Parameterized Complexity Applied in Algorithmic Game Theory

Mahdi Parsa
Bsc., Msc., Amirkabir University

Institute for Integrated and Intelligent Systems
Science, Environment, Engineering and Technology
Griffith University

Submitted in fulfilment of the requirements of the degree of Doctor of Philosophy

June 2011

Abstract

The modern mathematical treatment of the study of decisions taken by participants whose interests are in conflict is now generally labeled as “game theory”. To understand these interactions the theory provides some solution concepts. An important such a concept is the notion of Nash equilibrium, which provides a way of predicting the behavior of strategic participants in situations of conflicts. However, many decision problems regarding to the computation of Nash equilibrium are computationally hard. Motivated by these hardness results, we study the parameterized complexity of the Nash equilibrium.

In parameterized complexity one considers computational problems in a two-dimensional setting: the first dimension is the usual input size n, the second dimension is a positive integer k, the parameter. A problem is fixed-parameter tractable (**FPT**) if it can be solved in time $f(k)n^{O(1)}$ where f denotes a computable, possibly exponential, function.

We show that some decision problems regarding to the computation of Nash equilibrium are hard even in parameterized complexity theory. However, we provide FPT-algorithms for some other problems relevant to the computation of Nash equilibrium.

STATEMENT OF ORIGINALITY

This work has not previously been submitted for a degree or diploma in any university. To the best of my knowledge and belief, the thesis contains no material previously published or written by another person except where due reference is made in the thesis itself.

(Signed) ______________________________

Mahdi Parsa

June 2010

Approval

Name:	Mahdi Parsa
Title:	Parameterized Complexity Applied in Algorithmic Game Theory
Submission Date:	June 2011
Supervisor:	Professor Vladimir Estivill-Castro School of Information and Communication Technology, Griffith University Australia
Co-supervisor:	Professor Rodney Topor School of Information and Communication Technology, Griffith University Australia

Acknowledgements

I would like to make special mention of individuals and groups, without whom, I would never have completed this work. I would like to sincerely thank my supervisors, Professor Vladimir Estivill-Catro, and Professor Rodney Topor. Vlad's guidance, encouragement, support, and constructive feedback has been exemplary and tireless for the period of my PhD. Rodney's thoughts and input were always very useful and helped to broaden the scope of many aspects of the thesis and its components. I am grateful to my friends, particulary Apichat Heednacram for helping me with the initial stages of the PhD and sharing the same passion about parameterized complexity theory.

Finally, I thank my wife for her encouragement and support, my son who brought joy to our life. Also my parents whose help was always there when it was needed.

To my wife and my parents

List of Publications Arising from this Thesis

International conferences with papers fully refereed

1. V. Estivill-Castro and M. Parsa. Computing Nash equilibria gets harder: new results show hardness even for parameterized complexity. In Downey, R. and Manyem, P., editors, *CATS-09, Proceedings of the Fifteenth Australasian Symposium on Computing: The Australasian Theory*, volume 94, pages 83-90, Australian Computer Society, Inc., Darlinghurst, Australia, 2009.

2. V. Estivill-Castro and M. Parsa. Single parameter FPT-algorithms for non-trivial games. In C. S. Iliopoulos and W. F. Smyth, editors, *IWOCA-10, Proceedings of the 21st International Workshop On Combinatorial Algorithms*, London, UK, volume 6460 of Lecture Notes in Computer Science, pages 121-124. Springer, 2011.

3. V. Estivill-Castro and M. Parsa. On the parameterized complexity of dominant strategies. *The Thirty-Fifth Australasian Computer Science Conference, ACSC 2012*. (Accepted for the publication.)

Submitted to international journals with papers fully refereed

1. V. Estivill-Castro and M. Parsa. The parameterized complexity of Nash equilibria in win-lose games. Journal of Discrete Algorithms (JDA), 2010. Submitted.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 A brief review of game theory

Game theory is a mathematical framework for the study of conflict and cooperation between intelligent agents. This theory offers models to study decision-making situations and proposes several long-standing solution concepts. Therefore, it has many applications in economics and politics, business, advertising, and information routing. A game consists of a set of players, a set of strategies for each player where each strategy is called pure strategy, and a specification of payoffs [1] for each combination of strategies. Players want to optimize their payoff which depends both on their own choices and also the choices of others.

Here, we use the Prisoners' Dilemma, a classical example in game theory, to illustrate how interactions between players can be modeled as a game. In this game two prisoners, the row player and the column player, are collectively charged with a crime and held in separate cells with no way of communicating. Each prisoner has two choices, cooperate (C) which means not defect his partner or defect (D), which means betray his partner. The punishment for the crime is ten years of prison. If both

[1] "In any game, payoffs are numbers which represent the motivations of players. Payoffs may represent profit, quantity, utility, or other continuous measures, or may simply rank the desirability of outcomes. In all cases, the payoffs reflect the motivations of the particular player" [Sho08].

Table 1.1: Payoff matrix of the players in the Prisoners' Dilemma.

		column player	
		C	D
row player	C	-1,-1	-10,0
	D	0,-10	-9,-9

prisoners cooperate, they will receive light sentences (one year). If both prisoners betray, then they will be prosecuted but with a recommendation for a sentence below the maximum (nine years). If one confesses and the other does not, then the confessor will be free while the other will be prosecuted for the maximum sentence (ten years). This situation can be summarized in Table 1.1. The table represents a game with two players, namely row player and column player. Each player has two possible choices, the row player chooses a row and the column player chooses a column and these two choices are to be made independently and simultaneously. The numbers represent the payoff for the players and there are two payoffs at each position: by convention the first number is the payoff for the row player and the second number is the payoff for the column player. For example the position with numbers (-10,0) means that if the row player chooses to cooperate (C) and the column player chooses to defect (D) the payoff to the row player is -10 and to the column player is 0. Game theory studies what strategies are rational if both players want to minimize the time they spend in prison or maximize their payoffs.

One of the key solution concepts in game theory is the Nash equilibrium, a set of strategies, one for each player, such that all players have no incentive to unilaterally change their decision. For example, in Prisoners' Dilemma the outcome (D, D) of both players defecting is the game's Nash equilibrium, i.e., it is the outcome from which each player could only do worse by unilaterally changing its move. This equilibrium is also called pure Nash equilibrium as the player only used the pure strategies. However, many games do not have a Nash equilibrium in pure strategies. Therefore, the notion of mixed strategies has been defined. A mixed strategy is an assignment of a probability to each pure strategy. For example assigning probability 1/3 to the strategy C and probability $2/3$ to the strategy D in the Prisoners' Dilemma for the row player is a mixed strategy for the player.

Nash [Nas50] proved that every game with a finite number of players and finite number of strategies for each player has an equilibrium (Nash equilibrium) maybe with mixed strategies. But Nash's existence proof does not suggest how to compute an equilibrium. This fact motivates the question: when can we compute Nash equilibria efficiently?

This question is important for two reasons. First, Nash equilibria represent the outcome of many scenarios, many of which have been previously used to model the behavior of participants (governments, unions, individuals) in many social models [TV07]. Second, resolving the complexity of computing Nash equilibrium has economic implications: an efficient algorithm for computing an equilibrium is a very important step for establishing its credibility, while an intractability result forms doubt on its modeling power [Rou10].

Since Nash's paper was published, many researchers developed algorithms for finding Nash equilibria [GW03, LH64]. However, all of them are known to have worst-case running times that are exponential [SvS04, PNS04]. So, the question that naturally arises is : what is the inherent complexity of finding a Nash equilibrium in n-player games?

Computer scientists usefully convert computational problems into decision problems to fit in known complexity classes. A decision problem is a problem where all the answers are YES or NO. For example, questions likes "Does x divide y without remainder?" is a decision problem. To study the complexity of a problem, computer scientists either show that a problem is tractable by explicitly giving a polynomial time algorithm for the problem or they show that the problem is hard. To show hardness of a problem regarding to a complexity class $\mathcal{C}$, they show that if this problem can be solved efficiently then so can every member of the class $\mathcal{C}$ of problems. This is usually done using the notion of reduction among the problems in a class. In classical complexity theory the study of two complexity classes was the birth of this type of hardness results. The class **P** denotes all decision problems that can be decided in polynomial time. The larger class **NP** is the class of all decision problems that can be decided by a non-deterministic Turing machine in polynomial time. Equivalently, **NP** is the class of decision problems for which it can be verified on a deterministic Turing machine in polynomial time whether a candidate solution is actually a solution. A decision problem D is **NP**-complete if it is in the class **NP** and also any problem in **NP** can be reduced into D by a transformation of the inputs in polynomial time. It is generally believed **P** $\neq$ **NP**, but many problems have been shown to be **NP**-complete. Showing a problem D is **NP**-complete is a statement of intractability because if we could show D is solvable in polynomial time, then so would be many interesting problems in **NP** not having such known polynomial algorithm.

The problem of finding a Nash equilibrium in n-player games does not fall into a standard complexity class [Pap94], because it cannot be characterized as an appropriate decision problem as a result of Nash existence theorem (the answer to the decision version of the Nash problem is always "yes"). However, Papadimitriou introduced the notion of **PPAD**-completeness. A **PPAD**-completeness result for a problem implies that there is not too much hope for a polynomial algorithm for the problem [Pap94]. The **PPAD** stands for Polynomial Parity Arguments on Directed graphs.

Nevertheless, the first complexity results for computing Nash equilibria used classic notions (**NP**-hardness) of complexity theory [GZ89]. Further research shows that many attempts to turn this problem into a decision problem, either by imposing restrictions on the kind of Nash equilibrium that is desired, or by asking questions about properties of potential Nash equilibria, resulted in variants of Nash problems that are also computationally hard (**NP**-hard) [AKV05, BIL08, CS05a, CS08]. Therefore, the quest for efficient algorithms for these hard problems (decision or search version) started with studying special cases, approximation algorithms [DMP06, BBM07, TS07, TS10], or using heuristics [CLR06, KLS01, CDT06a, WPD04].

Here, we apply the theory of parameterized complexity, a relatively new approach for dealing with computationally hard problems, to study the complexity of computing Nash equilibria. This theory is based on the observation that many computational problems are associated with a parameter that varies within a small range. For small parameters, this approach has resulted in the engineering of many useful algorithms in different fields [DF98]. Formally, we say that a problem is fixed parameter tractable (**FPT**) with respect to a parameter k if there exists a solution running in $f(k) \cdot n^c$ time, where f is a function of k which is independent of n (the size of problem instance) and c is a constant. In other words, if the parameter k is small and fixed, then the problem can be solved in polynomial time. On the other side, in order to characterize those problems that do not seem to admit a fixed parameter-tractable algorithm a **W** hierarchy collection of computational complexity classes has been defined [DF98]. The largest class, **XP**, is an analogue of **NP**.

A review of the literature in game theory reveals that many decision problems regarding to the computation of Nash equilibria can be answered in $n^{O(k)}$ time, where n is the size of the game (input size) and k is a parameter that is related to a property of the Nash equilibria in the problem. The $n^{O(k)}$ time complexity indicates these problems are in the class **XP** and thus applying parameterized complexity techniques may result in tractability.One of these classes of decision problems are problems related to

the support of Nash equilibria. The support of a mixed strategy is the set of strategies assigned positive probability and the size of this set is known as support size. It is an important concept, because if we know the support, then the corresponding Nash equilibria can be found in polynomial time [vS02]. For example, we can decide whether there exists a Nash equilibrium in a two-player game where each player plays at most k strategies with positive probability in $n^{O(k)}$ time by trying all subsets of size at most k of strategies of players for a potential Nash equilibrium of size at most k. Hence, we focus on this question: can we remove the dependency on the parameter k from the exponent (that is $f(k) \cdot n^{O(1)}$) in these classes of problems and achieve fixed-parameter tractability?

In this thesis we establish the parameterized complexity of some decision problems relating to the computation of Nash equilibria. We achieve both positive results (FPT results) and negative results (proofs of hardness). We particularly focused on studying the most fundamental decision problems relating to the computation of Nash equilibria [AKV05, Pap07, CS08, CS05b, GKZ93, GZ89, GLM^{+}04, KP09].

We summarize the decision problems in this thesis, the reasons for their study, and our classifications relating to parameterized complexity (see also Table 1.2 on Page 16). For each problem, first we define the problem and the motivation for its study, then we summarize our contributions.

1.2 Computing Nash equilibria in two-player games

In this section we define decision problems relevant to the computation of Nash equilibria in two-player games.

k-UNIFORM NASH

> ***Instance*** : A two-player game $\mathcal{G}$.
>
> ***Parameter***: An integer k.
>
> ***Question*** : Does there exist a Nash equilibrium with all the strategies in the support having the same probability (uniform Nash equilibrium) and the size of support is k?

Here, we study the parameterized complexity of the k-UNIFORM NASH problem on a subclass of two-player games, namely win-lose games. A win-lose game is a two-player game where the payoff values of both players are 0 or 1. We focus on the study of this class of games due to the fact that every general two-player game with rational-payoff matrices can be mapped into a win-lose game where the mapping preserves the Nash equilibria in an efficiently recoverable form [AKV05]. This mapping enables us to show that finding a sample Nash equilibrium in win-lose game is as hard as for general two-player games (**PPAD**-completeness) . We also specialize our games into imitation games. An imitation game is a two-player game where the players have the same set of pure strategies and the goal of the column player is to choose the same pure strategy as the row player. Moreover, an imitation game is called symmetric if the payoff matrix of the column player is symmetric. We study the parameterized complexity of finding uniform Nash equilibria in this class of games for the following reasons:

- It seems that a uniform mixed strategy is probably the simplest way of mixing pure strategies [BIL08].
- Symmetric imitation win-lose games can be represented by simple graphs and determining Nash equilibria corresponds to an analysis of (or a search for) special structures in graphs [BIL08, CS05a].
- There is a corresponding one-to-one relation between Nash equilibria of two-player games and Nash strategies for the row player in an imitation game [CS05a].
- Finding the support is perhaps the easiest milestone in computing Nash equilibria and the support size seems a reasonable parameter to parameterize the quest for Nash strategies.

Results:

- Theorem 4.1.3: We show that the k-UNIFORM NASH is **W[2]**-complete, even for win-lose games, therefore, it is unlikely to be fixed-parameter tractable.

- Lemma 3.1.7: We show that a sample uniform Nash equilibrium in a subclass of win-lose games, symmetric imitation win-lose games, can be found in polynomial time.

- Theorem 3.1.11: We show that, on symmetric imitation win-lose games where the payoff matrices have at most r non-zero entries in each row and each column, finding a uniform Nash equilibrium with maximum support size can be found in FPT-time with r as the parameter. The games where the payoff matrices have at most r non-zero entries in each row and each column also known as r-sparse.

- Theorem 4.1.2: We show that the k-UNIFORM NASH is in **FPT** on r-sparse symmetric imitation win-lose games.

. . . .

GUARANTEED PAYOFF

Instance : A two-player game $\mathcal{G}$.

Parameter: An integer k.

Question : Does there exist an uniform Nash equilibrium in $\mathcal{G}$ where the column player obtains an expected payoff of at least k?

The answer to this question is more useful than it might appear at first glance. For example, a FPT-time algorithm for this decision problem could be used to find, in FPT-time, the maximum expected payoff that can be guaranteed in a Nash equilibrium [CS08].

Result:

- Theorem 4.2.2: We show that the GUARANTEED PAYOFF is **W[2]**-hard, therefore, it is unlikely to be fixed-parameter tractable.

. . . .

k-MINIMAL NASH SUPPORT**:**

> ***Instance*** : A two-player game $\mathcal{G}$.
>
> ***Parameter***: An integer k.
>
> ***Question*** : Does there exist a Nash equilibrium such that each player uses at most k strategies with positive probability?

This problem is a natural relaxation of the k-UNIFORM NASH problem as we remove the requirements on the uniformity of the distribution and the strict bound on the support size.

Result:

- Theorem 4.3.2: We adapt an earlier **NP**-hardness [GZ89] proof to show that the k-MINIMAL NASH SUPPORT problem is **W[2]**-hard, therefore, it is unlikely to be fixed-parameter tractable in general two-player games.

. . . .

NASH EQUILIBRIUM IN A SUBSET

> ***Instance*** : A two-player game $\mathcal{G}$. A subset of strategies $E_1 \subseteq \{1, \ldots, m\}$ for the row player and a subset of strategies $E_2 \subseteq \{1, \ldots, n\}$ for the column player.
>
> ***Parameter***: $k = \max\{|E_1|, |E_2|\}$
>
> ***Question*** : Does there exist a Nash equilibrium of $\mathcal{G}$ where all strategies not included in E_1 and E_2 are played with probability zero?

Algorithms that answer certain existence questions may shed light on the design of algorithms that construct a Nash equilibrium. For example, an algorithm that could tell us whether there exists any equilibrium where a player plays on a certain set of strategies, could be useful in eliminating possibilities in the search for a Nash equilibrium. Therefore, we study the parameterized complexity of NASH EQUILIBRIUM IN A SUBSET. The NASH EQUILIBRIUM IN A SUBSET problem has been shown to be **NP**-complete [GZ89].

Result:

- Theorem 3.2.1: We show that the NASH EQUILIBRIUM IN A SUBSET is fixed parameter tractable.

1.3 Computing pure Nash equilibria on routing games

We study the parameterized complexity of finding the optimum Nash equilibria for simple routing games where the games consisting of m parallel links and a collection of n users (see Definition 3.3.1). Each user sends its traffic over a link, to control the routing of its own traffic. In a Nash equilibrium, each user selfishly routes its traffic on those links that minimize its latency cost, given the network congestion caused by the other users. The social cost (makespan) of a Nash equilibrium is the expectation of the maximum latency incurred by the accumulated traffic.

BEST NASH FOR ROUTING WITH IDENTICAL LINKS:

Instance : A simple routing game $\mathcal{G}$.

Parameter: An integer k.

Question : Does there exist a pure Nash equilibrium in simple routing game $\mathcal{G}$ such that the social cost (makespan) is bounded by k?

The non-cooperative individual optimization of utility does not always lead to a social optimal outcome, e.g., in the Prisoners' Dilemma the sum of the players' costs is arbitrarily large for the Nash equilibrium with respect to the optimal cost (the social cost of playing (D, D) is -9+(-9)=-18 while it is -1+(-1)=-2 when playing (C, C)). Therefore, game theorists study the inefficiency of Nash equilibria. For example, they introduced the *price of stability* [ADK+08], an inefficiency measure of equilibria, to differentiate between games where all Nash equilibria are inefficient from games where only some of their equilibria are inefficient. Formally, the price of stability of a game is the ratio between the optimum objective function value of a

Nash equilibrium of the game and the optimal outcome. Hence, the computational complexity of finding the optimum Nash equilibria has received attention in the literature [KP09, LA97, GLM+05, GHK+09]. Moreover, the focus of most research was on network congestion games and on routing games.

Here, we focus on routing games. In this class of games the existence of a pure Nash equilibrium is guaranteed [FPT04, Ros73] and a pure Nash equilibrium can be found in polynomial time. However, the BEST NASH FOR ROUTING WITH IDENTICAL LINKS is a **NP**-complete problem [FKK+09].

Result:

- Theorem 3.3.3: We show that the BEST NASH FOR ROUTING WITH IDENTICAL LINKS is fixed-parameter tractable.

. . . .

NUMERICAL THREE DIMENSIONAL MATCHING (**NTDM**)

Instance : Disjoint sets X, Y, and Z, each containing n elements, a weight $w(a) \in \mathbb{N}$ for each element $a \in X \bigcup Y \bigcup Z$.

Parameter: $k \in \mathbb{N}$.

Question : Does there exist a partition of $X \bigcup Y \bigcup Z$ into n disjoint sets $A_1, A_2, \ldots A_n$, such that each A_i contains exactly one element from each of X, Y, and Z, and, for $1 \leq i \leq n$, $\sum_{a \in A_i} w(a) = k$?

When we were exploring the parameterized complexity of finding extreme Nash equilibria, best or worst case Nash equilibria, we found out many **NP**-hardness results are based on a reduction from the NUMERICAL THREE DIMENSIONAL MATCHING problem. Therefore, we became interested in determining the parameterized complexity of this problem. Moreover, some scheduling and game theory problems can be modeled as the NUMERICAL THREE DIMENSIONAL MATCHING problem [APS99]. We need to mention that this parametrization of the problem is different form Jia, Zhang, and Chen's [JZC04] parametrization as they considered the size of partitions as the parameter. They provided a faster FPT algorithm for the known classical **NP**-complete problem, SHORT 3DIMENTIONL MATCHING, where its parameterized complexity has been studied earlier [DF98]. Recently, we noticed that a paper has been published in the proceedings of IPEC 2010 that provides a FPT algorithm for NUMERICAL THREE DIMENSIONAL MATCHING [FGR10]. Our result is an independent result from theirs, and it has been submitted for the journal of discrete algorithm before the appearance of the mentioned paper.

Result:

- Theorem 3.4.2: We show that the NUMERICAL THREE DIMENSIONAL MATCHING is fixed-parameter tractable and as a consequence many partitioning problems such as 3-PARTITION is fixed-parameter tractable.

1.4 Computation of dominant strategies

Next, we apply our parameterized complexity analysis to the study of problems regarding to the notion of dominant strategies. A strategy for a player is (strictly) dominant if, regardless of what any other player does, the strategy earns a better payoff than any other. This is a more elementary notion than Nash equilibrium and it can be used as a preprocessing technique for computing Nash equilibria. If the payoff is strictly better, the strategy is named *strictly dominant* but if it is simply not worse, then it is called *weakly dominant*. For example, in the Prisoners' Dilemma, the strategy defecting (D) is a strictly dominant strategy of the game for each player [2].

Gilboa, Kalai and Zemel [GKZ93] used classical complexity theory and showed that many decision problems regarding computation of domination are **NP**-complete.

