Cost Sharing and Clustering under Distributed Competition

Dissertation

zur Erlangung des akademischen Grades des
Doktors der Naturwissenschaften (Dr.rer.nat.)
an der Universität Konstanz
Fachbereich Informatik und Informationswissenschaft

vorgelegt von

Martin Hoefer

Tag der mündlichen Prüfung: 20. September 2007

Referenten:
Prof. Dr. Ulrik Brandes
Prof. Dr. Dietmar Saupe

Inhalte aus dieser Arbeit wurden bereits wie folgt veröffentlicht bzw. sind zur Veröffentlichung angenommen:

Chapter 4:	Hoefer [Hoe06a], Cardinal und Hoefer [CH06]
Chapter 5:	Hoefer [Hoe06b, Hoe07b]
Chapter 9:	Brandes, Delling, Gaertler, Görke, Hoefer, Nikoloski und Wagner [BDG^{+}07a, BDG^{+}07b]

Deutsche Zusammenfassung

Das Internet ist mittlerweile ein zentraler Bestandteil der weltweiten wirtschaftlichen und gesellschaftlichen Entwicklung. Daher ist es wichtig, die grundlegenden Prozesse zu verstehen, die bei der Entwicklung, Verwaltung und Nutzung des Internet eine Rolle spielen. Im Gegensatz zu anderen Netzwerken, die in der Informatik und der Mathematik seit Jahrzehnten untersucht werden, wird das Internet nicht zentral geplant oder verwaltet. Firmen, Behörden und andere Institutionen arbeiten gemeinsam und verteilt daran, Server einzurichten, Verbindungen zu legen, Daten zu verschieben oder Webangebote bereitzustellen. Viele dieser Akteure sind wirtschaftlicher Natur, ihr Hauptanliegen ist das Kaufen und Verkaufen von Produkten. Das Internet ist daher mehr als nur ein Netzwerk, es ist ein Markt. Wichtige Aspekte wie z.B. Investitionen in Topologie und Server, Verwaltung und Updates, Absicherung von Subsystemen etc. müssen daher unter marktwirtschaftlichen und insbesondere spieltheoretischen Gesichtspunkten analysiert werden.

Das Internet ist nicht nur wirtschaftlichen Prozessen ausgesetzt, es bietet auch eine Plattform für Kommunikation und Vernetzung. Auf dieser Grundlage entstehen soziale Netzwerke zwischen den beteiligten Akteuren. Die Analyse sozialer Netzwerke ist ein wichtiger Forschungsgegenstand in der Soziologie mit Auswirkungen auf angrenzende Fachgebiete wie Wirtschaftswissenschaften, Mathematik und Informatik. Ein neues, wichtiges Teilgebiet dieser Forschung besteht darin, die Wechselwirkung sozialer Netzwerke mit (wirtschaftlicher) Entscheidungsfindung zu verstehen.

In dieser Arbeit werden zwei grundlegende Modelle für nicht-kooperative Spiele vorgestellt, mit denen sich wirtschaftliche Interessen im Internet und Einflüsse sozialer Netzwerke auf Entscheidungen der Akteure analysieren lassen. Aspekte wie Existenz, Berechenbarkeit und soziale Güte stabiler Zustände wie reiner Nash Gleichgewichte sowie approximativer Nash Gleichgewichte stehen bei der Analyse der Modelle im Vordergrund.

Im ersten Teil der Arbeit werden Spiele betrachtet, in denen die Spieler eine Menge von Rohstoffeinheiten kaufen und deren Kosten aufteilen müssen. Jeder Spieler hat eine bestimmte Anforderung an die Art und Menge der gekauften Einheiten. Die Spieler können beliebige Beiträge für den Kauf von Rohstoffen anbieten. Sobald der Gesamtbeitrag aller Spieler für eine Rohstoffeinheit die Kosten übersteigt, gilt die Einheit als gekauft. Ein Spieler möchte dabei seine Anforderung mit möglichst geringem eigenen Kostenbeitrag erfüllen. Das Modell wird in Kapitel 3 formal eingeführt und in den Kapiteln 4 und 5 auf mehrere Probleme aus dem Bereich Service Installation, Facility Location und Netzwerkdesign angewendet, die einfache Modelle für fundamentale Fragestellungen im Internet liefern. Generell gibt es schon für sehr kleine Instanzen der Spiele keine oder nur sehr teuere reine Nash Gleichgewichte. Dagegen gibt es für eine Reihe von Teilklassen der Spiele approximative Nash Gleichgewichte mit kleinen konstanten Gütegarantien. Diese Zustände können ausserdem effizient berechnet werden. In den Algorithmen werden bestehende Techniken aus der linearen Optimierung sowie neue kombinatorische Ansätze verwendet. Kapitel 6 skizziert eine interessante, realistische Erweiterung des Modells auf nutzungsbasierte Rohstoffkosten.

Im zweiten Teil der Arbeit wird ein Ansatz betrachtet, der es erlaubt, Clustering von Graphen spieltheoretisch zu modellieren. Dadurch lassen sich zum Beispiel Zugehörigkeits- und Mitgliedschaftsentscheidungen von Akteuren im Kontext sozialer Netzwerke untersuchen. Jeder Spieler ist ein Knoten im Graph und entscheidet, welchem von mehreren möglichen Vereinen, Gruppen oder Clustern er angehört. Der Wert dieser Entscheidung für den Spieler hängt dabei von der Entscheidung aller anderen Spieler und der Struktur des Netzwerkes ab. Die betrachteten Spiele sind Teilklassen von Polymatrix Spielen, in denen die Summe der Bewertungen eines Zustands durch die Spieler intuitive und bekannte Indizes für Clusterings nachbilden. Alle in diesem Teil der Arbeit betrachteten Spiele sind Potenzialspiele, d.h. sie haben mindestens ein reines Nash Gleichgewicht, das durch einen Prozess des iterativen Strategiewechsels gefunden werden kann. Der Schwerpunkt der Analyse liegt in der Dauer dieses Prozesses und der Berechenbarkeit von guten Nash Gleichgewichten und sozial optimalen Zuständen. In Kapitel 7 wird das Modell formal eingeführt. In Kapitel 8 werden Spiele untersucht, die sich aus zwei Teilspielen zusammensetzen, einem für verbundene und einem für nicht verbundene Spieler. Es werden besondere Bedingungen an die Bewertungen in Spielen mit 2 Strategien hergeleitet. Unter diesen Bedingungen können das beste Nash Gleichgewicht und sozial optimale Zustände einfach charakterisiert und effizient berechnet werden. In Kapitel 9 werden Spiele auf Basis des populären Index *Modularity* betrachtet. Das Hauptresultat ist ein Beweis der NP-Härte für das Finden des sozial optimalen Zustands und des besten Nash Gleichgewichts. Dies liefert die ersten grundlegenden theoretischen Einsichten in die Modularity-Optimierung und bestätigt eine zuvor in der Literatur formulierte Vermutung über den Komplexitätsstatus.

Acknowledgements

First and foremost I would like to express my sincere gratitude to my supervisor Ulrik Brandes. The encouragement, advice, and enduring support I received from him made this thesis possible. His clear thinking was most inspiring, and he gave me the freedom to choose and tackle the research questions of my interest. I am deeply thankful to him for all this.

In addition, I am indebted to a number of people who supported me during the preparation of this work. I am grateful to Dietmar Saupe, who readily agreed to be a supervisor of this thesis, and to Carlos Alós-Ferrer for joining my thesis committee.

The credit for introducing me to algorithmic game theory is given to Piotr Krysta. His continuous support and advice were very much appreciated. I am also thankful for enlightening research visits to Dortmund and Liverpool.

After a long hike to Schafberg mountain on a rainy and foggy November day Jean Cardinal came up with the idea to play vertex cover games. I am much indebted to him for sharing this idea with me, for numerous discussions that led to some results of this thesis, and for a fabulous research visit to Brussels.

Clustering was great fun with Marco Gaertler, Robert Görke, and Daniel Delling. Thanks to them for several research visits and many hours of modularization.

I was lucky to be a doctoral student within our departmental DFG Graduiertenkolleg[1], which was a creative and stimulating research environment. Thanks to everybody who contributed to it and to DFG for financial support. I particularly enjoyed the regular spring workshops and summer schools, and I thank Christian Pich, who repeatedly agreed to share a room with me. Moreover, it was a great pleasure to share our de-facto double office with him. Additional thanks go to all members of the Algorithmics Group for making my working place so enjoyable.

I thank Melanie Badent and Tim Fischer for proofreading parts of this thesis.

My special thanks go to my family, Hannelore and Claudia Hoefer, and my apologies to all the people whose help and support I forgot to mention.

Evgeniya, thank you for proofreading and for being моето съкровище.

[1]DFG Graduiertenkolleg "Explorative Analysis and Visualization of Large Information Spaces"

Contents

Chapter 1

Introduction

One of the unique artefacts and dynamic driving forces in modern society is the Internet. It represents not only a network but a virtual platform for exchange that is jointly developed, built and maintained by millions of users worldwide. It offers unique opportunities in human history in all parts of everyday life. Therefore it is important to understand the dynamics and forces that underlie the development of the Internet. A straightforward way for computer scientists is to model and analyze it as a graph, which allows to study networking problems. This has been done intensively for many networks and network problems during the last decades, which came from fields like operations research, computer science, and mathematics. The study of these model has generated valuable insights into applications from domains like public traffic, supply chains, scheduling, programming, parallel computing, etc. However, unlike most of these networks, the Internet is not centrally built or maintained. Instead, there are a number of decentralized public and private parties that jointly create, develop and improve it. In fact, many of the parties concerned with development and use of the Internet are companies that in essence buy and sell products and thus act as economic agents in a market. To interpret their behavior and to analyze their incentives, it is necessary to understand the Internet as a market, a place of competition and cooperation. This leads to an economic and, in particular, a game-theoretic approach to study the behavior of agents and the results of their actions on the network.

For example, consider the topology of the Internet. At present it is still subject to change and development. A lot of independent parties are concerned with building and maintaining connections. Some of them, like global players in business and industry, hold a lot of servers and connections throughout different parts of the world. Some interesting questions with respect to the topology development of the Internet are the following:

– Why are certain links in the Internet established?

– Who is interested in building and maintaining them, who will profit from them?

– Are players motivated to cooperate in the creation of a link?

– How do these interests change if the topology changes?

– What would be the most profitable topology, and who would establish the links?

Answers to these questions are very important for improving the structure of the Internet. Insights could be used to guide public authorities, telecommunications firms and other parties concerned with building computer networks. They could see how to establish cheap networks that other selfish parties and companies will be motivated to pay for, maintain and use. This could lead to an improved topology. But maybe cheap networks are not stable - i.e. some parties might always be motivated to change a given network and establish other links. One can certainly think of many interesting insights, and at present there is still only very little understanding of this development and the underlying dynamics.

The development of the Internet concerns many more aspects than topology, for instance package routing and congestion, service installation, facility and server location, maintenance, security and cryptography, etc. All these are similarly influenced by economic decisions of the involved parties. More generally, the Internet poses fundamental questions about the incentives and optimization processes of economic agents in competitive environments. Similar questions have been studied for decades in economics and game theory, and there is a large body of mature research, of well-motivated and established mathematical concepts that a rigorous investigation can rely on. The distinctive and new aspect is that the Internet is a network and an inherently computational environment, so algorithmic, graph-theoretic and game-theoretic aspects need to be combined.

The Internet is influenced by economic agents, and in turn, it creates affiliations and social ties between them. In fact, such social ties and social networks always evolve between humans and crucially influence our lives. Thus, not surprisingly, there has been an enormous research interest in these concepts in many disciplines. Only recently, however, scientists are starting to understand the influence of social networks on economic decision making. Typical research questions concern structural properties and topology of equilibrium networks, or the influence of a given network on existence and structure of equilibria in a market or a game.

1.1 Game Theory, Computation, and Networks

The connection between computer science and game theory has been explored for many years. One of the most central concepts in game theory, which we will study in this thesis, are non-cooperative strategic games. They were proposed and studied by Nash [Nas51] in the late 1940s, who showed the existence of a mixed-strategy equilibrium in any game. A central problem has been to compute such an equilibrium for a game represented in normal form by a large payoff table. For games with two players Lemke and Howson [LH64] proposed a solution algorithm in 1964, which has striking similarities with the simplex algorithm [Dan63] for linear programming (LP) [NW88]. While the simplex algorithm is known to be inefficient [KM72], it was only recently that a similar result was shown for the Lemke-Howson algorithm [vSS06]. Papadimitriou [Pap94] defined the complexity class PPAD of search problems, for which a solution is guaranteed to exist by a parity argument. Recently, finding a mixed Nash equilibrium in strategic games for any number of players has been shown to be PPAD-complete [DGP06, CD06]. It is now widely believed that *any* algorithm for finding mixed Nash equilibria must be inefficient. For the case of two players this is surprising, because if the payoffs of every state sum to 0, a simple application of LP can be used to find a mixed equilibrium in polynomial time. More generally, the evidence for hardness of finding mixed equilibria stands in marked contrast to existing efficient algorithms for LP [Kha79]. This is even more remarkable in light of the fact that many results from the last decades on various equilibrium concepts rely on LP techniques. For example, in cooperative games [MvN47] Deng et al. [DIN97] present and review a variety of these connections. In fact, a significant portion of the results in this thesis is also proven using LP duality.

While problems of computing equilibria have a long history on the edge of game theory and computer science, the recently emerging field of *algorithmic game theory* is driven by problems dealing with networks. A prominent way to capture game-theoretic networking problems is by formulating a non-cooperative strategic game with selfish agents. Stable outcomes (e.g. networks, allocations, connections) of interactions of these agents then correspond to Nash equilibria. Typically, introducing the dynamics of selfish behavior gives rise to a variety of new issues. In particular, outcomes representing Nash equilibria can be much more costly than social optimum solutions. Papadimitriou [Pap01] refers to this cost increase due to selfish behavior as the *price of anarchy*. The price of anarchy has been studied in a large number of games dealing especially with networking issues, such as load balancing [CKV02, CV02, Rou04], routing and congestion [AAE05, CK05b, CDR03, Rou03, RE02], facility location [DGK$^+$05, Vet02], and flow control [AKP$^+$02, DGH03, She95]. As there might be more than one Nash equilibrium in a game, it is natural to consider how costly an outcome of a game must be to be stable. This minimum cost increase that is needed for any Nash equilibrium has been termed the *price of stability* [ADK$^+$04]. The price of stability has been considered most notably in network design [ADTW03, ADK$^+$04, FKL$^+$06, SM04] and congestion games [CK05a].

The results on non-cooperative games and their equilibria build the basis for the analysis of incentives in more regulated environments. In particular, the area of *computational mechanism design* treats problems of how to design and compute rules and regulations for games to ensure certain agent behavior. Mechanism design is a traditional branch in standard game theory with important applications in economics and business. An institution, called *the mechanism*, collects private information from the agents, determines an outcome, and specifies a payment for each agent. The agents have private valuation functions, which determine the value of the outcome to them. They are hoping to get an outcome maximizing their personal utility, which depends on the valuation and the payments. The mechanism, however, tries to implement some social goal, which can be contrary to the agents private interests. Thus, the agents can be motivated to report wrong or biased information to influence the decision of the mechanism. The use of this concept was motivated by the problem to design efficient protocols for Internet applications with selfish participants. Moreover, the concept has been applied to computational problems from a diverse set of domains like combinatorial auctions and e-commerce [dVV03, Par01], scheduling [HMU07, NR01], broadcasting network design [FPS01, JV01, BdFFM04], or routing [FPSS02, NR01]. An extensive overview of important results in algorithmic game theory and computational mechanism design is given in a forthcoming book by Nisan et al. [NTRV07].

In addition to these rather computational issues, there is an evolving branch of economics and sociology to study game-theoretic models for social network creation. Some of these works regard the network as a fixed parameter of the game, while others consider network creation as an outcome of strategic interaction. This line of research was initiated by Myerson [Mye77], who imposed a decomposition for a cooperative game based on connectivity of players in a graph. He provided an allocation rule now known as Myerson value to allocate welfare to groups of players. The extensions of this model are numerous, and an exposition of the most important developments is given in the books of Slikker and van den Nouweland [SvdN01] and Jackson and Dutta [DJ03]. Jackson [Jac04] provides an excellent introduction and overview of more recent work.

1.2 Overview of the Thesis

In this thesis we consider strategic games that model important aspects of networking under distributed competition. Chapter 2 provides some fundamentals of game theory to facilitate the access to the contents of later chapters. We study two frameworks of games, which have quite different characteristics and applications. Hence, the thesis is divided into two parts, and each one treats a single framework for a class of games.

Cost Sharing and Service Installation

In the first part we study games in which players are to specify investments for establishing a feasible set of resources for usage. In equilibrium the strategies of players then represent a cost sharing of a feasible set of established resources.

In Chapter 3 we present the formal model of *investment games*, outline some basic properties, and remark on how our model relates to other prominent cost sharing games.

Chapter 4 contains a treatment of *covering games* in which the game is based on the formal model of a covering integer program with non-negative coefficients [Vaz00, Chapter 13.2]. Important examples that fall into this class are for instance variants of Vertex Cover or Set Cover problems. In addition, we treat an extension based on facility location problems. These games can be used as simple models for a variety of distributed resource and service installation scenarios. We provide tight bounds for prices of anarchy and stability in general and special cases, and study the existence and computation of exact and approximate Nash equilibria.

Investment games are a generalization of the *connection game*, which has been proposed by Anshelevich et al. [ADTW03] in the context of Steiner network creation. These games were proposed as a simple model for the topology development in the Internet. A drawback is that players in this game can have incentives to create disconnected networks, while a crucial feature of the importance of the Internet is global connectivity. In Chapter 5 address this issue by considering tree connection games, in which the created network must be connected. We show hardness and approximation results for Nash equilibria in this game. Furthermore, a more complex network creation game, the *backbone game*, is introduced and initial results on the cost and complexity of Nash equilibria are derived.

In Chapter 6 an adjustment to investment games is outlined. In the *wholesale investment game* each player is required to specify her investments and the set of resources she intends to use. The cost of a resource unit then increases with the number of players that decide to use it. The model generates an economy of scale, a reasonable and frequent property of standard economic models. We provide tight bounds on the price of anarchy but leave a deeper study of this game for future work.

Clustering and Affiliations

In the second part we study games in which players correspond to vertices in a graph. The strategies of players represent clustering decisions, and the utility functions compose well-known clustering indices. These games serve as models for distributed graph clustering problems. They can also be used to model incentives in affiliation decisions of players embedded in a social network.

Chapter 7 describes a suitable formal framework for the study of such *clustering games*. This enables us in Chapter 8 to consider games corresponding to unweighted MaxCut and MaxAgree problems. Also, a more general class of games is studied involving symmetric (2×2) games played by pairs of players. As prices of anarchy and stability might not be well-defined, we focus on the complexity of computing Nash equilibria.

In Chapter 9 we consider games that correspond to optimizing the recently popular modularity clustering index [GN04]. We provide bounds on the prices of anarchy and stability and study computational issues of Nash equilibria. In particular, while finding a Nash equilibrium can be done in polynomial time, finding the best Nash equilibrium and the social optimum state is NP-hard. This resolves a recent conjecture about the complexity of finding clusterings that maximize the modularity index.

For a more detailed summary and evaluation of the results see Chapter 10 and, in particular, Table 10.1 on page 144.

Chapter 2

Preliminaries

This chapter introduces formal concepts and measures used for the analysis in this thesis. In the next Section 2.1 the concepts of a strategic game and a Nash equilibrium are defined along with the important classes of congestion and potential games. Section 2.2 presents ideas to measure social welfare and cost in games. In particular, prices of anarchy and stability are defined and the connections to performance ratios for approximation algorithms are outlined. Section 2.3 introduces concepts and classes from complexity theory to study the complexity of computing Nash equilibria. A focus is put on tools to characterize the behavior of best-response iterations. Finally, graphs are a central concept in our results, and for reference Section 2.4 outlines the notation used throughout.

This chapter concentrates on concepts relevant for the results in this thesis. In addition to the references in the text we refer the reader to standard literature for further discussions. For a cumulative introduction into various aspects of game theory see [AH02]. A treatment of complexity theory and approximation algorithms is given in [GJ79, Hoc96, Vaz00].

2.1 Strategic Games

The last decades have seen a variety of game-theoretic models and interesting solution concepts [AH02, OR94, vD91], but each of these concepts has certain advantages as well as drawbacks. In this thesis we consider concepts that arise from some of the most foundational and most frequently explored game-theoretic ideas. We consider strategic games, which are a model to capture the interaction of a number of non-cooperative competitive selfish agents. In the simplest setting each player has a set of actions or strategies, of which she can pick one. The vector of chosen strategies, called a strategy profile or state, yields a certain value to each player. Hence, as we assume each player is selfish, the values of different states influence her choice of action. Changing her action causes changes in the state and might motivate other players to reevaluate their choices. The aim of game theory is to characterize the properties and outcomes of this dynamic decision making process.

In general, there are many parameters that can crucially influence the players in their decisions. Hence, in standard non-cooperative game theory the model of a strategic game usually comes with a number of limiting assumptions, which we also use here. First, players are assumed to be myopic, i.e. at any given point of time they only evaluate their strategy choices against the current choices of the other players. They do not use any memory, learning, or foresight to induce or anticipate certain behavior of other players. Second, players are assumed to be non-cooperative, i.e. they do not form any coalitions and do not evaluate or take a coordinated action. Third, nothing is assumed on the process of how players are coordinated in their decision making. Instead, at any point of time any player is allowed to switch to any different strategy. Sometimes we deviate from this assumption and consider an iterative process, in which in each iteration one player is allowed to deviate to her best strategy choice at the moment. Such a process is called a *best response iteration.* Of interest is to find and characterize states from which no player has an incentive to unilaterally move away. Under the outlined assumptions these are the stable states of the game and are thus most likely to evolve if the game is played over a longer period of time. A game considered in this thesis can formally be defined as follows.

Definition 2.1 *A* strategic game $\Gamma = ([k], \mathcal{S}, \mathsf{util})$ *is a tripel, in which*

- $[k]$ *is the set* $[k] = \{1, \dots, k\}$ *of the* k players*,*
- $\mathcal{S}_p$ *is a set of* strategies $s_p \in \mathcal{S}_p$ *for player* $p \in [k]$ *and* $\mathcal{S} = \mathcal{S}_1 \times \ldots \times \mathcal{S}_k$ *is the set of* strategy profiles *or* states $s \in \mathcal{S}$*, and*
- $\mathsf{util}_p : \mathcal{S} \to \mathbb{R}$ *is a* utility function *indicating the preference over the states for player* $p \in [k]$*.*

A game is called symmetric *if* $\mathcal{S}_p = \mathcal{S}_q$ *for any players* $p, q \in [k]$*, and for any two states* $s, s' \in \mathcal{S}$ *such that* s' *results from permuting the strategies of* s*, we have* $\mathsf{util}_p(s') = \mathsf{util}_q(s)$ *if* $s'_p = s_q$*.*

For convenience s_{-p} denotes a profile excluding p, hence for $s \in \mathcal{S}$ and $s = (s_1, \dots, s_k)$ the profile $s_{-p} = (s_1, \dots, s_{p-1}, s_{p+1}, \dots, s_k)$. Similarly, $\mathcal{S}_{-p}$ denotes the set of all profiles s_{-p}. Note that in a symmetric game it suffices to specify the utility function for just one player e.g. util_1, since the utility values for all other players can be easily extracted from the values of util_1 for symmetric states. In games, in which the strategy spaces of the players are finite, $l_p = |\mathcal{S}_p|$ denotes the number of strategies. If all players have the same number of strategies, their number is denoted by l.

