solution could induce on this subtree. In the $\mathcal{N}$DDS$^\alpha$(convex, $\geqslant r$) problem, naturally we want to store information about edges added inside the subtree and vertices from the subtree which we may later connect to something outside.

Naively, this would take exponential space, as it seems we have to store at least the degrees of the selected vertices in the subtree. However, the following theorem from [FSS20], which is the central technical result of this section, helps greatly.

Lemma 1. *[FSS20, Theorem 4] For any integer k, any forest T on at least $k+1$ vertices can be completed to a graph of minimum degree k by adding at most*

$$\left\lceil \frac{1}{2} \sum_{v \in V(T)} \max\{0, k - \deg(v)\} \right\rceil$$

or

$$\left\lceil \frac{1}{2} \sum_{v \in V(T)} \mathrm{df}(v) \right\rceil$$

edges, and this cannot be done with less edge additions. Moreover, in the case $k \geqslant 4$, it can be done in a way that the added edges form a connected graph on the vertices they cover.

The above lemma gives us the following observations for $\mathcal{N}$DDS$^\alpha$:

Observation 7. *Given α and input graph H on at least $\alpha+1$ vertices as an instance of $\mathcal{N}$DDS$^\alpha$(convex, $\geqslant r$), all the players in H can be made to invest by adding k edges where:*

$$k = \left\lceil \frac{1}{2} \sum_{v \in V(H)} \mathrm{df}(v) \right\rceil$$

, and this cannot be done with lesser edge additions.

For an optimal algorithm for forests algorithm, Theorem 1 means that whenever we fix the subset of vertices H, we have to add exactly $\lceil \mathrm{df}(T[H])/2 \rceil$ edges in order to induce a k-core on H. Thus it is enough to find a subset of vertices H of size at least p with the smallest possible $\mathrm{df}(T[H])$. This objective turns out to be simple enough for the bottom-top dynamic programming. Namely, for a subtree T_v rooted at v, it is enough to store the size of $H \cap T_v$, the total deficiency of these vertices, whether v is in H and how many neighbors in $H \cap T_v$ it has. Since v separates T_v from the rest of the tree, the deficiency of other vertices in $H \cap T_v$ is unchanged no matter how H looks like in the rest of the tree.

The discussion above ultimately leads to a polynomial time algorithm, stated formally in the next theorem.

Theorem 13. *[FSS20, Theorem 5]* EDGE-K-CORE *is solvable in time $O(kn^2)$ on the class of forests, where k is required minimum degree of the induced solution subgraph.*

Observation 8. *$\mathcal{N}$DDS$^\alpha$(convex, $\geqslant r$) is solvable in time $O(\alpha n^2)$ on the class of forests.*

Now we employ the reduction from Theorem 12, to give an algorithm for $\mathcal{N}$DDS$^\alpha$(convex, $\geqslant r^*$) parameterized by the minimum size of a vertex cover of the input graph G. We employ the results on EDGE-K-CORE from [FSS20, Section 4], which establishes that EDGE-K-CORE admits an FPT algorithm. Secondly, they prove that this problem does not admit polynomial kernel unless the polynomial hierarchy collapses. The high level description of the main ideas behind the ILP algorithm are as follows: In order to prove that EDGE-K-CORE is FPT parameterized by the vertex cover number of the input graph, construct an FPT-time Turing reduction from EDGE-K-CORE to an instance of integer linear program (ILP) whose number of variables is bounded by some function of the vertex cover. While reducing to ILP is a common approach in the design of parameterized algorithms, see [CFK+15, Chapter 6], the reduction for EDGE-K-CORE is not straightforward and needed development of new combinatorial results.

Theorem 14. *[FSS20, Theorem 14]* EDGE-K-CORE *admits an* FPT *algorithm when parameterized by the vertex cover number. The running time of this algorithm is* $2^{\mathcal{O}(\mathrm{vc}\cdot 3^{\mathrm{vc}})} \cdot n^{\mathcal{O}(1)}$*, where* vc *is the minimum size of a vertex cover of the input* n*-vertex graph.*

Observation 9. $\mathcal{N}$DDS$^{\alpha}$*(convex,* $\geqslant r^*$*) admits an* FPT *algorithm when parameterized by the vertex cover number. The running time of this algorithm is* $2^{\mathcal{O}(\mathrm{vc}\cdot 3^{\mathrm{vc}})} \cdot n^{\mathcal{O}(1)}$*, where* vc *is the minimum size of a vertex cover of the input* n*-vertex graph.*

We believe that this algorithm nicely employs a classical FPT framework as well as involves classical graph-theoretical results, that are tweaked to fit in the paradigm of parameterized complexity.