[2] For the row player, if the column player chooses to confess, the payoff of defecting for the first player is 0 that is strictly better than cooperating with payoff -10. Similarly, if the column player chooses to defect, then the defect gives payoff -9 to the row player that is strictly bigger than cooperating with payoff -10.

Later, Conitzer and Sandholm [CS05b] extended their hardness results to other classes of games. Here, we study the parameterized complexity of some of the problems that others studied earlier using classical complexity.

MINIMUM MIXED DOMINATING STRATEGY SET

Instance : Given the row player's payoffs of a two-player game $\mathcal{G}$ and a distinguished pure strategy i of the row player.

Parameter: An integer k.

Question : Is there a mixed strategy $\mathbf{x}$ for the row player that places positive probability on at most k pure strategies, and dominates the pure strategy i?

Note that a strategy may fail to be dominated by a pure strategy, but may be dominated by a mixed strategy. Here, we study the parameterized complexity of this problem and focus on some specialization of two-player games as the problem is unlikely to be in **FPT** for general two-player games. Here, we focus on win-lose games and r-sparse games. A game $\mathcal{G}$=(A, B) is called r-sparse if there are at most r nonzero entries in each row and each column of the matrices A and B.

Results:

- Lemma 3.5.4: We show the MINIMUM MIXED DOMINATING STRATEGY SET problem can be solved in polynomial time in win-lose games.
- Theorem 3.5.5: We show that MINIMUM MIXED DOMINATING STRATEGY SET is fixed-parameter tractable for r-sparse games.

. . . .

ITERATED WEAK DOMINANCE (IWD)

Instance : A two-player game and a distinguished pure strategy i.

Parameter: An integer k.

Question : Is there a path of at most k-step of iterated weak dominance that eliminates the pure strategy i?

It is well-known that iterated strict dominance is path-independent, that is, the elimination process will always terminate at the same point, and the elimination procedure can be executed in polynomial time [GKZ93]. In contrast, iterated weak dominance is path-dependent and it is known that whether a given strategy is eliminated in some path is **NP**-complete [CS05b].

Results:

- Theorem 4.5.1: We show that the ITERATED WEAK DOMINANCE problem is **W[2]**-hard, therefore it is unlikely to be fixed-parameter tractable.

. . . .

SUB GAME

Instance : Games $\mathcal{G}$= ($A_{n\times n}$, $B_{n\times n}$) and $\mathcal{G}'$ = ($A'_{k\times k}$, $B'_{k\times k}$).

Parameter: An integer k.

Question : Is there a path of iterated strict dominance that reduces the game $\mathcal{G}$ to the game $\mathcal{G}'$?

Finding the smallest sub-game of a given game $\mathcal{G}$=(A, B) using iterated elimination of strict dominated strategies can be performed in polynomial time. However, if whether a distinguished sub-game of $\mathcal{G}$=(A, B) can be reached (using strict domination) is a **NP**-complete problem [GKZ93].

Results:

- We verify that the **NP**-hardness [GKZ93] result can be extended to a **W[1]**-hardness result for the problem using the same reduction (see Section 3.6).

- We provide some preprocessing rules that verifies the no-instances for the game with payoffs in $\{0, 1\}$.

. . . .

The rest of the thesis is organized as follows. In Section 2 we give formal definitions for game, graph and complexity and review some theorems. In Chapter 3, we show the fixed-parameter tractability results. In Chapter 4 we show our parameterized hardness results. In Chapter 5 we discuss further the implications of our results and some open problems.

Table 1.2: The summary of thesis parameterized results

Problem	Parameter	Class of games	Results
k-Uniform Nash	k the size of uniform Nash equilibrium	symmetric imitation win-lose games	**W[2]**-complete
	k the size of uniform Nash equilibrium and the r	r-sparse symmetric imitation win-lose games	**FPT**
Sample uniform Nash equilibrium	-	symmetric imitation win-lose games	P
Guaranteed payoff	k the bound on payoff	general two-player games	**W[2]**-hard
k-Minimal Nash support	k the upper bound on the size of Nash equilibria	zero-sum games	**W[2]**-hard
Nash Equilibrium In A Subset	k the largest size of the given sets of strategies	general two-player games	**FPT**
Best Nash for Routing with Identical Links	k upper bound on social cost	simple routing games	**FPT**
Numerical Three Dimensional Matching	k the given weight	-	**FPT**
Minimum Mixed Dominating Strategy Set	k the size of support	win-lose games	**P**
Minimum Mixed Dominating Strategy Set	k the size of support	r-sparse games	**FPT**
Iterated Weak Dominance (IWD)	k the number of steps	two-player games	**W[2]**-hard
Sub Game	k the size of sub-game	two-player games	**W[1]**-hard

Chapter 2

Preliminaries

In this chapter, we review relevant concepts from graph theory, computational complexity theory including parameterized complexity, and game theory. This chapter also provides a high-level overview of the objects, related to this thesis, and the techniques used to analyze them. The main purpose is to establish the formal framework for the results that will appear in later chapters. For additional details

a) on graph theory, the reader should refer to Diestel [Die97],

b) on complexity theory, the reader may consult Papadimitriou [Pap93, Pap94], Goldreich [Gol10], and Downey and Fellows [DF98], Chen and Meng [CM08], and Cesati [Ces03], and

c) on game theory, the reader can review Myerson [Mye97], Osborne [Osb03], and Nisan et al [NRTV07].

2.1 Graph theory

A *graph* G is a pair (V, E) of sets where $E \subseteq \{\{u, v\} : u, v \in V\}$. The members of V are called *vertices* and the sets $\{u, v\} \in E$ are called *edges* of G. In this thesis we consider only finite graphs, that is graphs for which V and E are both finite.

Definition 2.1.1. An *induced subgraph* of G= (V, E) is a graph G' = (V', E') such that $V' \subseteq V$, $E' \subseteq E$ and for any v_1, v_2 vertices of V', if $\{v_1, v_2\}$ is in E, then $\{v_1, v_2\}$ is in E'. For a given subset V' of V, the induced subgraph by V' is denoted by $G_{V'}$.

Definition 2.1.2. The *degree* of the vertex v is given by $d(v) = |\{u \in V : \{u, v\} \in E\}|$. A graph G= (V, E) is called *r-regular* if all vertices have degree r. A graph is called *regular* if it is r-regular for some r.

Definition 2.1.3. A graph G= (V, E) is called *complete* if for all $v_1, v_2 \in V$, $\{v_1, v_2\} \in E$. An induced complete subgraph of a graph G= (V, E) is called a *clique* of G. A *maximal clique* is a clique whose vertices are not a proper subset of the vertices of any other clique.

Lemma 2.1.4. *Let G= (V, E) be a graph and V' be a subset V such that $G_{V'}$ has at least one edge. Let $u \in V \setminus V'$. If $G_{V'}$ and $G_{V' \cup \{u\}}$ are regular, then both are cliques.*

Proof. Let $d \geq 1$ be the degree of $G_{V'}$. Then, from the regularity of $G_{V' \cup \{u\}}$, it must have positive degree and there must be an edge $\{u, v\}$ involving u. Then, the degree of u must be the same as the degree of v which is $d + 1$. That is, u is a neighbor of all the vertices of V'. This implies that any vertex of $G_{V' \cup \{u\}}$ has the same degree as u. Hence every vertex in $G_{V'}$ has degree d because in this graph u is not involved. Thus, both graphs are cliques. □

A tree decomposition is a mapping of a graph into a tree that can be used to speed up solving certain problems on the original graph. The treewidth measures the number of graph vertices mapped onto any tree node in an optimal tree decomposition.

Definition 2.1.5. Let G= (V, E) be a graph. A *a tree decomposition* of G is a pair $(\{X_i : i \in I\}, T)$ where each X_i is a subset of V, called a *bag*, and T is a tree with the element of I as nodes, such that

1. $\bigcup_{i \in I} X_i = V$;

2. for every edge $\{u, v\} \in E$, there is an $i \in I$ such that $\{u, v\} \subseteq X_i$; and

3. for all $i, j, k \in I$, if j lies on the path between i and k in T then $X_i \bigcap X_k \subseteq X_j$.

The *width* of $(\{X_i : i \in I\}, T)$ equals $\max\{|X_i| : i \in I\} - 1$. The *treewidth* of G is the minimum k such that G has a tree decomposition of width k and we denote it by $tw(G)$.

2.2 Polynomial time algorithms

Computational complexity studies how much work is required to solve different problems. It provides a useful classification tool for practitioners, especially when tackling discrete deterministic problems. This theory assigns problems to different complexity classes. It characterizes problems by several parameters: the underlying computational model, a computational paradigm, a resource, and an upper bound on this resource.

The model describes the basic operations that can be used in a computation. A well-known example is the deterministic Turing machine. It provides a simple, useful formal model of the informal concept of effective computation. Moreover, any sequential computation performed on a real computer can be simulated on a Turing machine with only a polynomial-time slowdown.

The paradigm determines how the computation is performed. For example, in a deterministic Turing machine, every intermediate state of a computation has exactly one follow up state. Finally, a particular resource, like the time or space required for the computation, can characterize a complexity class.

Computational tasks refer to objects that are represented in some canonical way, where such canonical representation provides explicit and full description of the corresponding objects. Here, we consider only finite objects like numbers, sets, and graphs. Theses objects can be represented by finite strings.

Definition 2.2.1. A *string* x is a finite sequence of symbols that are chosen from a finite alphabet set Σ. For a natural number n, we denote the set of all strings of length n by Σ^n.

We denote the set of all strings by Σ^*, that is, $\Sigma^* = \bigcup_{n \in \mathbb{N} \cup \{0\}} \Sigma^n$. We also denote the length of a string $x \in \Sigma^*$ by $|x|$.

Definition 2.2.2. An *algorithm* for computing a function $f : \Sigma^* \to \Sigma^*$ consists of a finite set of rules that describes how we can obtain $f(x)$ for an arbitrary input $x \in \Sigma^*$.

Before we introduce our computational model, we illustrate the idea of the model by an example. Consider the process of an algebraic computation by a human using paper. In such a process, at each time, the human looks at some location on the paper, and depending on what he sees and what he has in mind (which is little), he modifies the contents of this location and shifts his look to an adjacent location. The following definition formalizes the idea in the example.

Definition 2.2.3. A (k-tape) Turing machine is given by a tuple $M = (Q, \Sigma, \delta, b, q_s, q_h)$, where Q is a finite set of states, Σ is a finite alphabet, $\delta : Q \times \Sigma^k \to Q \times \Sigma^k \times \{L, R\}^k$ is a finite transition function, $b \in \Sigma$ is a specific blank symbol, and $q_s \in Q$ and $q_h \in Q$ are start and halting state, respectively.

The Turing machine is equivalent to a notepad with k-tapes of infinitely many cells, each of which contains a symbol in Σ. We consider the last tape as output tape that stores the result of the computation. A tape head determines the current position for each tape. A *configuration* consists of a state, the content of each tape, and the position of the heads. The initial configuration is the one with state q_s, the input x in the first $|x|$ cells of the first tape, and heads on the leftmost cell of each tape. The (infinitely many) remaining cells to the right of the input are filled with blanks.

Consider the Turing machine is in the state $q \in Q$, and σ_i is the symbol at the position of the i-th head. Furthermore, consider $\delta(q, \sigma_1, \sigma_2, \ldots, \sigma_k) = (q', \sigma'_1, \sigma'_2, \ldots, \sigma'_k, z_1, \ldots, z_k)$, which describes a new state q'. At this point the entry σ_i replaced by the entry σ'_i. If $z_i = L$, then i-th head will move one cell to the left if such a cell exists. Also, if $z_i = R$, then the head will move one cell to the right. We also assume the transition function δ never to leave the halting state q_h once it has entered it, and not to modify the content of the tapes while in q_h. We interpret the state q_h as *halting*. We are now ready to define what it means for a Turing machine to compute a function under resource constraints. The running time of a Turing machine will be the number of steps before it halts.

Definition 2.2.4. Let $f : \Sigma^* \to \Sigma^*$ and $T : \mathbb{N} \to \mathbb{N}$ be two functions. Then, f can be computed in T-time if there exists a Turing machine M with the following property: for every $x \in \Sigma^*$ if M is started in the initial configuration with input x, then after at most $\max(1, T(|x|))$ steps it halts with $f(x)$ written on its output tape.

Many models of computation allow to speed up computation by a constant factor with respect to the Turing machine model [Gol10]. This motivates ignoring constant factors in stating time complexity upper bounds, and leads to use of the big-O notation. Also in practice, the implementation of an algorithm on a particular computer shows the same order of growth as the input grows for another implementation in another computer.

Definition 2.2.5. Let $f : \mathbb{N} \to \mathbb{N}$ and $g : \mathbb{N} \to \mathbb{N}$ be nonnegative functions on the positive integers. We write $f(n) = O(g(n))$ and say that $f(n)$ is *of order at most* $g(n)$ if there exist constants C_1 and N_1 such that

$$\forall n \geq N_1 \qquad f(n) \leq C_1 g(n).$$

We also say that g is an *asymptotic upper bound* for f.

Now we use two relations to define complexity classes, **P** and **NP**.

Definition 2.2.6. Let S be a finite set. The *characteristic function* χ_S of the set S is a mapping $\chi_S : S \to \{0, 1\}$, given by

$$\chi_S(x) = \begin{cases} 1, & if x \in S; \\ 0, & otherwise. \end{cases}$$

Definition 2.2.7. We say that a relation $R \subseteq \Sigma^* \times \Sigma^*$ is *polynomial-time recognizable* if its characteristic function can be computed in polynomial time by a Turing machine. We call R *polynomially balanced* if there exists a polynomial function $p : \mathbb{N} \rightarrow \mathbb{N}$ such that $(x, y) \in R$ implies $|y| \leq p(|x|)$.

Associated with a relation R, there are three different problems. The decision problem, asks for a given instance $x \in \Sigma^*$, whether there exists a solution $y \in \Sigma^*$ such that $(x, y) \in R$. We usually illustrate the decision problems as follows:

DECISION PROBLEM

Instance : A string $x \in \Sigma^*$ and a relation R.

Question : Does exist a string $y \in \Sigma^*$ such that $(x, y) \in R$.

The search problem is to find a solution, i.e., an element $y \in \Sigma^*$ satisfying $(x, y) \in R$. Finally, the counting problem asks for the number $|\{y \in \Sigma^* : (x, y) \in R\}|$ of solutions for x.

2.3 Decision problems and NP-completness

The decision problem associated with a polynomially balanced relation $R \subseteq \Sigma^* \times \Sigma^*$ can alternatively be interpreted as a language $L_R = \{x \in \Sigma^* : (x, y) \in R \quad \text{for some} \quad y \in \Sigma^*\}$. Moreover, we say that a Turing machine decides a language L if it computes its characteristic function $\chi_L(x) : \Sigma^* \rightarrow \{0, 1\}$. Now, we define the class **P** as the set of all languages L where the number of transitions performed that decides L is bounded by a polynomial function of the length of the input. The class **P** is often used synonymously with tractability (efficient solvability).

Definition 2.3.1. For a function $T : \mathbb{N} \rightarrow \mathbb{N}$, let $DTIME(T)$ be the set of all languages that can be decided in $O(T)$-time. Then, $P = \bigcup_{k \geq 1} DTIME(n^k)$.

We proceed to define the class **NP** of decision problems that can be verified efficiently.

Definition 2.3.2. **NP** is the class of all decision problems associated with polynomial-time recognizable and polynomially balanced relations. In other words, a language L is in **NP** if there is a polynomial-time recognizable and polynomially balanced relation $R \subseteq \Sigma^* \times \Sigma^*$ such that $L_R = L$. The name **NP**, short for nondeterministic polynomial time, derives from the way the class has traditionally been defined.

The relative complexity of different decision problems can be captured in terms of *reductions*. Intuitively, a reduction from one problem to another transforms every instance of the former into an equivalent instance of the latter, where equivalence means that both of them yield the same decision. For this transformation to preserve the complexity of the original problem, the reduction should of course have less power than is required to actually solve the original problem. For comparing problems in **NP**, the type of reduction most commonly used is the one that can itself be computed in (deterministic) polynomial time.

Definition 2.3.3. . A language $P \subseteq \Sigma^*$ is *called polynomial-time (many-one) reducible* to a language $Q \in \Sigma^*$, denoted $P \leq_p Q$, if there exists a function $f : \Sigma^* \to \Sigma^*$ computable in polynomial time such that for every $x \in \Sigma^*$, $x \in P$ if and only if $f(x) \in Q$. A language Q is called **NP**-hard if for every language P in **NP**, $P \leq_p Q$. Moreover, a problem Q is called **NP**-complete if it is **NP**-hard and also contained in **NP**.

The class of **NP**-complete problems is important because if one of them had a polynomial solution then every problem in **NP** has.

2.4 Search problems and PPAD-completeness

In this section, we focus on the complexity of search problems. We define **FNP** [3] as the class of search problems associated with polynomial-time recognizable and poly-

[3] The letter **F** here stands for Function (Function Nondeterministic Polynomial). The complexity class FNP is the function problem extension of the decision problem class NP. The name is somewhat of a misnomer, since technically it is a class of binary relations not functions.

nomially balanced relations. Furthermore, we say that a search problem of a relation R is *self-reducible* if it can be reduced, using an appropriate type of reduction, to the corresponding decision problem of R. The self-reducibility holds for a large class of natural problems, and especially for any search problem such that the corresponding decision problem is **NP**-complete [Gol10]. Therefore, it implies that in many cases we need to consider only the decision version of a problem. However, there are many other search problems where the self-reducibility does not hold [BG94]. For example, **TFNP** [4], a subclass of **FNP**, is the class of all search problems where every instance is guaranteed to have a solution. **TFNP** contains many interesting problems such as the Nash problem, the problem of finding a Nash equilibrium of a given two-player game, because Nash proved that every game has a solution [Nas51]. However, **TFNP** does not have any complete problem [Pap94]. Therefore, a useful and very elegant classification of the problems in **TFNP** was proposed by Papadimitriou [Pap94]. The main idea is based on the following observation:

> "If the problem is total, then there should be a proof showing that it always has a solution. In fact, if the problem is not known to be tractable, then existence proof should be *non-constructive*; otherwise it would be easy to turn this proof into an efficient algorithm."

Here, we group the problems in **TNFP** into complexity classes, according to the non-constructive proof that is needed to establish their totality. These classes are most conveniently defined via complete problems. One of those complexity classes is **PPAD** . It captures the complexity of finding a Nash equilibrium in two-player games [CD05, CD06, CDT06b, DGP05]. The class **PPAD** is based on the following existence proof.

Observation 2.4.1. *(Parity argument on direct-graph): In a directed graph, a node is unbalanced if it has in-degree different from its out-degree. In a directed graph, if there is an unbalanced node, then it must have another unbalanced node.*

[4] The letter **T** here stands for Total (Total Function Nondeterministic Polynomial).

The main issue here is in what sense is proving the observation a non-constructive proof. For example, is finding an unbalanced node in a graph not an easy problem? Here, the answer depends on the way the graph is given. For instance, if the input graph is specified implicitly by a circuit [5] the problem is not as easy as it sounds.

As we mentioned earlier these classes are defined via complete problems. Therefore, we need to define the notion of reducibility among search problems.

Definition 2.4.2. (reducibility between search problems). A search problem $P \in \Sigma^* \times \Sigma^*$ is called *polynomial-time (many-one) reducible* to a search problem $Q \in \Sigma^* \times \Sigma^*$, denoted $P \leq_p Q$, if there exist two functions $f : \Sigma^* \to \Sigma^*$ and $g : \Sigma^* \to \Sigma^*$ computable in polynomial time, such that for every $x \in \Sigma^*$ and for every $y \in \Sigma^*$ such that $(f(x), y) \in Q$, it also holds that $(x, g(y)) \in P$.

Now we introduce the $LEAFD$ problem as our base problem for the definition of the class **PPAD** . Moreover, this problem corresponds to the parity argument on directed graphs.

Definition 2.4.3. ($LEAFD$). The input of the problem is a pair $(M, 0^n)$ where M is a Turing machine which satisfies

- for every $v \in \{0,1\}^n$, $M(v)$ is an ordered pair (u_1, u_2) where $u_1, u_2 \in \{0,1\}^n \cup \{no\}$;
- $M(0^n) = (no, 1^n)$ and the first component of $M(1^n)$ is 0^n.

This instance defines a directed graph $G = (V, E)$ where $V = \{0,1\}^n$, and a pair $(u, v) \in E$ if and only if v is the second component of $M(u)$ and u is the first component of $M(v)$. The output of this problem is a directed leaf of G other than 0^n, where a vertex of V is a directed leaf if its out-degree plus in-degree equals one.

In other words, a $LEAFD$ instance is defined by a directed graph (of exponential size) $G = (V, E)$, where both the in-degree and the out-degree of every node are

[5] For example, a graph on 2^n vertices can be represented by a function (circuit) $S : \{0,1\}^n \to \{0,1\}^n$ that determines the neighborhood of each vertex.

at most 1, together with a pair of polynomial-time functions, $P : V \to V \cup \{no\}$ and $S : V \to V \cup \{no\}$ that compute the predecessor and successor of each vertex, respectively. A pair (u, v) appears in E only if $P(v) = u$ and $S(u) = v$. In addition, a starting source vertex 0^n with in-degree 0 and out-degree 1 is given. The required output is another vertex with in-degree 1 and out-degree 0 (a sink) or with in-degree 0 and out-degree 1 (another source).

Definition 2.4.4. **PPAD** is the set of total **FNP** search problems (**TFNP**) that are polynomial-time reducible to $LEAFD$.

A search problem in **PPAD** is said to be complete if there is a polynomial time reduction from $LEAFD$ to it. For example, finding the Nash equilibria on a two-player game is **PPAD**-complete [CD06].