We will always assume that players want to *maximize* their utility and we are interested in characterizing the stable states of the game. If for player $\mathtt{p}$ and $\mathtt{s}_\mathtt{p}, \mathtt{s}'_\mathtt{p} \in \mathcal{S}_\mathtt{p}$ the strategy $\mathtt{s}_\mathtt{p}$ delivers better utility than strategy $\mathtt{s}'_\mathtt{p}$ for every profile $\mathtt{s}_{-\mathtt{p}}$, then $\mathtt{p}$ will never pick $\mathtt{s}'_\mathtt{p}$. In this case $\mathtt{s}'_\mathtt{p}$ is *dominated* by $\mathtt{s}_\mathtt{p}$.

Definition 2.2 *For a player* $\mathtt{p} \in [\mathtt{k}]$ *and strategies* $\mathtt{s}_\mathtt{p}, \mathtt{s}'_\mathtt{p} \in \mathcal{S}_\mathtt{p}$ *in a strategic game, if*

$$\mathtt{util}_\mathtt{p}(\mathtt{s}_\mathtt{p}, \mathtt{s}_{-\mathtt{p}}) \geq \mathtt{util}_\mathtt{p}(\mathtt{s}'_\mathtt{p}, \mathtt{s}_{-\mathtt{p}}) \textit{ for all } \mathtt{s}_{-\mathtt{p}} \in \mathcal{S}_{-\mathtt{p}},$$

then $\mathtt{s}_\mathtt{p}$ dominates $\mathtt{s}'_\mathtt{p}$*.* $\mathtt{s}_\mathtt{p}$ *is a* dominant strategy *if it dominates all* $\mathtt{s}'_\mathtt{p} \in \mathcal{S}_\mathtt{p}$*.*

.

Clearly, a state composed of dominant strategies is a stable state of the game. The existence of dominant strategies, however, is a very strong assumption as players must have a universal preference in the game. Also, dominant strategies are not necessary for stability within a group of myopic non-cooperative players. The necessary and sufficient condition is that there is a state, which encompasses all the *local* preferences of players. Each player is motivated to pick a strategy that returns the best utility only against the *current choices* of other players. These strategy choices are captured by a best-response function, which for a profile $\mathtt{s}_{-\mathtt{p}}$ returns the set of strategies $\mathtt{s}_\mathtt{p} \in \mathcal{S}_\mathtt{p}$ resulting in optimal utility for player $\mathtt{p}$.

Definition 2.3 *The* best-response function $\mathtt{br}_\mathtt{p} : \mathcal{S}_{-\mathtt{p}} \longrightarrow 2^{\mathcal{S}_\mathtt{p}}$ *of player* $\mathtt{p}$ *in a game* Γ *is defined as*

$$\mathtt{br}_\mathtt{p}(\mathtt{s}_{-\mathtt{p}}) = \{\, \mathtt{s}_\mathtt{p} \in \mathcal{S}_\mathtt{p} \mid \forall \mathtt{s}'_\mathtt{p} \in \mathcal{S}_\mathtt{p} : \ \mathtt{util}_\mathtt{p}(\mathtt{s}_\mathtt{p}, \mathtt{s}_{-\mathtt{p}}) \geq \mathtt{util}_\mathtt{p}(\mathtt{s}'_\mathtt{p}, \mathtt{s}_{-\mathtt{p}}) \,\}$$

An element $\mathtt{s}_\mathtt{p} \in \mathtt{br}_\mathtt{p}(\mathtt{s}_{-\mathtt{p}})$ *is called a* best-response strategy *or* best response *for* $\mathtt{p}$ *against* $\mathtt{s}_{-\mathtt{p}}$*. We denote by* $\mathtt{util}_\mathtt{p}(\mathtt{br}_\mathtt{p}, \mathtt{s}_{-\mathtt{p}})$ *the utility* $\mathtt{util}_\mathtt{p}(\mathtt{s}_\mathtt{p}, \mathtt{s}_{-\mathtt{p}})$ *for any strategy* $\mathtt{s}_\mathtt{p} \in \mathtt{br}_\mathtt{p}(\mathtt{s}_{-\mathtt{p}})$*.*

The stable states of the game are composed of a collection of best responses. They were first studied by John Nash [Nas51], are called pure Nash equilibria, and will be denoted NE throughout.

Definition 2.4 *A state* $\mathtt{s}$ *is called a* pure Nash equilibrium (NE) *if each player plays a best-response* $\mathtt{s}_\mathtt{p} \in \mathtt{br}_\mathtt{p}(\mathtt{s}_{-\mathtt{p}})$ *for all* $\mathtt{p} \in [\mathtt{k}]$*, or equivalently*

$$\mathtt{util}_\mathtt{p}(\mathtt{s}_\mathtt{p}, \mathtt{s}_{-\mathtt{p}}) \geq \mathtt{util}_\mathtt{p}(\mathtt{s}'_\mathtt{p}, \mathtt{s}_{-\mathtt{p}}) \textit{ for all } \mathtt{p} \in [\mathtt{k}] \textit{ and } \mathtt{s}'_\mathtt{p} \in \mathcal{S}_\mathtt{p} \tag{2.1}$$

It is simple to note that there can be more than one NE in a game. In addition, some games have no NE, e.g. the symmetric game of *Matching Pennies* for two players. The intuition is that there are two players, and each one has a penny. Each

	Head	Tail
Head	`1,-1`	`-1,1`
Tail	`-1,1`	`1,-1`

Figure 2.1: Matching Pennies for two players

player decides, which side of her penny should show up. If both players choose the same side, player 1 receives the penny from player 2. Otherwise, player 2 receives the penny from player 1. The cost values are given in Figure 2.1. This notation is standard for so-called (2×2) games, games with 2 players and 2 strategies per player. The strategy of player 1 yields the row, the strategy of player 2 yields the column. From the cell that indicates the current state player 1 gets a cost given by the first number, and player 2 gets a cost given by the second number.

The existence of games without NE raises natural questions that we will address repeatedly in later chapters.

– Does a given game have one/more than one NE?

– What structural properties are exhibited by the NE of a game?

Congestion and Potential Games

There are several classes of games that have at least one NE. One of the most prominent classes are congestion games [Ros73]. They have attracted an enormous amount of research interest, because they yield a natural general model to study the competitive usage of a set R of resources in a large variety of contexts.

Definition 2.5 *A* congestion game $([k], \mathsf{R}, \mathcal{S}, \mathtt{late}, \mathtt{util})$ *is a strategic game with a set of resources* R *and*

- *each strategy* $s_p \in \mathcal{S}_p$ *is a subset* $s_p \subseteq \mathsf{R}$*, for all* $p \in [k]$*,*
- *there is a* latency function $\mathtt{late}_r : \mathbb{N} \to \mathbb{R}$ *for each* $r \in \mathsf{R}$*, and*
- *the utility of player* p *is* $\mathtt{util}_p(s) = -\sum_{r \in s_p} \mathtt{late}_r(|\{q \in [k] \mid r \in s_q\}|)$.

In this model players are to pick a set of resources for usage (e.g. a path to route through a traffic network). The disutility for a player is the sum of latencies incurred by the resources. A resource creates a different (usually a larger) latency if more players are using it in their strategies. Interestingly, a congestion game is guaranteed to possess a NE. This is easier to understand for the class of potential games [MS96].

	C	D
C	-3,-3	-18,-1
D	-1,-18	-12,-12

(a)

	C	D
C	-2	0
D	0	6

(b)

Figure 2.2: (a) A Prisoner's Dilemma game; (b) corresponding potential Φ

Definition 2.6 *A strategic game is called a* potential game *if there is a function* $\Phi : \mathcal{S} \to \mathbb{R}$ *such that for every player* p*, every state* $s \in \mathcal{S}$*, and every strategy* $s'_p \in \mathcal{S}_p$

$$\mathrm{util}_p(s'_p, s_{-p}) - \mathrm{util}_p(s_p, s_{-p}) > 0 \Leftrightarrow \Phi(s'_p, s_{-p}) - \Phi(s_p, s_{-p}) > 0.$$

Φ *is called* ordinary potential *or simply* potential *of the game.* Φ *is called* weighted potential *if*

$$\mathrm{util}_p(s'_p, s_{-p}) - \mathrm{util}_p(s_p, s_{-p}) = w_p \cdot (\Phi(s'_p, s_{-p}) - \Phi(s_p, s_{-p}))$$

for some positive numbers $w_p > 0$*. In case* $w_p = w_q$ *for all* $p, q \in [k]$*, function* Φ *is called* exact potential.

Rosenthal [Ros73] showed that congestion games are potential games by providing a potential function. Monderer and Shapley [MS96] proved equivalence of both classes. For an example consider the symmetric (2×2) game in Figure 2.2. It is a variant of the famous *Prisoner's Dilemma*. Two players have jointly committed a crime, were captured, and are now interrogated by the police separately and simultaneously. Each player has two strategies: cooperation (C) or defection (D). If they both cooperate, the police has no substantial evidence against them and they are imprisoned for a short period of 3 months. If one defects by confessing and the other one does not, the confessant receives a mild verdict of 1 month and the other one a harsh verdict of 18 months. If they both confess, they both go to jail for one year each. The remarkable property in the Prisoner's Dilemma is that the only NE is (D,D), as D is a dominant strategy for both players. The NE, however, is catastrophal, because the players are sentenced to a much higher penalty than in the case of cooperation. Note that the Prisoner's Dilemma is a potential game, and the exact potential is given in Figure 2.2(b). In fact, every symmetric (2×2) game has an exact potential [Blu93, You98], and we use this property to define potentials for clustering games in Chapter 8.

In potential games a single function Φ is able to indicate for *every state* and *every player* whether there is a profitable strategy switch. Thus, if the state s is no

NE, there is still a player who can strictly improve her utility. By switching to such a strategy the potential strictly decreases. Hence, a sequence of strictly improving best-response steps can never return to a state s encountered previously. If the state space $\mathcal{S}$ is finite, there must exist a state of minimum potential, which must be a NE. The following theorem from [MS96] is now obvious.

Theorem 2.1 *A potential game with finite $\mathcal{S}$ has at least one NE.*

The proof suggests that best-response iteration can be used to obtain a NE. This process, however, can take a large number of steps. We make the details of this observation more precise in Section 2.3.

Pure and Mixed Nash Equilibria

Another way to guarantee the existence of an equilibrium is to create a continuous strategy space by allowing each player p to play a probability distribution over $\mathcal{S}_\mathsf{p}$. In particular, the new strategy space for a player p becomes the set of probability distributions over the elements $\mathsf{s}_\mathsf{p} \in \mathcal{S}_\mathsf{p}$, which are in this context called *pure* strategies. A distribution over pure strategies is called *mixed* strategy. In his seminal work John Nash [Nas51] proved the existence of an equilibrium in mixed strategies for every strategic game. The concept is referred to as *mixed* Nash equilibrium and is denoted mixed NE throughout. In the consideration of mixed strategies players pick a pure strategy only with a probability. It is assumed that players specify their distributions knowing only the distributions of other players. Then the game is played like a random experiment, whose outcomes cannot be changed individually. In a mixed NE no player can improve her expected utility of this random experiment by changing only her personal probability distribution. While such a model can be a reasonable choice in some situations, the widespread use in mathematics and economics stems from appealing theoretical properties, in particular, from guaranteed existence. The main question is, however, whether mixed strategies make sense. What does it mean for a person to ”play a mixed strategy” in a decision context? For the games considered in this thesis, we have come to the conclusion that mixed strategies are an interesting, but sometimes unintuitive concept. Thus, our analyses consider states in pure strategies that exactly or approximately satisfy the NE inequality (2.1). More details on mixed NE are discussed in Chapters 3 and 7.

2.2 Social Cost and Approximation

In the definition of a strategic game given in the last section there is no consideration of social cost or welfare of states. The only cost that is considered is specified by the player-specific utility functions, and these functions merely serve to define a preference relation over the states for each player. However, in many scenarios it is natural to consider social cost or welfare to evaluate the performance of states with respect to certain societal criteria (e.g. total/maximum amount of utility, fairness of utility distribution etc). In particular, the discussion of the Prisoner's Dilemma in the last section yielded the total number of months spent in prison by both of the players as a natural choice for the definition of a social cost. This definition - social cost or welfare as sum of player utilities - is a prominent and intuitive way to measure social quality. Throughout this thesis we only measure social quality of a state as the sum of player utilities. A reasonable choice for a social cost or welfare function has to take the considered scenario and the intended application of the model into account. Our choice is therefore not necessarily a good choice for every game, but for the games considered in forthcoming chapters it bears an intuitive and straightforward meaning.

Definition 2.7 *The* social quality *of a state* $\mathsf{s} \in \mathcal{S}$ *is measured by a function* $\mathsf{welfare} : \mathcal{S} \to \mathbb{R}$ *or* $\mathsf{cost} : \mathcal{S} \to \mathbb{R}$*. We define the* $\mathsf{welfare}(\mathsf{s}) = \sum_{\mathsf{p}\in[\mathsf{k}]} \mathsf{util}_\mathsf{p}(\mathsf{s})$ *and* $\mathsf{cost}(\mathsf{s}) = -\sum_{\mathsf{p}\in[\mathsf{k}]} \mathsf{util}_\mathsf{p}(\mathsf{s})$*. The* social optimum s^* *is the state of minimum social cost and maximum social welfare*

We use cost or welfare depending on the context of the game. The games of Part I involve players minimizing their cost when making an investment. In these games we strive to minimize social cost, i.e. the total disutility of all players. The games of Part II involve players maximizing payoffs generated by the graph structure when making a clustering decision. In these games we strive to maximize social welfare, i.e. the total payoffs for all players. For the remainder of this section we concentrate on social cost. It is easy to adjust the definitions to social welfare maximization.

It is natural to consider structural and quantitative questions about the game and the social cost of certain states.

– Which one is the state s^* minimizing cost?

– Which one is the best/worst NE in terms of cost?

– How much worse is the best/worst NE than s^* in terms of cost?

Especially the last question creates a connection to typical questions that are considered frequently in theoretical computer science and the analysis of approximation algorithms. In particular, let us consider a minimization problem in an abstract formulation, which encompasses a large number of problems regularly considered in computer science. The following presentation is mainly restricted to the consideration of minimization problems, and we only remark how we adjust the definitions to maximization problems.

Definition 2.8 *A* minimization problem $\Pi = (\mathcal{I}, \mathcal{Z}, \mathsf{feas}, \mathsf{c})$ *is given by*

- *a set of* instances $\mathcal{I}$ *and a set of possible solutions* $\mathcal{Z}$*,*
- *a function* $\mathsf{feas} : \mathcal{I} \to 2^{\mathcal{Z}}$*, which returns the set of feasible solutions to an instance* $I \in \mathcal{I}$*, and*
- *a cost function* $\mathsf{c}(I, \mathsf{feas}(I)) \in \mathbb{R}$ *assigning a cost to every feasible solution to any instance* $I \in \mathcal{I}$*.*

The goal is to find for each instance $I \in \mathcal{I}$ *the feasible solution* $z^* \in \mathsf{feas}(I)$ *such that* $\mathsf{c}(I, z^*) \leq \mathsf{c}(I, z)$ *for all* $z \in \mathsf{feas}(I)$*.*

We generally assume that for each instance there is a non-empty set of feasible solutions, otherwise there is nothing to optimize. An example for a minimization problem is the Vertex Cover problem [GJ79].

Problem 2.2 (Vertex Cover) *Given a graph* $G = (V, E)$ *and a cost function* $c : V \to \mathbb{R}_0^+$*, find a set of vertices* $V' \subseteq V$ *containing for each edge at least one incident vertex, for which* $\sum_{v \in V'} c(v)$ *is minimal.*

Here an instance I is a simple undirected graph G. A feasible solution to an instance is given by a set of vertices that contains for each edge in graph G at least one incident vertex. The set of feasible solutions is composed of all such subsets satisfying this property. The cost function measures the cardinality of a feasible vertex set z, and thus is optimized by a set of minimum cardinality.

In complexity theory, it is assumed that the representation of any instance I of a minimization problem and any feasible solution z to I can be given by strings over a finite alphabet. Then an algorithm, i.e. a program on a generic model of computation like the Turing machine, is used to obtain the desired solution z with minimum $\mathsf{c}(I, z)$. Most interesting minimization problems turn out to be NP-hard, i.e. most likely not solvable in a number steps that is polynomial in the size of the representation of I. We defer to [GJ79] and the next section for more complexity theoretic considerations. Hence, the theory of algorithms has shifted to consider tractable *approximation algorithms*[Vaz00]. They run in time polynomial in the input size and return solutions that have a cost within a guaranteed factor of the optimum cost. This serves to reveal the trade-off between tractability and cost optimization that is inherent in an approximation algorithm. In order to render the following concepts well-defined, it is assumed that the cost function satisfies $\mathsf{c}(I, z) \geq 0$ for every feasible solution z to the instance I and $\mathsf{quality}(s) \geq 0$ for every state s of the strategic game under consideration. In addition, it is tacitly assumed that $\frac{0}{0} := 1$ for all ratios to be defined. This serves to consistently obtain a ratio of 1 whenever numerator and denominator are equal.

Definition 2.9 *For a solution* z *to an instance* I *of a minimization problem the* approximation ratio $\beta(I, z)$ *is given by*

$$\beta(I, z) = \frac{c(I, z)}{c(I, z^*)}.$$

For maximization problems we define $\beta(I, z)$ *similarly using the inverse ratio.*

Definition 2.10 *A deterministic (approximation) algorithm for a minimization or maximization problem defines a function* $\mathsf{algo} : \mathcal{I} \to \mathcal{Z}$ *that yields for each instance* I *exactly one solution* z*. The* performance ratio *of the algorithm is given by the largest approximation ratio of any computed solution*

$$\sup_{I \in \mathcal{I}} \beta(I, \mathsf{algo}(I)).$$

The performance ratio can be given as a function of the length of the representation of I.

To characterize the degradation of social cost in a NE we take a similar straightforward approach and define what has been termed the *coordination ratio* [KP99] or the *price of anarchy* [Pap01]. This serves to capture the inherent trade-off in the social cost obtained by competitive optimization on the one hand and by centralized and coordinated optimization on the other hand.

Definition 2.11 *For a state* s *of a strategic game* Γ *and a social cost function* cost *the* approximation ratio $\beta(s)$ *is given by*

$$\beta(s) = \frac{\mathsf{cost}(s)}{\mathsf{cost}(s^*)}.$$

For $\mathsf{welfare}$ *functions we define* $\beta(s)$ *similarly using the inverse ratio.*

Definition 2.12 *The* coordination ratio *or the* price of anarchy *for a strategic game* Γ *and a social cost or welfare function is the largest approximation ratio of any NE*

$$\sup_{s \in \mathcal{S} \text{ is NE}} \beta(s).$$

We drop the argument s from the approximation ratio β whenever context allows. In comparison to the performance ratio some differences remain. Most notably, the concepts of game and NE are used similarly to instance and solution, however, for a given game there might be several NE. A deterministic algorithm, however, outputs exactly one solution to a given problem instance. The fact that several NE can be present in a game is a motivation to consider the best NE in terms of social cost. This measures how good a stable solution can get in terms of social cost and is thus termed the *price of stability*.

Definition 2.13 *The* price of stability *for a strategic game* Γ *and a social cost or welfare function is the smallest approximation ratio of any NE*

$$\inf_{s \in \mathcal{S} \text{ is NE}} \beta(s).$$

It is easy to see that both prices of anarchy and stability can easily become unbounded even in (2×2) games. An example is the Prisoner's Dilemma (see Figure 2.2), in which the only NE is arbitrarily worse in social cost than the optimum state. Another problem with the definition is that some games do not have NE. One alternative is to consider the measures for mixed NE, which are guaranteed to exist. In this thesis, however, we take a different approach. We consider the price of anarchy and stability only for the subset of games in which NE exist. For the study of general games we introduce a different notion that captures for a state s the difference from a NE. This concept captures a scenario, in which there must be a significant improvement to motivate a player to change her strategy. For example, this can be due to a cost for strategy switches or a limited ability to obtain strategies that yield optimal utility values. For the following definition we assume $\mathtt{util}_p(\mathtt{br}_p, s_{-p}) \geq 0$ for all utility and best-response functions $\mathtt{util}_p$ and $\mathtt{br}_p$.

Definition 2.14 *For a state* s *of a strategic game* Γ *with utility minimizing players the* stability ratio $\alpha(s)$ *is given by*

$$\alpha(s) = \sup_{p \in [k]} \frac{\mathtt{util}_p(s)}{\mathtt{util}(\mathtt{br}_p, s_{-p})}.$$

For utility maximizing players we define $\alpha(s)$ *as the supremum over the inverse ratio.*

Again, we drop the argument s from α whenever context allows. The stability ratio measures by how much state s allows selfish users to unilaterally decrease their utility. The ratio is 1 for a NE. This is the basis to formulate the following approximate equilibrium concept.

Definition 2.15 *A state* s *of a strategic game* Γ *with stability ratio at most* α *and approximation ratio at most* β *is called an* (α, β)-approximate NE.

The concept of (α, β)-approximate NE poses an obvious two-parameter optimization problem: On the one hand one strives to find a state with social cost as small as possible, on the other hand this might exclude stable solutions if the price of stability is high (or the game has no NE). In these cases it is interesting to study the properties of states s with a good trade-off between these two objectives, which are represented by the stability and approximation ratios of s. A NE is a $(1, \beta)$-approximate NE, the state s^* with optimum social value is a $(\alpha, 1)$-approximate NE.

The price of anarchy (stability) for a game with social cost or welfare is the largest (smallest) β such that there exist $(1, \beta)$-approximate NE in the game.

The concept of (α, β)-approximate NE is closely related to the more prominent concept of ϵ-NE [Eve57]. While our ratios are *relative* parameters, ϵ-NE are states that violate the Nash inequality (2.1) by an *absolute* value of at most ϵ. ϵ-NE are invariant to adding a constant to all payoffs, (α, β)-approximate NE are invariant to scaling all payoffs by a positive factor. As the games in this thesis are studied in combination with approximation algorithms for combinatorial optimization problems, we use (α, β)-approximate NE as a stability concept with relative performance measures.

2.3 Computational Complexity

In the previous sections we have introduced tools to study existence and approximation problems for NE in strategic games with social cost function. In this section we describe concepts to study complexity theoretic questions in games.

At the very heart of complexity theory in computer science lies the concept of the Turing machine, the definition of classes P and NP, and the notions of completeness and hardness with the most famous variants, the NP-completeness and NP-hardness. For a detailed introduction to these concepts, the underlying assumptions and interesting extensions we refer the reader to the classic textbook by Garey and Johnson [GJ79]. In the context of games some interesting questions concerning complexity are:

- How hard is it to decide whether a given game has a NE?
- How hard is it to compute a NE for a given game if it exists?
- How hard is it to compute the NE with best/worst social cost?
- How hard is it to compute the state of best/worst social cost?

Note that the answers to these questions depend on the representation of the game and can become quite trivial. Suppose the game Γ is represented in normal form as in Figures 2.1 and 2.2 by a large table listing all utility and social cost values for all possible states $s \in \mathcal{S}$. Then for each of the mentioned tasks there is a polynomial time algorithm. In particular, the algorithm works by scanning for every state all the states that differ in exactly one strategy and comparing the utilities for each player. As the length of the input is in $\Theta(k|\mathcal{S}|)$, we can then find all NE of the game in time at most quadratic of the input size. By evaluating the social cost we can also find best/worst states and/or NE in the same time. This is in contrast to the completeness results of finding mixed NE outlined in Section 1.1 that assume a representation in normal form.