Following this, we again exploit the FPT-algorithm for EDGE-K-CORE parameterized by $\mathrm{tw} + k$, where k is the required minimum degree of the solution subgraph, given by [FSS20]. Their work improves upon the following result on FPT for EDGE-K-CORE by Chitnis and Talmon which runs in time $(k+\mathrm{tw})^{\mathcal{O}(\mathrm{tw}+b)} \cdot n^{\mathcal{O}(1)}$ by employing their algorithm as a subroutine. Again to avoid any confusion because of difference in naming convention of parameters for the EDGE-K-CORE and $\mathcal{N}$DDS$^{\alpha}$problem, we point out that here b implies the budget, i.e., the allowed size of set of edges to be added, and k implies the required minimum degree of the solution subgraph. Intuitively, they start with the central combinatorial result of this section which allows the algorithmic improvement establishing that whenever the total deficiency of a graph G exceeds a polynomial in k, G can be completed to a graph of minimum degree k using the minimum possible number of edges with the required edge additions identifiable in polynomial time. This result is interesting on its own, since it considerably simplifies the problem whenever the budget is sufficiently high compared to k. If we are trying to identify the best vertex set H which induces a k-core, we have to only care about the total deficiency of $G[H]$, and not of any particular structure on it. The final result is as follows:

Theorem 15. *[FSS20, Theorem 21]* EDGE-K-CORE *admits an* FPT *algorithm when parameterized by the combined parameter* $\mathrm{tw} + k$*, where* k *is the minimum degree of the required solution subgraph.*

The above theorem, along with the reduction from Theorem 12 gives the following results for $\mathcal{N}$DDS$^{\alpha}$:

Observation 10. $\mathcal{N}$DDS$^{\alpha}$*(convex,* $\geqslant r^*$*) admits an* FPT *algorithm when parameterized by the combined parameter* $\mathrm{tw} + \alpha$*, where* α *is the minimum degree of the required solution subgraph.*

9 Conclusion and Future Directions

In this paper, we exploit the problem of Network Design for BNPG games using tools of parameterized complexity. As discussed in Prior Works and Introduction, $\mathcal{N}$DDS plays a crucial role

in numerous infrastructures not only a part of computer science but also economics, game theory and network design. We adapted the problem from Kempe et al. who considered network modifications in the form of edge editions (deletion or addition) for inducing a targeted PSNE on the network. We considerably notched up the results taking into account the parameterized complexity considering the natural as well as structural parameters, establishing hardness results as well as XP-algorithms for the same. Our hardness results, even when we combine upto three parameters, emphasize intuitively on the parameterized hardness of the problem, potentially eliminating any hope of FPT-algorithms for the same. Since FPTs are ruled out, we further put a cap on the possible algorithmic results for the problem by providing XP-algorithms w.r.t k and r.
Following this, we consider the *homogeneous* variant of the problem to study the problem under relaxed constrains. Again we establish W[1]-hard for most of the parameters using the hardness results of the general variant itself. We then establish the notion of deficiency in the input graph or its subgraphs, formulating the problem statement of $\mathcal{N}\mathrm{DDS}^{\alpha}(\text{convex}, \geqslant r^*)$ in a second perspective. Then we establish the bounds on the number of edges to be added corresponding to the optimal solution for general graphs. We then establish a reduction to EDGE-K-CORE. This essentially, results in strictly bounding the number of edges in the optimal solution for the case of forests. Morever, using the same reduction, we extend the results of EDGE-K-CORE to obtain an FPT for $\mathcal{N}\mathrm{DDS}^{\alpha}(\text{convex}, \geqslant r^*)$ w.r.t parameters $(tw + \alpha)$. We also obtain an FPT for $\mathcal{N}\mathrm{DDS}^{\alpha}(\text{convex}, \geqslant r^*)$ w.r.t. vc as parameter, and complement this result by ruling out the existence of a polynomial kernel using the reduction to EDGE-K-CORE.