2.5 Parameterized complexity theory

The theory of **NP**-completeness provides a structure for the study of computationally hard problems. However, the theory does not provide any recommendations for solving hard problems even though many of these hard problems are of great theoretical and practical importance. To solve these hard problems several approaches have been proposed such as polynomial time approximation algorithms [ACG+99], randomized algorithms [MR96] and heuristic algorithms [MF04]. None of these approaches has satisfied all needs requested from applications: polynomial time approximation algorithms can only provide approximate solutions while certain applications may require precise solutions; the success of a randomized algorithm on a problem in general heavily depends on the probabilistic distribution of the problem instances; and heuristic algorithms in general do not have formal performance guarantees.

The theory of parameterized computation and complexity is an approach to deal with hard computational problems arising from industry and applications. The theory is based on the observation that many intractable computational problems in practice are associated with a parameter that varies within a small or moderate range. Therefore, by taking the advantages of the small parameters, many theoretically intractable problems can be solved effectively and practically. The following is an example.

VERTEX COVER

> ***Instance*** : A graph G= (V, E) and a nonnegative integer $k \leq |V| = n$.
>
> ***Parameter***: k.
>
> ***Question*** : Is there a subset $V' \subseteq V$ with k or fewer vertices such that each edge in E has at least one of its endpoints in V'?

The problem is **NP**-complete in general. There is an algorithm of running time $O(2^k|V|)$ for this problem, which is practical for solving the problem when the value of the parameter k is small.

The above algorithm's running time is bounded by a polynomial of the input size n and a function $f(k)$ that depends only on the parameter k. Therefore, the running time becomes acceptable when the parameter value k is small. Now a natural question arises, whether we can find such an algorithm for a computationally intractable problem with small parameter values. Consider the following problem.

DOMINATING SET

> ***Instance*** : A graph G= (V, E) and a nonnegative integer $k \leq |V| = n$
>
> ***Parameter***: An integer k.
>
> ***Question*** : Does G have a dominating set of size at most k? (A dominating set is a set of vertices $V' \subseteq V$ such that for every vertex $v \in V$, there exist a vertex $v' \in V'$ such that $\{v, v'\} \in E$).

In this problem the parameter k, the size of the dominating set, is generally much smaller than the graph size n. The best known algorithm for this important problem runs in time $O(n^{k+1})$.

Let us compare these two algorithms, the one with running time $O(2^k n)$ for the

VERTEX COVER problem and the second with running time $O(n^{k+1})$ for the DOMINATING SET in Table 2.1.

	largest k for $2^k n$	largest k for n^{k+1}
n=10	48	15
n=10^2	45	7
n=10^3	41	5
n=10^6	31	2
n=10^9	21	1

Table 2.1: Largest parameter size that can be handled by a 1-teraflop machine in an hour for different values of n. A 1-teraflop machine takes time 10^{-12} for a single operation [JS03].

Therefore, though both being **NP**-complete, VERTEX COVER and DOMINATING SET seem to be significantly different in terms of their computational complexity on small parameter values. To distinguish these two different kinds of parameterized problems, a formal framework has been established in the theory of fixed-parameter tractability. The framework divides parameterized problems into two fundamental classes: the class of fixed-parameter tractable (**FPT**) problems, such as VERTEX COVER, and the class of fixed-parameter intractable problems, such as DOMINATING SET (see Table 2.2). This classification has successfully been used to establish computational lower bounds for a large number of problems of practical importance [DF98].

VERTEX COVER	DOMINATING SET
Hard problem (**NP**-complete)	Hard problem (**NP**-complete)
fixed-parameter tractable problem	fixed-parameter intractable problem
Early **FPT** alg. $O(2^k n)$	Exhaustive search alg. $O(n^{k+1})$

Table 2.2: Comparing the VERTEX COVER problem vs. the DOMINATING SET problem.

Now we review the fundamental definitions for parameterized tractability and intractability.

The theory of parameterized computation and complexity mainly considers decision problems. Let us start with the definition of parameterized problem. In contrast with classical complexity in parameterized complexity the decision problem is organized in two parts, namely, the input and the parameter.

Definition 2.5.1. A *parameterized problem* is a language $L \subseteq \Sigma^* \times \mathbb{N}$. The second part of the problem is called the parameter.

We point out that a parameterized problem is simply a decision problem in classical complexity theory in which each instance is associated with an identified integer called the parameter. Thus, we can talk about the membership of a parameterized problem in a general complexity class (such as **NP**) and about algorithms solving a parameterized problem.

Definition 2.5.2. A parameterized problem L is *fixed-parameter tractable* if there is an algorithm that decides in $f(k)|x|^{O(1)}$ time whether $(x, k) \in L$, where f is an arbitrary computable function depending only on k. **FPT** denotes the complexity class that contains all fixed-parameter tractable problems.

In order to characterize those problems that do not seem to admit an FPT algorithm, Downey and Fellows [DF98] defined a *parameterized reduction* and a hierarchy of classes **W[1]** $\subseteq$ **W[2]** $\subseteq \ldots \subseteq \ldots$ including likely fixed parameter intractable problems. Each W-class is the closure under parameterized reductions with respect to a kernel problem, which is formulated in terms of special mixed-type boolean circuits in which the number of input lines set to "true" is bounded by a function of the parameter. The following is the formal definition of those ideas.

Definition 2.5.3. Let $L, L' \subseteq \Sigma^* \times \mathbb{N}$ be two parameterized problems. We say L *reduces* to L' by a standard parameterized (many-to-one) reduction if there is a function Φ which transforms an instance (x, k) of L into an instance (x', k') such that

- $(x, k) \in L$ if and only if $(x', k') \in L'$.

- k' depends on k (there exists a function g such that $k' = g(k)$);

- Φ is computable in $f(k)|x|^c$, for some arbitrary function f and constant c.

To discuss the parameterized intractability (or fixed-parameter intractability), we describe a group of satisfiability problems on circuits of bounded depth.

Definition 2.5.4. A Boolean formula in *conjunctive normal form* (CNF) is a conjunction of clauses, each consisting of the disjunction of literals, where a literal is either a Boolean variable or the negation of it.

A circuit C of n variables is a directed acyclic graph, in which each node of in-degree 0 is an input gate and is labeled by either a positive literal x_i or a negative literal $\neg x_i$, where $1 \leq i \leq n$. All other nodes in C are called *gates* and labeled by a Boolean operator either *AND* or *OR*. A designated gate of out-degree 0 in C is the output gate. The size of the circuit C is the number of nodes in C, and the depth of C is the length of the longest path from an input gate to the output gate in C. The circuit C is a Π_t-circuit if its output is an AND gate and its depth is bounded by t. An *assignment* τ to the input variables of the circuit C satisfies C if τ makes the output gate of C have value 1. The *weight* of an assignment is the number of variables assigned value 1 by τ.

The parameterized problem WEIGHTED SATISFIABILITY on Π_t-circuits, abbreviated $WCS[t]$, is defined as follows,

WEIGHTED SATISFIABILITY

Instance : A Π_t-circuit C.

Parameter: An integer k.

Question : Does C have a satisfying assignment of weight k ?

The parameterized problem WEIGHTED CNF-SATISFIABILITY, abbreviated $WCNF - SAT$, is defined as follows, WEIGHTED CNF-SATISFIABILITY

Instance : A CNF Boolean formula F.

Parameter: An integer k.

Question : Does F have a satisfying assignment of weight k ?

Finally, the WEIGHTED 3-CNF-SATISFIABILITY problem, abbreviated $WCNF-3SAT$, is the $WCNF-SAT$ problem whose instances satisfy a further condition that every clause in the CNF Boolean formula F contains at most three literals.

"Extensive computational experience and practice have given strong evidence that the problem $WCNF - 3SAT$ and the problems $WCS[t]$ for all $t \geq 1$ are not **FPT**" [CM08]. The theory of fixed-parameter intractability is built on this working hypothesis, which classifies the levels of fixed-parameter intractability in terms of the parameterized complexity of the problems $WCNF - 3SAT$ and $WCS[t]$.

Definition 2.5.5. The class **W[1]** contains all problems that can be reduced to $WCNF - 3SAT$ by a parameterized reduction.

Definition 2.5.6. A parameterized problem (L, k) is called **W[1]**-hard if the parameterized problem $WCNF - 3SAT$ can be reduced to (L, k) by a parameterized reduction.

A problem in **W[1]** that is **W[1]**-hard is called **W[1]**-complete.

Definition 2.5.7. For each integer $t \geq 2$, the class $W[t]$ consists of all parameterized problems that are reducible to the problem $WCS[t]$ by a parameterized reduction. Note that $WCNF - 3SAT$ is subsumed by $WCS[1]$.

Definition 2.5.8. A parameterized problem (L, k) is called **W[t]**-hard for $t \geq 2$ if the parameterized problem $WCS[t]$ can be reduced to (L, k) by a parameterized reduction.

A problem in **W[t]** that is **W[t]**-hard is called **W[t]**-complete.

Theorem 2.5.9. ***FPT*** $\subseteq$ ***W[1]*** $\subseteq$ ***W[2]*** $\subseteq \ldots$ ***W[t]*** $\subseteq \ldots \subseteq \ldots$ [*DF98*].

In the rest of this section we review some parameterized problems and their parameterized complexity as we may refer to them in this thesis.

SET COVER

Instance : A family $S = \{S_1, \ldots, S_r\}$ of r subsets of the set N = $\{1, ..., n\}$ that covers N, that is $\bigcup_{i=1,\ldots,r} S_i = N$.

Parameter: A positive integer $k \leq r$.

Question : Does S have a subset of size at most k that covers N?

Theorem 2.5.10. *The* SET COVER *problem is not only* ***NP****-complete, but also* ***W[2]****-complete [DF98, GJ79].*

CLIQUE

Instance : A graph G= (V, E).

Parameter:An integer k.

Question : Is there a set $V' \subseteq V$ of size k that forms a clique?

Theorem 2.5.11. *The* CLIQUE *problem is not only* ***NP****-complete but also* ***W[1]****-complete [DF98, GJ79].*

MAX CLIQUE

Instance : A graph G= (V, E).

Parameter: An integer k.

Question : Does G have a maximal clique of size k?

Theorem 2.5.12. *The* MAX CLIQUE *problem is not only* ***NP****-complete but also* ***W[2]****-complete [DF98, GJ79].*

2.6 Games and Nash equilibria

A two-player *normal form* game $\mathcal{G}$ consists of two matrices $A = (a_{ij})_{m\times n}$ and $B = (b_{ij})_{m\times n}$, where a_{ij} denotes the payoff for the first player and b_{ij} denotes the payoff for the second player when the first player plays his i-th strategy and the second player plays his j-th strategy. We identify the first player as the row player and the second player as the column player. The set S = $\{1, 2, \ldots, m\} \times \{1, 2, \ldots, n\}$ is called the *pure strategy space* of the game $\mathcal{G}$ and each member of S is called a pure strategy profile.

We illustrate this notions with the popular game named *Rock, Paper, Scissors* (Fig. 2.1). Opponents face each other, and simultaneously each chooses one and only one of the options scissors, paper or rock. If players choose equally, then both get payoff zero. However, rock gets 1 against scissors that receives a payoff of 0. Scissors wins 1 over paper that gets 0, but paper gets 1 when facing a rock that in this case gets 0. The normal form of this game is usually presented with the matrices A and B in a single table, with each entry showing the payoffs a_{ij}, b_{ij} to each player. Therefore, a row is a pure strategy for the row player while a column is a pure strategy for the column player. Players select their strategy without knowledge of their opponent's choices and the objective of each player is to maximize their payoff individually.

	Rock	Paper	Scissors
Rock	0,0	0,1	1,0
Paper	1,0	0,0	0,1
Scissors	0,1	1,0	0,0

Figure 2.1: Schema showing winners and losers for Rock, Paper, Scissors.

Definition 2.6.1. A pure strategy profile $(i,\, j)$ is called pure Nash equilibrium if for every strategy i' of the row player and strategy j' of the column player we have $a_{ij} \geq a_{i'j}$ and $b_{ij} \geq b_{ij'}$.

In Chapter 1, the Prisoner's Dilemma of Table 1.1 has a pure Nash equilibrium with profile (D, D), as we mentioned earlier. Note that not every game has a pure Nash equilibrium. For example, in Rock, Paper, Scissors, there is no pure strategy Nash equilibria. If the row player plays Paper and the column player plays Paper as well then the row player can improve his payoff by playing Scissors. But if the row player plays Scissors, the column player would rather play Rock. If the column player plays Rock then the row player would rather play Paper , and so on. Because of this fact, the notion of mixed strategies is relevant.

Definition 2.6.2. An ordered n-tuple $\mathbf{x} = (x_1, ..., x_n)$ with $\sum_{i=1}^{n} x_i = 1$ and $\mathbf{x} \geq 0$ is a mixed strategy. We let $\Delta(A)$ be the probability space over the rows of A.

Thus, a mixed strategy is a probability distribution over the pure strategy space and can be considered a way to participate in the game with a randomized algorithm. The *support* (denoted $supp(\mathbf{x})$) of mixed strategy $\mathbf{x}$ is the set of pure strategies which are played with positive probability, that is $\{i : 1 \leq i \leq n, x_i > 0\}$. A *mixed strategy profile* is an ordered pair $(\mathbf{x}, \mathbf{y})$ where $\mathbf{x}$ is a mixed strategy of the row player and $\mathbf{y}$ is a mixed strategy for the column player. If a player uses a mixed strategy to play, then the notion of payoff extends to the notion of expected payoff.

Definition 2.6.3. Let ($\mathbf{x}$, $\mathbf{y}$) be a mixed strategy profile of the game $\mathcal{G}$ = (A, B), then the *expected payoff* of the row player is $\mathbf{x}^T A\mathbf{y}$ and the expected payoff of the column player is $\mathbf{x}^T B\mathbf{y}$.

It is important to note that every finite game has a Nash equilibrium where each player uses a mixed strategy [Nas50].

Definition 2.6.4. A strategy profile $(\mathbf{x}^*, \mathbf{y}^*)$ is a Nash equilibrium of game $\mathcal{G} = (A, B)$ if $\mathbf{x}^{*T} A\mathbf{y}^* \geq \mathbf{x}^T A\mathbf{y}^*$ holds $\forall \mathbf{x} \in \Delta(A)$, and $\forall \mathbf{y} \in \Delta(B)$, we get $\mathbf{x}^{*T} B\mathbf{y}^* \geq \mathbf{x}^{*T} B\mathbf{y}$; and it is called ϵ-Nash equilibrium if $\mathbf{x}^{*T} A\mathbf{y}^* + \epsilon \geq \mathbf{x}^T A\mathbf{y}^*$ holds $\forall \mathbf{x} \in \Delta(A)$, and $\forall \mathbf{y} \in \Delta(B)$, we get $\mathbf{x}^{*T} B\mathbf{y}^* + \epsilon \geq \mathbf{x}^{*T} B\mathbf{y}$.

The above definition implies that a strategy profile $(\mathbf{x}^*, \mathbf{y}^*)$ is a Nash equilibrium if and only if the strategy $\mathbf{x}^*$ of the row player is a best response to the strategy $\mathbf{y}^*$ of the column player and vice versa. Moreover, Theorem 2.6.5 implies that a Nash equilibrium $(\mathbf{x}^*, \mathbf{y}^*)$ can be verified in polynomial time.

Theorem 2.6.5. *[vS02] In a two-player game $\mathcal{G}$=(A, B), the strategy* $\mathbf{x}^*$ *of the row player is a best response to the column player's* $\mathbf{y}^*$ *strategy if and only if*

$$\forall i[i \in supp(\mathbf{x}^*) \Longrightarrow \mathbf{e_i}^T A\mathbf{y}^* \overset{def}{=} (A\mathbf{y}^*)_i = \max_{j=1,\ldots,n} (A\mathbf{y}^*)_j].$$

One of the well-known methods for computing Nash equilibrium is the support searching method. This method is based on the observation that if the support of a potential Nash equilibrium is known, then the computation of Nash equilibrium is equivalent to solving a system of linear inequalities (in case of two-player games). The following definitions formalize this idea.

Definition 2.6.6. For a given game $\mathcal{G}$ = $(A$, $B)$ an ordered pair S'=(S'_1, S'_2) is called a *support profile* if the set $S'_1 \subseteq \{1, \ldots, m\}$ and $S'_2 \subseteq \{1, \ldots, n\}$. In other words, for every $p \in \{1, 2\}$, S'_p is a subset of strategies of player p.

The following Feasibility Program is a direct consequence of Theorem 2.6.5. For a given support profile (S'_1, S'_2) of two-player game $\mathcal{G}$, if there is a Nash equilibrium (**x**, **y**), where $supp(\mathbf{x})$ = S'_1 and $supp(\mathbf{y})$ = S'_2, then it returns such Nash equilibrium.

Feasibility Program

Input: A game $\mathcal{G}$ = $(A$, $B)$ and a support profile S'=(S'_1, S'_2).

Output: A Nash equilibrium (**x**, **y**) such that $supp(\mathbf{x}) = S'_1$ and $supp(\mathbf{y}) = S'_2$ if such a Nash equilibrium exists.

This is equivalent to finding ($\mathbf{x}$, $\mathbf{y}$) and two real values v_1 and v_2 such that

$$\begin{array}{l}
\forall i \in S'_1 \quad \sum_{j=1}^{n} y_j a_{ij} = v_1, \forall i \notin S'_1 \quad \sum_{j=1}^{n} y_j a_{ij} \leq v_1, \\
\forall j \in S'_2 \quad \sum_{i=1}^{m} x_j b_{ij} = v_2, \forall i \notin S'_1 \quad \sum_{j=1}^{n} y_j b_{ij} \leq v_2, \\
\sum_{i=1}^{m} x_i = 1, \forall i \in S'_1 \quad x_i \geq 0, \forall i \notin S'_1 \quad x_i = 0, \\
\sum_{j=1}^{n} y_j = 1, \forall j \in S'_2 \quad y_j \geq 0, \forall i \notin S'_2 \quad y_j = 0.
\end{array}$$

For example, consider the game Rock, Paper, Scissors. The payoff matrix of players are summarized in Figure 2.1. One may ask, does the strategies Rock and Paper for the row player and the strategies Rock and Paper for the column player constitute a Nash equilibrium where only these two strategies are played with positive probability? To answer this question we set up the FEASIBILITY PROGRAM.

Step 1: Suppose S'_1={Rock, Paper} and S'_2={Rock, Paper} are support profiles for the row player and the column player, respectively.

Step 2: Are there a real value v_1, a mixed strategy **x**=(x_1, x_2, x_3) for the row player and a real value v_2, a mixed strategy **y**=(y_1, y_2, y_3) for the column player that satisfy the following linear inequalities?

$$\begin{array}{llll}
 & \text{Rock:} & 0 \cdot y_1 + 0 \cdot y_2 + 1 \cdot y_3 = v_1 & \text{or } y_3 = v_1; \\
\textit{Row player} & \text{Paper:} & 1 \cdot y_1 + 0 \cdot y_2 + 0 \cdot y_3 = v_1 & \text{or } \quad y_1 = v_1; \\
 & \text{Scissors:} & 0 \cdot y_1 + 1 \cdot y_2 + 0 \cdot y_3 \leq v_1 & \text{or } y_2 \leq v_1;
\end{array}$$

$$\begin{array}{llll}
 & \text{Rock:} & 0 \cdot x_1 + 0 \cdot x_2 + 1 \cdot x_3 = v_2 & \text{or } x_3 = v_2; \\
\textit{Column player} & \text{Paper:} & 1 \cdot x_1 + 0 \cdot x_2 + 0 \cdot x_3 = v_2 & \text{or } \quad x_1 = v_2; \\
 & \text{Scissors:} & 0 \cdot x_1 + 1 \cdot x_2 + 0 \cdot x_3 \leq v_2 & \text{or } x_2 \leq v_2;
\end{array}$$

$$x_1 + x_2 + x_3 = 1; \quad x_1, x_2, x_3 \geq 0; \quad x_3 = 0;$$
$$y_1 + y_2 + y_3 = 1; \quad y_1, y_2, y_3 \geq 0; \quad y_3 = 0.$$

Step 3: The above system of inequalities has no solution. Because, when we simplify the inequalities we will end with $1 \leq 0$ which is a contradiction. Therefore, the support profile (S'_1, S'_2) is not a equilibrium profile.

2.7 Imitation Games and Uniform Nash equilibria

McLennan and Tourky initiated the study of computing Nash equilibria in imitation games [MT10]. In an imitation game, the row player receives a payoff 1 if he plays the same strategy as the column player, otherwise he receives 0. In other words, an imitation game can be expressed as a pair of matrices $(I,\ M)$, where I is the identity matrix and M, the payoff matrix of the column player, is a square matrix. The following lemma shows that in any imitation game the row player imitates the column player, that is, the support for the Nash strategy of the row player is a subset of Nash strategy of the column player.

Lemma 2.7.1. *Let* $(\mathbf{x}^*,\ \mathbf{y}^*)$ *be a Nash equilibrium of the imitation game* $(I,\ M)$*, then* $supp(\mathbf{x}^*) \subseteq supp(\mathbf{y}^*)$.

Proof. By contradiction, we let $i \in supp(\mathbf{x}^*)$ and $i \notin supp(\mathbf{y}^*)$ (thus, $y_i^* = 0$). Since $(\mathbf{x}^*, \mathbf{y}^*)$ is a Nash equilibrium, we know $\mathbf{x}^*$ is a best response to $\mathbf{y}^*$. That is,

$$(I\mathbf{y}^*)_i = \max_{j=1,\ldots,n} (I\mathbf{y}^*)_j \quad \text{or,}$$

$$y_i^* = \max_{j=1,\ldots,n} y_j^* \neq 0.$$

This contradicts $y_i^* = 0$. □

The study of imitation games is also justified, because any two-player game $\mathcal{G}$ can be transformed into an imitation game with a one to one relation between Nash equilibria of $\mathcal{G}$ and the Nash strategies of the row player in the corresponding imitation game [CS05a].