If the game is represented in some succinct manner, however, the questions for NE can become much harder to answer. For each game considered in this thesis we will outline such a succinct representation. It is easily observed that most of the computational tasks involving NE can become NP-hard problems. For example, for the games in Chapters 4 and 5 evaluating the best-response functions $\mathsf{br}_p(s)$ for a state s can already pose an NP-hard minimization problem. Thus, given a game and a state s, even the recognition of s as a NE becomes an NP-complete problem. The problem of deciding the existence of a NE is not even in NP, and neither is any of the problems involving best/worst NE. In addition, for some of the games in Chapters 4 and 5 deciding whether a game allows a NE is NP-hard. For special classes of games, however, it is possible to draw a more differentiated picture. For each class of games considered in this thesis, we present specific answers in the corresponding chapters.

While for some games even deciding the existence of a NE is an NP-hard problem, for congestion and potential games we have already seen that the decision problem is trivial. It is not surprising that computing the best or worst NE in a succinctly represented potential game can be NP-hard. This can be shown easily, for instance for a class of congestion games with social cost as sum of latencies [IMN$^+$05]. But the problem of finding just a single (not necessarily best or worst) NE belongs to a different complexity class. NE correspond to states, which are local optima with respect to the potential function and a neighborhood of single player strategy switches. If there are polynomial time algorithms to evaluate the potential function and to find for each player and each state a state with improved utility (if it exists), then the problem of finding a NE is in the class PLS defined for local search problems. We outline some details on this connection and introduce the complexity class formally. The starting point is the underlying concept of a local search problem. We present the definition for minimization problems, and as before it can be adjusted easily to maximization problems.

Definition 2.16 *A* local search (minimization) problem $\Lambda = (\mathcal{I}, \mathcal{Z}, \mathsf{feas}, \mathsf{neigh}, \mathsf{c})$ *is given by*

- *a set of* instances $\mathcal{I}$ *and a set of possible solutions* $\mathcal{Z}$*,*
- *a function* $\mathsf{feas} : \mathcal{I} \to \mathcal{Z}$*, which returns the set of feasible solutions for an instance* $I \in \mathcal{I}$*,*
- *a* neighborhood $\mathsf{neigh}(I, z) \in 2^{\mathsf{feas}(I)}$ *assigning a set of* neighbors *to every feasible solution* $z \in \mathsf{feas}(I)$ *to any instance* $I \in \mathcal{I}$*, and*
- *a cost function* $\mathsf{c}(I, z) \in \mathbb{R}$ *assigning a cost to every feasible solution* $z \in \mathsf{feas}(I)$ *for every instance* $I \in \mathcal{I}$*.*

The goal is to find for each instance $I \in \mathcal{I}$ *a feasible solution* $z^*_I \in \mathsf{feas}(I)$ *such that* $\mathsf{c}(I, z^*_I) \leq \mathsf{c}(I, z)$ *for all neighbors* $z \in \mathsf{neigh}(I, z^*_I)$*. The solution* z^*_I *is called a* local optimum*.*

For instance, we can consider a class of finite potential games as instances, the states of a game as feasible solutions, the set of states that differ in the strategy choice of at most one player as the neighborhood of a state, and the potential as cost function. Then finding a NE in this class of potential games satisfies the definition of a local search problem. For complexity considerations it is important, whether for a given instance I and a feasible solution z the evaluation of neighborhood and cost functions and the check for local optimality of z can be done in polynomial time. Formally, we again assume that instances and solutions are represented as character strings over a finite alphabet. In addition, the representation of a feasible solution z is polynomial in the representation of the corresponding instance I, i.e. $|z| \leq \mathsf{poly}(|I|)$ where poly is a polynomial of constant degree.

Definition 2.17 *A local search problem* Λ *is in the class* PLS *(*P*olynomial time* L*ocal* S*earch) if there are three polynomial time algorithms* A_Λ*,* B_Λ *and* C_Λ *with the following properties.*

- *Given a string* x*, algorithm* A_Λ *determines if* x *is an instance of* Λ *and in this case produces some feasible solution* $z \in \mathsf{feas}(x)$*.*
- *Given an instance* I *and a string* x*, algorithm* B_Λ *determines if* x *is a feasible solution* $x \in \mathsf{feas}(I)$ *and in this case calculates the cost* $\mathsf{c}(I, x)$*.*
- *Given an instance* I *and a solution* z*, algorithm* C_Λ *determines if* z *is a local optimum. If this is not the case,* C_Λ *returns a neighbor* $z' \in \mathsf{neigh}(I, z)$ *with strictly better cost* $\mathsf{c}(I, z') < \mathsf{cost}(I, z)$*.*

PLS was introduced by Johnson et al. [JPY88]. It captures the intuition that there is a local search algorithm that efficiently computes a starting solution. Then it iteratively uses efficient steps to move to better neighboring solutions. This is not necessarily a polynomial time algorithm to solve the local search problem, because it might use an exponential number of iterations to reach a local optimum. Hence, it is of interest to characterize the hardest local search problems in PLS, which is done by reduction and a completeness definition.

Definition 2.18 *For* $\Lambda_1, \Lambda_2 \in$ PLS *a* reduction *from* Λ_1 *to* Λ_2 *consists of two polynomial time computable functions* g *and* h *such that*

- h *maps instances* I *of* Λ_1 *to instances* $\mathsf{h}(I)$ *of* Λ_2

- g *maps a solution to the mapped instance* $z_2 \in \mathsf{feas}(\mathsf{h}(I))$ *and the corresponding instance* I *of* Λ_1 *to a solution* $\mathsf{g}(z_2, I) \in \mathsf{feas}(I)$.
- *for all instances* I *of* Λ_1*, if* z_2 *is a local optimum for* $\mathsf{h}(I)$ *of* Λ_2*, then* $\mathsf{g}(z_2, I)$ *is a local optimum for* I *of* Λ_1.

We say that Λ_1 reduces to Λ_2.

Reductions for PLS are similar to standard reductions for NP. For each instance of Λ_1 we can construct with h in polynomial time an instance of Λ_2. If Λ_2 is solved, then we can use g to turn the local optimum z_2 in polynomial time into a local optimum for Λ_1. Hence, in this case Λ_2 is at least as hard as Λ_1. Note that the reduction is transitive.

Lemma 2.1 *If* Λ_1 *reduces to* Λ_2 *and* Λ_2 *reduces to* Λ_3*, then* Λ_1 *reduces to* Λ_3.

Thus, if every problem of PLS reduces to Λ_1 and Λ_1 reduces to Λ_2, then every problem of PLS reduces also to Λ_2. We call such a property PLS-*hardness.*

Definition 2.19 *A problem is called* PLS-hard *if all problems in* PLS *reduce to it. A problem* $\Lambda \in$ PLS *is called* PLS-complete *if all problems in* PLS *reduce to* Λ.

This yields similar notions as the well-known NP-hardness and NP-completeness. If a PLS-hard problem can be solved in polynomial time, then with the help of the reductions every problem in PLS can be solved in polynomial time. The difference between hardness and completeness is that the corresponding problem must be part of PLS. The class of PLS-hard problems can include any sorts of problems, e.g. decision problems, optimization problems, or search problems. The PLS-complete problems represent in this sense the hardest local search problems in PLS. Johnson et al. [JPY88] introduced PLS and provided a PLS-complete local search problem called CIRCUIT FLIP. In addition, there are a number of intuitive PLS-complete problems with natural neighborhood functions, e.g. the TRAVELLING SALESMAN Problem with the k-OPT neighborhood for any fixed k [Kre89]. PLS-completeness means the problem is one of the hardest in a class including a set of intuitive and frequently studied local search problems. It was shown that for any PLS-complete problem and any standard local search algorithm with arbitrary pivoting and tie-breaking rules there exist instances for which the algorithm needs exponential running time [Yan97, Theorem 13]. Thus, PLS-completeness implies that the standard local search approach can fail to provide a local optimum in reasonable time. However, nothing has been said about different solution algorithms, and in fact it is unknown, whether a local optimum can be found in polynomial time for all problems in PLS. However, there is some evidence [JPY88] that no problem in PLS is inherently intractable in the usual sense.

Theorem 2.3 *If a problem in* PLS *is* NP*-hard, then* $\mathsf{NP} = \mathsf{co\text{-}NP}$.

For the results in this thesis, the MaxCut problem is of special interest.

Problem 2.4 (MaxCut) *Given a simple undirected graph* $G = (V, E)$ *and a function* $c : E \to \mathbb{R}_+$*, find a partition of* V *into two sets* V_1 *and* V_2 *such that the value of the edge cut* $\sum_{e \in E \cap (V_1 \times V_2)} c(e)$ *is maximal.*

A natural representation for a solution is to encode the partition as a bitstring of length $|V|$, which considers one vertex for each position and encodes as 0 or 1 the partition side on which the vertex is located. For the local search maximization problem we add a neighborhood function called Flip. For any solution it consists of all bitstrings of Hamming distance at most 1. This naturally corresponds to all partitions that differ only in the assignment of at most one vertex. The following theorem has been shown in [SY91].

Theorem 2.5 *Finding a local optimum for* MaxCut *with the* Flip *neighborhood is* PLS*-complete.*

MaxCut can be interpreted easily as a game by considering each vertex as a player that chooses the partition as a strategy. This game has been previously considered as parity affiliation game [FPT04] or cut game [CMS06]. The utility for a player p in a given state s is given by $\mathrm{util}_p(s_p, s_{-p}) = \frac{1}{2} \sum_{s_q \neq s_p} c(p, q)$. The social welfare function is given by the sum of players utilities. In a best response step a player tries to increase the values of cut edges. It is easy to verify that the utility increase of a player in such a step corresponds to half the increase of the welfare function. Therefore, the sum of player utilities yields both, the welfare and a potential for the game. The following corollary is now immediate.

Corollary 2.1 *Finding a NE in a potential game is* PLS*-hard.*

Recall that this result is dependent on the representation of the game. It holds for games that allow a succinct representation like the MaxCut game, which can be encoded using the graph and the associated edge values. Thus, for potential games a best-response iteration can take a number of steps exponential in the size of a succinct representation of the game until arriving at a NE. In general, utility and best-response functions might not be polynomial time computable. Thus, the existence of the required algorithms in Definition 2.17 is not guaranteed, and the local search problems posed by finding a NE in arbitrary congestion and potential games are not necessarily in PLS. For all potential games considered in this thesis, however, the corresponding local search problems are in PLS. Hence, for the MaxCut game the problem is PLS-complete. More information about the complexity of classes of potential games and properties that lead to polynomial or exponential time best-response iterations can be found in [FPT04, ARB06].

2.4 Graphs

Let us introduce some concepts from graph theory and their notation that is used throughout the thesis. A *graph* or *undirected graph* $G = (V, E)$ is a pair of two multisets, the *vertices* $V = \{v_1, \dots, v_n\}$ and *edges* $E = \{e_1, \dots, e_m\}$, for $n, m \in \mathbb{N}$. Each edge $e \in E$ is a two-element subset $e = \{v, w\} \subseteq V$. For a *digraph* or *directed graph* edges are ordered pairs of vertices. A graph is called *simple* if there are no edges $\{v, v\} \in E$ for any $v \in V$ and each edge appears at most once in E. A simple graph that includes all possible edges is called *complete* or a *clique*.

The following concepts for graphs generalize to directed graphs in the obvious way. An edge $\{u, v\}$ and a vertex u are called *incident*. In a digraph an edge (u, v) is an *outgoing* edge for u and an *ingoing* edge for v. Two vertices u and v are *adjacent* if there is an edge $\{u, v\} \in E$. The *degree* of a vertex v is the number of edges incident at v and is denoted by $\deg(v)$. The *neighborhood* of a vertex u is the set of adjacent vertices $N(u) = \{v \in V \mid \{u, v\} \in E\}$. A graph is called *regular* if every vertex has the same degree.

A *subgraph* $G' = (V', E')$ is a graph such that $V' \subseteq V$ and $E' \subseteq E$. An *induced subgraph* $G[E'] = (V', E')$ is a graph such that $E' \subseteq E$ and V' is exactly the set of incident vertices of the edges in E'. Similarly, the induced subgraph $G[V'] = (V', E')$ is a graph such that $V' \subseteq V$ and E' is exactly the set of edges incident only with vertices from V'. A simple graph is called a *path* $\mathcal{P}$, if the vertices can be numbered such that v_i is adjacent to v_{i+1} for $i = 1, \dots, n - 1$, and to v_{i-1} for $i = 2, \dots, n$, and there are no other edges. n is called the *length* of the path. A *cycle* is a path of length at least 3 with an additional edge between v_1 and v_n. A *tree* $\mathcal{T}$ is a graph for which no subgraph is a cycle. If a tree has one vertex that is connected to all other vertices, it is called a *star*. A graph is called *connected* if it contains a path for every pair of vertices and *unconnected* otherwise. A *component* C of a graph G is a maximal connected subgraph, i.e. for a component C there is no other connected subgraph of G that contains C as a subgraph.

A *cluster* C of a graph G is a subset of the vertices V. A *clustering* $\mathcal{C}$ of G is a partition of V into mutually disjoint clusters. A x-*clustering* of G is a partition of V into at most x mutually disjoint clusters.

Part I

Investment Games

Chapter 3

Formal Framework

Service installation, facility and server location, and network creation are some important aspects in the development of the Internet. In the first part of this thesis we study a general class of *investment games*, which allows to derive strategic games for optimization processes concerning all these aspects. More generally, our model allows to turn a large class of optimization problems with a covering aspect into a strategic cost sharing game.

An investment game can be outlined as follows. There are k players in the game, and there is a set R of resources. Each player p has an associated constraint $\mathsf{valid}_p \subset \mathbb{N}^R$ that specifies possible combinations of resources that should be bought. She insists on fulfilling this constraint. Each resource $r \in R$ is available for purchase in costly integer units, in particular, for each resource there is a unit cost $c(r) \in \mathbb{R}_{\geq 0}$. A unit of resource r is considered *bought* if the corresponding cost $c(r)$ is paid for. A strategy for a player is an investment function $s_p : R \to \mathbb{R}_{\geq 0}$, which specifies for each resource r how much payment player p is willing to contribute to units of r. For simplicity we assume that the number of bought units of resource r is determined by the total amount of payment offered by all players to resource r. The bought units of a resource are considered by *all* players to satisfy their constraints, no matter whether this player contributes to the cost or not. A player p insists on satisfying her constraint valid_p. If there are several strategies that would do, she chooses the one minimizing her total investment $\sum_{r \in R} s_p(r)$.

Investment games are models for cost and usage sharing of resources. In these games we assume that players make investment decisions to purchase certain resources. For such decisions one is usually required to specify a concrete investment and not to take a randomized action. As was argued in [ADTW03] treating these games as a random experiment is rather unnatural given the motivation for the model. Hence, the following chapters address the existence and computability of exact and approximate NE in pure strategies. Our intuition does not rule out that a convincing application of mixed NE in investment games exists. At least from a technical perspective it is an interesting direction for future work to explore the properties of mixed NE in these games.

Investment games are closely related to non-cooperative cost sharing games based on the Shapley value [ADK+04]. In these games a strategy is not an investment function but a subset of resources. The cost of a resource unit is split between players requesting it in a pre-defined way, e.g. equally [ADK+04] or based on player weights [CR06]. Games with equal cost sharing scheme are easily shown to be potential games and have NE, the price of anarchy is $\Theta(k)$ and the price of stability $\Theta(\log k)$. In addition, they possess a fairness aspect to exclude *free riders*, players that get their constraint satisfied without contributing to the cost. However, this is achieved by sacrificing generality and allowing players only a much smaller set of actions. Also, the techniques used for analysis rely in large parts on characterizing best responses and potential functions, which is fundamentally different from our approaches to investment games in the next chapters.

Another recently prominent model for competitive resource allocation has been proposed by Johari and Tsitsiklis [JT04]. In the simplest model players submit bids, and a resource is then distributed proportionally to the size of the players bids. The utility of a player is a function of the acquired share minus the bid. Using the fact that utility functions are concave and strategy spaces are continuous, these games can be shown to possess unique, but inefficient NE. The price of anarchy is $\frac{4}{3}$, and this is best possible for a certain class of fixed-price resource sharing policies [JT05]. In contrast to our investment games, this model represents an approach to model distribution of established resources to players. It is also technically different because it assumes divisible resources and concave utility functions, which are major ingredients for the proof techniques.

Considering our analysis and proof techniques, the closest relations exist to mechanisms and cooperative games for cost sharing and combinatorial optimization. We outline these connections and related work in the following chapters, which treat games based on specific problems in detail. In addition, investment games are related to numerous works from the last decades on various cost sharing models for selfish agents. As a starting point the reader is referred to [You94].

Definition and Initial Observations

Definition 3.1 *An* investment game $([k], R, \mathcal{S}, \mathtt{util}, c, \mathtt{valid})$ *is given by*

- *a set* $[k] = \{1, \ldots, k\}$ *of* k *players, and a set* R *of resources,*
- *the state space* $\mathcal{S} = \mathcal{S}_1 \times \ldots \times \mathcal{S}_k$*, and for each player* $p \in [k]$ *a set of* strategies $\mathcal{S}_p$*, for which* $s_p \in \mathcal{S}_p$ *is a function* $s_p : R \to \mathbb{R}_{\geq 0}$*,*

- *the* utility functions $\mathtt{util}_p$ *defined as*

$$\mathtt{util}_p(s_p, s_{-i}) = \begin{cases} -|s_p|, & \textit{if } \mathtt{bought}(p) \in \mathtt{valid}_p \\ -\infty & \textit{otherwise} \end{cases}$$

  with $|s_p| = \sum_{r \in R} s_p(r)$*, and*

- $c : R \to \mathbb{R}_{\geq 0}$ *specifies for each resource* $r \in R$ *a non-negative unit cost* $c(r)$*,*

- $\mathtt{bought} : \mathcal{S} \to \mathbb{N}^R$ *specifies for each state* s *the number of bought resource units and is given by*

$$\mathtt{bought}_r(s) = \left\lfloor \frac{\sum_{p \in [k]} s_p(r)}{c(r)} \right\rfloor \quad \textit{for each resource } r \in R,$$

  and

- $\mathtt{valid}_p \subseteq \mathbb{N}^R$ *is the set of feasible combinations of resources for player* p*.*

In these games a player is meant to make a costly investment into certain resources. Hence, utility represents cost for a player, and by maximizing her utility a player *minimizes* her cost contribution. Consequently, we consider social cost $\mathtt{cost}(s) = -\sum_{p \in [k]} \mathtt{util}_p(s)$. Note that any state s of finite social cost must pay for a set $\mathcal{R}$ of resource units that satisfies all constraints $\mathtt{valid}_p$. We will implicitly assume in all our games that at least one such set exists. Then in every NE a feasible set of resource units is purchased, as each player prefers to purchase completely any feasible solution rather than leaving her constraint unsatisfied. For a NE and a social optimum state s^* we can derive the following properties:

- Each bought resource unit is required by at least one player for constraint satisfaction.
- For each resource unit of a resource r the total amount offered for its purchase is either $c(r)$ or 0.
- In a NE a player contributes only to resource units that are mandatory to satisfy her constraint.

In particular, in a state s^* minimizing function $\mathtt{cost}$ a valid set $\mathcal{R}$ of resource units of minimum total cost is exactly purchased. We denote such a set by $\mathcal{R}^*$ and note

$$\mathtt{cost}(s^*) = -\sum_{p=1}^{k} \mathtt{util}_p(s^*_p, s^*_{-p}) = \sum_{p=1}^{k} |s^*_p| = c(\mathcal{R}^*). \tag{3.1}$$

Finding a solution $\mathcal{R}^*$ poses a minimization problem of Definition 2.8. A NE represents a cost sharing of a (not necessarily optimal) solution to the underlying optimization problem.

Chapter 4

Covering and Facility Location Games

This chapter studies *covering games* as a special class of investment games. For introduction consider the important special case of vertex covering. Recall the definition of VERTEX COVER as Problem 2.2 in Section 2.2. Based on this covering problem a *vertex cover game* for k players can be described as follows. In a graph G each player p owns a set $E_p \subseteq E$ of edges. Each player strives to cover her edges. For each vertex $v \in V$ there is a non-negative cost $c(v)$, and a strategy for a player p is a function $s_p : V \to \mathbb{R}_{\geq 0}$ specifying an offer to costs of each vertex. The cost of a strategy s_p for player p is the sum of all money she offers for the vertices. Once the sum of offers of all players for vertex v exceeds its cost it is considered *bought*. Bought vertices are considered to be in the cover and can be used by all players to cover their incident edges. Each player strives to minimize her cost, but she insists on covering her edges. Similarly to the optimization problem we call a game *unweighted* if all vertices have equal costs, and *weighted* otherwise. We refer to games with a planar graph G as *planar games*.

Now consider an instance of VERTEX COVER as an integer linear program [NW88]. For each vertex $v \in V$ there is a binary variable x_v indicating whether it is in the cover or not. Furthermore, for each edge $e = \{u, v\}$ there is a constraint $x_u + x_v \geq 1$ ensuring that e is covered. The cost of a vertex is given by the cost value $c(v)$, which appears in the objective function.

$$\begin{array}{lll} \text{Min} & \sum_{v \in V} c(v) x_v & \\ \text{subject to} & x_v + x_u \geq 1 & \text{for all } \{u, v\} \in E \\ & x_v \in \{0, 1\} & \text{for all } v \in V. \end{array}$$

Suppose for a vertex cover game the underlying optimization problem is described by the above integer program. Then a player strives to purchase resources represented by vertices to satisfy the constraints corresponding to her edges. Here a variable corresponds to $x_u = \mathsf{bought}_u(s)$, and is thus raised to 1 if the sum of money offered to vertex u by the players exceeds $c(u)$. Thus, valid_p for player p contains the vectors x such that for each edge $\{u, v\} \in E_p$ the inequality $x_u + x_v \geq 1$ holds. Hence, we can specify the remaining parts of the Definition 3.1 for investment games to get a formal description of the vertex cover game.

Definition 4.1 *A* vertex cover game *is an investment game in which resources and constraints are given as follows.*

- *The resource set $R = V$ is the vertex set of a simple undirected graph $G = (V, E)$.*
- *Each player p has an edge set $E_p \subseteq E$. For the constraint $\mathsf{bought}(s) \in \mathsf{valid}_p$ if and only if $\mathsf{bought}_u(s) + \mathsf{bought}_v(s) \geq 1$ for all $(u, v) \in E_p$.*

For a succinct representation it is sufficient to encode the graph G, the cost function c, and the edge sets E_p of the players. In this game a social optimum solution corresponds to an optimum solution to the underlying instance of VERTEX COVER. This formulation of the game allows a straightforward translation to games based on arbitrary integer covering problems [Vaz00, Chapter 13.2].

Problem 4.1 (INTEGER COVERING PROBLEM) *Given two index sets R and T and non-negative constants $a(t, r)$, $b(t)$, $c(r) \geq 0$ for all $t \in T$ and $r \in R$, find an optimum solution to the following covering integer program (CIP):*

$$\begin{array}{lll} \text{Min} & \sum_{r \in R} c(r) x_r & \\ \text{subject to} & \sum_{r \in R} a(t, r) x_r \geq b(t) & \text{for all } t \in T \\ & x_r \in \mathbb{N} & \text{for all } r \in R. \end{array}$$

A *covering game* is based on an instance of the INTEGER COVERING PROBLEM. Each player p owns a subset of the constraints which she strives to satisfy. To reveal similarities with investment games for facility location and tree connection, we denote this set by T_p. Integral units of a resource r have cost $c(r)$. In accordance with the vertex cover game we use n for the number of resources and variables and m for the number of constraints.