The significance of our work follows from the fact that we first prove W[1]-hardness (in some cases W[2]-Hardness), giving a lower bound and ruling out FPT and along with providing an upper bound in the form XP-algorithms as well as FPT algorithms with a combination of parameters, making the analysis complete.

Our work provides an intuitive dead-end w.r.t the parameterized complexity of considered parameters but leaves open several research questions to be explored in Approximation algorithms or even FPT-Approximation Algorithms. Plausible directions to work on the problem in the future include:

1. **ε-PSNE**: Considering the multitude of harndess results w.r.t most natural parameters as discussed above, one can pivot from parameterized complexity and focus towards finding approximation algorithms for the $\mathcal{N}\mathrm{DDS}$. There are two broad directions in terms of designing approximation algorithms of $\mathcal{N}\mathrm{DDS}$ (1) Relaxing the budget by an additional factor of ε (i.e. the target budget is $(1+\varepsilon).k$ in this case) or (2)Relaxing the PSNE constraints by an ε factor. The later case is a well studied problem in mechanism design termed as ε-PSNE. A joint strategy profile x is said to be an ε-PSNE for BNPG iff $U_i(x_i, n_i^x) \geqslant (1+\varepsilon)U_i(1-x_i, n_i^x)$ (in case of equality breaking ties in favor of investing). We strongly think that the problem of finding ε-PSNE is worth studying for BNPG games, and may yield polynomial time algorithms for the same.

2. **Structural Parameters**: The innate property of problem can be realized in terms of much more complex structural parameters such as feedback vertex set or feedback arc set of the input graph. We already have initiated these results by establishing an FPT algorithm w.r.t vertex cover number, motivating us to further analyze other structural parameters for $\mathcal{N}\mathrm{DDS}$.

3. **Line Graph**: In graph theoretic problems, it is often helpful to look at the corresponding line graph of the input instance and formulate the problem on the resulting graph. We believe that this may allow us to exploit the property of line graph and may help in

looking at the problem in a different formulation. Another motivation behind considering line graphs is that it aids in realizing the edges of original instance as vertices in the line graph. The problem of edge edition can now be consider from a perspective of a vertex edition problem and may render achievable results.

4. **Trivial Graph Classes**: We already have presented a polynomial time algorithm running in time $O(\alpha n^2)$ for $\mathcal{N}DDS^{\alpha}(\text{convex}, \geqslant r)$ on forests. For the other W-Hard variants, we can try solving the problem on relatively easier input graph classes like, cycles, paths, caterpillars, etc. These graph classes have a much simpler structure and we believe that a dynamic program can be designed for trees/forests to solve the problem optimally similar to the dynamic program for $\mathcal{N}DDS^{\alpha}(\text{convex}, \geqslant r)$.

5. **Parameterization by distance to trivial graph classes**: Following the last point, if we are able to devise algorithms for trivial graph classes (trees, paths, cycles, cluster graphs), we can identify simpler substructures in a general input instance to obtain algorithms parameterized by distance to these trivial graph classes.

6. **Color/Chromatic Coding**: Randomized approaches considering color coding may be employed to reduce the relative size of input graph by coloring the solution. Though, According to our intuition $\mathcal{N}DDS$ is potentially hard w.r.t to the color coding or chromatic coloring techniques. Even though it is easy to identify the vertices of the edges in the solution set, it is inherently difficult to devise a way to obtain the corresponding edges without assumptions (such as a bound on the maximum size of investment degree sets $D_{v \in V[G]}$ yields an algorithm with FPT runtime).

7. **The 2-approximation Heuristic**: Another approach would be to utilize the bounds on optimal solution size to get a heuristic which may even perform near-optimally on real data corresponding to BNPG networks.

References

[BE17] Robert Bredereck and Edith Elkind. Manipulating opinion diffusion in social networks. In *IJCAI*, pages 894–900, 2017.

[BK+07] Yann Bramoullé, Rachel Kranton, et al. Public goods in networks. *Journal of Economic Theory*, 135(1):478–494, 2007.