Lemma 2.7.2. *Let* $\mathcal{G}$*=(*$A_{n\times n}$*,* $B_{n\times n}$*) be a two-player game and consider the matrix* C *defined by blocks as* $C = \begin{pmatrix} 0 & B \\ A^T & 0 \end{pmatrix}$*. If* C *does not have a zero row, then there is a one-to-one relation between Nash equilibria of* $\mathcal{G}$ *and Nash strategies for the row player in the imitation game (*I*,* C*), where* I *is the identity matrix of size* $2n$ *[CS05a].*

Definition 2.7.3. A mixed strategy $\mathbf{x}$ is called a uniform mixed strategy if for all $i \in supp(\mathbf{x})$, we have $x_i = 1/|supp(\mathbf{x})|$.

A Nash equilibrium ($\mathbf{x}$, $\mathbf{y}$) is called a *uniform* Nash equilibrium if both mixed strategies $\mathbf{x}$ and $\mathbf{y}$ are uniform mixed strategies. For example, the unique Nash equilibrium in Rock, Paper, Scissors game is uniform.

2.8 Dominant strategies

In a two-player game $\mathcal{G}$=(A, B), a strategy i of the row player is said to *weakly dominate* strategy i' of the row player if for every strategy j of the column player we have $a_{ij} \geq a_{i'j}$. The strategy i is said to *strictly dominate* strategy i' if in addition $a_{ij} > a_{i'j}$. A similar definition is used to define the domination relation of the column player, but now using the payoff matrix B.

If a strategy is dominated, the game (and thus the problem) can be simplified by removing it. Eliminating a dominated strategy may enable elimination of another pure strategy that was not dominated at the outset, but is now dominated. The elimination of dominated strategies can be repeated until no pure strategies can be eliminated in this manner. In a finite game this will occur after a finite number of eliminations and will always leave at least one pure strategy remaining for each player. This process requires polynomial time for strict domination and it is called *iterated dominant strategies* [GKZ93]. It is well-known that no pure strategy that is iteratively strictly dominated can be part of any Nash equilibrium. Weakly dominated strategies can be part of a Nash equilibrium, hence the iterated elimination of weakly dominated strategies may discard one or more Nash equilibria of the game. However, at least one Nash equilibrium of the game survives the iterated elimination of weakly dominated strategies. If only one pure strategy survives for each player, this constitutes a Nash equilibrium. For example, in the Prisoners' Dilemma, the strategy defecting (D) is a strictly dominant strategy of the game for each player. Therefore, we eliminate the dominated strategies C for both players. The remaining strategies D for each player constitute a Nash equilibrium.

Note that a strategy may fail to be strongly eliminated by a pure strategy, but may be dominated by a mixed strategy.

Definition 2.8.1. Consider strategy i of the row player in two-player game (A, B). We say that the strategy i is dominated by a mixed strategy $\mathbf{x}$ = $(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$ of the row player, if the following holds for every strategy j of the column player

$$\sum_{i' \neq i} x_{i'} a_{i'j} \geq a_{ij}.$$

Chapter 3

FPT results

In this chapter, we present our FPT results regarding the computation of Nash equilibria and the notion of dominant strategies. Furthermore, we show the fixed-parameter tractability of the NUMERICAL THREE DIMENSIONAL MATCHING problem.

3.1 A FPT result for win-lose games

In this section, we focus on win-lose games. A win-lose game $\mathcal{G}$ is a game where the payoff values of both players are 0 or 1. We discussed the importance of this class of games earlier in Chapter 1. Here, we just focus on the FPT proofs.

The first natural parameter to consider for win-lose games for the computation of a Nash equilibrium is the maximum number of non-zero entries in all rows and all columns of the payoff matrices of the players. This parameter represents the notion of sparsity in graph theory, matrix algebra, and also game theory.

Definition 3.1.1. A win-lose game $\mathcal{G}$=(A, B) is called r-*sparse* if there are at most r nonzero entries in each row and each column of the matrices A and B.

Chen, Deng and Teng [CDT06b] showed that the problem of finding an ϵ-approximate equilibrium for a 10-sparse game is **PPAD**-complete. Therefore, it is unlikely to find a FPT-time algorithm, if we only consider the value r as the parameter. Hence, to reach tractability results, we analyze a restricted version of this class of games. We prove the parameterized tractability of computing Nash equilibria in a subclass of r-sparse games, namely r-sparse imitation symmetric win-lose games.

Definition 3.1.2. Let ISWLG be the class of all *I*mitation *S*ymmetric *W*in-*L*ose *G*ames ($I_{n\times n}$, $M_{n\times n}$) that have the following properties.

- The matrix M is a symmetric matrix,
- for every strategy j in the set of strategies $\{1, \dots, n\}$, m_{jj} is equal to 0 (the matrix has diagonal equal to zero).

A game (I, M) in ISWLG represents a simple undirected graph G= (V, E) where the matrix M corresponds with G's adjacency matrix. Intuitively, a game in ISWLG can be represented as a pursuit-evasion game on graphs: the row player wants to catch the column player, whereas the column player wants to keep an eye on the row player (e.g. be in a vertex adjacent to the one, the row player is in), but does not want to be caught. The following discussion shows a link between games in ISWLG and their corresponding graph representation. In other words, we illustrate that a maximal clique (Definition 2.1.3) in the graph representation of a game $\mathcal{G}$ = (I, M) corresponds to a uniform Nash equilibrium, but the reverse is not true.

Example 3.1.3. *Consider the imitation win-lose game $\mathcal{G}$ = (I, M) where the payoff matrix of the column player is defined as follows:*

$$M = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}.$$

The game $\mathcal{G}$ has a uniform Nash equilibrium of support size four (both players play all their strategies with probability 1/4), but the corresponding graph does not have a maximal clique of size four.

A k-uniform mixed strategy is a mixed strategy profile (x, x) where the mixed strategy x is uniform and the size of support is equal to k.

Observation 3.1.4. *Let G be the graph representation of a game $\mathcal{G}$=(I, M) in ISWLG. Because I is the payoff matrix of the row player, in a Nash equilibrium, the row player always places pure strategies played by the column player in its support (see Lemma 2.7.1). In a k-uniform mixed strategy, the row player has expected payoff $1/k$ always and therefore will maximize its expected payoff by minimizing its support to the support of the column player.*

Observation 3.1.5. *Let G be the graph representation of a game $\mathcal{G}$=(I, M) in ISWLG. Because M defines the payoff of the column player, for the column player, a uniform strategy has an expected payoff equal to $(p-1)/p = 1-1/p$ if the support is a clique of size p (the payoff is 1 everywhere except 0 in the diagonal, and the row player is playing all his strategies with probability $1/p$; this is $(p-1)/p$).*

Lemma 3.1.6. *Let G be the graph representation of a game $\mathcal{G}$=(I, M) in ISWLG and $G_{\mathbf{x}}$ be a maximal clique of size k in graph G. Then the mixed strategy profile* (x, x) *constitutes a uniform Nash equilibrium of the game $\mathcal{G}$ where* x *is defined as follows:*

$$x_i = \begin{cases} 1/k, & \textit{if } i \textit{ is a vertex of } G_{\mathbf{x}}; \\ 0, & \textit{otherwise.} \end{cases}$$

Proof. Let $G_{\mathbf{x}}$ be a maximal clique of size k (we have k-entries in x equal to $1/k$ and all others are 0). The expected payoff of each pure strategy j of the column player in $supp(\mathbf{x})$ is equal to $(\mathbf{x}^T M)_j = 1 \cdot (k-1)/k$. Therefore, all pure strategies in the support of the mixed strategy x for the column player have the same expected payoff. If $k = n$, then G is a complete graph and by Theorem 2.6.5, the mixed strategy profile (x,x) is a uniform Nash equilibrium. If $k < n$, then for any pure strategy $t \notin supp(\mathbf{x})$ there exists at least one pure strategy $j \in supp(\mathbf{x})$ where $\{j, t\} \notin E$. Therefore, the expected payoff of playing the pure strategy t is at most $1 \cdot (k-1)/k$. Theorem 2.6.5 applies again to conclude the mixed strategy profile (x,x) is a Nash equilibrium. □

Clearly, every graph has a maximal clique. Therefore, Lemma 3.1.6 guarantees that every game in ISWLG has a uniform Nash equilibrium. Consequently, the problem of finding a Nash equilibrium of a game $\mathcal{G}$ in ISWLG reduces to a search for a maximal clique in its graph representation. The following lemma shows that a sample (uniform) Nash equilibrium can be found in polynomial time for games in ISWLG.

Lemma 3.1.7. *A sample (uniform) Nash equilibrium in ISWLG can be found in polynomial time.*

Proof. As we discussed in the above paragraph, the problem of finding a sample Nash equilibrium of a game in ISWLG reduces to finding a sample maximal clique in its graph representation. Therefore, this search can be done in polynomial time as follows.

Simply, we pick an edge and consider all other vertices in an attempt to enlarge the clique to a triangle. In each iteration we have a clique and test all vertices outside the clique for enlargement. We keep repeating this process until the clique is maximal. With simple data structures, this process terminates after $O(n^3)$ steps, in the worst case. Therefore, we can find a sample Nash equilibrium in polynomial time. □

Our next step is the search for uniform Nash equilibria that maximize the payoff of the column player for a game in ISWLG. To reach this goal, we need to show that the largest maximal clique in the graph representation of the games guarantees the maximum payoff of playing uniform Nash equilibria. The following theorem and corollary beside Observation 3.1.4 and Observation 3.1.5 approve this claim.

Theorem 3.1.8 ([MS65])**.** *If f^* denotes the optimal objective value of the following indefinite quadratic program*

$$\begin{aligned} \max \quad f(x) &= x^T M x \\ \sum_{i=1}^{n} x_i &= 1, \\ x_i &\geq 0. \end{aligned} \tag{3.1}$$

where M is the adjacency matrix of a graph G. Then, the size of a maximal clique of G is $(1-f^*)^{-1}$.

Corollary 3.1.9. *If (x_0, x_0) is a uniform Nash equilibrium of an imitation game $\mathcal{G}$=(I, M) in ISWLG, then the expected payoff of the column player is at most the payoff of playing a maximum clique.*

Proof. Let (x_0, x_0) be a uniform Nash equilibrium, then the expected payoff of the column player is equal to $x_0^T M x_0$. According to the quadratic program 3.1 the value of $x_0^T M x_0$ is smaller than the optimum value f^* of the program. Therefore, we have

$$x_0^T M x_0 \leq f^*.$$

Furthermore, Theorem 3.1.8 shows that the size of a maximum clique of a graph G with adjacency matrix M is $(1-f^*)^{-1}$. Therefore, the payoff of playing a uniform Nash equilibrium on a maximum clique gives the column player a payoff of $1 - 1/(1-f^*) = f^*$ which is an upper bound for the payoffs of playing any uniform Nash equilibria for the column player. □

However, the problem of finding the largest maximal clique is known as the MAXIMUM CLIQUE problem, and it is a **NP**-complete problem even on sparse graphs [GJ79]. Therefore, the search for such Nash equilibria is a computationally hard problem (**NP**-complete) even on sparse games. We illustrate this problem is **FPT** on r-sparse games in ISWLG by showing that the MAXIMUM CLIQUE problem is **FPT** on r-sparse graphs. Here, we consider the r as the parameter.

Theorem 3.1.10. *A uniform Nash equilibrium that maximizes the payoff for the column player on sparse games in ISWLG can be found in FPT-time with r as the parameter.*

Proof. As we mentioned earlier, the search for a uniform Nash equilibrium that maximizes the payoff of the column player on sparse games can be reduced to search for a maximum clique in its graph representation. Moreover, in this case the search would be in a graph with bounded degree (the degree of each vertex is at most r). This can be performed with complexity in **FPT** time as follows.

Asking if a graph of maximum degree $f(k)$ has a clique of size k is in **FPT** for any fixed-function f [DF98, page 454]. However, because we are dealing with maximal cliques, we adapt this algorithm as follows. For each vertex, we test as many as $\binom{r}{k-1}$ subsets of neighbors if together they are a k-clique. If they are a k-clique, we test the $r - k$ supersets of neighbors to see if they are a containing clique. This testing is needed for k with $0 \leq k \leq r$, because any clique size is bounded by r. This has complexity in **FPT** because it only depends linearly on $|V|$. In summary, we have an algorithm that requires polynomial time in the size of the game and exponential time in r. □

We have used the link between the graph representation of a game and its Nash equilibria to show our tractability results. Now, we can provide many results regarding families of graphs where finding a maximal clique is in **FPT**. For example, we show that a uniform Nash equilibrium that maximizes the payoff of the column player can be found in FPT-time where the tree-width of the graph is considered as the parameter (Theorem 3.1.12). To prove our result, we will use Lemma 3.1.11. Its proof is an adapted version of a proof of a similar Lemma for finding cliques of a given size [Nie06].

Lemma 3.1.11. *For a graph G with given tree decomposition $(\{X_i : i \in I\}, T)$ a maximum clique can be found in $O(2^{\omega} \cdot \omega \cdot |I|)$ time, where ω denotes the width of tree decomposition.*

Proof. It is known that any clique of a graph G is completely contained in a bag of G's tree decomposition [Nie06]. Therefore, to find a maximum clique, the idea is to check all $|I|$ many bags of size at most ω. For each bag X_i, there is at most $2^{|X_i|}$ possibilities to obtain a maximal clique for the subgraph G_{X_i} of G induced by the vertices from X_i. These processes can be performed in $O(2^{\omega} \cdot \omega \cdot |I|)$ time. □

Theorem 3.1.12. *Let $\mathcal{G}$=(I, M) be an imitation symmetric win-lose game with graph representation G. If G has bounded tree-width, then a uniform Nash equilibrium of the $\mathcal{G}$ that maximizes the payoff of the column player can be found in $O(2^{\omega} \cdot \omega \cdot |I|)$ time respect to the width parameter ω.*

Proof. The proof is a direct consequence of Lemma 3.1.6 and Lemma 3.1.11. □

3.2 FPT results when searching Nash equilibrium on a set

In this section, we show the fixed-parameter tractability of the NASH EQUILIBRIUM IN A SUBSET problem.

NASH EQUILIBRIUM IN A SUBSET

Instance : A two-player game $\mathcal{G}$. A subset of strategies $E_1 \subseteq \{1, \ldots, m\}$ for the row player and a subset of strategies $E_2 \subseteq \{1, \ldots, n\}$ for the column player.

Parameter: $k = \max\{|E_1|, |E_2|\}$

Question : Does there exist a Nash equilibrium of $\mathcal{G}$ where all strategies not included in E_1 and E_2 are played with probability zero?

There is a FEASIBILITY PROGRAM [vS02], which is a linear program, and, if the support of a Nash equilibrium is known, then the computation of corresponding Nash equilibrium can be performed in polynomial time (see Section 2.6). We use this FEASIBILITY PROGRAM to prove the following theorem.

Theorem 3.2.1. NASH EQUILIBRIUM IN A SUBSET *is in* ***FPT*** .

Proof. An FPT algorithm proceeds as follows. Given an instance of NASH EQUILIBRIUM IN A SUBSET, (that is a game $\mathcal{G}$ = (A, B) and a subset of strategies E_1 of size at most k for the row player and a subset of strategies E_2 of size at most k for the column player), the algorithm enumerates all subsets E_1' and E_2' of E_1 and E_2, and for each

E_1' it enumerates all subsets E_2' of E_2. It then applies the FEASIBILITY PROGRAM to test if there exists a Nash equilibrium, using E_1' as the support for the row player and E_2' as the support of the column player. If any of these tests of the FEASIBILITY PROGRAM succeeds, we reply indicating it is a yes-instance. If all tests fail, we reply indicating it is a no-instance. The algorithm is correct, because, we have tested all subsets of the input sets E_1 and E_2 exhaustively.

The complexity of the algorithm is $2^{|E_1|\cdot|E_2|}p(|\mathcal{G}|)$ time where p is the polynomial time complexity of the FEASIBILITY PROGRAM. □

3.3 A FPT result for congestion games

In congestion games (also *routing games*), players choose several links, one link to route their traffic.

Definition 3.3.1. A *simple routing game* $\mathcal{G}$ consists of:

- a set of m parallel links from a source node s to a terminal node t and a rate c^l for each link $l \in \{1, 2, \ldots, m\}$,
- a set $N = \{1, 2, \ldots, n\}$, of n users,
- a vector of traffic weights, w_1, w_2, ..., w_n, where the i-th user has traffic $w_i > 0$.

A pure strategy for a user i is a link l in $\{1, 2, \ldots, m\}$. Analogously, a pure strategy profile is an n-tuple $(l_1, l_2, \ldots, l_n)$, when user i chooses link l_i in $\{1, 2, \ldots, m\}$. The *cost* for a user i, when users choose a pure strategy profile $P = (l_1, l_2, \ldots, l_n)$ is $C_i(P) = \sum_{k:l_k=l_i} w_k/c^{l_i}$. In other words, the cost for a user i is the cost of the link it chooses (see Figure 3.1).

A pure strategy profile $P = (l_1, l_2, \ldots, l_n)$ is a pure Nash equilibrium, if no user has incentive to deviate unilaterally from his/her strategy and improve his/her own cost. Here, we consider the makespan as the social objective function that measures the inefficiency of equilibria.

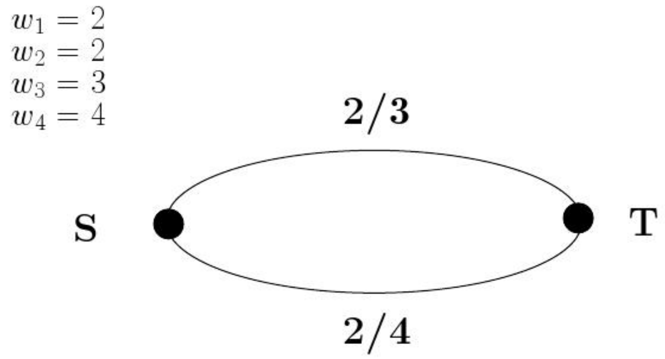

Figure 3.1: A simple congestion game with two links and four users. The costs for users who uses the top link is (2+3 = 5) and for those players using the bottom link is (4+2 = 6). This sitting also is a pure Nash equilibrium.

Definition 3.3.2. The makespan of a strategy profile $P = (l_1, l_2, \dots, l_n)$ is defined as:

$$C_{max}(P) = \max_{i \in \{1,2,\dots,n\}} C_i(P).$$

The BEST NASH FOR ROUTING WITH IDENTICAL LINKS problem on identical links is a **NP**-hard problem [FKK+09], but we show now it is fixed-parameter tractable.

BEST NASH FOR ROUTING WITH IDENTICAL LINKS

Instance : A simple routing game $\mathcal{G}$ with identical links.

Parameter: $k \in \mathbb{N}$.

Question : Is there a pure Nash equilibrium P with $C_{max}(P) \leq k$?

In order to prove our tractability result, we will produce a parameterized reduction to INTEGER LINEAR PROGRAMMING. The INTEGER LINEAR PROGRAMMING problem (with a number of variables bounded by the parameter) is FPT [Nie06].

INTEGER LINEAR PROGRAMMING

Instance : A matrix $A_{m \times k}$ and a vector $\vec{b}_{m \times 1}$ whose entries are integer numbers.

Parameter: $k \in \mathbb{N}$.

Question : Does there exist a non-negative integral vector $\vec{x}$ such that $A \cdot \vec{x} \leq \vec{b}$?

Theorem 3.3.3. BEST NASH EQUILIBRIUM *is in **FPT**.*

Proof. Consider instance I of BEST NASH FOR ROUTING WITH IDENTICAL LINKS and $k \in \mathbb{N}$. If there exists a user i, where $w(i) > k$, then the instance I does not have a solution; hence I is a no-instance. Thus, without loss of generality, we can assume that all weights are equal or less than k. A *pattern* assigned to a link l is a k-tuple $\bar{a}_l = (a_{l,1}, a_{l,2}, \ldots, a_{l,k})$, where $a_{l,i} \in \{0, 1, \ldots, k\}$. A pattern $\bar{a}_l$ indicates that, exactly $a_{l,i}$ users with traffic i placed their traffic to link $l \in \{1, 2, \ldots, m\}$ (for $i \in \{1, 2, \ldots, k\}$).

The idea of proof is to introduce integer variables that indicate, for a given pattern $\bar{a}$, how many links share the same pattern $\bar{a}$. But, we need some consideration regarding the assignment of links. First, the cost of each pattern should not exceed the parameter k. Second, the solution must be a Nash equilibrium.

To satisfy the first condition, we just consider patterns with cost of size at most k. We denote the cost of a pattern $\bar{a} = (a_1, a_2, \ldots, a_k)$ as $C(\bar{a})$ and $C(\bar{a}) = \sum_{i=1}^{k} i \cdot a_i$. Let $\mathcal{C}$ be the set of all patterns $\bar{a} = (a_1, a_2, \ldots, a_k)$ with cost less or equal to the k (i.e $C(\bar{a}) \leq k$). It is easy to see that the cardinality of $\mathcal{C}$ is bounded by $(k+1)^k$. That is, the size of $\mathcal{C}$ is bounded by a function that only depends to the parameter k.

For the second condition, we introduce the notion of *dominance* in the set of patterns. We say that a pattern $\bar{a} = (a_1, a_2, \ldots, a_k)$ is *dominated* by a pattern $\bar{b} = (b_1, b_2, \ldots, b_k)$ (denotes $\bar{a} \prec \bar{b}$) if $\exists i \in \{1, 2, \ldots, k\}$ with $a_i \neq 0$ and $c(\bar{b}) + i < c(\bar{a})$. In other words, if $\bar{a} \prec \bar{b}$, there is a user with traffic i in a link that has an incentive to move to another link.