Definition 4.2 *A* covering game *is an investment game in which constraints are given as follows.*

- *For each game there is a covering integer program with* n *variables and* m *constraints.*
- *Each player* p *has a subset of constraints* $T_p \subseteq T$*. For the constraint* $\mathtt{bought}(s) \in \mathtt{valid}_p$ *if and only if* $\sum_{r \in R} a(t,r) \cdot \mathtt{bought}_r(s) \geq b(t)$ *for all* $t \in T_p$*.*

For a succinct representation it is sufficient to encode the constraint parameters $a(t,r)$ and $b(t)$, the cost function c, and the constraint sets T_p of the players.

The rest of this chapter is organized as follows. In the following Section 4.1 we shed some light on how covering games are embedded into recent developments in the literature. Section 4.2 presents some initial observations in the covering game. In Section 4.3 we characterize cost and complexity of exact NE. There are upper and lower bounds of $\Theta(k)$ on the prices of anarchy and stability (Theorems 4.2 and 4.3), and it is NP-hard to determine the existence of a NE (Theorem 4.5). Section 4.4 presents efficient algorithms for computing approximate NE. For set cover games it is possible to compute (f, f)-approximate NE in polynomial time (Theorem 4.7), in which f denotes the maximum frequency of any elements in the sets. $(f, 1)$-approximate NE exist in any set cover game (Theorem 4.8), and for vertex cover the ratio of $f = 2$ is tight (Theorem 4.9). In Section 4.5 we argue that for the subclasses of set cover games with integrality gap 1 and singleton players optimal NE exist (Theorems 4.10 and 4.11), and the proofs provide polynomial time algorithms to compute optimal NE or near-optimal approximate NE, respectively. In the case of singleton players the results can be extended to set multi-cover games (Corollary 4.2), which have a slightly different constraint structure. In addition, for this class of games an exact (but possibly expensive) NE can be computed in polynomial time (Theorem 4.12). Section 4.6 translates almost all techniques and results for covering games to a class of facility location games (Theorems 4.14 - 4.19). The underlying problems do not stem from covering integer programs and represent another extension of the constraint structure. Finally, Section 4.7 contains a discussion on how our techniques are connected to results obtained in related games. In particular, it displays the close relations to a class of recently proposed cooperative games based on the same covering and facility location problems.

4.1 Previous and Related Work

Covering games represent a new way to model cost sharing between selfish agents in this scenario. They are most closely related to recent variants of cooperative games and mechanism design problems based on optimization. Devanur et al. [DMV05] considered cost sharing mechanisms for set cover and facility location problems. In these models every player corresponds to a single element or terminal and has a private utility (i.e. a willingness to pay) for being in the cover. The goal of the mechanism is to collect information about utility values, pick a subset of elements to be covered, find a minimum cost cover for the subset, and distribute costs to covered players such that no coalition can be covered at a smaller cost. A truthful or strategyproof mechanism allows no player to lower her cost by misreporting her utility value. In [DMV05] truthful mechanisms for set cover and facility location games were presented. For set cover games this work was extended [SLWC05, LSW05] to the consideration of different social desiderata like fairness aspects and model formulations with elements or sets being agents.

The mechanism design scenario models selfish service receivers who can either cooperate to an offered cost sharing or manipulate. Players may also be excluded from the game depending on their utility. A major goal has been to derive good cost sharing schemes that guarantee truthfulness or budget balance. Our game, however, is strategic and non-cooperative in nature and allows players a much richer set of actions. In our game each player is motivated to participate in the game. We investigate distributed uncoordinated service installation scenarios rather than a coordinated environment with a mechanism choosing customers, providing service and charging costs.

Closer to our covering game are cooperative games proposed in [DIN97, GS04, IMM05] based on covering and facility location. In these games each player has a single constraint or terminal, and each subset of players has a cost value associated with it. This value is the cost of an optimum cover for this subset of players only. A solution concept is a distribution of cost to the players. In these games each player contributes generally, and does not need to specify exactly how much is paid to which resource. Recently, Immorlica et al. [IMM05] presented bounds for cross-monotonic cost sharing schemes. For each coalition of players these schemes distribute the cost to the players such that every player is better off if the coalition expands. The authors showed that for vertex cover no more than $O(n^{-\frac{1}{3}})$, for set cover no more than $O(\frac{1}{n})$, and for uncapacitated facility location no more than $\frac{1}{3}$ of the cost can be charged to the agents with a cross-monotonic scheme, respectively.

When cross-monotonicity and budget balance cannot be achieved, a different desirable concept in a cooperative game is a *core* solution [PS03]. Specifically, the core is the foremost stability concept in cooperative games, and it includes cost allocations that assign any coalition of players at most the cost associated with this coalition. Deng et al. [DIN97] and Goemans and Skutella [GS04] showed that the core of cooperative integer covering and facility location games is non-empty if and only if the integrality gap of the underlying problem is 1. The main arguments rely on LP-duality, and a deeper discussion of the connections to these works is given in Section 4.7.

4.2 Initial Observations

The following initial observations can be used to simplify a covering game. Suppose a constraint of the integer program is not included in any of the constraint sets of the players. This constraint has no influence on the game. Hence, in the following w.l.o.g. we assume that the sets T_p form a partition of the constraint set. In particular, for the vertex cover game this means $E = \bigcup_{p=1}^{k} E_p$.

Suppose a constraint t is owned by a player p and a set of players $Q \subset [k]$, $p \notin Q$. Now consider a NE for an adjusted game in which the constraint is owned only by player p. In this NE a player $q \in Q$ has no better strategy to satisfy the constraints in $T_q - t$. However, t is satisfied as well, potentially by a different player. If t is added to T_q again, q has no incentive to deviate from her strategy as her covering requirement only increases. The NE for the adjusted game yields a NE in the original game. Hence, we assume that all constraint sets T_p are mutually disjoint, as all our results for NE and approximate NE continue to hold if the sets T_p are allowed to overlap. In particular, in the vertex cover game we assume all edge sets are mutually disjoint.

For the vertex cover game there is a decomposition property. Suppose the graph $G[E_p]$ induced by edge set E_p of player p is not connected. The player has to cover edges in each component, and her optimum strategy decomposes to cover both components independently at minimum cost. Hence, we can form an equivalent game in which the edges for each of the k_p components are owned by a different subplayer $p_1, \ldots, p_{k_p}$. Any approximate NE of this equivalent game can be translated to the original game, and the stability ratio can only improve. Hence, for deriving approximate NE we assume that the edges of each player form only a single connected component. This property can be translated to other covering games, however, it does not seem to have a similar intuitive meaning.

4.3 Cost and Complexity of Nash Equilibria

In this section we consider bounds on the quality of NE in vertex cover games and on the hardness of deciding their existence. In general it is not possible to guarantee their existence, they can be hard to find or expensive. At first observe that the price of anarchy in the vertex cover game is exactly k. This result continues to hold for general covering games.

Theorem 4.2 *The price of anarchy in the covering game is exactly* k.

Proof. Consider a star in which each vertex has cost 1 and each player owns a single edge. The centralized optimum cover $\mathcal{R}^*$ is the center vertex of cost 1. If each player purchases the vertex of degree 1 incident to her edge, we get a NE of cost k.

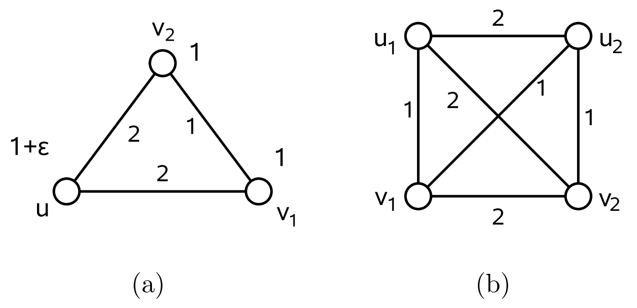


Figure 4.1: Vertex cover games for two players without NE. (a) Weighted game; (b) unweighted game. Numbering of edges indicates player ownership. For the weighted game numbers at vertices indicate vertex costs.

Therefore, the price of anarchy is at least k. On the other hand, k is a simple upper bound. If there is a NE, in which $\mathcal{R}$ with $c(\mathcal{R}) > kc(\mathcal{R}^*)$ is purchased, there is at least one player p who pays more than $c(\mathcal{R}^*)$. She could unilaterally improve by purchasing $\mathcal{R}^*$ all by herself. As the argumentation for the upper bound does not use specific properties of the vertex cover game, it continues to hold for all covering games. □

Note that the price of anarchy is k even for very simple games in which every player owns only one edge and G is a tree. Hence, in the following we consider existence and quality of the best NE in a game.

Lemma 4.1 *There are planar vertex cover games for two players without NE.*

Proof. We consider the game for two players in Figure 4.1(a) for an $\epsilon > 0$. For this game we consider the four possible covers. A cover including all three vertices cannot be bought in a NE, because vertex u is not needed by any player to fulfill the covering requirement. Hence, any player contributing to the cost of u could feasibly improve by removing these payments. Suppose the cover purchased in a NE includes v_1 and v_2. If player 1 contributes to v_1, she can remove these payments, because she only needs v_2 to cover her edge. With the symmetric statement for v_2 it follows that player 1 does not contribute in such a NE. Player 2, however, cannot purchase both v_1 and v_2, because buying u offers a cheaper alternative to cover her edges. Finally, suppose u and v_1 are in the cover. In a NE, in which this cover is purchased, player 1 does not contribute to the cost of u. Player 2, however, cannot purchase u completely, because v_2 offers a cheaper alternative to cover the edge (u, v_2). With the symmetric observation for the cover of u and v_2, we conclude that there is no feasible cover that can be purchased in a NE. With similar arguments we can prove that the game on K_4 depicted in Figure 4.1(b) has no NE. This proves the lemma. □

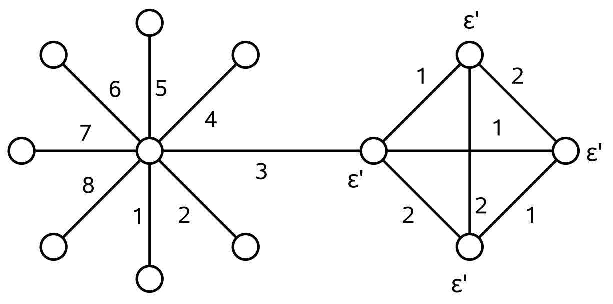


Figure 4.2: A game with $k = 8$, for which the price of stability is close to $(k - 1)$. Numbering of edges indicates player ownership. Indicated vertices have cost $\epsilon' \ll 1$, vertices without labels have cost 1.

Note that every game with less edges, vertices, or players than the game in Figure 4.1(a) is guaranteed to have a NE.

Theorem 4.3 *The price of stability in the weighted vertex cover game is at least* $k - 1$. *The price of stability in the unweighted vertex cover game is at least* $\frac{k+2}{4}$.

Proof. Consider a game as depicted in Figure 4.2. The social optimum cover includes the center vertex of the star and three vertices of the K_4-gadget yielding a total cost of $1 + 3\epsilon'$. If the center vertex of the star is in the cover, and if we assume that there is a NE, in which such a cover is purchased, none of the other players can contribute to vertices of the K_4-gadget incident to edges of player 1 and 2. For this network structure, however, we noted before that players 1 and 2 cannot agree on a set of vertices covering their edges. Therefore, in all NE of the game the star center must not be bought. In turn, this requires all other adjacent star vertices to be in the cover. Under these conditions the best feasible cover includes the vertex that connects K_4 to the star yielding a cost of $k - 1 + 3\epsilon'$. Such a NE is obtained by assigning each player to purchase a leaf vertex of the star - including the vertex that also belongs to K_4. Players 1 and 2 are assigned to purchase one of the additional K_4 vertices, respectively. With $\epsilon = \frac{3\epsilon'(k-2)}{1+3\epsilon'}$ we get a bound of $k - 1 - \epsilon$. Thus, if ϵ tends to 0 the bound becomes arbitrarily close to $k - 1$, which proves the first part of the theorem. For the unweighted case we simply consider the game graph with all vertex costs equal to 1. A similar analysis delivers the stated bound and proves the second part of the theorem. □

The existence of games without NE raises the question, whether we can decide for a given game that is has a NE or not. We prove that this decision problem is NP-hard using a reduction from 3-SATISFIABILITY (3SAT) [GJ79].

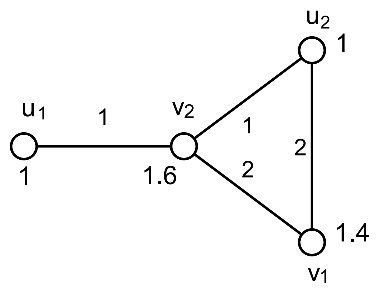


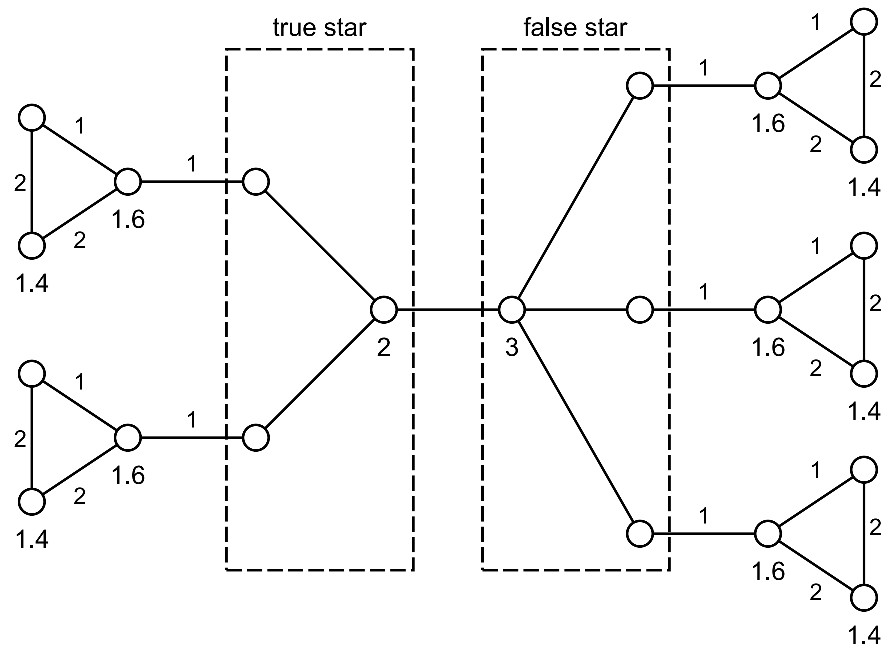
Figure 4.3: Extended triangle used in the proof of Theorem 4.5. This game does not have any NE. To stabilize the extended triangle in the gadgets described below, a third player can either buy u_1 completely, then the first two players can pay for u_2 and v_1; or reduce the cost of v_1 of 0.5, allowing the first two players to pay 0.9 for v_1 and 1.6 for v_2.

Problem 4.4 (3-Satisfiability) *Given a set of boolean variables and a set of clauses over the variables in conjunctive normal form with exactly three variables per clause, is there an assignment of variables such that every clause evaluates to true?*

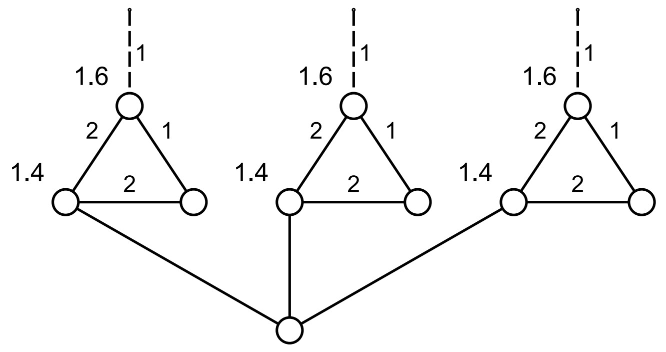
Theorem 4.5 *It is* NP*-hard to determine whether (1) an unweighted vertex cover game or (2) a weighted vertex cover game for 2 players has a NE, even if the graphs* $G[E_p]$ *are forests.*

Proof. We present the reduction for weighted games and then show how to adjust it for unweighted games. Given an instance of 3SAT we introduce for every variable a gadget with a *decision player.* This player owns the edges of two stars, denoted as the *true* star and the *false* star. The number of leaves of the (false) true star is equal to the number of (negated) non-negated occurrences of the variable in the clauses of the instance. The cost of each center vertex is equal to the number of leaves, the cost of the leaves is equal to 1. In addition, we include a direct connection between the centers of the stars. At the leaves of the stars we add extended triangle gadgets depicted in Figure 4.3. This gadget represents a game without a NE, which can be shown along the lines of the proof of Lemma 4.1. At each leaf vertex of the stars we install an extended triangle gadget. As no pair of gadgets is directly connected, this introduces only two new *triangle players.* The leaf vertices of the stars become the u_1-vertices of the gadgets. An example variable gadget is depicted in Figure 4.4(a).

For each clause we introduce a new *clause player.* She owns a star of three edges connecting a new center vertex of cost 1 to three extended triangle gadgets. We let the edges connect to triangles of the false or true star of a variable gadget depending on whether the variable appears negated or non-negated in the clause, respectively.



(a) Gadget for a variable occurring non-negated in two clauses and negated in three clauses.



(b) Gadget for a clause.

Figure 4.4: Variable and clause gadgets. The edges owned by triangle players are numbered, while edges owned by decision and clause player are unlabeled. Vertex labels represent corresponding costs, all unlabeled vertices have cost 1.

In particular, the edges connect to the v_1-vertices of the extended triangles. As we have installed a sufficient number of these gadgets, we construct the network such that no two edges of different clause players are incident at the same vertex. An example of a clause gadget is depicted in Figure 4.4(b).

Suppose there is a satisfying assignment for the instance of 3SAT. Then we construct a NE as follows. If a variable is true in the assignment, we pick the center vertex of her false star and all leaf vertices of the true star of its gadget to be in the cover and let the decision player pay for it. All extended triangles incident to the false star then allow a stable cost distribution, in which u_2 and v_1 are bought by the triangle players (see Figure 4.3). In the case a variable is false, we pick the leaf vertices of the false star and adjust the assigned payments accordingly. As we have a satisfying assignment for the 3SAT instance, this stabilizes at least one triangle gadget per clause. So each clause player has the chance to reduce the cost of the vertices of the remaining two incident triangles by 0.5 each. The triangle players can then purchase the v_1 and v_2 vertices in the remaining unstabilized gadgets (see Figure 4.3). This assignment leaves no player an incentive to defect and forms a NE.

Now suppose there exists a NE. Then a decision player can either purchase one or both of the star centers. Once she purchases the center of a star, she is not willing to contribute anything to the leaf vertices of the star. Thus, if she does not contribute to the extended triangles attached to a star, the clause players must help the triangle players agree upon a cover. However, a clause player can only contribute a total cost of 1 to the triangle vertices, because otherwise she can pick her star center as a cheaper alternative. The minimum cost reduction that she can achieve at every v_1 vertex of her incident triangle gadgets is $1/3$, which is not enough to allow for a stable cost assignment in all triangles. Hence, we need to have the decision players purchase the stars such that they trigger a stable cover in at least one extended triangle from each clause. Furthermore, as they can only trigger a stable cover in triangle gadgets attached to one of their stars, this naturally translates to a satisfying assignment for the 3SAT instance.

Finally, we use the transformations mentioned in the introduction to obtain an equivalent game by merging all decision players into one player and all clause players into another player. Note that we have introduced only two triangle players, whose edges form a partition of all edges from the extended triangle gadgets. The class of decision players shares endpoints only with one of the triangle players. The same is true for clause players and the other triangle player. Hence, we can merge the players again forming an equivalent game with only two players. This proves NP-hardness for weighted games and two players, even in the case of the graphs induced by the set of edges of each player are forests.

For unweighted games we replace the extended triangles by the games on K_4 depicted in Figure 4.1(b). Vertices labeled u_1 and v_1 indicate where to connect the decision and clause player stars, respectively. In the variable gadgets the star centers have cost 1. In addition, we introduce a number of new players such that edges of the true and false stars are each owned by a different player. For a clause player we now install two stars instead of one star. The stars have different centers, but leaf vertices from the same K_4 gadgets. Observe that in every variable gadget players in equilibrium contribute only to the leaves of at most one star. Furthermore, a clause player must invest at least a cost of 1 to stabilize a K_4 gadget. Hence, if at least one gadget per clause is stabilized by the decision players, there exists a NE. On the other hand, this condition is also necessary, because the centers of the clause stars allow the clause players to stabilize at most two K_4 gadgets. This proves NP-hardness.

To show that the result holds even when the graphs $G[E_p]$ are forests, note that the two stars of a clause player can be shared among two distinct players and the above reasoning is still correct. It can be checked that all graphs $G[E_p]$ in this game are forests. In this case it can be checked in polynomial time whether a state is a NE, because Vertex Cover can be solved in polynomial time on trees. The problem of deciding whether a NE exists, when restricted to these instances, is also in NP and so is NP-complete. This proves Theorem 4.5. □

4.4 Approximate Nash Equilibria

The results of the previous section showed that cheap NE can be absent from the game, even from very simple variants of the vertex cover game. This section explores whether there exist weaker notions of stability in this game or not. In particular, it examines the trade-off between efficiency and stability by considering existence and algorithmic computation of approximate NE. Recall from Definition 2.15 that for an (α, β)-approximate NE the stability ratio $\alpha \geq 1$ specifies the violation of the NE inequality, and $\beta \geq 1$ is the approximation ratio of the social cost. We first outline an algorithm for approximate NE for set cover games. To clarify the relation with Vertex Cover, we use a similar notation for elements and edges.

Problem 4.6 (Set Cover) *Given a set E of elements, a set $\mathcal{M} \subseteq 2^E$ of sets $M \subseteq E$, and a cost function $c : \mathcal{M} \to \mathbb{R}_{\geq 0}$, find a subset $\mathcal{R} \subseteq \mathcal{M}$ such that $\bigcup_{M \in \mathcal{R}} M = E$ and $\sum_{M \in \mathcal{R}} c(M)$ is minimal.*

In terms of the vertex cover problem a vertex v would be represented as a set M of incident edges. A *set cover game* is a covering game based on an instance of Set Cover. We denote by E_p the elements player p strives to cover, and by

$$f = \max_{e \in E} |\{M \in \mathcal{M} \mid e \in M\}|$$

the maximum frequency of any element in the sets. Note that a vertex cover game is a set cover game with $f = 2$, because every edge is incident to exactly two vertices.

For set cover games we show that $(f, 1)$-approximate NE exist and (f, f)-approximate NE can be computed in polynomial time. For vertex cover games the former result is shown to be tight.

Algorithm 1: (f, f)-approximate NE for set cover games

```
1 $s_p(M) \leftarrow 0$ for all players $p$ and sets $M$
2 $s_p(e) \leftarrow 0$ for all players $p$ and elements $e$
3 while there is an uncovered element $e$ do
4     Let $p$ be the player owning element $e$, and let $\gamma_p(e) \leftarrow \min_{e \in M} c(M)$
5     Increase payments: $s_p(M) \leftarrow s_p(M) + \gamma_p(e)$ for all $M$ with $e \in M$
6     Add all purchased sets to the cover
7     Reduce set costs: $c(M) \leftarrow c(M) - \gamma_p(e)$ for all $M$ with $e \in M$
```

Theorem 4.7 *Algorithm 1 returns an (f, f)-approximate NE for set cover games in polynomial time.*

Proof. The algorithm can clearly be implemented to run in polynomial time. To show the approximation ratio, we remark that any run of Algorithm 1 is also a run of the primal-dual f-approximation algorithm for minimum set cover (see for instance [Vaz00, chapter 15]). So it remains to show that the stability ratio is equal to f as well.