[BKD14] Yann Bramoullé, Rachel Kranton, and Martin D'amours. Strategic interaction and networks. *American Economic Review*, 104(3):898–930, 2014.

[Buc20] Elisabeth Buchwald. Why do so many Americans refuse to wear face masks? Politics is part of it — but only part. https://tinyurl.com/hu6rxzfv, 2020.

[CFG20] Matteo Castiglioni, Diodato Ferraioli, and Nicola Gatti. Election control in social networks via edge addition or removal. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 1878–1885, 2020.

[CFK+15] Marek Cygan, Fedor V Fomin, Łukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. *Parameterized algorithms*, volume 4. Springer, 2015.

[CT18] Rajesh Chitnis and Nimrod Talmon. Can we create large k-cores by adding few edges? In *International Computer Science Symposium in Russia*, pages 78–89. Springer, 2018.

[FG06] Jörg Flum and Martin Grohe. Parameterized complexity theory. 2006. *Texts Theoret. Comput. Sci. EATCS Ser*, 2006.

[FSS20] Fedor V. Fomin, Danil Sagunov, and Kirill Simonov. Building large k-cores from sparse graphs. *CoRR*, abs/2002.07612, 2020.

[GGJ$^+$10] Andrea Galeotti, Sanjeev Goyal, Matthew O Jackson, Fernando Vega-Redondo, and Leeat Yariv. Network games. *The review of economic studies*, 77(1):218–244, 2010.

[HS16] Ashish R Hota and Shreyas Sundaram. Optimal network topologies for mitigating security and epidemic risks. In *2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1129–1136. IEEE, 2016.

[IKMW07] Nicole Immorlica, Jon Kleinberg, Mohammad Mahdian, and Tom Wexler. The role of compatibility in the diffusion of technologies through social networks. In *Proceedings of the 8th ACM conference on Electronic commerce*, pages 75–83, 2007.

[Kle07] Jon Kleinberg. Cascading behavior in networks: Algorithmic and economic issues. *Algorithmic game theory*, 24:613–632, 2007.

[KYV20] David Kempe, Sixie Yu, and Yevgeniy Vorobeychik. Inducing equilibria in networked public goods games through network structure modification. *arXiv preprint arXiv:2002.10627*, 2020.

[LKGM18] Vadim Levit, Zohar Komarovsky, Tal Grinshpoun, and Amnon Meisels. Incentive-based search for efficient equilibria of the public goods game. *Artificial Intelligence*, 262:142–162, 2018.

[MCWG$^+$95] Andreu Mas-Colell, Michael Dennis Whinston, Jerry R Green, et al. *Microeconomic theory*, volume 1. Oxford university press New York, 1995.

[MD20] Arnab Maiti and Palash Dey. On parameterized complexity of binary networked public goods game, 2020.

[Mor00] Stephen Morris. Contagion. *The Review of Economic Studies*, 67(1):57–78, 2000.

[Rou07] Tim Roughgarden. Routing games, algorithmic game theory, noan nisan, tim roughgarden, eva tardos, cijay v. vazirani, 2007.

[Sam54] Paul A Samuelson. The pure theory of public expenditure. *The review of economics and statistics*, 36(4):387–389, 1954.

[SDE$^+$12] Daniel Sheldon, Bistra Dilkina, Adam N Elmachtoub, Ryan Finseth, Ashish Sabharwal, Jon Conrad, Carla P Gomes, David Shmoys, William Allen, Ole Amundsen, et al. Maximizing the spread of cascades using network design. *arXiv preprint arXiv:1203.3514*, 2012.

[SHKW14] Liat Sless, Noam Hazon, Sarit Kraus, and Michael Wooldridge. Forming coalitions and facilitating relationships for completing tasks in social networks. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, pages 261–268, 2014.

[Won20] Tessa Wong. Coronavirus: Why some countries wear face masks and others don't. https://www.bbc.com/news/world-52015486, 2020.

[YZBV20] Sixie Yu, Kai Zhou, P Jeffrey Brantingham, and Yevgeniy Vorobeychik. Computing equilibria in binary networked public goods games. In *AAAI*, pages 2310–2317, 2020.