Now, we can describe a solution by determining for each pattern $\bar{a}$, how many links follow the pattern $\bar{a}$, under the constraint that no two different patterns dominate each other. Therefore, for every pattern $\bar{a} \in \mathcal{C}$, we introduce an integer variable $x_{\bar{a}}$, where the variable $x_{\bar{a}}$ denotes the number of links that follow the pattern $\bar{a}$. Let b_j be the number of users $i \in \{1, 2, \ldots, n\}$ with traffic $w_i = j$, for $j \in \{1, 2, \ldots, k\}$.

The following integer quadratic program is such that if there is a solution to the instance I of BEST NASH FOR ROUTING WITH IDENTICAL LINKS, then the program has a solution.

(1) $\sum_{\bar{a} \in \mathcal{C}} x_{\bar{a}} = m,$

(2) $\forall i \quad (1 \leq i \leq k \Rightarrow \sum_{\bar{a}=(a_1,a_2,\ldots,a_k)} a_i \cdot x_{\bar{a}} = b_i),$

(3) $\forall \bar{a}, \bar{b} \in \mathcal{C} \quad (\bar{a} \preceq \bar{b} \Rightarrow (\text{at least one of} \quad x_{\bar{a}} \quad \text{and} \quad x_{\bar{b}} \quad \text{should be equal to zero.})),$

(4) $\forall \bar{a} \in \mathcal{C} \quad x_{\bar{a}} \in \mathbb{N} \cup \{0\}.$

This is because when I has a solution, we have m links and all are assigned exactly one pattern from $\mathcal{C}$. Since $x_{\bar{a}}$ is the number of links that follows pattern $\bar{a}$, this satisfies Equation (1). Also, Equation (2) would be satisfied if there is a solution, because we are covering all the users' traffic. Moreover, Equations (3) should be satisfied, since it encodes the dominance constraint. In other words, Equation (3) says that if a pattern is dominated by another pattern (a user has the motivation to move to another link), then at least one of those two patterns should not appear in an equilibrium setting.

Now, if the above integer quadratic program has a solution, it would be the case that I has a solution. This is because, Equation (1) ensures that each link is assigned only one pattern. Equation (2) ensures that we assign the right number of users for each value of traffic $j \in \{1, 2, \ldots, k\}$. Hence, a solution of the integer program determines patterns $\bar{a}$'s and the number $x_{\bar{a}}$ of repetitions of each pattern $\bar{a}$ where each pattern in the solution is not dominated by any other pattern in the solution.

At the first glance, conditions in Equation (3) seems not to be linear. However, we can replace each condition with a combination of two linear inequalities by introducing new integer variables z_{ab} as follows:

$$x_{\bar{a}} \leq m \cdot z_{ab} \text{ and } x_{\bar{b}} \leq m \cdot (1 - z_{ab}) \text{ where } z_{ab} \in \{0, 1\} \ .$$

The size of the set $\mathcal{C}$ is bounded by a function of the parameter k. Therefore, the number of combination of choosing two members in set $\mathcal{C}$ is bounded by a function that is only depending to the parameter k. Hence, the number of integer variables z_{ab} is bounded by a function that only depends to the parameter k.

By this substitution, we obtain an integer linear program where the number of variables of the integer program is bounded by the parameter. Hence, this is a reduction to INTEGER LINEAR PROGRAMMING. Thus, the problem is in **FPT** because INTEGER LINEAR PROGRAMMING is in **FPT**, when parameterized by the number of variables. □

3.4 A FPT result on a partitioning problem

In this section, we show the fixed-parameter tractability of the NUMERICAL THREE DIMENSIONAL MATCHING problem. It is an independent result from the computation of Nash equilibrium. However, this problem plays an important role to show **NP**-hardness of many decision problems regarding computation of Nash equilibria as well as many other partitioning problems [GJ79]. The NUMERICAL THREE DIMENSIONAL MATCHING is a **NP**-complete problem [GJ79].

NUMERICAL THREE DIMENSIONAL MATCHING (**NTDM**)

Instance : Disjoint sets X, Y, and Z, each containing n elements, a weight $w(a) \in \mathbb{N}$ for each element $a \in X \bigcup Y \bigcup Z$.

Parameter: $k \in \mathbb{N}$.

Question : Does there exist a partition of $X \bigcup Y \bigcup Z$ into n disjoint sets $A_1, A_2, \ldots A_n$, such that each A_i contains exactly one element from each of X, Y, and Z, and, for $1 \leq i \leq n$, $\sum_{a \in A_i} w(a) = k$?

In this problem, each A_i is called a component of the partition. In order to prove our tractability result, we will produce a parameterized reduction to INTEGER LINEAR PROGRAMMING. Before proving the main theorem, we review an observation.

Observation 3.4.1. *Consider the set* $\mathcal{B} = \{(a, b, c)|a + b + c = k$ *and* $a, b, c \in \mathbb{N}\}$. *We will find the size of* $\mathcal{B}$. *Note that each element (a, b, c) of* $\mathcal{B}$ *is equivalent to splitting* $k - 3$ *co-linear points by two dividers. Thus, the cardinality of* $\mathcal{B}$ *is exactly* $\frac{(k-1)!}{2!(k-3)!}$=$(k - 1)(k - 2)/2$=$\Theta(k^2)$.

Theorem 3.4.2. NUMERICAL THREE DIMENSIONAL MATCHING *is in* ***FPT****.*

Proof. Consider an instance I of NUMERICAL THREE DIMENSIONAL MATCHING and $k \in \mathbb{N}$ be given. If there exists $a \in X \bigcup Y \bigcup Z$, where $w(a) > k$, then instance I does not have a solution, hence I is a no-instance. Therefore, without loss of generality we can assume that all weights are less or equal to k.

For every $i \in \{1, 2, \ldots, k\}$, we consider X_i, the number of elements in X, that have weight equal to i. Similarly, we consider $Y_i = |\{y \in Y|w(y) = i\}|$ and $Z_i = |\{z \in Z|w(z) = i\}|$. Let $\mathcal{C}$ be the set of all triples $\bar{a} = (a_1, a_2, a_3)$ in $\{1, 2, \ldots, k\}^3$, where $\sum_{i=1}^{3} a_i = k$. We call a member of $\mathcal{C}$ a *pattern*. In other words, pattern $\bar{a}$ indicates that numbers a_1, a_2, and a_3 that are assigned to some A_i. That is, there must be x in X and y in Y and z in Z, where $w(x) = a_1$, $w(y) = a_2$, and $w(z) = a_3$, respectively and one of the components is $\{x, y, z\}$.

Observation 3.4.1 assures that the size of $\mathcal{C}$ is bounded by a function that only depends on the parameter k. A solution to NUMERICAL THREE DIMENSIONAL MATCHING can be described by saying, for each pattern $\bar{a} \in \mathcal{C}$, how many of the components A_i have their member according to pattern $\bar{a}$. Therefore, for every pattern $\bar{a} \in \mathcal{C}$, we introduce an integer variable $x_{\bar{a}}$, where the variable $x_{\bar{a}}$ denotes the number of components A_i's that follow the pattern $\bar{a}$.

The following integer linear program is such that if there is a solution to the instance I of NUMERICAL THREE DIMENSIONAL MATCHING, then the program has a solution.

$$\begin{array}{rrcll}
 & \sum_{\bar{a}\in\mathcal{C}} x_{\bar{a}} & = & n, & (3.2)\\
\forall i\in\{1,2,\ldots,k\} & \sum_{\bar{a}=(a_1,a_2,a_3),a_1=i} x_{\bar{a}} & = & X_i, & (3.3)\\
\forall i\in\{1,2,\ldots,k\} & \sum_{\bar{a}=(a_1,a_2,a_3),a_2=i} x_{\bar{a}} & = & Y_i, & (3.4)\\
\forall i\in\{1,2,\ldots,k\} & \sum_{\bar{a}=(a_1,a_2,a_3),a_3=i} x_{\bar{a}} & = & Z_i, & (3.5)\\
\forall \bar{a}\in\mathcal{C} & x_{\bar{a}} & \in & \mathbb{N}_0=\mathbb{N}\cup\{0\}. & (3.6)
\end{array}$$

This is because, when I has a solution, we have n sets A_i and each component A_i must follow exactly one pattern in $\mathcal{C}$. Since $x_{\bar{a}}$ is the number of A_i that follows pattern $\bar{a}$, this satisfies Equation (3.2). Also, equations (3.3)-(3.5) would be satisfied if there is a solution, because we are covering each of X, Y, and Z. Now, if the above integer linear program has a solution, it would be the case that I has a solution. This is because, Equation (3.2) ensures that there is exactly one pattern for every A_i. The equations (3.3)-(3.5) ensure that we assign the right number of elements of each set (i.e X, Y, and Z) of weight $j \in \{1, 2, \ldots, k\}$. Hence, a solution of the integer program determines patterns $\bar{a}$'s and the number $x_{\bar{a}}$ of repetition of each pattern $\bar{a}$.

Since the number of variables of the integer program is bounded by the parameter (the size of the set $\mathcal{C}$), this is a reduction to INTEGER LINEAR PROGRAMMING. Therefore, the problem is in **FPT**, since INTEGER LINEAR PROGRAMMING is in **FPT**, when parameterized by the number of variables. □

3.5 FPT results on mixed strategy domination

In the rest of this chapter, we discuss the parameterized complexity of decision problems regarding to the notion of domination among strategies.

Recall that sometimes a strategy is not dominated by any pure strategy, but is dominated by some mixed strategies. Example 3.5.1 illustrates the differences between these two types of strategies.

Example 3.5.1. *Consider the payoff matrix of the row player that is given as follows:*

$$\begin{pmatrix} 4 & 0 & 2 \\ 0 & 4 & 0 \\ 1 & 1 & 0 \end{pmatrix}.$$

In this situation, no pure strategy can eliminate any other. However playing the first and the second strategy with probability 1/2, dominates the third strategy. Because, the expected payoff of those two strategies is equal to $1/2 \cdot (4, 0, 2) + 1/2 \cdot (0, 4, 0) = (2, 0, 1) + (0, 2, 0) = (2, 2, 1)$.

Moreover, we can test in polynomial time, whether a given strategy of a player is dominated by a mixed strategy of the same player. The following proposition shows the tractability of this issue.

Proposition 3.5.2. *Consider a two-player game $\mathcal{G}$ = ($A_{m\times n}$, $B_{m\times n}$), a subset S' of the row player's pure strategies, and a distinguished strategy i for the row player. We can determine in polynomial time (in the size of the game) whether there exists a mixed strategy $\mathbf{x}$, that places positive probability only on strategies in S' and dominates the pure strategy i. Similarly, for the column player, a subset S' of the column player's pure strategies, and a distinguished strategy j for the column player. We can determine in polynomial time (in the size of the game) whether there exists a mixed strategy $\mathbf{y}$, that places positive probability only on strategies in S' and dominates the pure strategy j. This applies both for strict and weak dominance [CS05b].*

Nevertheless, finding such a mixed strategy that dominates a pure strategy with the smallest support size is computationally hard problem (**NP**-complete) [CS05b]. Therefore, we study the complexity of this problem under the framework of parameterized complexity theory.

MINIMUM MIXED DOMINATING STRATEGY SET

> ***Instance*** : Given the row player's payoffs of a two-player game $\mathcal{G}$ and a distinguished strategy i of the row player.
>
> ***Parameter***: k.
>
> ***Question*** : Is there a mixed strategy $\mathbf{x}$ for the row player that places positive probability on at most k pure strategies, and dominates the pure strategy i?

It is not hard to obtain a proof of **W[2]**-hardness for this problem. The original proof [CS05b] introduced a reduction from SET COVER, a **W[2]**-complete problem, to MINIMUM MIXED DOMINATING STRATEGY SET. We just need to verify that it is a parameterized reduction. In the proof, every instance of the SET COVER problem maps to an instance of MINIMUM MIXED DOMINATING STRATEGY SET, the parameter k in the first problem maps to the parameter k in the second problem and an instance of SET COVER problem has a sub-cover of size at most k if and only if the corresponding instance in MINIMUM MIXED DOMINATING STRATEGY SET problems has mixed strategy with support size at most k that eliminates the given pure strategy in the problem instance. This W-hardness result shows that it is unlikely to find an FPT algorithm for this problem by that parametrization.

Moreover, the review of the proof reveals that the constructed instances of the MINIMUM MIXED DOMINATING STRATEGY SET problem in the reduction have limited payoffs, which are $\{0, 1, k + 1\}$. Therefore, a natural question to ask next is whether it is possible to find an FPT algorithm by considering extra conditions on the problem instances. Our first step would be specializing the games to win-lose games. Recall that in win-lose games, the given payoffs are in $\{0, 1\}$. The following lemma shows that this restriction makes the problem easy.

Lemma 3.5.3. *In a win-lose game $\mathcal{G}$=(A, B) every pure strategy that is weakly dominated by a mixed strategy is also weakly dominated by a pure strategy.*

Proof. Consider a mixed strategy $\mathbf{x} = (x_1, x_2, \ldots, x_m)$ that dominates a pure strategy i (without loss of generality, both of course, of the row player). Clearly, for any strategy j of the column player where $a_{ij} = 0$, the expected payoff of playing the mixed strategy in the column j is at least 0. Therefore, we only need to consider columns j where $a_{ij} = 1$. Let j_0 be a first column where $a_{ij_0} = 1$. Because $\mathbf{x}$ dominates the strategy i there is a row (strategy) r in the mixed strategy $\mathbf{x}$ where $x_r > 0$ and $a_{rj_0} = 1$. We claim that row r weakly dominates row i. We just need to show that $a_{rj} = 1$ for any column j where $a_{ij} = 1$. However, if $a_{rj} = 0$, then for the j-th column we have $\sum_{i=1}^{m} a_{ij}x_i = \sum_{i \neq r} a_{ij}x_i + a_{rj}x_j < 1$. This contradicts the hypothesis that $\mathbf{x}$ dominates i. □

Lemma 3.5.4. MINIMUM MIXED DOMINATING STRATEGY SET *is in* **P** *(that is, it can be decided in polynomial time) if it is limited to win-lose games.*

Proof. By Lemma 3.5.3, if a pure strategy i is dominated by a mixed strategy $\mathbf{x}$, then there exits a pure strategy i' that dominates i. Therefore, the problem reduces to the problem of finding a pure strategy that dominates i. This can be done in polynomial time in the size of the game by using Proposition 3.5.2. □

Our first effort for specializing the problem makes it an easy problem (class **P**). Therefore, instead of limiting the payoffs, we will work on limiting r, the number of non-zero entries in each row and each column of the payoff matrix of the row player. The MINIMUM MIXED DOMINATING STRATEGY SET problem remains **NP**-complete even on r-sparse games with $r \geq 3$ [CS05b, GJ79].

Theorem 3.5.5. MINIMUM MIXED DOMINATING STRATEGY SET *problem for r-sparse games (when considering r and k as the parameters) is in the class* ***FPT***.

Proof. Consider an r-sparse instance of MINIMUM MIXED DOMINATING STRATEGY SET. Without loss of generality we can assume the last row of the first player is the strategy to be dominated by a mixture of another k strategies. Because of Proposition 3.5.2, finding a mixed strategy that weakly dominates the distinguished strategy reduces to the problem of determining the support of the mixed strategy. Consider the following procedure.

Step 1: We remove (in polynomial time) all columns where the last row has a zero payoff. Because, all payoffs are at least zero in each column, any mixed strategy that dominates those columns with positive entries of the distinguished strategy also does so where the distinguished strategy has zeros. As the game is r-sparse, this step reduces the size of the payoff matrix of the row player to a matrix with r columns.

Step 2: If there is a column where all entries in that column are less than the last entry in the column, then the instance is a no-instance.

Step 3: Now remove all rows that are made completely of zeros. Because there are at most r entries different than zero in each column, the matrix now has at most r^2 rows. We can test exhaustively all subsets of rows of size k of the first $r^2 - 1$ rows for domination of the now r^2-th row. If none of the tests results in domination, we have a no-instance, otherwise we have a yes-instance and a certificate of the domination.

The only step that is not polynomial is the exhaustive verification at the end; however, this is polynomial in r as there are $\binom{r^2-1}{k} = O(r^{2k})$ such subsets. This problem can be solved in $f(r)poly(n)$ because $k < r^2$. □

3.6 Preprocessing rules on a sub-game problem

For given games $\mathcal{G}$ and $\mathcal{G}'$ researchers have been interested on whether there is there a path of iterated elimination of dominated strategies applicable to $\mathcal{G}$ so that we reach the game $\mathcal{G}'$ or a game $\mathcal{G}''$ where the rows and columns of $\mathcal{G}''$ are permutations of the rows and the columns of $\mathcal{G}'$. If there exists such a path we say that the $\mathcal{G}$ reduces to the game $\mathcal{G}'$. finding a game $\mathcal{G}$ reduces to $\mathcal{G}'$ is a **NP**-complete problem. We denote this problem as SUB GAME problem and formalize it as follows.

SUB GAME

Instance : Games $\mathcal{G}$= ($A_{n\times n}$, $B_{n\times n}$) and $\mathcal{G}'$ = ($A'_{k\times k}$, $B'_{k\times k}$).

Parameter: An integer k.

Question : Is there a path of iterated strict dominance that reduces the game $\mathcal{G}$ to the game $\mathcal{G}'$?

The original hardness proof of SUB GAME problem introduced a reduction from CLIQUE, a **W[1]**-complete problem, to SUB GAME [GKZ93]. This proof also provides a parameterized hardness results for the SUB GAME problem. Therefore, the SUB GAME problem is **W[1]**-hard for general two-player games. This parameterized hardness result shows that it is unlikely to find a FPT algorithm for this problem by considering that parameter only. Furthermore, the review of the hardness proof of SUB GAME problem reveals that the constructed instances of the problem in the reduction have limited payoffs, which are $\{0, 1, 2, 3, 4\}$. Therefore, a natural question to ask next is whether it is possible to find an FPT algorithm by considering extra conditions on the problem instances. Our first step would be specializing the games to the win-lose games. Hence, the given payoffs in the matrices are in $\{0, 1\}$. We believe that this problem is still **NP**-complete even with that extra consideration and we are not aware of its classical complexity. Therefore, we just provide some preprocessing rules that determine no-instances in FPT-time and not any FPT algorithm for the problem.

Observation 3.6.1. *In a win-lose game $\mathcal{G}$=(A, B), if for every strategy j of the column player we have $a_{ij} = 1$ and $a_{i'j} = 0$, then the strategy i of the row player strictly dominates the strategy i' of the row player. Analogously, if for every strategy i of the row player we have $b_{ij} = 1$ and $b_{ij'} = 0$, then the strategy j of the column player strictly dominates the strategy j' of the column player.*

According to the definition of strict dominance, in win-lose games, the only possible elimination for the row player is the elimination of a row full of 0s by a row full of 1s. Similarly, the only possible elimination for the column player is the elimination of a column full of 0s by a column full of 1s. Therefore, if we have a yes-instance of the SUB GAME problem and $k < n$ then there must exist either at least a row full of 1s and also a row full of zeros in the payoff matrix of the row player or at least a column full of 1s and also a column full of zeros in the payoff matrix of the column player.

Observation 3.6.2. *Let [$\mathcal{G}$= (A, B), $\mathcal{G}'$= (A′, B′), k] be a yes-instance of the* SUB GAME *problem. Then, A' appears as a sub-matrix of A and also B' appears as a sub-matrix of B in the same position as A'.*

IFor the simplicity of discussion in the rest of this section, we assume that in every yes-instance of the SUB GAME problem the matrices A' and B' appear in the top left corner of the matrix A and the matrix B, respectively.

Lemma 3.6.3. *In a yes-instance [$\mathcal{G}$=(A, B), $\mathcal{G}'$=(A′, B′), k] of the* SUB GAME *problem we have*

a) if j is a strategy of the column player (with $1 \leq j \leq k$), and i is a strategy of the row player (with $k+1 \leq i \leq n$), then $a_{ij} = 0$, and

b) if i is a strategy of the row player (with $1 \leq i \leq k$), and j is a strategy of the column player (with $k+1 \leq j \leq m$), $b_{ij} = 0$ (see Table 3.1).

Proof. (by contradiction)—a) Assume there exists a strategy i, $k+1 \leq i \leq n$, of the row player and a strategy j, $1 \leq j \leq k$ of the column player such that $a_{ij} = 1$. We claim that the strategy i cannot be eliminated in any path of iterated elimination. This is true as a consequence of Observation 3.6.1. Similar argument proves the part b. □

Now, we can conclude the following corollaries from the above observation and lemma. We can use them as reduction rules to determine the yes-instances.

Corollary 3.6.4. *Consider a yes-instance [$\mathcal{G}$=(A, B), $\mathcal{G}'$=(A′, B′), k] of the* SUB-GAME *problem. Then following facts hold.*

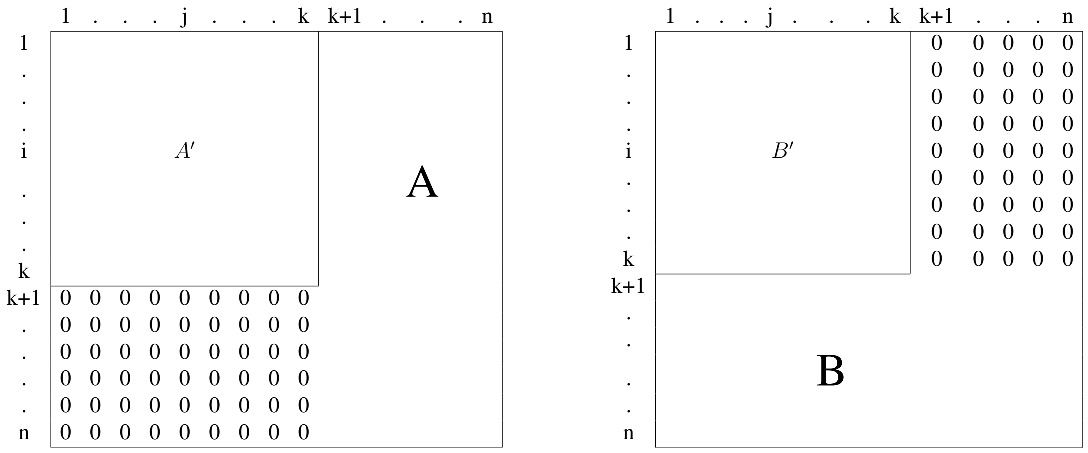
1 . . . j . . . k k+1 . . . n
1 . . . i . . . k k+1 . . . . n
A'
A
0 0 0 0 0 0 0 0 0
1 . . . j . . . k k+1 . . . n
1 . . . i . . . k k+1 . . . n
B'
B
0 0 0 0 0

Table 3.1: The sub-game in the top corner is a yes-instance.