After the execution of the algorithm, consider player p and her best move taking into account the payments of all other players $q \neq p$. For that purpose, we define new costs $c_p(M)$ for each set M, by letting $c_p(M) = c(M) - \sum_{q \neq p} s_q(M)$. The theorem is proven if the sum of the payments of player p is not greater than f times the cost of the cheapest set cover of E_p with respect to the costs c_p.

From the algorithm, we know that for any set M we have $\sum_{q \in [k]} s_q(M) \leq c(M)$, hence

$$s_p(M) \leq c(M) - \sum_{q \neq p} s_q(M) = c_p(M).$$

Also from the algorithm, we know that for any set S that includes one or more elements of E_p, we have $s_p(M) = \sum_{e \in M \cap E_p} \gamma_p(e)$, so for any such M,

$$\sum_{e \in M \cap E_p} \gamma_p(e) \leq c_p(M).$$

Now consider a set cover $\mathcal{R}_p^*$ of E_p that has minimum cost with respect to the cost function c_p. This yields:

$$\sum_{M \in \mathcal{R}_p^*} \sum_{e \in M \cap E_p} \gamma_p(e) \leq \sum_{M \in \mathcal{R}_p^*} c_p(M) = c_p(\mathcal{R}_p^*).$$

Since $\mathcal{R}_p^*$ is a set cover of E_p, the charge $\gamma_p(e)$ of each element e in E_p is counted at least once in the left-hand side above. Hence,

$$\sum_{e \in E_p} \gamma_p(e) \leq \sum_{M \in \mathcal{R}_p^*} \sum_{e \in M \cap E_p} \gamma_p(e) < c_p(\mathcal{R}_p^*),$$

and this implies

$$\sum_{M \in \mathcal{M}} s_p(M) = f \cdot \sum_{e \in E_p} \gamma_p(e) \leq f \cdot c_p(\mathcal{R}_p^*),$$

which proves the theorem. □

Our arguments are also implicitly used in [JV01], in which dual payments and core solutions in cooperative games are considered. Alternatively, it is possible to employ these results to show that the resulting stability ratio is f. A proof along these lines is given for the primal-dual Algorithm 3 for the UFL game below in Section 4.6.2. For remarks on the connection to cooperative games see Section 4.7.

The algorithm can also be used to show that any social optimum cover $\mathcal{R}^*$ can be purchased in an $(f, 1)$-approximate NE.

Theorem 4.8 *For every set cover game there is an $(f, 1)$-approximate NE.*

Proof. Suppose we have an optimal set cover $\mathcal{R}^*$ for the underlying instance. Consider an iteration of Algorithm 1 with an element e. The contribution $\gamma_p(e)$ of the player owning e to all sets M with $e \in M$ is raised until it matches the cost the set M_e including e having minimum cost. Suppose after this step we remove e from the instance and drop all contributions of e to sets $M \notin \mathcal{R}^*$. Consider the new cost function $c_{-e}(M) = c(M) - \gamma_p(e)$ for $M \in \mathcal{R}^*$ and $e \in M$ and $c_{-e}(M) = c(M)$ otherwise.

Lemma 4.2 *The cover $\mathcal{R}^*$ is optimal for covering the elements $E - e$ in the instance $(\mathcal{M}, c_{-e})$.*

Proof. Suppose there is a cheaper cover $\mathcal{R}$ to cover the elements in $E - e$ under cost function c_{-e}, i.e. $c_{-e}(\mathcal{R}) < c_{-e}(\mathcal{R}^*)$. If all sets $M \in \mathcal{R}^*$ with $e \in M$ are also in $\mathcal{R}$, then it directly follows that $\mathcal{R}$ is a cheaper cover for the original instance. This contradicts the optimality of $\mathcal{R}^*$. Thus, there must be at least one set $M \in \mathcal{R}^*$ with $e \in M$ that is not in $\mathcal{R}$. If M is the only set covering e in $\mathcal{R}^*$, this might leave

e uncovered when considering $\mathcal{R}$ as a cover for the original instance. Here we add to $\mathcal{R}$ the set M_e that was used to limit the contribution of e in Algorithm 1. This turns $\mathcal{R}$ into a feasible cover covering all elements including e. Consider the costs of $\mathcal{R}$ and $\mathcal{R}^*$ under c and c_{-e}. The costs of a cover differ by $\gamma_p(e)$ for every set from $\mathcal{R}^*$ including e. By assumption $\mathcal{R}$ had dropped one or more of these sets, and we added only one set M_e of cost $\gamma_p(e)$ to regain feasibility. In total this yields $c(\mathcal{R}^*) - c_{-e}(\mathcal{R}^*) \geq c(\mathcal{R}) - c_{-e}(\mathcal{R})$, which implies $c(\mathcal{R}) < c(\mathcal{R}^*)$. $\mathcal{R}$ is a cheaper feasible cover for the original instance. This is a contradiction to $\mathcal{R}^*$ being optimal and proves the lemma. $\square$

Algorithm 1 can be adjusted in this way by determining payments and keeping them assigned only to sets of $\mathcal{R}^*$ for every element. The remainder of $\mathcal{R}^*$ stays optimal for the reduced set of elements under the reduced cost functions after every iteration. In particular, when considering the last element, a simple observation shows that all sets in $\mathcal{R}^*$ finally can get fully purchased. The stability ratio of at most f follows, as dropping the payments to sets outside $\mathcal{R}^*$ can only decrease the stability ratio. This proves the theorem. $\square$

For lower bounds on the ratios we note that any algorithm to find an (α, β)-approximate NE in the set cover game can be used as an approximation algorithm for the set cover problem with approximation ratio $\min(\alpha, \beta)$. The argument follows simply by considering a game with one player. This observation can be combined with recent results on the complexity status of Set Cover. In particular, Set Cover cannot be approximated in polynomial time to a factor of $o(\log |\mathcal{M}|)$ unless $\mathsf{P} = \mathsf{NP}$ [AMS06]. Thus, a polynomial time algorithm for $(O(\log |\mathcal{M}|), O(\log |\mathcal{M}|))$-approximate NE is all we can hope for. For the special case of weighted Vertex Cover a recent result [KR03] suggests that if $\mathsf{P} \neq \mathsf{NP}$ and the unique games conjecture [Kho02] holds, there is no polynomial time algorithm to approximate Vertex Cover by a factor of $2 - \epsilon$. Thus, in this case our algorithm delivers the best factors that are (based on current complexity theoretic beliefs) achievable in polynomial time. Note that both bounds apply only to polynomial time computability. We now show that in vertex cover games the frequency $f = 2$ is also a lower bound for the stability ratio, in a much stronger sense.

Theorem 4.9 *For any $\alpha < 2$ and $\beta \geq 1$ there is an unweighted vertex cover game without (α, β)-approximate NE.*

Proof. The proof follows with a game on the complete graph K_{4x} for a given natural number $x \in \mathbb{N}$. We assume the vertices are numbered v_1 to v_{4x} and distribute the edges of the game to $2x^2+x$ players in $x+1$ classes as follows. In the first class there are $2x$ players. Every player p from this class owns only a single edge (v_p, v_{2x+p}). Then, for each integer $j \in [1, x-1]$ there is another class of $2x$ players. A player p in one of the classes owns a cycle of four edges (v_p, v_{p+j}), (v_{p+j}, v_{2x+p}), (v_{2x+p}, v_{2x+p+j})

and (v_{2x+p+j}, v_p). Finally, there are x players in the last class. Each player p in this class also owns a cycle of four edges (v_p, v_{x+p}), (v_{x+p}, v_{2x+p}), (v_{2x+p}, v_{3x+p}) and (v_{3x+p}, v_p). See Figure 4.5 for $x = 2$ and the distribution of the 10 players into 3 classes on K_8.

Any feasible vertex cover of a complete simple graph is composed of either all or all but one vertices. For a cover of all $4x$ vertices we can simply drop the payments to one vertex. This reduces the payment for at least one player. In addition, it increases the cost of some of the deviations as the players must now purchase the uncovered vertex in total. The stability ratio of the resulting state can only decrease. Therefore, the minimum stability ratio is obtained by purchasing a cover of $4x - 1$ vertices.

So w.l.o.g. consider a state in which a cover of $4x-1$ vertices is bought, including all but vertex v_{4x}. Note that some player subgraphs do not include v_{4x}, and there are only two types of player subgraphs - a single edge or a cycle of length 4. First, consider a player subgraph that consists of a single edge and both endvertices are in the cover. If the player contributes to the cost of the incident vertices, she can drop the maximum of both contributions. Thus, if she contributes more than 0 to at least one of the vertices, her incentive to deviate is at least a factor of 2. Second, consider a player subgraph that consists of a cycle of length four and all vertices are in the cover. Label the four included vertices along a Euclidean tour with u_1, u_2, u_3 and u_4. Let the contributions of the player to u_j be y_j for $j = 1, 2, 3, 4$, respectively. To optimally deviate from a given state, the player picks one of the possible minimum vertex covers $\{u_1, u_3\}$ or $\{u_2, u_4\}$ and removes all payments outside this cover. A

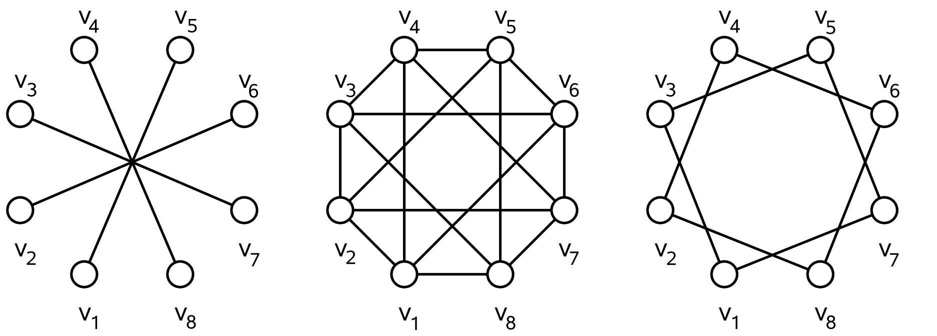


Figure 4.5: From left to right the edges owned by the players in the first, second, and third classes of players for K_8. The first and second class consist of four players each, the third class of two players. Players in the first class own a single edge, while players in other classes own cycles of length 4.

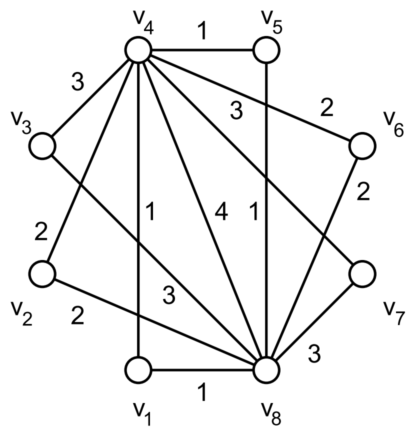


Figure 4.6: Edges of players that have an edge incident to v_8. Numbering of players as described in the text. Edge labels indicate player ownership.

factor of α bounding her incentives to deviate must thus obey the inequalities

$$\sum_{j=1}^{4} y_j \leq \alpha(y_1 + y_3) \qquad \text{and} \qquad \sum_{j=1}^{4} y_j \leq \alpha(y_2 + y_4).$$

Note that a player might also contribute to vertices outside her cycle. These additional contributions, however, would unnecessarily tighten the bounds and require an increase in α. Therefore, in order to find the minimum α that is achievable we assume the player contributes only to vertices inside her subgraph. Summing the two inequalities yields

$$(2-\alpha)\sum_{j=1}^{4} y_j \leq \sum_{j=1}^{4} y_j,$$

so either her overall contribution is 0 or $\alpha \geq 2$. Hence, to construct a state with stability ratio of less than 2, all $4x-1$ vertices in the cover must be purchased by the $2x$ players whose subgraph includes v_{4x}.

For the rest of the proof we concentrate on these $2x$ players. We will refer to player p, if she includes v_p in her subgraph, for $p = 1, \ldots, 2x-1$. All these players own cycle subgraphs. The player that owns the edge (v_{2x}, v_{4x}) is labeled player $2x$. See Figure 4.6 for an example on K_8. We consider the contribution $s_p(v_j)$ of player p to vertex v_j for all $p = 1, \ldots, 2x$ and $j = 1, \ldots, 4x-1$. Observe that for each player the set $\{v_{2x}, v_{4x}\}$ forms a feasible vertex cover. To achieve a stability ratio α, we must ensure that each player can only reduce her payments by a factor of at most α when switching to this cover. In the case of player $2x$ only $\{v_{2x}\}$ is needed, so we must ensure that she can reduce her payments by at most α when dropping all payments but $s_{2x}(v_{2x})$. As v_{4x} is not part of the purchased cover, its' cost of 1 must be paid for completely by a player that strives to use it in a deviation. This yields the following set of $2x$ inequalities:

$$\begin{aligned} \sum_{j=1}^{4x-1} s_p(v_j) &\leq \alpha(s_p(v_{2x})+1) \qquad \text{for } p = 1,\dots,2x-1 \\ \sum_{j=1}^{4x-1} s_{2x}(v_j) &\leq \alpha s_{2x}(v_{2x}) \end{aligned}$$

We again strive to obtain the minimum ratio α possible. In the minimum case no vertex gets overpaid, i.e. $\sum_{p=1}^{2x} s_p(v_j) = 1$ for all $j = 1,\dots,4x-1$. Using this property in the sum of all the inequalities gives

$$4x-1 = \sum_{j=1}^{4x-1}\sum_{p=1}^{2x} s_p(v_j) \leq \alpha\left(2x-1+\sum_{p=1}^{2x} s_p(v_{2x})\right) \leq 2x\alpha,$$

which finally yields $\alpha \geq 2 - \frac{1}{2x}$. This proves that in the presented game no (α,β)-approximate NE with $\alpha < 2 - \frac{1}{2x}$ exists. Thus, for every $\epsilon > 0$ we can pick $x \geq (2\epsilon)^{-1}$, which then yields a game without $(2-\epsilon,\beta)$-approximate NE for any $\beta \geq 1$. □

This lower bound is best possible for the considered class of games. For the players that include v_{4x} in their subgraph, assign player p to contribute $1 - \frac{1}{2x}$ to v_p and v_{2x+p} for $p = 1,\dots,2x-1$. Player $2x$ is assigned v_{2x} completely and the remaining cost of $\frac{1}{2x}$ at every other vertex. This yields a $(2-\frac{1}{2x},1)$-approximate NE.

It would be interesting to see if this lower bound is connected to the integrality gap of the above given integer program for Vertex Cover. Such a relation exists for approximate budget balanced core solutions in the cooperative game [JV01]. Our result, however, is mainly due to the fact that the majority of players is sufficiently overcovered leaving only a small number of contributing players. It seems that under these conditions a relation to the integrality gap is more complicated to establish. A discussion on the relation between our games and cooperative games based on covering problems can be found in Section 4.7.

Some classes of Vertex Cover can be approximated to a better extent. For example, there is a PTAS for Vertex Cover on planar graphs [Bak94]. It is therefore natural to explore if it is possible for planar games to find covers with approximation and stability ratio arbitrarily close 1. The bad news is that in general there are also limits to the existence of cheap approximate NE even for planar games. In particular, Theorem 4.9 provides a lower bound of 1.5 on the stability ratio for unweighted planar games. For weighted planar games there is an additional Pareto relationship between stability and approximation ratios that yields a stability ratio close to 2 for states in which near-optimal covers are bought.

Corollary 4.1 *For any $\alpha < 1.5$ and $\beta \geq 1$ there is a planar unweighted vertex cover game without (α, β)-approximate NE. For any $\alpha < \frac{2}{2\beta-1}$ and $\beta < \frac{7}{6}$ there is a planar weighted vertex cover game without (α, β)-approximate NE.*

Proof. With the planarity of K_4 and Theorem 4.9 the first part follows. For the second part consider a game from Figure 4.1(a) with $\epsilon > 0$. Here every state with finite stability ratio and $\beta < \frac{2+\epsilon}{2}$ returns $\mathcal{R}^* = \{v_1, v_2\}$. How good can this cover be in terms of the stability ratio? If player 1 contributes, she can always drop payments to the one vertex to which she contributes the most. If her contribution is greater than 0, her deviation incentive is at least a factor of 2. If we assign player 2 to purchase the whole cover, this delivers $\alpha = \frac{2}{1+\epsilon} < 2$ for all $\epsilon > 0$. Hence, once an algorithm returns (α, β)-approximate NE with $\beta < \frac{2+\epsilon}{2}$, then for this game any such cover has $\alpha > \frac{2}{1+\epsilon}$. Solving for ϵ we get the bound, which proves the second part of the corollary. □

So the better an algorithm is required to be in terms of social cost, the more it allows for selfish improvement by a factor close to 2. Note that these lower bounds apply directly to any algorithm with or without polynomial running time.

4.5 Games with Cheap Nash Equilibria

In contrast to general covering games there are some classes that have optimal NE. The first class are integral set cover games, in which the integrality gap of the linear programming relaxation is 1. The second class are singleton set cover games, in which each player owns only a single element. The results for this class can be extended to set multi-cover games, in which each player strives to cover her element by a certain number of sets, but each set is available for purchase only once.

Theorem 4.10 *If a set cover game has an underlying CIP with integrality gap 1, the price of stability is 1 and an optimal NE can be found in polynomial time.*

Proof. Consider the LP-relaxation of the underlying CIP derived by setting $x_M \geq 0$ instead of $x_M \in \mathbb{N}$. The dual of this relaxation is

$$\begin{array}{lll} \text{Min} & \sum\limits_{e \in E} \gamma_e & \\ \text{subject to} & \sum\limits_{e \in M} \gamma_e \geq c(M) & \text{for all } M \in \mathcal{M} \\ & \gamma_e \geq 0 & \text{for all } e \in E. \end{array}$$

We can find the optimum primal solution x^* and the optimum dual solution γ^* in polynomial time. Note that x^* is integral and thus defines a feasible cover due to an integrality gap of 1. Both x^* for the primal and γ^* for the dual have the same value. Now assign each player to pay $s_p(M) = \sum_{e \in E_p \cap M} \gamma_e^* x_M^*$. The theorem follows if the cover is purchased and every player plays a best-response. We first show that the cover is purchased. If $x_M^* > 0$, then due to complementary slackness the inequality $\sum_{e \in E} \gamma_e^* \leq c(M)$ is tight, hence by this assignment all the purchased sets get exactly paid for. Now we prove that x^* is a collection of best responses. For a player p consider an adjusted game, which is derived by iteratively removing elements and payments of other players. Upon removing an element e, we remove its contribution from the costs of sets M including e. This yields a cost function

$$c_p(M) = c(M) - \sum_{e \notin E_p, e \in M} \gamma_e^* x_M^*.$$

It captures the reduced problem, in which an optimum solution corresponds to a best response for player p. Note that for this reduced problem the solution x^* is still feasible. By obtaining the dual for the reduced problem we can set the covering requirement to 0 for every removed element e for $e \notin E_p$. Then γ^* still represents a feasible solution to the LP-dual of the reduced problem. It has the same value as x^* for the primal. LP-duality yields that both x^* and γ^* must be optimal solutions to the reduced primal and dual problems. Thus, the payment of p is a best response. This proves that the optimum solution can be paid for in a NE. As there is a polynomial time algorithm to solve linear programs [Kha79], it is possible to compute an optimal NE in polynomial time. This proves the theorem. □

For illustration of the arguments consider a vertex cover game on a bipartite graph G, which is known to have an integrality gap of 1, see Figure 4.7(a). To obtain an optimum dual solution one can employ a flow network using a standard construction [KT06] by adding a source and a sink vertex, see Figure 4.7(b). Each of these two vertices is then connected by directed edges to all vertices from one partition of the graph. The additional edges are directed away from the source to the sink and receive as capacity the cost of the incident vertex from G. All edges from G receive infinite capacity. A maximum flow in this network yields an optimum solution to the corresponding LP-dual of VERTEX COVER in G. The dual variables correspond to the flow values on the edges. They can be used to construct a cost sharing of an optimum vertex cover. To show that this represents a NE, Figure 4.7(c) illustrates the problem of finding a best response for player 2. For the reduced problem the flow over her edges is still feasible, hence by LP-duality it lower bounds the optimum cover cost. As the flow also yields a feasible strategy, player 2 plays a best response.

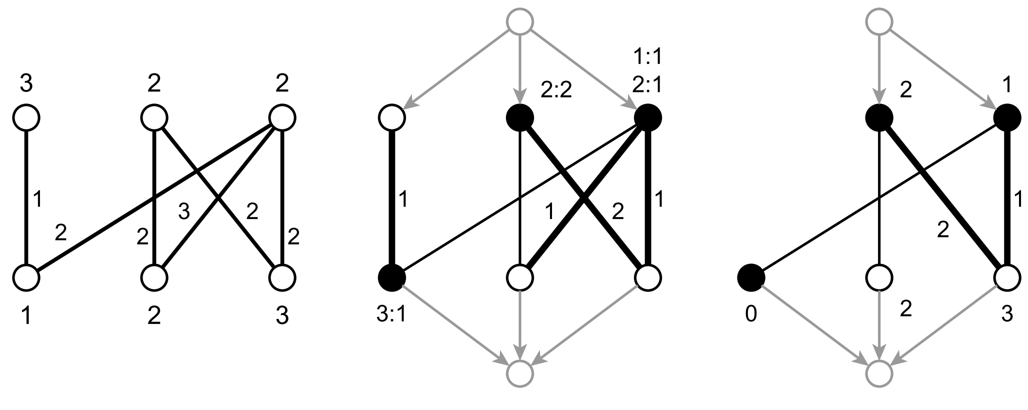


Figure 4.7: (a) A vertex cover game with an optimal NE. Edge numbering indicates player ownership, vertex numbering indicates cost. (b) Dual variables correspond to maximum flow values in an extended graph. It yields a cost sharing of an optimum cover indicated by filled vertices. Edge numbering represents top-down flow, labels of a vertex v are $p : s_p(v)$. (c) Player 2 plays a best response, because in the reduced problem the flow over her edges stays feasible. Edge numbering represents top-down flow, vertex numbering indicates cost, filled vertices constitute an optimum cover.

For singleton set cover games we prove a similar result. The proof, however, is along different lines. It does not immediately yield an efficient algorithm to compute an optimal NE, however, it allows us to obtain $(1+\epsilon, O(\log|\mathcal{M}|))$-approximate NE in time polynomial in $|\mathcal{M}|$, k and ϵ^{-1}, for any constant $\epsilon > 0$.

Theorem 4.11 *If a set cover game has singleton players with* $|E_p| = 1$*, the price of stability is 1 and* $(1+\epsilon, O(\log|\mathcal{M}|))$*-approximate NE can be computed in polynomial time, for any constant* $\epsilon > 0$*.*

Proof. Consider a solution $\mathcal{R}$ to the underlying instance of SET COVER and a set $M \in \mathcal{R}$. Consider the elements that remain covered if M is dropped out of $\mathcal{R}$. The corresponding players can be assumed to contribute nothing to the cost of M. Now consider the set of remaining players Q_M that are covered in $\mathcal{R}$ only by M. For each $p \in Q_M$ independently consider the case, in which M is unavailable. Her cheapest strategy to cover her element must then purchase a different set than M. We denote the cost of this other set by c_p^M. A set is called *stabilized* if $c(M) \leq \sum_{p \in Q_M} c_p^M$. For a stabilized set M we assign the players to pay

$$s_p(M) = \left(\frac{c_p^M}{\sum_{p \in Q_M} c_p^M} \right) c(M) \qquad \text{for } M \text{ with } p \in Q_M, \tag{4.1}$$

and $s_p(M) = 0$ otherwise. This obviously yields a best response for p. Thus, stabilized sets can be purchased by the players without incentives to deviate.

A solution $\mathcal{R}$ is called *exchange minimal* if all sets $M \in \mathcal{R}$ are stabilized. It is possible to use the above mentioned proportional assignment of the costs for stabilized sets M to players in Q_M. This will give them no incentive to deviate. In addition, note that overcovered players, whose elements appear in more than one set of $\mathcal{R}$, are never included in any Q_M, and are thus assigned no payments. This implies that this cost distribution of an exchange minimal solution is a NE. Note that it is possible to exchange unstabilized sets with the sets purchased by player deviations. This results in a feasible cover of reduced cost. In particular, this implies that $\mathcal{R}^*$ is an exchange minimal solution, and thus the price of stability is 1.

This property suggests a procedure that converges to a NE. We start with an approximate cover, exchange unstabilized sets, and decrease the cost of the cover until it can be purchased in a NE. The problem with this approach is the running time, as there might be an exponential number of exchange steps. Here we borrow a trick from Anshelevich et al. [ADTW03] and adjust the set costs such that each exchange operation guarantees a minimum improvement.

Using a well known greedy approximation algorithm [Hoc96, Chapter 3], we obtain a starting solution $\mathcal{R}$ with $\beta = H_n$, in which we denote $n = |\mathcal{M}|$ and $H_n \in \Theta(\log n)$ is the n-th harmonic number. Then consider an exchange step, in which this cover is transformed into a different cover $\mathcal{R}'$. In order to guarantee a minimum improvement of the cost, let $\epsilon > 0$ and define $\kappa = \frac{\epsilon c(\mathcal{R})}{(1+\epsilon)n\beta}$. Before determining the best responses of a player, we reduce the cost of every set currently in the cover by κ (assuming ϵ small enough such that $\kappa < \min_{M \in \mathcal{M}} c(M)$). However, for sets outside the cover we assume that they still have full cost. If there is an unstabilized set under these conditions, the exchange step improves the cost by at least κ. After the step we reduce costs of the sets entering the cover and reinstall the original cost of the set leaving the cover. The algorithm runs until no further improving exchange steps are possible. Hence, in total the algorithm makes at most $\frac{(1+\epsilon)n\beta}{\epsilon}$ such exchange steps. This proves a polynomial bound on the running time.

If the algorithm has terminated, we assign players to purchase the cost of the cover as described above. Note, however, that due to our cost reduction each set in the cover has a remaining unpaid cost of κ. Suppose in the final solution $\mathcal{R}'$ we have $n' = |\mathcal{R}'|$ sets. This creates a remaining cost of at most $\kappa n'$ to be paid for. These costs are distributed to the players such that each player gets a global share proportional to the costs already assigned, i.e. player p contributes a share of $|s_p|/(c(\mathcal{R}') - n'\kappa)$ of the remaining costs. Note that this might require a player to pay for sets not needed for covering her element. To establish the stability ratio of $(1 + \epsilon)$, note that the increase for player p in this step is only

$$|s_p|\frac{\kappa n'}{c(\mathcal{R}') - \kappa} \leq \frac{\epsilon c(\mathcal{R})|s_p|}{\alpha(1+\epsilon)(1-\epsilon)c(\mathcal{R}')} \leq \epsilon|s_p|.$$

Hence, even if all additional payments of player p contribute to sets she does not need for coverage, she cannot decrease her payments by more than a factor of $(1+\epsilon)$. This proves the bound on the stability ratio and establishes the second part of the theorem. $\square$

In fact, finding an exchange minimal solution poses a local search problem in PLS. Note that the described procedure is a local search algorithm. It is in essence similar to an adjusted best response iteration, in which in each iteration the players from a set $[k]_M$ for an unstabilized M are scheduled to simultaneously choose their best responses.

More generally, one might wonder if the existing FPTAS to compute an approximate local optimum for any problem in PLS [OPS04] can be used to obtain approximate NE with stability ratios arbitrarily close to 1. It computes a solution such that the *social cost* cannot be improved by more than a factor ϵ by moving to a solution from the neighborhood. Instead, we strive to find a cost sharing such that the *utility* of every player cannot be improved by more than a factor of ϵ by changing her strategy. This turns out to be a significant difference. Consider for instance the class of congestion games, for which finding a NE is a problem in PLS. In this case, the FPTAS computes an approximate local optimum of the potential. However, the state only approximates the value of the potential and does not yield the same result for the stability ratio. In fact, for any given polynomial time computable α, finding a state with stability ratio at most α in these games is PLS-complete. Furthermore, there are starting states from which the standard local search algorithm requires an exponential number of steps until the stability ratio drops below α [SV07].

It is straightforward to extend Theorem 4.11 to singleton set multi-cover games. An instance of Set Multi-Cover is given by an instance of Set Cover and an additional covering requirement $b(e) \in \mathbb{N}$ for each element $e \in E$. The goal is to find a set $\mathcal{R} \subseteq \mathcal{M}$ of minimum cost such that each element e is included in at least $b(e)$ sets from $\mathcal{R}$. A set multi-cover game is an investment game based on an instance of Set Multi-Cover. In particular, each player owns a subset of elements and strives to cover each element e by $b(e)$ sets, which must be bought. Note that each set is available for purchase only once. This introduces constraints into the underlying integer programming formulation that violate the structure of a CIP. Nevertheless, we get the following corollary.

Corollary 4.2 *If a set multi-cover game has singleton players with $|E_p| = 1$, the price of stability is 1 and $(1+\epsilon, O(\log|\mathcal{M}|))$-approximate NE can be computed in polynomial time, for any constant $\epsilon > 0$.*

Proof. We show how to adjust the arguments of the proof of Theorem 4.11. Consider a feasible cover $\mathcal{R}$ and a set $M \in \mathcal{R}$. When removing M from $\mathcal{R}$ we consider only the set of players Q_M with elements e, for which the covering extent drops below $b(e)$. Then, if each player $p \in Q_M$ purchases the cheapest set $M_p \notin \mathcal{R}$ containing her element e, we obtain a new feasible cover. If this new cover has higher cost, the set M is assumed to be stabilized. Otherwise, the cover drops in cost, and hence it is possible to construct an iterative procedure that converges to an exchange minimal cover with only stabilized sets. Now consider an exchange minimal cover $\mathcal{R}$. The same assignment rule as in Equation (4.1) is used to distribute the cost to the players, i.e. the cost of a stabilized set M is assigned to the players in Q_M proportionally to the cost of their deviations. If an element e is covered more than $b(e)$ times, the corresponding player does not contribute, because she does not appear in any of the sets Q_M. Hence, she has no incentive to deviate. Consider a player p with and element e that is covered exactly $b(e)$ times. The cheapest set $M_p \notin \mathcal{R}$ that contains e is considered as her best deviation for each of the $b(e)$ sets of $\mathcal{R}$ that contain e. Thus, p is assigned to pay at most $\min(c(M), c(M_p))$ for each set $M \in \mathcal{R}$ with $e \in M$. This obviously forms a best-response for p. Thus, an exchange minimal cover can be purchased in a NE. This proves that the price of stability is 1. The rest of the theorem follows by constructing the same algorithm with exchange steps and adjusted set costs as in the proof of Theorem 4.11. In particular, there is a similar greedy approximation algorithm for Set Multi-Cover [Vaz00] as for Set Cover with a performance ratio of $H_{|\mathcal{M}|}$ to compute a starting solution. The corollary follows. □

Another possible extension of the singleton case is to consider a threshold τ_p for each player p. Player p would rather stay uncovered if her assigned payments exceed τ_p. The outlined procedure finds an approximate NE with stability ratio arbitrarily close to 1 for this case of set cover and set multi-cover games as well.

Finally, we contrast the pervious two theorems with a procedure to compute an exact (but not necessarily cheap) NE. Note that in a step of the local search procedure a player is assigned to fully purchase the unbought set M_p she is deviating to. Using a similar idea it is possible to construct an exact NE without cost sharing, i.e. every set is either bought completely by one player or not contributed to at all.

Theorem 4.12 *Algorithm 2 returns an exact NE for singleton set multi-cover games in polynomial time.*

Proof. Clearly, Algorithm 2 can be implemented to run in polynomial time. It constructs a directed acyclic graph G_s, which contains a directed edge between sets M_1 and M_2 if and only if there is a player that prefers M_1 over M_2. If at any point in time a set M is dropped from consideration, it represents a sink in G_s. Hence, at this point for each remaining player with $e \in M$ it is the most expensive set that contains her element. None of these players will contibute anything to M,

Algorithm 2: Exact NE for singleton set multi-cover games

```
1  $s_p(M) \leftarrow 0$ for all $p \in [k]$ and $M \in \mathcal{M}$
2  Construct directed graph $G_s = (\mathcal{M}, E)$ with $e = (M_1, M_2) \in E$ iff
   $M_1 \cap M_2 \neq \emptyset$ and $c(M_1) < c(M_2)$
3  while there remain uncovered elements do
4      for every remaining element $e$ do
5          if $e$ is uncovered and included in exactly $b(e)$ sets $\mathcal{M}_e$ then
6              for every unbought set $M \in \mathcal{M}_e$ do
7                  Assign $p$ owning $e$ to contribute $s_p(M) \leftarrow c(M)$
8                  Remove edges $(M', M)$ with bought sets $M'$
9                  Redirect edges $(M', M)$ to $(M, M')$ with unbought sets $M'$
10             Drop $e$ from consideration
11     Find a sink in $G_s$ and drop the corresponding set from consideration
```

as they all have sufficiently many and cheaper alternatives to satisfy their covering requirement. As previous players were not motivated to purchase M, no player has an incentive to contribute to M.

When a player p is assigned to contribute, she is left with the set $\mathcal{M}_e$ of exactly $b(e)$ sets to cover her element e. In this case all other sets, in which e appears, have been dropped because they were too expensive. Thus, p is not be motivated to contribute to any other sets than those of $\mathcal{M}_e$. Every other player q will only use the sets of $\mathcal{M}_e$ for free, and this does not allow p to lower her contribution. Thus, her contribution is a best response, which proves the theorem. □

Unfortunately, the proposed algorithm can compute worst-case NE. Reconsider the singleton vertex cover game with a star network used to obtain a lower bound for the price of anarchy. Suppose the fixed cost is 1 for the leaf vertices and $1 + \epsilon$ for the center vertex. Algorithm 2 assigns each player to purchase the leaf vertex incident to her edge. This obviously yields a NE which is arbitrarily close to k times more costly than s^*.

It is an interesting open problem to adjust the results of this section to cope with general covering games. In particular, the two main obstacles in generalizing the techniques are the presence of multi-unit resources and best-responses purchasing non-trivial combinations of resource units.

4.6 Facility Location Games

4.6.1 Introduction and Basic Properties

Game-theoretic models for facility location have a long history dating back to the work of Hotelling [Hot29]. In the area of *competitive location* game-theoretic models for spatial and graph-based facility location have received a high attention during the last decades [ELT93, MFT96]. These models consider facility owners as players that selfishly decide where to open a facility. Clients are behavioral, e.g. they are assumed to connect to the closest facility and represent a revenue a player gets from opening a facility. Recent examples of this kind of location games are also found in [Vet02, DGK+05]. According to our knowledge, however, none of these models consider the clients as players that need to create connections and facilities without central coordination. In this section we propose such a model within our framework of investment games. We first restrict ourselves to one of the most simple variants, the uncapacitated facility location problem.

Problem 4.13 (Uncapacitated Facility Location) *Given a graph* $G = (T \cup F, T \times F)$ *with vertex sets* F *of* n_f facilities *and* T *of* n_t clients *or* terminals*, a non-negative opening cost* $c(f)$ *for each facility* f*, and a non-negative connection cost* $c(t, f)$ *for each edge* (t, f)*, find a non-empty subset of facilities* $\mathcal{F} \subset F$ *such that* $\sum_{f \in \mathcal{F}} c(f) + \sum_{t \in T} \min_{f \in \mathcal{R}} c(t, f)$ *is minimal.*

The goal in Uncapacitated Facility Location is to pick a subset of opened facilities and to connect each terminal to exactly one opened facility at minimum total cost. We denote an optimum solution by $\mathcal{F}^*$ and refer to metric Uncapacitated Facility Location if the connection costs obey the triangle inequality in the graph G. The classic integer programming formulation is due to Balinski [Bal61].

$$\begin{array}{lll}
\text{Min} & \sum_{f \in F} c(f) y_f + \sum_{t \in T} c(t, f) x_{tf} & \\
\text{subject to} & \sum_{f \in F} x_{tf} \geq 1 & \text{for all } t \in T \\
& y_f - x_{tf} \geq 0 & \text{for all } t \in T, f \in F \\
& y_f, x_{tf} \in \{0, 1\} & \text{for all } t \in T, f \in F.
\end{array} \tag{4.2}$$

Even for this simple version the integer program is not a CIP, as there are negative coefficients in the constraints. We construct an investment game based on an instance as follows. Each of the players holds a set $T_p \subseteq T$ of terminals. The set of resources R is composed of all facilities F and all possible edges $T \times F$. For ease of notation, we split a strategy for player p into two components, capturing the offers to the connection and opening costs, respectively. In particular, $s_p = (s_p^c, s_p^o)$, in which $s_p^c : T \times F \to \mathbb{R}_{\geq 0}$ and $s_p^o : F \to \mathbb{R}_{\geq 0}$. For the variables we again suppose that $y_f = \mathtt{bought}_f(s)$ and $x_{tf} = \mathtt{bought}_{(t,f)}(s)$ using a similar split for function $\mathtt{bought}$ to refer to facilities and connections. Hence, if the total offers of all players exceed the cost of a connection or facility, the corresponding variable is raised to 1. In this case the connection or facility is considered bought or opened, respectively. Finally, for each player p the constraint $\mathtt{valid}_p(s)$ is set true if all the inequality constraints corresponding to any terminal $t \in T_p$ are satisfied. Each player thus insists on having all her terminals connected by a bought connection to at least one opened facility.

Definition 4.3 *A* UFL game *is an investment game in which resources and constraints are given as follows.*

- *The resource set is given by* $R = F \cup T \times F$, *in which* F *is a set of facilities and* T *a set of terminals*
- *Each player* p *has a terminal set* $T_p \subseteq T$. *For the constraint* $\mathtt{bought}(s) \in \mathtt{valid}_p$ *if and only if* $\sum_{f \in F} \mathtt{bought}_{(t,f)}(s) \geq 1$ *and* $\mathtt{bought}_f(s) - \mathtt{bought}_{(t,f)}(s) \geq 0$ *for all* $t \in T_p$ *and* $f \in F$.

In a metric UFL game *the connection costs satisfy the triangle inequality.*

For a succinct representation it is sufficient to encode the sets of facilities F and terminals T, the cost function c and the terminal sets T_p of the players.

It is possible to apply some of the simplifications observed Section 4.2. For the remainder of this section we assume that the sets T_p form a partition of the terminal set T. In particular, this means that in a NE connection costs are not shared between players. Note that the facility location game can be cast as a connection game studied in [ADTW03] and Chapter 5 in the following way. Consider the complete bipartite graph given by facilities and terminals and all edges directed from terminals to facilities. Then introduce a new source vertex s and introduce a directed edge from each facility f to s. The cost of (f, s) is the opening cost of $c(f)$. In this connection game each player strives to connect all her clients to s with bought edges at the minimum cost. In this way facility location games are a special case of directed single-source connection games [ADTW03]. Note that if we were building the connection game with undirected edges, the corresponding facility location game would allow players to connect to facilities using other players connections.

4.6.2 Exact and Approximate Nash Equilibria

This subsection presents results on exact and approximate NE for the metric UFL game. Lower bound constructions are mainly derived by using the following transformation to turn a vertex cover game with graph $G = (V, E)$ into a metric UFL game. The set of facilities F is given by the vertex set V of the graph G. For the opening costs $c(f) = c(v)$. The terminal set T is given by the edge set E. For each terminal t corresponding to $(u, v) \in E$ we specify the connection costs for edges between t and the two facilities corresponding to u and v. These edges are termed *basic* edges. All other edge costs are given by the shortest path metric over basic edges.

Even in the metric UFL game the price of anarchy is exactly k. The lower bound is derived by an instance with two facilities, f_1 with cost k and f_2 with cost 1. Each player p has one terminal t_p, and all connection costs are 0. The argumentation follows Theorem 4.2. The upper bound of k is also easily translated to metric and non-metric UFL games. To derive a bound on the price of stability, we note that there are games without NE.

Lemma 4.3 *There is a metric UFL game without NE.*

Proof. The proof follows by translating the game of Figure 4.1(a) into a metric UFL game. We set the cost of vertex u to 1.5 and the cost of each basic edge to 1. In NE no player will consider to pay a connection cost of 3 to connect a terminal to a facility because it is always possible to open another facility and connect the terminal with a total cost of less than 3. Hence, in NE only basic edges are bought and the total connection cost is 3. Then the opened facilities identify a feasible vertex cover for the original instance. This proves the lemma. $\square$

We can use this game to make the price of stability as large as $k - 2$.

Theorem 4.14 *The price of stability in the metric UFL game is at least* $k - 2$.

Proof. Consider the game in Figure 4.8. This game is in essence obtained by transformation from the game in Figure 4.2. In addition to the transformation there are two major adjustments. First, instead of the unweighted game in Figure 4.1(b) we attach the game in Figure 4.1(a) to the star of players $3, \ldots, k$. Second, after the transformation we must adjust opening and connection costs to ensure the property that in NE no non-basic edges are purchased. It is easy to verify that for the presented game this property holds. The argumentation then follows the proof of Theorem 4.3. In particular, suppose there is a NE in which the facility in the center of the star composed by players $3, \ldots, k$ is bought. In this case players 1 and 2 must agree on opening some of the cheap facilities. This is not possible due to Lemma 4.3. If, however, the center facility of the star is not bought, each of the players $4, \ldots, k$ pays for the connection and opening costs of the corresponding leaf facility resulting in a total cost of $1 + \epsilon$ for each player. Player 3 can connect to the cheap facility and contribute a cost of ϵ to the opening cost. Then player 1 can contribute the remaining cost of $\epsilon/2$ and connect her two terminals at a cost of 2ϵ. Player 2 purchases one of the facilities of cost ϵ and the connection to it. This yields a NE of cost $(k - 2)(1 + \epsilon) + 4.5\epsilon$. The social optimum solution has cost $1 + (k + 3)\epsilon$.

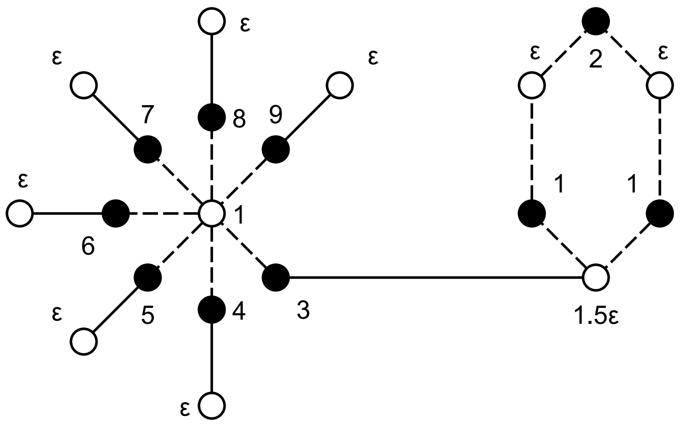


Figure 4.8: A metric UFL game with $k = 9$, for which the price of stability is arbitrarily close to $k-2$. Filled vertices are terminals, empty vertices are facilities. Numbering of terminals indicates player ownership, labels of facilities indicate opening costs. All basic solid edges have cost 1, all basic dashed edges cost ϵ. All other connection costs are given by the shortest path metric.

Thus, if ϵ tends to 0, the lower bound becomes arbitrarily close to $k-2$. □

The next theorem sheds light on the hardness of deciding NE existence.

Theorem 4.15 *It is* NP*-hard to decide whether a metric UFL game has a NE.*

Proof. The theorem follows directly by modification of the transformation for the vertex cover game. If we transform the variable and clause gadgets of Figure 4.3 using a cost of 1 for each basic edge, then every non-basic edge has cost at least 3. Thus, in NE no non-basic edge is purchased, as a player can always open another facility and connect a terminal with a basic edge with smaller cost. The set of opened facilities in NE identifies a feasible vertex cover for the original gadget. □

Thus, NE can be quite costly and hard to compute. For some classes of games, however, there is a cheap NE. In particular, as outlined in the introduction, results on the connection game [ADTW03] can be used to show that singleton UFL games (with a single terminal per player) allow for an iterative improvement procedure similar to the one presented for singleton games in Section 4.5. Hence, the price of stability is 1, and for metric games a $(1+\epsilon, 1.52)$-approximate NE can be found using a recent 1.52-approximation algorithm [MYZ02] to compute a starting solution. We show here that for games with integrality gap 1 optimal NE exist and can be computed efficiently.

Theorem 4.16 *For any metric UFL game, in which the underlying UFL problem has integrality gap 1, the price of stability is 1, and an optimal NE can be computed in polynomial time.*

Proof. The proof works by adjusting the proof of Theorem 4.10. Reconsider the IP formulation (4.2) and its corresponding LP-relaxation obtained by relaxing $y_f, x_{tf} \geq 0$. The integrality gap is assumed to be 1, so the optimum solution (x^*, y^*) to (4.2) is also optimal for the relaxation. The LP-dual is

$$\begin{array}{lll} \text{Min} & \sum_{t \in T} \gamma_t & \\ \text{subject to} & \gamma_t - \delta_{tf} \leq c(t,f) & \text{for all } t \in T, f \in F \\ & \sum_{t \in T} \delta_{tf} \leq c(f) & \text{for all } f \in F \\ & \gamma_t, \delta_{tf} \geq 0 & \text{for all } t \in T, f \in F. \end{array} \tag{4.3}$$

We can find the optimum dual solution (γ^*, δ^*) in polynomial time. It has the same value as (x^*, y^*) for (4.2). Now assign each player to pay $s_p^o(f) = y_f^* \left(\sum_{t \in T_p} \delta_{tf}^*\right)$ to the opening cost of each facility f. By complementary slackness this assignment purchases every facility exactly. In addition, if a terminal t is connected to facility f then $x_{tf}^* = 1$ and $\gamma_t - \delta_{tf} \leq c(t,f)$ is tight. As x^* is integral, for each terminal $x_{tf}^* = 1$ for exactly one opened facility, and we can let the owning player p contribute $s_p^c(t,f) = x_{tf}^*(\gamma_t^* - \delta_{tf}^*)$ to $c(t,f)$ for every facility f. This assignment exactly purchases each facility of $\mathcal{F}^*$.

To show that the assignment creates a set of best responses, we again consider the situation for a single player by removing terminals of other players along with the dual payments. Suppose we remove a terminal with all constraints in which appears. In addition, we remove the contribution of these constraints from the costs of resources. For a terminal t there are at most two dual variables greater than 0: δ_{tf}^* for t connected to f and γ_t^*. Hence, the reduced problem with the cost function

$$c_p(f) = c(f) - \sum_{t \notin T_p} x_{tf}^* \delta_{tf}^* \quad \text{and} \quad c_p(t,f) = 0 \quad \text{for all } f \in F, t \notin T_p,$$

and $c_p = c$ otherwise captures the problem of finding a best response for player p. (x^*, y^*) still gives a solution to this reduced problem. Furthermore, the remaining dual variables give a feasible solution to the LP-dual of the reduced problem. LP-duality shows that (x^*, y^*) is still optimal. This proves that the payment functions are best responses and form a NE. □

For general games we consider approximate NE.