Fact 1: *If a row i of the row player has less than k zero entries, then this row will survive elimination. Similarly, if a column j of the row player has less than k zero entries, then this column will survive elimination.*

Fact 2: *As a consequence of Fact 1, every row of the row player with all entries equal to 1 will survive from elimination. Analogously, a column of the column player with all entries equal to 1 will survive from elimination.*

Fact 3: *Every strategy $j \in \{1, \ldots, k\}$ of the column player with more than k no-zero entries in the payoff matrix of the row player should be eliminated or we have a no-instance. Similarly, every strategy $i \in \{1, \ldots, k\}$ of the row player with more than k no-zero entries in the payoff matrix of the column player should be eliminated or we have a no-instance.*

3.7 Summary of FPT results

We now evaluate the algorithms presented in this chapter. We consider seven algorithms for solving the decision version of the problem. Table 3.2 summarizes the time complexities of the seven algorithms.

Table 3.2: The summary of our FPT results

Problem	**Hardness on class of games**	**Parameter**	**FPT on class of games**	**Time complexity**
Sample Nash equilibrium	PPAD-complete: imitation win-lose games	-	symmetric imitation win-lose games	$O(n^3)$
Maximum payoff of the column player	**NP**-complete: symmetric imitation win-lose games	r	sparse symmetric imitation win-lose games	$r^r O(n)$
Nash Equilibrium In A Subset	**NP**-complete: two-player games	k	two-player games	$2^{2k} poly(n)$
Best Nash for Routing with Identical Links	**NP**-complete: simple routing games	k	simple routing games	$(k+1)^{4k(k+1)^{2k}} poly(n)$
Numerical Three Dimensional Matching	-	k	-	$k^{4k^2} poly(n)$
Minimum Mixed Dominating Strategy Set	**NP**-complete and **W[2]**-hard: two-player games	k	win-lose games	$O(n^2)$
Minimum Mixed Dominating Strategy Set	**NP**-complete: sparse two-player games	k and r	r-sparse games	$r^{2k} poly(n)$

Chapter 4

Parameterized hardness result

In this chapter, we present our parameterized hardness results relevant to the computation of Nash equilibria.

4.1 Finding a uniform Nash equilibria of size k

Earlier we showed that a sample Uniform Nash equilibrium in a subclass of win-lose games (ISWLG) can be found in polynomial time. Moreover, we showed that a uniform Nash equilibrium that maximizes the payoff of the column player can be found in FPT-time on r-sparse games by considering r as the only parameter. Now, a natural question is what is the impact of the sparsity condition on the parameterized complexity of finding uniform Nash equilibria? To answer this question, we prove that it is unlikely to have an FPT algorithm to find a uniform Nash equilibrium of size k (the parameter) in the class ISWLG.

Recall that win-lose games have close links with the notion of clique. Therefore, we adapt and apply parameterized reductions from MAX CLIQUE (See 2.5.12) or closely related problems. First, we establish that the parameterized reduction places the problem in **W[2]**. Then, we show the **W[2]**-hardness of the problem. We use a specialization of win-lose games. Negative complexity results for a special case of a problem are useful because the hardness of the special problem implies the hardness of the general problem as the special problem reduces trivially to the general problem.

Definition 4.1.1. An imitation win-lose game (I, M) in the class ISWLG (see Definition 3.1.2) belongs to the class ISWLG(1) if the following condition holds,

- the matrix M has at least one column i so that, for all $j \neq i$, m_{ji} is equal to 1 (one column has all ones except in the diagonal).

Every graph in this class (ISWLG(1)) has a vertex that is adjacent to all other vertices in the graph. A general imitation win-lose game may not have a symmetric matrix M, and also may not have one column with all ones except in the diagonal.

We showed that sparsity resulted in the parameterized tractability of finding a maximum uniform Nash equilibrium in ISWLG. Therefore, it should not be surprising now that, for decision problems, sparsity enables us to show the problem is in **FPT** for the even more specialized ISWLG(1) games.

(r, k)-UNIFORM NASH (on sparse imitation symmetric win-lose games)

Instance : An imitation win-lose game $\mathcal{G}$=(I, M) in ISWLG(1), where each row and each column with more than r nonzero entries has exactly $n-1$ nonzero entries.

Parameter: Positive integers k and r.

Question : Is there a uniform Nash equilibrium $(\mathbf{x}, \mathbf{x})$ such that $\|supp(\mathbf{x})\| = k$?

The parameterized tractability of the (r, k)-UNIFORM NASH problem is a direct consequence of Lemma 3.1.10 as the degree of each vertex here is either at most r (as the lemma) or $n - 1$.

Theorem 4.1.2. (r, k)-UNIFORM NASH *in ISWLG(1) is in* ***FPT***.

However, we show that if we remove the sparsity condition, then the parameterized tractability will not hold.

k-UNIFORM NASH

Instance : An imitation win-lose game $\mathcal{G}$=(I, M) in ISWLG(1).

Parameter: Positive integers k

Question : Is there a uniform Nash equilibrium $(\mathbf{x}, \mathbf{x})$ such that $\|supp(\mathbf{x})\| = k$?

We now prove that it is unlikely this problem has a fixed-parameter tractable algorithm.

Theorem 4.1.3. k-UNIFORM NASH *(in imitation symmetric win-lose games) is* ***W[2]****-complete.*

Since, for proving **W[2]**-completeness we need to select a suitable **W[2]**-complete problem (note the analogy to a proof that a problem is **NP**-complete), we will produce a parameterized reduction from a suitable variant of MAX CLIQUE. In this specialized version of MAX CLIQUE, we know there is one vertex that has an edge to every other vertex, and we call it s-MAX CLIQUE. First we prove that the s-MAX CLIQUE problem is **W[2]**-complete.

Lemma 4.1.4. *Consider an input instance [graph G= (V, E), k] of* MAX CLIQUE. *We construct a new graph G′=(V′, E′) such that* $V' = V \cup \{u\}$ *where* $u \notin V$ *and* $E' = E \cup \{\{u, v\} : v \in V\}$. *The reduction* $[G{=}\ (V{,}E), k] \mapsto [G' = (V', E'), k' = k + 1]$ *is a parameterized reduction with the property that G has a maximal clique of size k if and only if G′ has maximal clique of size k′.*

Lemma 4.1.5. *Consider [G= (V, E), k] where there is a vertex* u *connected to every vertex* $v \in V$. *We construct a new graph G′=(V′, E′) as follows. The set V′ of vertices is* $V' = V \setminus \{u\}$ *and its edges are* $E' = E \setminus \{\{u, v\} : v \in V\}$. *The reduction* $[G{=}\ (V{,}E), k] \mapsto [G' = (V', E'), k' = k - 1]$ *is a parameterized reduction with the property that G has a maximal clique of size* k *if and only if G′ has maximal clique of size k′.*

The above two lemmas follow directly from Lemma 2.1.4 and prove the following lemma.

Lemma 4.1.6. *Let G= $(V,\ E)$ be a graph with a vertex connected to every vertex $v \in V$ and let k be a positive integer (the parameter). Deciding whether G has a maximal clique of size k is* ***W[2]****-complete.*

4.1.1 Showing our problem is in W[2]

We now show that k-UNIFORM NASH (for imitation win-lose games) is in **W[2]**. By definition, we must show that every instance of k-UNIFORM NASH (for imitation win-lose games) can be parametricly reduced to s-MAX CLIQUE.

We now describe this reduction. Assume an instance of k-UNIFORM NASH (an imitation win-lose game $\mathcal{G}$=$(I,\ M)$ in ISWLG and an integer k) is given. Label the strategies of both players in the input of k-UNIFORM NASH with $S_1 = S_2 = \{1, \ldots, n\}$. As the corresponding instance of s-MAX CLIQUE we consider a graph G= $(V$, $E)$ where the vertices of G are labeled with $V = \{1, \ldots, n\}$ and where the adjacency matrix of G is M (that is, the adjacency matrix is the payoff matrix of the column player).

Clearly for every given instance of a k-UNIFORM NASH (for imitation win-lose games) the corresponding instance of s-MAX CLIQUE can be found in polynomial time in n and with the parameter k' set to k. Thus, if we establish that every yes-instance corresponds to a yes-instance and every no-instance correspond to a no-instance, this would be a parameterized reduction. The following lemma about uniform Nash equilibrium $(\mathrm{x},\ \mathrm{x})$ establishes the link between corresponding instances in the two problems.

Lemma 4.1.7. *The profile $(\mathrm{x},\ \mathrm{x})$ is a k-uniform Nash equilibrium of the imitation game $(I,\ M)$∈ISWLG(1), if and only if G_{x} is a maximal clique of size k, where G_{x} is the induced subgraph over the support of the profile $(\mathrm{x},\ \mathrm{x})$.*

Proof. ($\Leftarrow$) Direct consequence of Lemma 3.1.6.

($\Rightarrow$) Assume the m-th vertex connects to all other vertices of G.

Claim 1 : $m \in \mathit{supp}(\mathbf{x})$.

(Proof of Claim 1:) Suppose $m \notin \mathit{supp}(\mathbf{x})$. This has interesting implications. First, $(\mathbf{x}^T M)_m = 1$ because m is adjacent to all other vertices and $m \notin \mathit{supp}(\mathbf{x})$. Now, let $j \in \mathit{supp}(\mathbf{x})$. Then, the expected payoff when playing j by the column player is $(\mathbf{x}^T M)_j = 1 - 1/\|\mathit{supp}(\mathbf{x})\|$. But then $(\mathbf{x}^T M)_m > (\mathbf{x}^T M)_j$ when $j \in \mathit{supp}(\mathbf{x})$. This contradicts Theorem 2.6.5 because the profile (**x**, **x**) is a Nash equilibrium.

Claim 2: $G_{\mathbf{x}}$ is a clique.

(Proof of Claim 2:) From Theorem 2.6.5, one can find that $\forall i, j \in \mathit{supp}(\mathbf{x})$ we have $(\mathbf{x}^T M)_i = (\mathbf{x}^T M)_j$. That is, their expected payoff are equal. If $d_{G_{\mathbf{x}}}(i)$ denotes the degree of vertex i in $G_{\mathbf{x}}$, we have found that, for every pair i, j in $\mathit{supp}(\mathbf{x})$ the payoff expressions $1 \cdot 1/\|\mathit{supp}(\mathbf{x})\| \cdot d_{G_{\mathbf{x}}}(i)$ and, $1 \cdot 1/\|\mathit{supp}(\mathbf{x})\| \cdot d_{G_{\mathbf{x}}}(j)$ are equal (each term in the expressions is the product of the payoff, times the probability of playing it, times the number of repetitions). Therefore, the degree of vertex i and the degree of j in $G_{\mathbf{x}}$ are the same. Hence we have a regular subgraph which has a vertex that connects to all others, therefore $G_{\mathbf{x}}$ is a clique.

Claim 3: The clique $G_{\mathbf{x}}$ is maximal.

(Proof of Claim 3:) Again, as for Claim 1, a vertex connected to every other vertex in the support that itself is not in the support is a contradiction. If t is connected to every vertex in $\mathit{supp}(\mathbf{x})$, but $t \notin \mathit{supp}(\mathbf{x})$, we have $1 = (\mathbf{x}^T M)_t > (\mathbf{x}^T M)_j$ for any $j \in \mathit{supp}(\mathbf{x})$. This contradicts that (**x**, **x**) is a Nash equilibrium because of Theorem 2.6.5.

□

4.1.2 Proving W[2]-hardness

Interestingly, the inverse of the reduction of the previous subsection works. Given an instance of s-MAX CLIQUE (an undirected graph G= (V, E) and an integer k), its graph has one vertex adjacent to all others. Label the vertices in the input instance of s-MAX CLIQUE with $V = \{1, \ldots, n\}$. As the corresponding instance of k-UNIFORM NASH we consider an imitation win-lose game $\mathcal{G}$=(I, M) in ISWLG such that M is the adjacency matrix of the graph G.

Again, the corresponding instance can be found in polynomial time in the size of G= (V, E) and more importantly, the parameter k' is set to k. To prove this is a reduction, we need to establish the correspondence between yes-instances to yes-instances, and no-instances to no-instances. However, because this is the inverse of the reduction from the previous subsection this follows directly from revisiting the lemmas that establish this correspondence.

4.2 Guaranteed payoff for the column player

Now we show that determining whether there exists a Nash equilibrium in a imitation game where the column player has a guaranteed payoff of at least k is **W[2]**-hard.

MAXIMUM PAYOFF FOR THE COLUMN PLAYER

Instance : An imitation game $\mathcal{G}$=(I, M).

Parameter : Positive integer k.

Question : Does $\mathcal{G}$ have a uniform Nash equilibrium $(\mathbf{x}, \mathbf{x})$ such that the payoff for the column player is at least k?

To prove our hardness result we reduce s-MAX CLIQUE to MAXIMUM PAYOFF FOR THE COLUMN PLAYER. Consider an instance of s-MAX CLIQUE, (G= (V, E), k). We construct an instance ($\mathcal{G}$=(I, M), k') of MAXIMUM PAYOFF FOR THE COLUMN PLAYER where I is identity matrix and the payoff matrix of the column player is defined as follows:

$$m_{ij} = \begin{cases} 2k & \text{if } \{i, j\} \in E, \\ 0 & \text{otherwise.} \end{cases}$$

Note that a clique of size t corresponds to a mixed strategy for the column player with payoff $2k(t-1)/t$ and thus, if the clique has size k, the payoff is $2k-2$. The claims in the proof of Lemma 4.1.7 directly imply the following results.

Corollary 4.2.1. *Let [G= (V, E), integer k] be an instance of* MAX CLIQUE. *The reduction* $[G{=}\ (V{,}E), k] \mapsto [\mathcal{G}(I, M), k' = 2k-2]$ *is a parameterized reduction from* MAX CLIQUE *to* MAXIMUM PAYOFF FOR THE COLUMN PLAYER. *Moreover, the graph G= (V, E) has maximal clique of size k if and only if the $\mathcal{G}$ has a Nash equilibrium in which the column player's payoff is at least k'.*

Theorem 4.2.2. *Let $\mathcal{G}$=(I, M) be an imitation game and k be an integer. Deciding whether $\mathcal{G}$ has an uniform Nash equilibrium in which the column player's payoff is at least k is* ***W[2]****-hard.*

4.3 Finding Nash equilibrium with smallest support in general games

We now show a different concern that also seems very hard to resolve, even with exponential time in the parameter and polynomial time for the size of the input. This will bring more light into the challenges faced when characterizing the complexity of computing Nash equilibria. We show that deciding whether a game has a Nash equilibrium with a support of size equal to or less than a fixed integer number is hard (in the sense of parameterized complexity).

We will prove this for a more specialized type of game, those where one player's loss is the opponent's gain; those are called zero-sum games. The negative result for zero-sum games propagates to more general games.

It is not unusual that a proof of **NP**-completeness does not result in a proof of hardness for parameterized complexity. The INDEPENDENT SET problem can be reduced to the VERTEX COVER problem to show that the later is **NP**-complete. However, the reduction is not a parameterized reduction and the INDEPENDENT SET problem is only known to be in **W[1]** while the VERTEX COVER problem lies in **FPT**. Some proofs of **NP**-complete problems related to Nash equilibria used SATISFIABILITY and CLIQUE as the problem that is reduced by constructing a game [CS08, GZ89], but they are not parameterized reductions. Gilboa and Zemel [GZ89] provide additional properties of the resulting game in the transformation and its Nash equilibria. We follow their reduction [GZ89] to prove our hardness results. We consider the following problem.

k-MINIMAL NASH SUPPORT

Instance : A zero-sum game $\mathcal{G}$=(A, $-A$).

Parameter: Positive integer k.

Question : Does $\mathcal{G}$ have a Nash equilibrium $(\mathbf{x}, \mathbf{y})$ such that $\max\{|supp(\mathbf{x})|, |supp(\mathbf{y})|\} \leq k$?

Our hardness result is based on a parameterized reduction from an instance of SET COVER. SET COVER is **W[2]**-complete [DF98] (see Section 2.5).

Theorem 4.3.1. k-MINIMAL NASH SUPPORT *is **W[2]**-hard.*

Let (N, S, r, k) be an instance of SET COVER. We construct a zero-sum game (A, $-A$) where $A_{(r)\times(n+1)}$ is the payoff matrix of the row player which is defined as follows:

$$a_{ij} = \begin{cases} 1 & \text{if } i \leq r, j \leq n, j \in S_i, \\ 0 & \text{if } i \leq r, i \leq n, j \notin S_i, \\ 1/k & \text{if } i \leq r, j = n+1. \end{cases} \tag{4.1}$$

Clearly, constructing this game given an instance of SET COVER requires polynomial time. Also, the parameter k' in k-MINIMAL NASH SUPPORT is set to k of the

	1	2	3	…	t	…	…	$n+1$
s_1					0			$1/k$
s_2					0			$1/k$
$\vdots$					$\vdots$			$1/k$
s_k					0			$1/k$
s_{k+1}								$1/k$
$\vdots$								$1/k$
s_r								$1/k$

Figure 4.1: The payoff of the strategy t in the proof of Theorem 4.3.2

SET COVER problem. Moreover, the following theorem shows that every yes-instance is mapped to and only to a yes-instances while no-instances are mapped to and only to no-instances. Therefore, this is a parameterized reduction.

Theorem 4.3.2. *The cover S of the set N has a sub-cover of size k or less if and only if the game $\mathcal{G}$=(A, $-A$) defined in Equation 4.1 has a Nash equilibrium such that the size of the support of the Nash strategy is at most k.*

Proof. (⇒) Assume $\mathcal{G}$ has a Nash equilibrium ($\mathbf{x}^*$, $\mathbf{y}^*$) in which the support of both players has at most size k. We claim that $\bigcup_{i\in J} S_i$ is a sub-cover of size at most k for the corresponding SET COVER instance, where $J = supp(\mathbf{x}^*)$. If contrary to the claim, then there exists a column t with $1 \leq t \leq n$ where for all $i \in J$, we have $a_{it} = 0$ (see Figure 4.1). Now we claim that for all $j \in supp(y^*)$ and $i \in supp(x^*)$, we have $a_{ij} = 0$, otherwise the column player has an incentive to play the strategy t because all the payoff values for the column player are either negative or zero, and playing the strategy t gives the column player the maximum payoff which is zero. Therefore, the expected payoff of the column player is equal to 0 and the expected payoff of the row player is also 0 as the game is a zero-sum game (expected payoff of the row player = - (expected payoff of the column player)).

Now consider a strategy $i_0 \notin supp(\mathbf{x}^*)$ for the row player and a strategy $j_0 \in supp(\mathbf{y}^*)$ for the column player where $a_{i_0 j_0} = 1$. The existence of such strategies are guaranteed, because of the above discussion (for all $j \in supp(\mathbf{j}^*)$ and $i \in supp(\mathbf{x}^*)$ we have $a_{ij} = 0$) and the fact that $\bigcup_{i=1}^{r} S_i = S$. Playing the strategy i_0 gives a non-zero payoff to the row player, therefore, the player has an intensive to move to the strategy i. This contradicts to ($\mathbf{x}^*$, $\mathbf{y}^*$) is a Nash equilibrium.

($\Leftarrow$) Let S_{i_1}, S_{i_2}, …, S_{i_k} constitute a sub-cover of size k for N. We define mixed strategies $\mathbf{x}$ and $\mathbf{y}$ for the row and column player as follows:

$$x_i = \begin{cases} 1/k & \text{if } i \in \{i_1, i_2 \dots, i_k\}, \\ 0 & \text{otherwise}, \end{cases} \quad \text{and,}$$

$$y_{n+1} = 1, y_j = 0 \text{ for } j \leq n.$$

Claim: The strategy profile ($\mathbf{x}$, $\mathbf{y}$) is a Nash equilibrium for the game $\mathcal{G}$.

In order to show that ($\mathbf{x}$, $\mathbf{y}$) is a Nash equilibrium, we recall Theorem 2.6.5. Note that $\mathbf{x}$ is the best response to $\mathbf{y}$ and vice versa, because for any $i \in supp(\mathbf{x})$ and for all $j \in supp(\mathbf{y})$ we have

$$\begin{aligned} (A\mathbf{y})_i &= \max_{t=1,\dots,r} (A(y))_t = 1/k \quad \text{and,} \\ -(\mathbf{x}^T A)_i &= \max_{t=1,\dots,n+1} -(\mathbf{x}^T A)_t = -1/k. \end{aligned}$$

□

4.4 Finding Nash equilibrium with smallest support in win-lose games

Theorem 4.1.3 shows that finding a uniform Nash equilibria of size k is unlikely to be **FPT** in win-lose games. Furthermore, Theorem 4.3.2 illustrates that finding Nash equilibria with the smallest support is **W[2]**-hard in two-player games. Therefore, our next step would be a combination of these two problems. That is, we just search for

Nash equilibria of size at most k (considering k as the only parameter) in win-lose games. We know that this problem is **FPT** on r-sparse games [HHKM] (when both k and r are the parameters).

The parameterized hardness proof of finding k-uniform Nash equilibria (Theorem 4.1.3) on ISWLG games (imitation symmetric win-lose games) arises the question whether all Nash equilibria of such games are uniform. We partially answer this question. We show that every Nash equilibrium of imitation games can be transformed to a Nash equilibrium where the mixed strategy of the column player is uniform. We also construct a game in ISWLG with a non-uniform Nash equilibrium.