Theorem 4.17 *For the metric UFL game there is an algorithm to derive* $(3,3)$*-approximate NE in polynomial time.*

In Algorithm 3 we denote a terminal by t, a facility by f, and the player owning t by p_t. The algorithm raises budgets for each terminal, which are offered for purchasing the connection and opening costs. Facilities are opened if the opening costs are covered by the total budget offered, and if they are located sufficiently far away from other opened facilities.

Algorithm 3: Primal-dual algorithm for (3,3)-approximate NE

In the beginning all terminals are unconnected, all budgets B_t are 0, and all facilities closed. Raise budgets of *unconnected* terminals at the same rate until one of the following events occurs. B denotes the current budget of unconnected terminals. A terminal t is *tight* with facility f if $B_t \geq c(t,f)$.

1. An unconnected terminal t goes tight with an opened facility f. In this case set t *connected* to f and assign player p_t to pay $s^c_{p_t}(t,f) \leftarrow c(t,f)$.
2. For a facility f not yet definitely closed the sum of the budgets of unconnected and indirectly connected terminals t pays for opening and connection costs: $\sum_t \max(B_t - c(t,f), 0) = c(f)$. Then stop raising the budgets of the unconnected tight terminals. Also,
   (a) if there are opened facility f' and terminal t' with $c(t',f) + c(t',f') \leq 2B$, set f *definitely closed* and all unconnected terminals t tight with f *indirectly connected.*
   (b) Otherwise open f and set all terminals *directly connected* to f, which are tight with f and not yet directly connected to some other facility. For each such terminal assign player p_t to pay $s^c_{p_t}(t,f) \leftarrow c(t,f)$ and $s^o_{p_t}(f) \leftarrow B_t - c(t,f)$.

In the end connect all indirectly connected terminals to the closest opened facility and assign the corresponding players to pay for the connection cost.

For the approximation ratio of 3 we note that the algorithm is a primal-dual method for metric Uncapacitated Facility Location [MP03, PT03]. For the analysis of the stability ratio consider a single player p and her payments. Note that the algorithm stops raising the budget of a terminal by the time it becomes directly or indirectly connected. First, we show that for the final budgets $\sum_{t \in T_p} B_t$ is a lower bound on the cost of any deviation for player p. For any terminal t we denote by $f(t)$ the facility t is connected to in the calculated solution.

Lemma 4.4 $c(t,f) \geq B_t$ *for any terminal* t *and any opened facility* $f \neq f(t)$.

Proof. Suppose there is such a facility for a terminal that is indirectly connected at the end of the algorithm. This is a contradiction, because then the terminal would have been tight to an opened facility during the run of the algorithm. If this happens, t gets directly connected to f. Otherwise suppose t is directly connected to $f(t)$. Then, f and $f(t)$ are within a distance of $2B_t$, which is too close for both of them to be open. As t is directly connected to $f(t)$, either $f(t)$ or both f and $f(t)$ are opened at a time when the current budget $B \geq B_t$. If f is opened first and the algorithm tries to open $f(t)$, then with t there is a terminal $c(t, f) + c(t, f(t)) = 2B_t \leq 2B$. Thus, $f(t)$ must stay closed. Otherwise, if the algorithm tries to open f after $f(t)$, then f must be closed for the same reason. □

Proof. [of Theorem 4.17] Lemma 4.4 shows that if a player has a deviation that improves upon B_t, it must open a new facility and connect some of her terminals to it. By opening a new facility, however, the player is completely independent of the cost contributions of other players. Using the argumentation of [PT03] the final budgets yield a feasible solution to the dual of the LP-relaxation. In the cooperative game they form a 3-approximately budget balanced core solution [JV01]. Now suppose there is a deviation for a player that opens a new facility f, connects a subset of her terminals T_f to f, and reduces her cost below the sum of the budgets, i.e. $c(f) + \sum_{t \in T_f} c(t, f) < \sum_{t \in T_f} B_t$. This, however, would mean that the coalition formed by T_f in the cooperative game can improve upon their budgets, which is a contradiction to B_t having the core property. Hence, we know that $\sum_{t \in T_p} B_t$ is a lower bound on every deviation cost. Finally, note that for every directly connected terminal $t \in T_p$ player p pays B_t. A terminal t becomes indirectly connected only if it is unconnected and tight to a facility f by the time f is definitely closed. Facility f becomes definitely closed only if there is another previously opened facility f' at distance $2B_t$ from f. Hence, there is an edge $c(t, f') \leq 3B_t$ by the triangle inequality. So in the end player p pays at most $3B_t$ when connecting an indirectly connected terminal to the closest opened facility. This establishes the bound on the stability ratio. □

Our proof uses results from cooperative games to lower bound the deviation possibilities of a player p that are not influenced by the contribution of other players than p. Note that it is possible to derive a self-contained proof as for Algorithm 1 before.

Finally, we discuss some observations regarding lower bounds on the stability ratio. There is no polynomial time algorithm for metric Uncapacitated Facility Location with an approximation ratio of 1.463 unless $\mathsf{NP} \subset \mathsf{DTIME}(n^{O(\log \log n)})$ [GK99]. This transfers to a lower bound for the stability ratio in terms of polynomial time computability. In addition, there is a game giving a constant lower bound in terms of existence. Reconsider the UFL game obtained from transforming the vertex cover game of Figure 4.1(a). The structure of the graph is fixed as well as all connection costs. Therefore, as there is no NE, any feasible solution can represent only an (α, β)-approximate NE with $\alpha > 1$. By appropriate adjustment of edge costs one can obtain a small bound of $\alpha > 1.097$ [Hoe06a].

4.6.3 Connection-Restricted Facility Location Games

This section presents an extension based on CONNECTION-RESTRICTED FACILITY LOCATION problems as considered in [GS04]. The problem is again to open a set of facilities and connect terminals to them at minimum total cost. Instead of the constraints $y_f - x_{tf} \geq 0$, which guarantee that terminals can only be connected to opened facilities, the connection restriction is generalized. In particular, for each facility f there is a set $\mathcal{A}_f$ of feasible subsets of terminals that can be connected simultaneously to f. The set of terminals that connect simultaneously to f must be included in $\mathcal{A}_f$. This formulation allows for instance capacity, quota, or incompatibility constraints and thus encompasses several well-known generalizations of the problem. In these cases the connection requirements are still expressable by a polynomial number of linear inequalities, which is not true for general CONNECTION-RESTRICTED FACILITY LOCATION. Nevertheless, for investment games based on this problem (denoted as CRFL games) some of the previous results can be extended to hold. In particular, lower bounds on the prices of anarchy and stability follow simply by extension. The upper bound of k on the price of anarchy is also valid. In the following we show that for the previously outlined subclasses of games cheap NE exist in the CRFL game.

Theorem 4.18 *For any CRFL game, in which a partially conic relaxation of the underlying CRFL problem has integrality gap 1, the price of stability is 1.*

Proof. We can formalize CONNECTION-RESTRICTED FACILITY LOCATION by an integer program as follows:

$$\begin{array}{lll}
\text{Min} & \sum_{f\in F} c(f)y_f + \sum_{t\in T} c(t,f)x_{tf} & \\
\text{subject to} & \sum_{f\in F} x_{tf} \geq 1 & \text{for all } t \in T \\
 & (y_f, x_{1f}, \ldots, x_{n_t f}) \in \mathcal{A}_f & \text{for all } f \in F \\
 & y_f, x_{tf} \in \{0,1\} & \text{for all } t \in T, f \in F.
\end{array}$$

Here $\mathcal{A}_f = \{(0,\ldots,0)\} \cup \{(1, \chi_{A_f}) \mid A_f \subseteq T \text{ feasible for } f\} \subseteq \{0,1\}^{n_t+1}$, and χ_{A_f} denotes the characteristic vector of the subset A_f. Following the argumentation in [GS04] it is possible to use the conic hull of the sets $\mathcal{A}_f$ to derive a linear relaxation:

$$\begin{array}{lll}
\text{Min} & \sum_{f\in F} c(f)y_f + \sum_{t\in T} c(t,f)x_{tf} & \\
\text{subject to} & \sum_{f\in F} x_{tf} \geq 1 & \text{for all } t \in T \\
 & (y_f, x_{1f}, \ldots, x_{n_t f}) \in \text{cone}(\mathcal{A}_f) & \text{for all } f \in F.
\end{array}$$

For this program a dual can be given by

$$\begin{array}{lll} \text{Max} & \sum_{t \in T} \gamma_t & \\ \text{subject to} & \sum_{t \in T'} \gamma_t \leq c(f) + \sum_{t \in A_f} c(t,f) & \text{for } f \in F \text{ and } A_f \in \mathcal{A}_f. \end{array}$$

Now we can apply the same arguments given before in Theorems 4.10 and 4.16. An integral optimum solution (x^*, y^*) to the LP-relaxation represents a partition of the terminal set T into a collection of feasible sets A_f^*, one for each facility f. The constraints corresponding to these sets hold with tightness, and we can assign each player p to pay for each of her terminals t the amount $s_p^c(t,f) = c(t,f)$ as connection cost to f with t connected to f. For the opening costs $s_p^o(f) = \sum_{t \in A_f^* \cap T_p} \gamma_t^* - c(t,f)$, in which γ^* is the optimum solution to the dual. In total this pays exactly for all costs of the solution by duality. Suppose now a player can connect her terminals differently at a cheaper cost. This means that she can reduce at least one of the costs γ_t corresponding to a terminal t by choosing at least one other set $A_{f'}$ for some facility f'. The new reduced payments would exactly pay for $c(f') + \sum_{t \in A_{f'}} c(t,f')$. Hence, the original payments coming from γ_t^* would violate the dual constraint, as they are strictly greater. This is a contradiction to γ^* being the optimal dual solution. $\square$

Note that in case one can optimize over the cones in polynomial time we can also compute the best NE in polynomial time. In particular, this is the case if the cone can be described by a polynomial number of linear inequalities.

Theorem 4.19 *For any CRFL game with a single terminal per player the price of stability is 1.*

Proof. Consider a solution $\mathcal{F}$ and an opened facility f. Now consider the set of terminals T_f that are connected to f in $\mathcal{F}$. We apply the exchange arguments outlined in Theorem 4.11. For each $t \in T_f$ consider the owning player independently and determine her cheapest strategy to connect her terminal to a different (possibly to be opened) facility than f. Naturally, the player considers only feasible strategies that create feasible sets of clients at the facility she is deviating to. Let the cost of this cheapest deviation be c_t^f. A facility is *stabilized* if $c(f) + \sum_{t \in T_f} c(t,f) \leq \sum_{t \in T_f} c_t^f$. In this case all connection and opening cost for f and the connecting terminals T_f can be purchased such that no player can improve by some other feasible strategy. Naturally, this could involve a single player paying for all connection and opening costs for all terminals connected to f, and this is also true for the cooperative game. A solution $\mathcal{F}$, in which all facilities are stabilized, is called *exchange minimal.* Obviously an exchange minimal solution is a NE. Furthermore, with iterative cost improvement steps we can in finite time converge to a NE. In particular, as $\mathcal{F}^*$

cannot be improved in this way, all facilities are stabilized, and it can be purchased in a NE. □

Adjusting the improvement steps to obtain approximate equilibria in polynomial time (c.f. Theorem 4.11) can be possible. Suppose we are given an oracle that determines in polynomial time for a player p and a facility f if p can be combined with the players currently connected to f into a set T' feasible for f. Then, the cost difference resulting from an exchange is easily calculated, and this allows us to determine optimum player deviations and unstabilized facilities in polynomial time. If the underlying CRFL problem furthermore allows an efficient β-approximation algorithm, we can use the output as a starting solution. Then, by scaling the costs of facilities, a guaranteed minimum cost reduction results in a polynomial number of iterations, and $(1 + \epsilon, \beta)$-approximate NE can be determined by the previously described local search algorithm polynomial time.

Corollary 4.3 *If the underlying CRFL problem allows a polynomial approximation algorithm with a ratio β, and if there is a polynomial time oracle to answer for each set A_f and each facility f whether $A_f \in \mathcal{A}_f$, then there is an algorithm to compute $(1 + \epsilon, \beta)$-approximate NE in polynomial time.*

The described oracle exists if we can describe the sets $\mathcal{A}_f$ by a polynomial number of linear inequalities. This is true for a large number of interesting and well-known cases of the game including e.g. quota or incompatibility constraints. For example the capacitated version can be stated with constraints of the form $k(f)y_f - x_{tf} \geq 0$, in which $k(f)$ is the maximum number of terminals (and players, as each player has a single terminal) that are allowed to connect simultaneously to facility f.

4.7 Discussion

This section clarifies the relations to the work on cooperative covering and facility location games. There is a strong connection between LP-duality and the core of a cooperative game. In particular, Deng et al. [DIN97] showed that the core of cooperative integer covering games is non-empty if and only if the integrality gap of the underlying problem is 1. Goemans and Skutella [GS04] showed a similar result for connection-restricted facility location games. The main arguments rely on LP-duality. In fact, they are reused and extended in the proofs of our Theorems 4.10 and 4.18. Given the results for the core, one could be tempted to think Theorems 4.10 and 4.18 result as immediate corollaries, because stability analysis for NE in our games needs to avoid cheap deviations only for a subset of the coalitions that are considered for the core – coalitions represented by single players. In this sense the NE is a weaker concept than the core. But our theorems are not at all corollaries, as in our game the investments of a player alter the cost of optimal solutions for other players. This feature leads to a different analysis for solution concepts. In our case overcoverage becomes a central problem that needs to be resolved in a way that provides cheap solutions with low incentives to deviate. The proofs employ the fact that if the integrality gap is 1, overcovered players do not contribute. The remaining analysis is then similar to the cooperative case. This actually shows that Theorems 4.10 and 4.18 extend to hold for coalition-proof NE in our game, in which every player coalition is allowed to reallocate payments.

Avoiding uncontrolled overcoverage is also the main idea of applying primal-dual algorithms to get approximate NE. Some of the arguments in the proof of Theorem 4.7 can be found in the context of cooperative games [JV01, PT03]. A proof using these arguments is given for Theorem 4.17: at first show that the overcovering problem is resolved in a feasible way; then the cost allocation part is taken care of, and the remaining analysis can be assembled from cooperative games.

Note that the way primal-dual algorithms cope with overcovering in set cover and UFL games is crucial to get the same bound for the stability ratio as for the approximation ratio. We have tested other primal-dual algorithms e.g. the primal-dual 2-approximation algorithm for Steiner Forest [AKR95, GW95] in the context of the connection game, see Chapter 5. In this case the underlying integer program is not an Integer Covering Problem, and in fact it is possible to observe that the stability ratio for the calculated payments can be strictly higher than 2. For the covering problems considered in this paper there are usually greedy algorithms that yield best approximation ratios, thus it would be appealing if their performance translates to the stability ratio as well. In the cooperative setting these algorithms yield the same factor for competitiveness, the analogous notion in cooperative games as our stability ratio. The main reason is that the method of dual fitting [Vaz00], which is used to analyze approximation ratios, can also be used for bounds on competitiveness (or equivalently budget balance) for solutions in an approximate core of cooperative games. In our non-cooperative case these algorithms might not handle overcovering well and introduce unbounded incentives to drop contributions. Thus, for recently proposed greedy methods the approximation ratio does not translate to the stability ratio.

Lemma 4.5 *The payments computed by the greedy* $O(\log |\mathcal{M}|)$*-approximation algorithm for* Set Cover *yield an unbounded stability ratio. The payments computed by recent greedy* $O(1)$*-approximation algorithms for* Uncapacitated Facility Location *[JMM+03, MYZ02] yield a stability ratio of* $\Omega(k)$.

Proof. For the first part consider a vertex cover game with a path of 4 vertices: $e_1 = \{v_1, v_2\}$, $e_2 = \{v_2, v_3\}$, and $e_3 = \{v_3, v_4\}$. The inner vertices have small cost $c(v_1) = c(v_2) = 1$, the other ones large cost $c(v_0) = c(v_3) = 10$. There are three players with $E_p = \{e_p\}$. Assume w.l.o.g. greedy picks v_1 first, then players 1 and 2

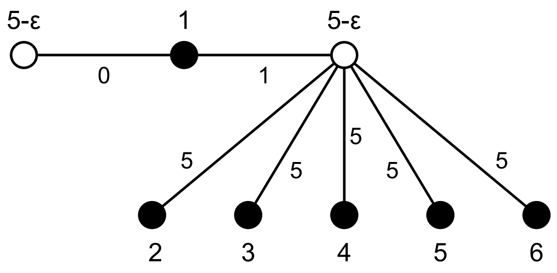


Figure 4.9: An example game yielding a high stability ratio for approximate NE using greedy algorithms. Facilities are empty vertices, labels indicate opening cost. Terminals are filled vertices, labels indicate player ownership. Edge labels indicate connection costs. All other connection costs are given by the shortest path metric.

are assigned to pay 0.5 to v_1. In the next iteration v_2 is picked, and player 3 must purchase it all by herself. Then player 2 has an unbounded stability ratio.

For the second part consider one of the recent greedy methods presented in [JMM+03]. These methods raise the budgets of each player simultaneously. If a facility is paid for, it is opened and contributing terminals are connected. A connected terminal t is possibly reconnected in later iterations if a facility is opened to which t has a smaller connection cost.

Consider a game outlined in Figure 4.9. There are two facilities, both have opening cost $k-1-\epsilon$. Player 1 can connect with a cost of 0 to the left facility and with a cost of 1 to the right one. All other players can connect with cost k to the left facility and cost $k-1$ to the right one. Using a dual ascending greedy algorithm the first facility that is opened is the left one at a time the budgets are $k-1-\epsilon$, and the cost of $k-1-\epsilon$ is paid completely by player 1. Later, when budgets are $k-\frac{\epsilon}{k-1}$, the right facility is also opened, and the costs are purchased by the other players. Now player 1 can reduce her costs by a factor of $k-1-\epsilon$ by dropping the contributions to the left facility and purchasing the connection to the right one. The stability ratio becomes at least $k-1-\epsilon$.

Recent variants of greedy algorithms use different techniques, e.g. the method of [MYZ02] combines scaled opening costs with a greedy method of [JMM+03] and a final greedy opening routine. With a moment of thought, however, one can verify that our argumentation can be adjusted to hold for all these variants. □

Chapter 5

Connection Games

This chapter studies investment games in the setting, in which they were proposed by Anshelevich et al. [ADTW03]. The *connection game* for k players can be described as follows. For each game there is an undirected graph $G = (V, E)$, and a nonnegative cost $c(e)$ associated with each edge $e \in E$. The set of resources $R = E$ is the set of edges of the graph. If $\sum_{p\in[k]} s_p(e) \geq c(e)$ for an edge e, it is considered bought. Each player wants to connect a subset of vertices of the graph by a network of bought edges with the least possible investment cost. We start by formulating the underlying combinatorial optimization problem. The Steiner Forest problem [Vaz00, GW95, AKR95] is defined as follows.

Problem 5.1 (Steiner Forest) *Suppose we are given a graph* $G = (V, E)$*, nonnegative edge costs* $c(e) \geq 0$ *for each* $e \in E$*, and a subset of vertices* $T \subseteq V$ *called the* terminals*. Furthermore, there is an equivalence relation on* T *expressed by the connection requirement function* $\mathsf{connect} : T \times T \to \{true,false\}$*. The problem is to find a forest* $\mathcal{T} \subseteq E$ *such that for each pair* (t, t') *with* $\mathsf{connect}(t, t') = true$ *there is a path between* t *and* t' *in* $\mathcal{T}$ *and the total cost* $\sum_{e\in T} c(e)$ *is minimal.*

Function $\mathsf{connect}$ naturally extends to every subset $V' \subseteq V$ of vertices to indicate if a solution must contain an edge from the cut $(V', V - V')$ in the graph. A special case of Steiner Forest is Steiner Tree, in which the function $\mathsf{connect}$ is true for every pair of terminals. Steiner Tree was one of the first problems shown to be NP-complete [Kar72]. It cannot be approximated by a factor of $\frac{96}{95}$ unless $\mathsf{P} \hat{=} \mathsf{NP}$ [CC02], and this extends to Steiner Forest. We can formally define the *connection game* as follows.

Definition 5.1 *A* connection game *is an investment game, in which resources and constraints are given as follows.*

- *The resource set* $R = E$ *is the edge set of a simple undirected graph* $G = (V, E)$*.*

- *Each player* p *has a set of terminals* $T_p \subseteq T \subseteq V$*. For the constraint* $\mathtt{bought}(s) \in \mathtt{valid}_p$ *if and only if there is a subnetwork* $\mathcal{T}_p \subseteq E$ *connecting the vertices* T_p *and* $\mathtt{bought}_e(s) \geq 1$ *for all* $e \in T_p$*.*

For a succinct representation it is sufficient to encode the graph G, the cost function c, and the terminal sets T_p of the players. As in the previous chapter we could formulate the connectivity requirement in the game definition using the constraints of an integer program for Steiner Forest [GW95], which involves the cuts of the graph and the function $\mathtt{connect}$. We then can apply similar simplifications as in the previous chapter, i.e. the function $\mathtt{connect}$ is assumed to encompass exactly the connection requirements of the players, and the set of terminals T is exactly the union of the player sets T_p. For this game a social optimum solution solves the underlying instance of Steiner Forest. Finding an optimum strategy for a single player poses an instance of Steiner Tree. In fact, there are several integer programming formulations for Steiner Tree and Steiner Forest [Pol03], each of them models the connection requirement in a different way. Primal-dual algorithms based on these programs are not directly applicable to obtain cheap approximate NE as it was the case for covering games. So we rather use combinatorial approaches to obtain cheap NE. For the rest of this chapter $\mathcal{T}^*$ denotes a social optimum forest. The subtree of $\mathcal{T}^*$ that player p uses to connect her terminals is denoted by $\mathcal{T}_p$.

5.1 Tree Connection Games

In this chapter we deal with the class of *tree connection games (TCG)*, which are games with tree connection requirements.

Definition 5.2 *In a connection game there are* tree connection requirements *if for any two terminals* $t_1, t_{j+1} \in T$ *there is a sequence of players* $p_1, \dots, p_j$ *and terminals* $t_2, \dots t_j$ *such that player* p_i *wants to connect the terminals* $t_i, t_{i+1} \in T_{p_i}$ *for* $i = 1, \dots, j$*.*

Note that tree connection requirements require every feasible solution network to be connected. A TCG can be thought of as a splitting of a single global player into k players, which preserves the overall connection requirements. For the subclass of TCGs with $|T_p| = 2$ we use the term path tree connection game (PTCG). A different subclass of the TCG are single source connection games (SSG) [ADTW03], in which there is one source terminal s with $s \in T_p$ for all players p. SSGs are closely related to facility location games as was discussed in Section 4.6.1.

Basic properties for investment games outlined in Chapter 3 yield that for a NE of the (tree) connection game the network of bought edges is a (tree) forest, the total contribution to each edge e is either $c(e)$ or 0, and each player contributes only to the subtree of bought edges that she needs to connect her terminals. Rather than using LP-duality the algorithms in this chapter rely on a player-based assignment technique. An outline is given in the following framework. The input is a feasible tree satisfying the connection requirements of all players. In each iteration it picks a player, assigns payments, removes the player, and reduces the edge costs by the amount she paid. The framework terminates if there is no player left. The exact assignment of payments is specified in several different ways in later sections of this chapter. As candidates for this elimination process we consider leaf players.

Definition 5.3 *A terminal* t *of a player* p *is a* lonely terminal *if* $t \in T_p$ *and* $t \notin T_q$ *for any* $q \neq p$*. A player* p *is a* leaf player *if she owns at least one lonely terminal and at most one non-lonely terminal.*

In a PTCG with at least two players each leaf player has exactly one lonely terminal. In this case we will use t_p to denote the lonely terminal of leaf player p.