Lemma 4.4.1. *In an imitation game $\mathcal{G}$=(I, M), if* (**x**, **y**) *is a Nash equilibrium, then there is a Nash equilibrium* (**x**, **y**$'$) *such that the strategy* **y**$'$ *is a uniform mixed strategy and* $supp(\mathbf{x}) = supp(\mathbf{y}')$.

Proof. By Lemma 2.7.1, if (**x**, **y**) is a Nash equilibrium in an imitation game $\mathcal{G}$, then $supp(\mathbf{x}) \subseteqq supp(\mathbf{y})$. Here we claim that the projection of **y** to the support of the row player constitutes a Nash equilibrium. Let **y**$'$ be a uniform distribution over the support of the row player for the column player. We claim that (**x**, **y**$'$) is a Nash equilibrium. Since (**x**, **y**) is a Nash equilibrium, Lemma 2.6.5 implies that for every j in the support of the column player we have $\mathbf{x}^T M\mathbf{e_j} = (\mathbf{x}^T M)_j = \max_{j=1,\dots,n}(\mathbf{x}^T M)_j$. In other words, every strategy j in the support of the column player is a best response to the mixed strategy of the row player. Moreover, from $supp(\mathbf{y}') \subseteqq supp(\mathbf{y})$, we can conclude that every strategy in the support of **y**$'$ is a best response to the strategy of the row player. Now we just need to show that the mixed strategy **x** is a best response to the mixed strategy **y**$'$ of the column player. To do so, we need to show that for every strategy i in the support of **x** we have $\mathbf{e_i}^T I\mathbf{y} = (I\mathbf{y}')_i = \max_{j=1,\dots,n}(I\mathbf{y}')_j$. This is correct as I is the identity matrix and **y**$'$ is a uniform mixed strategy. So, we showed that **x** is a best response to the **y**$'$ and vice versa. Therefore, the strategy profile (**x**, **y**$'$) is a Nash equilibrium. □

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	1	0	1	1	1	1	1	1	1	1	0	0	1	1	1	1	1	1	1	0	0	0	0	0
3	1	1	0	1	1	0	1	1	1	0	1	1	1	1	1	1	1	1	1	0	0	0	0	0
4	1	1	1	0	1	1	0	1	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
5	1	1	1	1	0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
6	1	1	0	1	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	1	1	1	1	1
7	1	1	1	0	1	1	0	0	1	1	1	1	0	0	0	0	0	0	0	1	1	1	1	1
8	1	1	1	1	0	1	0	0	1	1	1	1	0	0	0	0	0	0	0	1	1	1	1	1
9	1	1	1	0	1	1	1	1	0	0	1	1	0	0	0	0	0	0	0	1	1	1	1	1
10	1	1	0	1	1	1	1	1	0	0	1	1	0	0	0	0	0	0	0	1	1	1	1	1
11	1	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	1	1	1	1	1
12	1	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	1	1	1	1	1
13	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0
14	1	1	1	1	1	0	0	0	0	0	0	0	1	0	1	1	1	1	1	0	0	0	0	0
15	1	1	1	1	1	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0	0	0	0	0
16	1	1	1	1	1	0	0	0	0	0	0	0	1	1	1	0	1	1	1	0	0	0	0	0
17	1	1	1	1	1	0	0	0	0	0	0	0	1	1	1	1	0	1	1	0	0	0	0	0
18	1	1	1	1	1	0	0	0	0	0	0	0	1	1	1	1	1	0	1	0	0	0	0	0
19	1	1	1	1	1	0	0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0
20	1	0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1
21	1	0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	1	0	1	1	1
22	1	0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	1	1	0	1	1
23	1	0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	1	1	1	0	1
24	1	0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	1	1	1	1	0

Table 4.1: The payoff matrix of column player in Example 4.4.2.

Example 4.4.2. *In the imitation symmetric win-lose game $\mathcal{G}$=(I, M) with payoff matrix M as shown in Table 4.1 (see also Figure 4.2) , both players have the same number of strategies $\{1, 2, \ldots, 24\}$. In the graph representation of this game, all maximal cliques have size 12, which means all uniform Nash equilibria are size 12. Moreover, there is another Nash equilibrium of size 12 where the support of the row and the column player is equal to $\{1, 2, 3, \ldots, 12\}$. The row player plays his first strategy with probability 3/14 and plays the rest of the strategies in his support with probability 1/14. The column player plays uniformly 1/12 every strategies in his support, where $supp(\mathbf{y}) = \{1, 2, \ldots, 12\}$.*

The above lemma shows that in imitation games the finding of Nash equilibria can be reduced to the search for Nash strategies of the row player. Moreover, the example shows that there exists sub-class of imitation win-lose games such that the Nash equilibria are either uniform or *almost uniform*. A mixed strategy is called almost k-uniform if it is the uniform distribution on a multiset S of pure strategies, with $|S| = k$ [LMM03] [6]. Lipton, Markakis, and Mehta show that there always exists a k-uniform ϵ-Nash equilibrium with k logarithmic in the size of the game [LMM03].

[6] In the original text [LMM03] the term almost k-uniform appeared as k-uniform. Here, we named it almost uniform to prevent conflict with our k-uniform notion. Trivially, every k-uniform mixed strategy is an almost k-uniform.

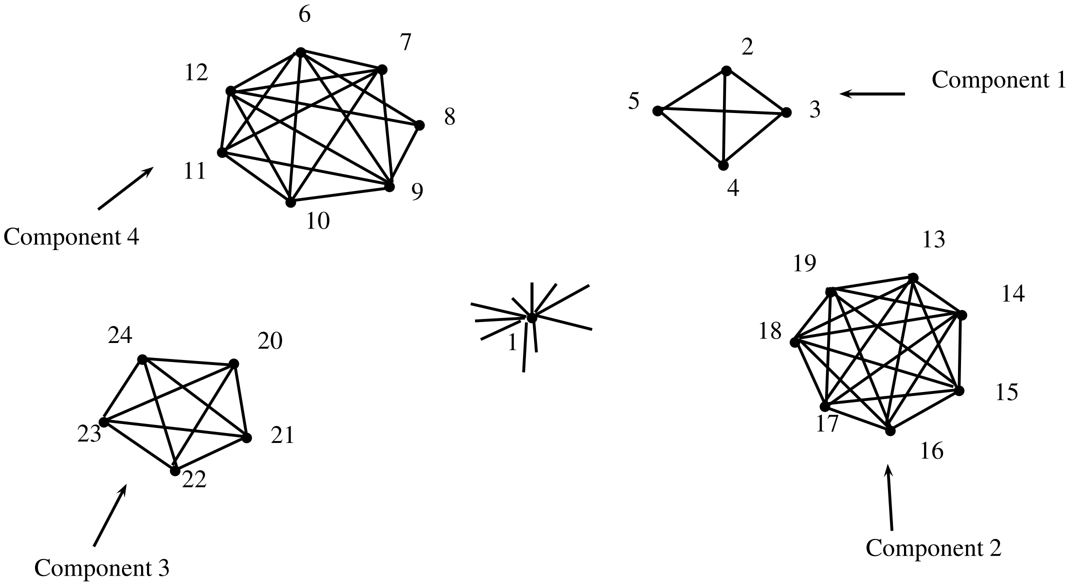

Figure 4.2: The graph represents the payoff of the column player in Example 4.4.2. Each vertex in Component 1 is adjacent to all vertices in Component 2. Also, each vertex in Component 1 is adjacent to at most five vertices in Component 4. Vertex 1 is connected to all vertices. Each vertex in Component 4 is adjacent to all vertices in Component 3.

Theorem 4.4.3. *[LMM03] Consider a game $\mathcal{G}$= ($A_{n\times n}$, $B_{n\times n}$) with all payoffs in* $[0,1]$. *Then for any Nash equilibrium* ($\mathbf{x}$, $\mathbf{y}$) *and any real number ϵ there exists a strategy profile* ($\mathbf{x}'$, $\mathbf{y}'$) *of almost k-uniform (for $k \geq \frac{12ln}{\epsilon^2}$) such that*

1. *($\mathbf{x}'$, $\mathbf{y}'$) is an ϵ-Nash equilibrium,*
2. $|\mathbf{x}'^T A\mathbf{y}' - \mathbf{x}^T A\mathbf{y}| \leq \epsilon$,
3. $|\mathbf{x}'^T B\mathbf{y}' - \mathbf{x}^T B\mathbf{y}| \leq \epsilon$.

Moreover, they show that such an almost k-uniform ϵ-Nash equilibrium can be found in $n^{2k+O(1)}$ time by enumerating all multisets of size at most $k = \frac{12ln}{\epsilon^2}$ for each of the players, and for each of those multisets, check whether it is an ϵ-Nash equilibrium. This result shows that the finding of almost k-uniform ϵ-Nash equilibrium is in XP. Therefore, determining the parameterized complexity of almost k-uniform ϵ-

Nash equilibria may shed light into the parameterized complexity of finding the Nash equilibria of size at most k. Of course, if we provide a parameterized hardness result for the almost k-uniform ϵ-Nash equilibria then the latter problem more likely to be a parameterized hard problem.

4.5 ITERATED WEAK DOMINANCE (IWD)

As discussed earlier, iterated elimination of strictly dominated strategies is conceptually straightforward in a sense that regardless of the elimination order the same set of strategies will be identified, and all Nash equilibria of the original game will be contained in this set. However, this process becomes a bit trickier with the iterated elimination of weakly dominated strategies. In this case, the elimination order does make a difference, that is, the set of strategies that survive iterated elimination can differ depending on the order in which dominated strategies are eliminated. Therefore, the problem such as deciding whether a strategy can be eliminated in a path of iterated weakly dominated absorbed more attention.

ITERATED WEAK DOMINANCE (IWD)

> ***Instance*** : A two-player game and a distinguished strategy i.
>
> ***Parameter***: k.
>
> ***Question*** : Is there a path of at most k-steps of iterated weak dominance that eliminates i?

ITERATED WEAK DOMINANCE is a **NP**-complete problem [CS05b] even in games with payoffs in $\{(0,0),(0,1),(1,0)\}$ [BBFH09]. Here, we show its hardness in terms of parameterized complexity theory.

Theorem 4.5.1. *The* IWD STRATEGY ELIMINATION *problem is* ***W[2]****-hard.*

We prove this by providing a parameterized reduction from SET COVER. Therefore, consider an instance of SET COVER. That is, we are given a set $S =$

$\{1, 2, \ldots, n\}$ and a family $\mathcal{F}$ of proper subsets $S_1, \ldots, S_r$ that cover S (that is, $S_i \subset S$, for $i = 1, \ldots, r$ and $S = \bigcup_{i=1}^{r} S_i$). The question is whether there is a sub-family of k or fewer sets in $\mathcal{F}$ that also covers S.

Our proof constructs, a game $\mathcal{G} = (A, B)$ and the question, whether the last row of the matrix A can be eliminated by iterated weak domination in $k + 1$ or fewer steps. Because k is the parameter of the SET COVER instance and $k' = k + 1$ is the parameter of the ITERATED WEAK DOMINANCE (IWD) this would be a parameterized reduction.

We start by describing the payoff matrices of the game $\mathcal{G} = (A, B)$. The number of rows of the matrices is $|\mathcal{F}| + 1 = r + 1$. The number of columns is $r + n + 1$.

We first describe the payoff matrix A of the row player. The last row of A will be

$$a_{r+1,j} = \begin{cases} 1, & j < n + r + 1, \\ 0, & \text{otherwise.} \end{cases}$$

That is, this row has a 1 everywhere except for the last column.

The last column of A has a similar form.

$$a_{i,n+r+1} = \begin{cases} 1, & i < r + 1, \\ 0, & \text{otherwise.} \end{cases}$$

That is, this column has a 1 everywhere except for the last row.

Now, the first block of r columns and r rows of A have a diagonal full with the value 0 and the value 1 everywhere else. We let the following entries of A defined by

$$a_{i,j} = \begin{cases} 1, & i \leq r \text{ and } j \leq r \text{ and } i \neq j, \\ 0, & i \leq r \text{ and } j \leq r \text{ and } i = j. \end{cases}$$

Finally, after the r-th column, the i-row has the characteristic vector of the set S_i scaled by k.

$$a_{i,j} = \begin{cases} k, & j - r \in S_i \; i \leq r \text{ and } r + 1 \leq j \leq r + n, \\ 0, & j - r \in S \setminus S_i \; i \leq r \text{ and } r + 1 \leq j \leq r + n. \end{cases}$$

We illustrate this construction with an example. Consider the set $S = \{1,2,3,4,5,6,7,8\}$, and the parameter $k = 2$. The family $\mathcal{F}$ is defined as follows

$$S_1 = \{1,2,3\}, \quad S_2 = \{3,5,7\}, \quad S_3 = \{4,5,6\},$$
$$S_4 = \{6,7,8\}, \quad S_5 = \{1,2,4\}, \quad S_6 = \{1,3,5,7\},$$
$$S_7 = \{2,4,6,8\}, \quad S_8 = \{3,4,5\}, \quad \text{and } S_9 = \{2\}.$$

Therefore, the matrix A is given by

$$\left(\begin{array}{ccccccccc|cccccccc|c}
0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 2 & 2 & 2 & 0 & 0 & 0 & 0 & 0 & 1 \\
1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 2 & 0 & 2 & 0 & 2 & 0 & 1 \\
1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 2 & 2 & 2 & 0 & 0 & 1 \\
1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 2 & 2 & 2 & 1 \\
1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 2 & 2 & 0 & 2 & 0 & 0 & 0 & 0 & 1 \\
1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 2 & 0 & 2 & 0 & 2 & 0 & 2 & 0 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 2 & 0 & 2 & 0 & 2 & 0 & 2 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 2 & 2 & 2 & 0 & 0 & 0 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
\hline
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0
\end{array}\right).$$

Observation 4.5.2. *In the resulting matrix A it is impossible to perform a row elimination to eliminate the $(r+1)$-st row.*

Any convex combination of strategies in $\{1,2,\dots,r\}$ would add to less than one in one column in $\{1,2,\dots,r\}$ because at each column there exists a zero entry. Thus, there would be one column blocking such elimination.

Observation 4.5.3. *Consider a yes-instance of the* Set Cover *problem, where I is the set of indices in $\{1,2,\dots,r\}$ such that $|I| \leq k$ and $S \subseteq \bigcup_{i \in I} S_i$. Removing the columns in I from A results in a configuration where the linear combination of rows in I with probability $1/|I|$ eliminate row $r+1$ in one step.*

To confirm this observation first note that any convex combination of rows in $\{1, 2, \ldots, r\}$ produces domination in the $r + n + 1$-th column.

Now we show that the removal of the columns in I causes no longer a blockage. First, consider a column $j \leq r$. Since j is not in I, when we consider the convex combination of rows in I, that combination will add to a payoff of 1, which is equal to the value in row $r + 1$ and column j.

Finally, consider a column j with $r < j < r+n+1$. Because I is the set of indices of a cover, all entries in the rows indexed by I have value k in column j. Therefore the linear combination with uniform probability $1/|I|$ on the rows with index I will have at least one entry with weight $k/|I| \geq 1$ since $|I| \leq k$.

To continue, we now need to describe the payoff matrix B for the column player. This matrix is made of two blocks. The first block is the first r columns, while the second block is the last $n + 1$ columns. All values are 0 for the first block and all values are 1 for the second block.

$$B = \left(0_{r+1 \times r} | 1_{r+1 \times n+1}\right).$$

Observation 4.5.4. *The only columns that can be eliminated by a column elimination are one of the first* r *columns.*

This observation follows trivially from the structure of B, since the only dominations are strict dominations from a column in the later block of columns full of the value 1 to a column in the first r columns full of the value 0.

Observation 4.5.5. *A row elimination cannot happen in matrix* A *until a set* $I \subseteq \{1, 2, \ldots, r\}$ *of columns is eliminated by column eliminations, and the set I defines a cover of* S*.*

We know the process of elimination must start with a column elimination. Because of the structure of the first r columns of A, the only row elimination possible after some columns eliminations must be a liner combination of a subset of indices of the already eliminated indices. However, this would be a possible row elimination only if the linear combination also implies a set cover because of the structure of the next n columns of matrix A.

Now clearly if there is a path of length $k+1$ (or less) that eliminates row $r+1$ in matrix A it must consist of k (or less) column eliminations defining k (or less) indices of the covering sets, and the last elimination is the corresponding row elimination with uniform weight on the same indices. This completes the proof.

4.6 Summary of the hardness results

We now summarize the hardness results presented in this chapter. These four hardeners results are for the decision version of the problem. Table 4.2 summarizes the complexity classes of the four problems.

Table 4.2: The summary of our parameterized hardness results

Problem	**Parameter**	**Class of games**	**Results**
k-UNIFORM NASH	k the size of uniform Nash equilibrium	symmetric imitation win-lose games	**W[2]**-complete
GUARANTEED PAYOFF	k the bound on payoff	general two-player games	**W[2]**-hard
k-MINIMAL NASH SUPPORT	k the upper bound on the size of Nash equilibria	zero-sum games	**W[2]**-hard
k-MINIMAL NASH SUPPORT	k the upper bound on the size of Nash equilibria	win-lose games	Open
ITERATED WEAK DOMINANCE (IWD)	k the upper bound on on the number of steps	two-player games	**W[2]**-hard

Chapter 5

Conclusion and further research proposal

We now conclude the thesis by summarizing its contributions and listing several open problems and potential future research directions.

5.1 Contribution

In this thesis, we have studied the complexity of computing Nash equilibrium, a central notion in algorithmic game theory, and the notion of dominant strategy, using the tools from parameterized complexity theory. Parameterized complexity provides an alternative approach to cope with computational hard problems by considering a small parameter k of the given problem and aiming for algorithms, which are exponential only in the parameter k, and not in the size of input. These types of algorithms are called FPT-algorithms. However, some problems seem not be solvable by a fixed-parameter tractable algorithm.

The bulk of the contributions in this thesis come in the form of W-hardness or FPT results. We studied the parameterized complexity of the problems listed in Table 1.2 and we established new complexity results for specific computational problems (see

Page 16).

1. The k-UNIFORM NASH problem,
2. The sample Nash equilibrium problem,
3. The GUARANTEED PAYOFF problem,
4. The k-MINIMAL NASH SUPPORT problem,
5. The NASH EQUILIBRIUM IN A SUBSET problem,
6. The BEST NASH FOR ROUTING WITH IDENTICAL LINKS problem,
7. The NUMERICAL THREE DIMENSIONAL MATCHING problem,
8. The MINIMUM MIXED DOMINATING STRATEGY SET problem,
9. The ITERATED WEAK DOMINANCE (IWD) problem.

5.2 Open problems and future direction

There are many research directions that one can take in extending the work in this thesis further. Let us start by recapitulating the main open questions.

1. We witnessed that finding Nash equilibria with the smallest support (k-MINIMAL NASH SUPPORT) in symmetric imitation win-lose games seems not to be in **FPT**. However, we did not provide a parameterized hardness result. We conjecture that this problem should be a parameterized hard problem.
2. Codenotti, Leoncini, and Resta showed [CLR06] that a sample Nash equilibrium in 2-sparse win-lose games can be found in polynomial time. Later, Chen, Deng, and Teng [CDT06b] show that finding an approximate Nash equilibrium in 10-sparse games is computationally hard (**PPAD**-complete). In this thesis,

we show that a sample Nash equilibrium (uniform Nash equilibrium) in symmetric imitation win-lose games can be found in polynomial time. Therefore, there are many other subclasses of games between 10-sparse games and symmetric imitation win-lose games to explore.

3. Extreme Nash equilibria (worst or best Nash equilibria) are another direction to explore. These solutions are very useful for studying the inefficiency of Nash equilibria [RT07, KP09, GLM$^+$05]. We show that best Nash equilibria can be found in simple routing games (BEST NASH FOR ROUTING WITH IDENTICAL LINKS) in FPT-time. We just touched the boundary of this problem where the underling topology is a parallel link. However, the parameterized complexity of this problem on other topologies (e.g. series-parallel graphs) is open. Moreover, the parameterized complexity of worst Nash equilibria is still open even on simple routing games.

4. There are many interesting decision problems regarding to the notion of dominant strategies [GKZ93, CS05b]. We show that some of those problems are parameterized hard problem (e.g. MINIMUM MIXED DOMINATING STRATEGY SET). Furthermore, we show that special cases of those problems are in **P** or **FPT**. However, many other problems are still open. For example, the paper "On the Complexity of Iterated Weak Dominance in Constant-Sum Games" by F. Brandt, M. Brill, F. Fischer and P. Harrestein [BBFH09] indicates that ITERATED WEAK DOMINANCE problem is in **P** for Constant-Sum games, but still **NP**-complete for win-lose games. In fact, the paper is more precise. If of each pair of actions (i, j), where i is for the row player and j is for the column player, the corresponding entries in A and B are only $(1, 0), (0, 1), (0, 0)$ the problem remains **NP**-complete. Disallowing $(0, 0)$ makes it constant-sum and thus becomes a problem in **P**. However, our parameterized hardness proof uses other entries different from $\{0, 1\}$. We do not know the fixed-parameter complexity of ITERATED WEAK DOMINANCE in win-lose games or in the more restricted

class of games where only $(1,0),(0,1),(0,0)$ are allowed.

5. In this thesis, we mainly focused on two-player games as most of the decision problems regarding to computation of Nash equilibria or notion of domination is **NP**-complete even on two-player games. Therefore, another direction of research would be the study of multi-player games.

6. Most of the FPT algorithms in this thesis are the first FPT algorithms for the problems. Thus, there are many opportunities to improve their time complexity.

7. Developing efficient algorithms for the computation of Nash equilibria in the non-trivial subclasses of win-lose games. Because, every general two-player game with rational-payoff matrices can be mapped into a win-lose game where the mapping preserves the Nash equilibria in an efficiently recoverable form [AKV05]. We have shown that some decision problems regarding to the computation of Nash equilibria are hard in terms of parameterized complexity even on very restricted version of win-lose games. We suspect that many decision problems may be computationally hard in the theory of parameterized complexity even on win-lose games.

8. In terms of the solution concepts considered, we have focused exclusively on the Nash equilibrium and the notion of dominant strategies, as they have historically played a central role in non-cooperative game theory. However, many other solution concepts have been investigated such as correlated equilibria. For completeness, a possible starting point is the investigation of the parameterized complexity of correlated equilibria for the classes of games discussed in this thesis, since we have provided natural parametrization and our results illustrate the effects of these parameters on the problems' complexity class membership.

Finally, one of the most important future directions is to develop new fixed-prameterize techniques that would be helping us to design FPT algorithms in the field of algorithmic game theory. As, topics like *mechanism design* have not been touched with parameterized analysis yet.

Notation

$\mathcal{G}$ — a two-player game.

$A_{m\times n}$ — the payoff matrix of the row player in a two-player game.

$B_{m\times n}$ — the payoff matrix of the column player in a two-player game.

$\Delta(A)$ — $\Delta(A)$ is the probability space over the rows of A.

m — the number of pure strategies for the row player.

n — the number of pure strategies for the column player.

S — the space set of pure strategies, i.e $S = \{1, \dots, m\} \times \{1, \dots, n\}$.

$I_{n\times n}$ — the identity matrix of size n.

$i \in \{1, \dots, m\}$ — a row (pure strategy) for the row player.

$j \in \{1, \dots, n\}$ — a column (pure strategy) for the column player.

$(i, j) \in S$ — a pure strategy profile.

$\mathbf{x} = (x_1, \dots, x_m)$ — a mixed strategy for the row player, where $\sum_{i=1}^{m} x_i = 1$ and $\forall i \in \{1, \dots, m\} \quad x_i \geq 0$.

$\mathbf{y} = (y_1, \dots, y_n)$ — a mixed strategy for the column player, where $\sum_{j=1}^{n} y_j = 1$ and $\forall j \in \{1, \dots, n\} \quad y_j \geq 0$.

$supp(\mathbf{x})$ — the support of a mixed strategy $\mathbf{x}$.

$(\mathbf{x}, \mathbf{y})$ — a mixed strategy profile, i.e. $\mathbf{x}$ is a mixed strategy for the row player and $\mathbf{y}$ is a mixed strategy for the column player.

e_i^T — a vector where its i-th coordinate is equal 1, and otherwise the entry is zero.

ISWLG — the class of all *I*mitation *S*ymmetric *W*in-*L*ose *G*ames.

ISWLG(1) — the class of all *I*mitation *S*ymmetric *W*in-*L*ose *G*ames with one strategy for the column player who has all payoffs equal to 1 except for the diagonal.

SISWLG — the class of all *S*parse *I*mitation *S*ymmetric *W*in-*L*ose *G*ames.

SISWLG(1) — the class of all *S*parse *I*mitation *S*ymmetric *W*in-*L*ose *G*ames with one strategy for the column player who has all payoffs equal to 1 except the diagonal.

r — the sparsity, that is, the maximum number of non-zero entries in all rows and all columns of a matrix.

c^l — the capacity of a link l in a routing games.

C_{max} — the makespan.

G — a graph.

V — the set of vertices of a graph G.

E — the set of edges of a graph G.

$d(v)$ — the degree of a vertex v in a graph G.

ω — the treewidth of a graph G.

$\{0, 1\}^*$ — the set of all strings with alphabet 0 or 1.

x — a string in $\{0, 1\}^*$.

L — a language, that is $L \subseteq \{0, 1\}^*$.

$|x|$ — the length of a string x.

$\leq_p$ — polynomial time reduction.

$poly(.)$ — a polynomial function.

P — the class of polynomially solvable problems by a deterministic Turing machine.

NP — the class of polynomially solvable problems by a nondeterministic Turing machine.

FPT — the class of fixed-parameter tractable problems.

W[1] — an intractable parameterized complexity class, the class of all parameterized problems that can be reduced by a parameterized reduction to the short Turing machine acceptance problem— deciding if a given nondeterministic single-tape Turing machine accepts within k steps.

W[2] — an intractable parameterized complexity class, the class of all parameterized problems that can be reduced by a parameterized reduction to the short multi-tape Turing machine acceptance problem— deciding if a given nondeterministic multi-tape Turing machine accepts within k steps.

D — a parameterized problem.

k — a positive integer used as the parameter of parameterized problems.

Glossary of Terms

Game: A game consists of a set of players, a set of actions for each player (pure strategies), and a specification of payoffs for each combination of strategies.

Payoff: In any game, payoffs are numbers which represent the motivations of players. Payoffs may represent profit, quantity, utility, or other continuous measures, or may simply rank the desirability of outcomes. In all cases, the payoffs are a model to reflect the motivations of the particular player.

Mixed strategy: A mixed strategy is an assignment of a probability to each pure strategy.

Nash equilibrium: A set of strategies, one for each player, such that all players have no incentive to unilaterally change their decision.

FPT: This denotes the class of fixed-parameter tractable problems.

Computationally hard problems: The problems for which no polynomial-time algorithms are known. In this thesis, the problems discussed are in the class of **NP**-complete or **PPAD**-complete .

Win-lose game: A win-lose game $\mathcal{G}$ is a game where the payoff values of both players are 0 or 1.

Sparse game: A two-player game $\mathcal{G}$=(A, B) is called r-*sparse* if there are at most r nonzero entries in each row and each column of the matrices A and B.

Imitation game: An imitation game is a two-player game where the players have the same set of pure strategies and the goal of the column player is to choose the same pure strategy as the row player.

Index

Bibliography

[ACG$^+$99] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, and M. Protasi. *Complexity and approximation: Combinatorial optimization problems and their approximability properties*. Springer, New York, 1999. ISBN 3540654313.

[ADK$^+$08] E. Anshelevich, A. Dasgupta, J. M. Kleinberg, É. Tardos, T. Wexler, and T. Roughgarden. The price of stability for network design with fair cost allocation. *SIAM Journal on Computing*, 38(4):1602–1623, 2008. ISSN 0097-5397.

[AKV05] T. G. Abbott, D. Kane, and P. Valiant. On the complexity of two-player win-lose games. In *FOCS-05, Proceeding of the 46th Annual IEEE Symposium on Foundations of Computer Science*, pages 113–122. IEEE Computer Society, Pittsburgh, PA, USA, 2005. ISBN 0769524680. Program chair: Tardos, E.

[APS99] C. Arbib, D. Pacciarelli, and S. Smriglio. A three-dimensional matching model for perishable production scheduling. *Discrete Applied Mathematics*, 92(1):1–15, 1999. ISSN 0166-218X.

[BBFH09] F. Brandt, M. Brill, F. Fischer, and P. Harrenstein. On the complexity of iterated weak dominance in constant-sum games. In *SAGT-09, Proceedings of the 2nd International Symposium on Algorithmic Game Theory*, pages 287–298. Springer-Verlag, Berlin, Heidelberg, 2009. ISBN 978-3-642-04644-5.

[BBM07] H. Bosse, J. Byrka, and E. Markakis. New algorithms for approximate nash equilibria in bimatrix games. In X. Deng and F. C. Graham, editors, *WINE-07: Proceedings of the Third International Workshop on Internet and Network Economics, San Diego, CA, USA*, volume 4858 of *Lecture Notes in Computer Science*, pages 17–29. Springer, 2007. ISBN 978-3-540-77104-3.

[BG94] M. Bellare and S. Goldwasser. The complexity of decision versus search. *SIAM Journal on Computing*, 23(1):97–119, 1994. ISSN 0097-5397.

[BIL08] V. Bonifaci, U. D. Iorio, and L. Laura. The complexity of uniform Nash equilibria and related regular subgraph problems. *Theoretical Computer Science*, 401(1-3):144–152, 2008. ISSN 0304-3975.

[CD05] X. Chen and X. Deng. 3-Nash is PPAD-complete. *Electronic Colloquium on Computational Complexity (ECCC)*, 12(134), 2005. ISSN 1433-8092. URL `http://eccc.hpi-web.de/eccc-reports/2005/TR05-134/index.html`.

[CD06] X. Chen and X. Deng. Settling the complexity of two-player Nash equilibrium. In *FOCS-06, Proceddings of 47th Annual IEEE Symposium on Foundations of Computer Science*, pages 261–272. IEEE Computer Society, Berkeley, California, USA, 2006. ISBN 0-7695-2720-5. Program chair: Arora, S.

[CDT06a] X. Chen, X. Deng, and S. H. Teng. Computing Nash equilibria: Approximation and smoothed complexity. *Electronic Colloquium on Computational Complexity (ECCC)*, 13(23), 2006. ISSN

1433-8092. URL `http://eccc.hpi-web.de/eccc-reports/2006/TR06-023/index.html`.

[CDT06b] X. Chen, X. Deng, and S. H. Teng. Sparse games are hard. In P. G. Spirakis, M. Mavronicolas, and S. C. Kontogiannis, editors, *WINE-06, Proceedings of the Second International Workshop on Internet and Network Economics*, volume 4286 of *Lecture Notes in Computer Science*, pages 262–273. Springer, Patras, Greece, 2006. ISBN 3540681388.

[Ces03] M. Cesati. The Turing way to parameterized complexity. *Journal of Computer and System Sciences*, 67(4):654–685, 2003. ISSN 0022-0000.

[CLR06] B. Codenotti, M. Leoncini, and G. Resta. Efficient computation of Nash equilibria for very sparse win-lose bimatrix games. In Y. Azar and T. Erlebach, editors, *ESA-06, Proceedings of the 14th Annual European Symposium on Algorithms*, volume 4168 of *Lecture Notes in Computer Science*, pages 232–243. Springer, 2006. ISBN 3540388753.

[CM08] J. Chen and J. Meng. On parameterized intractability: Hardness and completeness. *The Computer Journal*, 51(1):39–59, 2008. ISSN 0010-4620.

[CS05a] B. Codenotti and D. Stefankovic. On the computational complexity of Nash equilibria for (0,1) bimatrix games. *Information Processing Letters*, 94(3):145–150, 2005. ISSN 0020-0190.

[CS05b] V. Conitzer and T. Sandholm. Complexity of (iterated) dominance. In J. Riedl, M. J. Kearns, and M. K. Reiter, editors, *EC-05, Proceedings of the 6th ACM Conference on Electronic Commerce*, pages 88–97. ACM, Vancouver, BC, Canada, 2005. ISBN 1595930493.

[CS08] V. Conitzer and T. Sandholm. New complexity results about nash equilibria. *Games and Economic Behavior*, 63(2):621 – 641, 2008. ISSN 0899-8256.

[DF98] R. Downey and M. R. Fellows. *Parameterized Complexity*. Monographs in Computer Science. Springer, New York, 1998. ISBN 038794883X.

[DGP05] C. Daskalakis, P. W. Goldberg, and C. H. Papadimitriou. The complexity of computing a Nash equilibrium. *Electronic Colloquium on Computational Complexity (ECCC)*, 12(115), 2005. ISSN 1433-8092. URL `http://eccc.hpi-web.de/`.

[Die97] R. Diestel. *Graph theory*. Springer, New York, 1997. ISBN 0387982108.

[DMP06] C. Daskalakis, A. Mehta, and C. H. Papadimitriou. A note on approximate nash equilibria. In P. G. Spirakis, M. Mavronicolas, and S. C. Kontogiannis, editors, *WINE-06: Proceedings of the Second International Workshop on Internet and Network Economics, Patras, Greece*, volume 4286 of *Lecture Notes in Computer Science*, pages 297–306. Springer, 2006. ISBN 3-540-68138-8.

[FGR10] M. R. Fellows, S. Gaspers, and F. A. Rosamond. Parameterizing by the number of numbers. In V. Raman and S. Saurabh, editors, *IPEC-10: Processings of the 5-th International Symposium on Parameterized and Exact Computation*, volume 6478 of *Lecture Notes in Computer Science*. Springer, Chennai, India, 2010. ISBN 978-3-642-17492-6.

[FKK$^+$09] D. Fotakis, S. Kontogiannis, E. Koutsoupias, M. Mavronicolas, and P. Spirakis. The structure and complexity of Nash equilibria for a selfish routing game. *Theoretical Computer Science*, 410(36):3305–3326, 2009. ISSN 0304-3975.

[FPT04] A. Fabrikant, C. Papadimitriou, and K. Talwar. The complexity of pure Nash equilibria. In L. Babai, editor, *STOC-04, Proceedings of the thirty-sixth Annual ACM Symposium on Theory of Computing*, pages 604–612. ACM, 2004. ISBN 1581138520.

[GHK$^+$09] E. Gassner, J. Hatzl, S.O. Krumke, H. Sperber, and G. J. Woeginger. How hard is it to find extreme Nash equilibria in network congestion games? *Theoretical Computer Science*, 410(47-49):4989–4999, 2009. ISSN 0304-3975.

[GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, USA, 1979. ISBN 0716710447.

[GKZ93] I. Gilboa, E. Kalai, and E. Zemel. The complexity of eliminating dominated strategies. *Mathematics of Operations Research*, 18(3):553–565, 1993. ISSN 0364765X.

[GLM$^+$04] M. Gairing, T. Lücking, M. Mavronicolas, B. Monien, and M. Rode. Nash equilibria in discrete routing games with convex latency functions. In J. Diaz, J. Karhumäki, A. Lepist, and D. Sannella, editors, *ICALP-04, Proceedings of the 31st International Colloquium on Automata, Languages and Programming*, volume 3142 of *Lecture Notes in Computer Science*, pages 645–657. Springer, Turku, Finland, 2004. ISBN 3540228497.

[GLM$^+$05] M. Gairing, T. Lücking, M. Mavronicolas, B. Monien, and P. Spirakis. Structure and complexity of extreme Nash equilibria. *Theoretical Computer Science*, 343(1-2):133–157, 2005. ISSN 0304-3975.

[Gol10] O. Goldreich. *P, NP, and NP-completeness: The Basics of Computational Complexity*. Cambridge University Press, Cambridge; New York, 2010. ISBN 052119248X.

[GW03] S. Govindan and R. Wilson. A global Newton method to compute Nash equilibria. *Journal of Economic Theory*, 110(1):65–86, 2003. ISSN 0022-0531.

[GZ89] I. Gilboa and E. Zemel. Nash and correlated equilibria: Some complexity considerations. *Games and Economic Behavior*, 1(1):80–93, 1989. ISSN 0899-8256.

[HHKM] D. Hermelin, C. C. Huang, S. Kratsch, and Wahlsrtöm M. Parameterized two-player Nash equilibrium. personal communication.

[JS03] R. Johnsonbaugh and M. Schaefer. *Algorithms*. Prentice Hall, Upper Saddle River, NJ, USA, 2003. ISBN 0023606924.

[JZC04] W. Jia, C. Zhang, and J. Chen. An efficient parameterized algorithm for m-set packing* 1. *Journal of Algorithms*, 50(1):106–117, 2004. ISSN 0196-6774.

[KLS01] M. Kearns, M. Littman, and S. Singh. Graphical models for game theory. In J. S. Breese and D. Koller, editors, *UAI-01: Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence*, pages 253–260. Morgan Kaufmann, University of Washington, Seattle, Washington, USA, 2001. ISBN 1558608001.

[KP09] E. Koutsoupias and C. Papadimitriou. Worst-case equilibria. *Computer science review*, 3(2):65–69, 2009. ISSN 1574-0137.

[LA97] R. J. La and V. Anantharam. Optimal routing control: game theoretic approach. In G. G. Yen and O. R. González, editors, *CDC-97, Proceedings of the 36th IEEE Conference on Decision and Control*, volume 3, pages 2910–2915. IEEE Computer Society, Hyatt Regency San Diego, San Diego, USA, 1997. ISBN 0780341872.

[LH64] C. E. Lemke and J. T. Howson. Equilibrium points of bimatrix games. *Journal of the Society for Industrial and Applied Mathematics*, 12(2): 413–423, 1964. ISSN 03684245.

[LMM03] R. J. Lipton, E. Markakis, and A. Mehta. Playing large games using simple strategies. In *EC-2003, Proceedings of the 4th ACM Conference on Electronic Commerce*, pages 36–41. ACM, New York, NY, USA, 2003. ISBN 1-58113-679-X. Program chair: Nisan, N.

[MF04] Z. Michalewicz and D. B. Fogel. *How to solve it: modern heuristics*. Springer, Berlim, 2004. ISBN 3540224947.

[MR96] R. Motwani and P. Raghavan. Randomized algorithms. *ACM Computing Surveys*, 28(1):33–37, 1996. ISSN 0360-0300.

[MS65] T.S. Motzkin and E.G. Straus. Maxima for graphs and a new proof of a theorem of Turan. *Canadian Journal of Mathematics*, 17:533–540, 1965. ISSN 0008-414X.

[MT10] A. McLennan and R. Tourky. Simple complexity from imitation games. *Games and Economic Behavior*, 68(2):683–688, 2010. ISSN 0899-8256.

[Mye97] R. B. Myerson. *Game theory: analysis of conflict*. Harvard University Press, Cambridge, Massachusetts, 1997. ISBN 0674341163.

[Nas50] J. F. Nash. Equilibrium Points in N-Person Games. *Proceedings of the National Academy of Sciences of the United States of America*, 36(1): 48–49, 1950. ISSN 0027-8424.

[Nas51] J. F. Nash. Non-cooperative games. *The Annals of Mathematics*, 54(2): 286–295, 1951. ISSN 0003486X.

[Nie06] R. Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Oxford University Press, Oxford, 2006. ISBN 0198566077.

[NRTV07] N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani. *Algorithmic Game Theory*. Cambridge University Press, New York, NY, USA, 2007. ISBN 0521872820.

[Osb03] M. J. Osborne. *An introduction to game theory*. Oxford University Press, New York, 2003. ISBN 0195128958.

[Pap93] C. H. Papadimitriou. *Computational complexity*. Addison-Wesley, Reading, Massachusetts, 1993. ISBN 0201530821.

[Pap94] Christos H. Papadimitriou. On the complexity of the parity argument and other inefficient proofs of existence. *Journal of Computer and System Sciences*, 48(3):498 – 532, 1994. ISSN 0022-0000.

[Pap07] C. H. Papadimitriou. The complexity of finding Nash equilibria. In A. Nisan, R. Roughgarden, E. Tardos, and V. V. Vazirani, editors, *Algorithmic Game Theory*, pages 29–51. Cambridge University Press, NY, 2007. ISBN 0521872820.

[PNS04] R. Porter, E. Nudelman, and Y. Shoham. Simple search methods for finding a Nash equilibrium. In D. L. McGuinness and G. Ferguson, editors, *AAAI-04, Proceedings of the Nineteenth National Conference on Artificial Intelligence, Sixteenth Conference on Innovative Applications of Artificial Intelligence*, pages 664–669. AAAI Press / The MIT Press, San Jose, California, USA, 2004. ISBN 0262511835.

[Ros73] R. W. Rosenthal. A class of games possessing pure-strategy Nash equilibria. *International Journal of Game Theory*, 2(1):65–67, 1973. ISSN 0020-7276.

[Rou10] T. Roughgarden. Algorithmic game theory. *Communications of the ACM*, 53(7):78–86, 2010. ISSN 0001-0782.

[RT07] T. Roughgarden and E. Tardos. Introduction to the inefficiency of equilibria. In A. Nisan, R. Roughgarden, E. Tardos, and V. V. Vazirani, editors, *Algorithmic Game Theory*, pages 29–51. Cambridge University Press, NY, 2007. ISBN 0521872820.

[Sho08] M. Shor. Payoff. Dictionary of Game Theory Terms, Game Theory .net, 17 November 2008. http://www.gametheory.net/dictionary/Payoff.html.

[SvS04] R. Savani and B. von Stengel. Exponentially many steps for finding a Nash equilibrium in a bimatrix game. In *FOCS-04, Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*, pages 258–267. IEEE Computer Society, Rome, Italy, 2004. ISBN 0769522289. Program chair: Upfal, E.

[TS07] H. Tsaknakis and P. G. Spirakis. An optimization approach for approximate nash equilibria. In X. Deng and F. C. Graham, editors, *WINE-07: Proceedings of the Third International Workshop on Internet and Network Economics, San Diego, CA, USA*, volume 4858 of *Lecture Notes in Computer Science*, pages 42–56. Springer, 2007. ISBN 978-3-540-77104-3.

[TS10] H. Tsaknakis and P. G. Spirakis. Practical and efficient approximations of nash equilibria for win-lose games based on graph spectra. In A. Saberi, editor, *WINE-10: Proceedings of the sixth International Workshop on Internet and Network Economics, Stanford, CA, USA*, volume 6484 of *Lecture Notes in Computer Science*, pages 378–390. Springer, 2010. ISBN 978-3-642-17571-8.

[TV07] E. Tardos and V. V. Vazirani. Basic solution concepts and computational issues. In A. Nisan, R. Roughgarden, E. Tardos, and V. V. Vazirani, editors, *Algorithmic Game Theory*, pages 3–28. Cambridge University Press, NY, 2007. ISBN 0521872820.

[vS02] B. von Stengel. Computing equilibria for two-person games. In R. J. Aumann and S. Hart, editors, *Handbook of Game Theory*, volume 3, pages 1723–1759. Elsevier, North-Holland, Amsterdam, 2002. ISBN 9780444894281. Chapter 45.

[WPD04] W. E. Walsh, D. C. Parkes, and R. Das. Choosing samples to compute heuristic-strategy Nash equilibrium. In P. Faratin, D. C. Parkes, J. A. Rodriguez-Aguilar, and W. E. Walsh, editors, *Agent-Mediated Electronic Commerce V*, volume 3048 of *Lecture Notes in Computer Science*, pages 109–123. Springer, Berlin, 2004. ISBN 3540226745.