Algorithm 4: Algorithmic Framework

```
  Input: A feasible tree $\mathcal{T}$
  Output: Strategies $s_1, \ldots, s_k$ distributing the cost of $\mathcal{T}$
1 $c^1(e) = c(e)$ for all $e \in E$
2 for $iter \leftarrow 1$ to $k$ do
3     $p$ is a leaf player if possible; otherwise an arbitrary player
4     Call a procedure, which either determines $s_p$ or improves $\mathcal{T}$
5     if procedure returns a new tree then
6         exit and restart the framework with new tree
7     Set $c^{iter+1}(e) \leftarrow c^{iter}(e) - s_p(e)$ for all $e \in E$
8     Remove $p$, contract edges $e$ of cost $c^{iter+1}(e) = 0$
```

For an intuitive understanding of this framework and the notion of a leaf player consider a PTCG with two terminals per player. Construct a *connection requirement graph* G_{crg} as follows. The vertex set of G_{crg} is T. The edge set is created by introducing a single edge for each player between her terminals. Hence, in G_{crg} each player is associated with an edge. If we run the framework, each player will be removed, some players under the label "leaf players", and some as "arbitrary players". The players picked as arbitrary players during the run of the framework compose a set of edges, which breaks every cycle in G_{crg}. They will be removed without getting payments assigned. A leaf player in an iteration corresponds to an edge incident to a "leaf" vertex of degree 1. In the special case of a PTCG the framework picks players in an ordering that is similar to an inverse BFS ordering in G_{crg}.

The rest of this chapter is organized as follows. In the next Section 5.2 we outline how TCGs connect to related work on network creation games. In Section 5.3 we study exact NE and show that every PTCG has an optimal NE, hence the price of stability is 1 (Theorem 5.2). For three or more terminals per player, however, the price of stability is at least $k - 2$ (Theorem 5.3) and it is NP-hard to decide NE existence (Theorem 5.4). Section 5.4 describes an algorithm to obtain $(2, 1)$-approximate NE for TCGs (Theorem 5.6). The ideas can be used to get $(2 + \epsilon, 1.55)$- and $(3.1 + \epsilon, 1.55)$-approximate NE in polynomial time for PTCGs and TCGs, respectively, and any constant $\epsilon > 0$. A discussion of the proof techniques is given in Section 5.5. Section 5.5.1 reveals that the analysis for our algorithm for approximate NE is tight (Theorem 5.9), and our algorithm is shown to improve upon a previously proposed method (Theorem 5.10). Also, there cannot be any algorithm with better performance based only on the tools we used for design and analysis in this thesis. Finally, Section 5.5.2 discusses the possibility to use our techniques to analyze more complicated network design games. For a *backbone game* we show that in a special case the price of stability is 1 (Theorem 5.11) and approximate NE with stability ratio $(1 + \epsilon)$ can be obtained in polynomial time, for any constant $\epsilon > 0$ (Theorem 5.12).

5.2 Previous and Related Work

The connection game was introduced and studied by Anshelevich et al. [ADTW03]. Both prices of anarchy and stability are $\Theta(k)$. It is NP-hard to determine if a given game has a NE. There is a polynomial time algorithm that finds a $(4.65 + \epsilon, 2)$-approximate NE, for any constant $\epsilon > 0$. For SSGs, in which each player needs to connect a single terminal to the source, a polynomial time algorithm finds a $(1 + \epsilon, 1.55)$-approximate NE, for any constant $\epsilon > 0$. We denote these algorithms by ADTW-SS for the single source and ADTW for the general case. More recently, we have studied connection games in a geometric setting. In [HK05] we showed bounds on the price of anarchy and the minimum incentives to deviate from an assignment purchasing the social optimum network. The case of two players and two terminals per player was characterized in terms of prices of anarchy and stability, approximate NE, and best-response dynamics.

The Shapley cost sharing variant of the investment game [ADK$^+$04] mentioned earlier in Chapter 3 was proposed in the context of directed connection games, and prices of anarchy and stability were shown to be $\Theta(k)$ and $\Theta(\log k)$, respectively. Several additional special cases have been considered, mostly related to the open problem of characterizing the price of stability for undirected networks. The price of stability for a variant of undirected minimum spanning tree creation was shown to be $O(\log \log k)$ in [FKL$^+$06]. A splittable connection game and the price of anarchy for an iterative player arrival protocol were treated in [CCLE+06].

Network creation as a cost sharing problem has been studied intensively in cooperative games and mechanism design. Cooperative games based on Steiner Tree and Minimum Spanning Tree problems have been considered for decades, see for instance [Bir76, Meg78, GH81]. In these games each terminal is a player and the cost for a coalition is the cost of a minimum spanning or minimum Steiner tree for their terminals only. The existence of core solutions in these games was considered for instance in [Meg78, SK95]. While the core can be empty, the existence of a core solution depends on the integrality gap of a corresponding integer programming formulation. However, in contrast to covering and facility location games [DIN97, GS04] an integrality gap of 1 is not necessary for the existence of core solutions. Also, in the connection game every player corresponds to a certain connection requirement and not to a terminal. A player might not be satisfied with a connection to some arbitrary terminal. Thus, there is no simple applicability of LP methods to finding (approximate) NE in TCGs.

More recently, Jain and Vazirani [JV01] studied cost sharing mechanisms for the Steiner Tree problem. In particular, they considered a multicast perspective: there is a single source and each player has a private valuation for being connected to it. The mechanism should pick a subset of terminals, pick a tree connecting them to the source, and distribute the costs to players such that certain social desiderata are met. They showed how to derive cross-monotonic cost sharing schemes, which translated into a 2-approximate budget balanced group-strategyproof cost sharing mechanism. This work was extended recently in [KLS05, GKPR07] to similar cost sharing mechanisms for variants of Steiner Forest with multiple source-sink pairs.

The TCG is related to network creation models for social networks [Jac04] mentioned in the introduction. However, all these network creation games take a social network perspective and associate each player with a vertex. In this way they are similar to cooperative games based on Minimum Spanning Tree and Steiner Tree. Players can only create incident edges and do not consider more complex connection requirements as in the connection game. Thus, properties and analyses of these games are very different from our approach here. They are more closely related to clustering games considered in Part II of this thesis, and we mention some recent directly related works in Section 8.1. Closely related to these models is a network creation game studied in [FLM+03, CP05, AEED+06, DMHZ07], in which the utility for a player is given by the costs of the links established by the player plus the sum of shortest path lengths to all other players.

5.3 Cost and Complexity of Nash Equilibria

In this section we examine cost and complexity properties of NE in TCGs. We first observe that the price of anarchy is as large as k, even in the PTCG. Consider a graph with two vertices and two parallel edges e_1 and e_2, in which each player wants to connect both vertices. Edge e_1 has cost k, e_2 cost 1. If each player is assigned to purchase a share of 1 of e_1, the state is a NE and the price of anarchy becomes k. Note that k is also an upper bound using the arguments in the proof of Theorem 4.2. In contrast, the price of stability for the PTCG is 1 as every such game has a NE that allows to distribute the cost of $\mathcal{T}^*$.

Theorem 5.2 *For any social optimum tree $\mathcal{T}^*$ in a PTCG there exists a NE in which $\mathcal{T}^*$ is exactly purchased. The price of stability in the PTCG is 1.*

Proof. We use the framework of Algorithm 4 with the optimum tree $\mathcal{T}^*$ as input to construct an optimal NE. In line 4 we use the following Procedure ExactNash. As the optimum $\mathcal{T}^*$ is provided as input, the procedure only outputs a strategy s_p. The resulting algorithm is similar to ADTW-SS [ADTW03] for SSGs.

In the description $\mathcal{T}^e$ denotes the part of $\mathcal{T}^*$ rooted at s, which is located below edge e. When assigning the cost of e to player p the procedure uses cost functions c_p^e for G with $c_p^e(e') = s_p(e')$ for $e' \in \mathcal{T}^e$, $c_p^e(e') = 0$ for $e' \in \mathcal{T}^* \backslash \mathcal{T}^e$ and $c_p^e(e') = c(e')$ otherwise. The definition of c_h^e is accordingly. Note that the creation of player h (line 9) and building her function s_h (lines 23-25) have no influence on the output and can be dropped from the procedure. The consideration of player h, however, is useful for proving correctness of the algorithm.

We must show that the algorithm composed of the framework Algorithm 4 using Procedure ExactNash yields a NE in which $\mathcal{T}^*$ is bought. As a first observation it is possible to create an equivalent game as follows. For the task of distributing the cost of $\mathcal{T}^*$ we can drop all vertices outside of $\mathcal{T}^*$ from consideration. Instead, we can consider the complete graph $G_{\mathcal{T}^*}$ on the vertices of $\mathcal{T}^*$. Edge costs c_{sp} are determined by the cost of shortest paths in G with respect to c. Note that each player considers only paths as deviations. Furthermore, as no player is assigned a contribution to an edge outside $\mathcal{T}^*$, a player has to purchase the full cost of every such edge she uses in a deviation. Hence, replacing G and c by $G_{\mathcal{T}^*}$ and c_{sp} does not alter the cost of best deviations at any point in the algorithm. Thus, in the following we assume that c satisfies the triangle inequality and that $\mathcal{T}^*$ is a minimum spanning tree of G.

Trivially, the algorithm works correctly for every PTCG with only one player. Hence, we assume as our induction hypothesis that the algorithm works correctly for every PTCG with $k-1$ players. Then consider a PTCG with k players. Our induction step is represented by the first iteration of the framework. We must show that the result of this iteration is a PTCG with $k-1$ players such that $\mathcal{T}^*$ is an

Procedure `ExactNash(`$\mathcal{T}^*, c, p$`)` - NE cost distribution of $\mathcal{T}^*$

```
Input: The social optimum tree $\mathcal{T}^*$, a cost function $c$ and a selected player $p$
Output: A strategy $s_p$ for $p$
9  Create global player $h$ accumulating all players except $p$
10 Set $s_p(e) \leftarrow s_h(e) \leftarrow 0$ for all $e \in E$
11 if $p$ has no lonely terminal then
12     return $s_p$
13 if $p$ has no non-lonely terminal then
14     Set $s_p(e) = c(e)$ for all $e \in \mathcal{T}_p$ and return $s_p$
15 Let $s$ be the vertex with the non-lonely terminal of $p$
16 for each edge $e \in \mathcal{T}^*$ in reverse BFS order from $s$ do
17     if $e$ is a bridge then
18         Assign $s_q(e) \leftarrow c(e)$ to some player $q$, for which $e \in \mathcal{T}_q$
19     else
20         if $e \in \mathcal{T}_p$ then
21             Find the cheapest path $A_p$ excluding $e$ for player $p$ under $c_p^e$
22             Assign $s_p(e) \leftarrow \min(c(e),\ c_p^e(A_p) - s_p(\mathcal{T}^e))$
23         if $e \in \mathcal{T}_h$ then
24             Find the cheapest tree $A_h$ excluding $e$ for player $h$ under $c_h^e$
25             Assign $s_h(e) \leftarrow \min(c(e) - s_p(e),\ c_h^e(A_h) - s_h(\mathcal{T}^e))$
26 return $s_p$
```

optimum solution under the resulting cost function c^2. The induction hypothesis can then be used to argue that it allows a NE cost distribution for the remaining $k-1$ players. If player p has no incentive to deviate from s_p, a NE evolves that distributes the cost of $\mathcal{T}^*$. Thus, it remains to show the following two properties for the induction step:

1. Strategy s_p assigned to player p allows her no cheaper deviation.
2. $\mathcal{T}^*$ is optimal for the remaining $k-1$ players under cost function c^2.

We attack the proof by further adjusting the game. All costs of edges in $\mathcal{T}^*$ not purchased by p must be purchased by some of the other players. Thus, we suppose that all players except p are represented by the global player h, who accumulates all terminals except those of p. Procedure ExactNash assigns costs to both players p and h. Naturally, if at the end of the procedure player h has no incentive to deviate from s_h, then property 2 is fulfilled. Hence, both properties are fulfilled if there is a NE distributing the cost of $\mathcal{T}^*$ in the game for players h and p. It remains to prove the following lemma.

Lemma 5.1 *For the reduced game with two players p and h Procedure ExactNash computes a NE in which $\mathcal{T}^*$ is purchased.*

Proof. We show that at the end of the procedure no player has a possibility to lower her contributions by changing her strategy, and that the calculated strategies yield connections of bought edges - i.e. $\mathcal{T}^*$ is fully paid for. Note that the game actually represents a single source game for players p and h with a single source vertex s. For the rest of the proof we consider $\mathcal{T}^*$ rooted at s and use terms *higher* and *lower* to refer to edges and vertices at a closer and further distance from s in $\mathcal{T}^*$, respectively.

The first argument concerns the question if a player can lower her contributions.

Lemma 5.2 *The payments computed by Procedure ExactNash allow no cheaper feasible deviation for players p and h.*

Proof. With the similarity of Procedure ExactNash to ADTW-SS the lemma follows directly from [ADTW03, Theorem 3.2] for player p. For player h, however, the argument is more complicated.

Recall that $\mathcal{T}^e$ denotes the part of $\mathcal{T}^*$ below edge e. We denote by $\mathcal{T}^u$ the part of $\mathcal{T}^*$, which is below a vertex u. Note that we assume $e \notin \mathcal{T}^e$, but $u \in \mathcal{T}^u$. Consider s_p and s_h at the end of an iteration, in which Procedure ExactNash has assigned the cost of some edge $e = (u, v)$ for which we assume u is higher than v. It is not obvious that the construction of c_h^e leads to an equilibrium strategy for player h. Consider a vertex u where multiple subtrees join. We assume that for each edge

e_j below u the contribution of h to $\mathcal{T}^{e_j} + e_j$ is small enough, i.e. $\mathcal{T}^{e_j} + e_j$ is the cheapest way under c_h^e to connect the terminals of h in $\mathcal{T}^{e_j}$ to $\mathcal{T}^* \backslash \mathcal{T}^{e_j}$. But player h owns terminals in possibly *all* subtrees $\mathcal{T}^{e_j}$, and when constructing the payments for edges of a single $\mathcal{T}^{e_j} + e_j$ the contribution to the respective other subtrees was considered to be 0. Can h pick a different, cheaper tree to connect her terminals in $\mathcal{T}^u$ to $\mathcal{T}^* \backslash \mathcal{T}^u$ that improves upon her calculated contribution? We give a negative answer to this question as follows. Assume the procedure assigned payments for each $\mathcal{T}^{e_j} + e_j$, and that u is the first vertex at which $s_h(\mathcal{T}^u)$ is not optimal for h. Cost function c_h^u with $c_h^u(e') = s_h(e')$ for $e' \in \mathcal{T}^*$ and $c_h^u(e') = c(e')$ for $e' \in E \backslash \mathcal{T}^*$ captures the optimization problem faced by player h. In fact, we will see that $\mathcal{T}^*$ is indeed the optimum network under c_h^u, hence player h sticks to her contribution.

Suppose u is the first vertex, at which player h has a deviation A_h that is cheaper than $\mathcal{T}^*$ under c_h^u (i.e. the current contribution of h to $\mathcal{T}^*$). W.l.o.g. A_h includes all edges of cost 0, in particular all edges purchased completely by p and all edges of $\mathcal{T}^*$ outside $\mathcal{T}^u$. Let $\mathcal{T}^{e_j}$ be a tree that is not completely part of A_h. Consider for each terminal t of h located in $\mathcal{T}^{e_j}$ the path from t to u in A_h. We denote this set of paths by $\mathcal{P}^{e_j}$. Let $\mathcal{P}_1^{e_j}$ be the set of subpaths from $\mathcal{P}^{e_j}$ containing for every $\mathcal{P} \in \mathcal{P}^{e_j}$ the first part between the terminal of h and the first vertex $w \notin \mathcal{T}^{e_j}$. This vertex always exists because $u \notin \mathcal{T}^{e_j}$. It is in $\mathcal{T}^*$ because in our adjusted game $\mathcal{T}^*$ covers all vertices in G. The network $A_h^{e_j} = \bigcup_{\mathcal{P} \in \mathcal{P}_1^r e_j} \mathcal{P}$ was considered as a feasible deviation when constructing the payments for $\mathcal{T}^{e_j} + e_j$, as it connects every terminal in $\mathcal{T}^{e_j}$ to a vertex of $\mathcal{T}^* \backslash \mathcal{T}^{e_j}$. Furthermore, the payments of p were the same, hence the cost of $A_h^{e_j}$ was the same. Using the assumption that u is the first vertex for which $\mathcal{T}^u$ is not optimal for h, we know that $c_h^u(A_h^{e_j}) \geq c_h^u(\mathcal{T}^{e_j} + e_j)$. After substituting $A_h^{e_j}$ by $\mathcal{T}^{e_j} + e_j$ in A_h, the new network is not more costly than $\mathcal{T}^{e_j} + e_j$. To show that this new network is also feasible, suppose we iteratively remove a path $\mathcal{P} \in \mathcal{P}_1^{e_j}$. Then there might other terminals whose connections to u use parts of $\mathcal{P}$. The last vertex w of $\mathcal{P}$ is the first vertex of $\mathcal{P}$ outside of $\mathcal{T}^{e_j}$, and it stays connected to u as $\mathcal{P}$ is the first part of a path to u. All other vertices of $\mathcal{P}$ are in $\mathcal{T}^{e_j}$ and are connected by $\mathcal{T}^{e_j}$ and e_j. Hence, all terminals affected by the removal of $\mathcal{P}$ are finally reconnected to u. In this way A_h can be transformed into $\mathcal{T}^*$ without cost increase. This contradicts the assumption that A^h is cheaper than $\mathcal{T}^*$ and proves that $\mathcal{T}^*$ is optimal under c_h^u. Therefore, player h cannot lower her contribution. □

Figure 5.1 depicts the argument for player h. Vertex u has two subtrees for which the payments were assigned independently. The subtree $A_h^{e_1}$ of A_h, which is assumed to be cheaper than $\mathcal{T}^{e_1}$, is drawn in bold. A vertex w represents the first vertex outside $\mathcal{T}^{e_1}$ on a path in A_h, and it can be either completely outside $\mathcal{T}^u$ (like w_1 for t_1) or in another $\mathcal{T}^{e_j}$ (like w_2 for t_2). Replacing $A_h^{e_1}$ by $\mathcal{T}^{e_1}$ and e_1 yields a feasible network that is not more expensive.

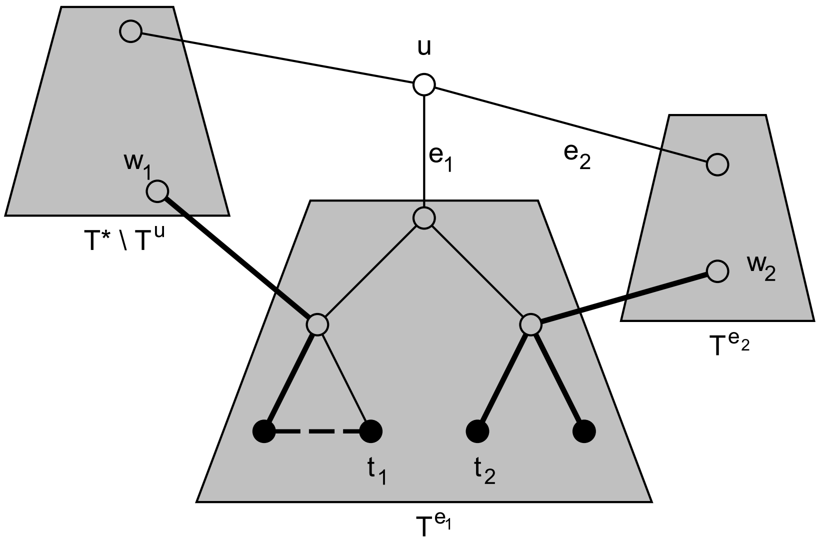


Figure 5.1: Transformation of $A_h^{e_1}$ into $\mathcal{T}^{e_1}$. Solid edges in the marked area belong to $\mathcal{T}^{e_1}$, bold edges to $A_h^{e_1}$. Replacing $A_h^{e_1}$ in A_h by $\mathcal{T}^{e_1}$ and e_1 yields a feasible network for player h that is not more expensive than A_h under c_h^u.

We have shown that the players have no incentive to move away from their assigned payments due to cost improvement. In addition, we need to show that the payments suffice to pay for the cost of $\mathcal{T}^*$.

Lemma 5.3 *The payments computed by Procedure ExactNash purchase $\mathcal{T}^*$.*

First, consider the structure of A_h in an iteration when assigning costs of edge e. Consider a terminal t_j of player h, and let $\mathcal{T}_j$ and A_j be the paths between s and t_j in $\mathcal{T}^*$ and A_h, respectively. The following lemma about the structure of A_h generalizes [ADTW03, Lemma 3.4] to player h with more than two terminals. It similarly holds for player p and her minimum cost deviation A_p.

Lemma 5.4 *If edge e cannot be paid for using the assignment procedure of the algorithm, there is a minimum cost alternative tree A_h for player h with the following property. For any t_j there are two vertices v_j and w_j on A_j such that all edges on A_j from t_j to v_j are in $\mathcal{T}_j \cap \mathcal{T}^e$, all edges between v_j and w_j are in $E \backslash (\mathcal{T}_j \cap \mathcal{T}^e)$, and all edges between w_j and s are in $\mathcal{T}^* \backslash \mathcal{T}^e$.*

Proof. The proof is by contradiction, i.e. if A_h violates this lemma, it can be transformed into a tree for player h that satisfies the properties of the lemma and is not more expensive. Suppose edge e is the first edge that cannot be paid for. Consider the path A_j from a terminal t_j to s. Once it reaches a vertex outside $\mathcal{T}^e$, there is a connection of cost 0 to s, because all vertices from the graph are in $\mathcal{T}^*$. In this case we can adjust A_h to satisfy the lemma without cost increase.

Now suppose A_j leaves $\mathcal{T}_j$ to $\mathcal{T}^e \backslash \mathcal{T}_j$ and re-enters $\mathcal{T}_j$ below e at another vertex. Consider the edges $e' \in \mathcal{T}_j \cap A_j$ of $\mathcal{T}^e$ such that A_j excludes edges from $\mathcal{T}^{e'}$. Let e'_{low} be the lowest of these edges. We denote by $f \notin A_j$ the edge directly below e'_{low} on $\mathcal{T}_j$ (see Figure 5.2(a)). Recall our assumption that e was the first edge that could not be paid for. By the time the algorithm was trying to purchase $T^f + f$, it found that the contribution of player h to T^f was optimal to connect all terminals of h in T^f to $\mathcal{T}^* \backslash \mathcal{T}^f$. As the payment functions and the adjusted cost function c_h^e are built adaptively, we know that this is still true in the present iteration. We can use the repairing construction from Lemma 5.2 to replace the respective parts of A_h with $T^f + f$. This yields a new feasible network that is not more expensive and uses all edges of $\mathcal{T}_j$ from t_j to e'_{low}. Hence, in the new network e'_{low} is not considered anymore as one of the edges $e' \in \mathcal{T}_j \cap A_j$, for which A_j excludes edges from $\mathcal{T}^{e'}$. Thus, iteratively A_h can be transformed without cost increase into a network obeying the lemma. □

Note that the argument can be extended to see that the vertices v_j build a "frontier" in $\mathcal{T}_e$ in the sense that A_h does not use any edge of $\mathcal{T}^e$ above any of the vertices v_j. The vertices v_j are similar to the *deviation points* used in the proof for SSGs in [ADTW03].

Proof. (of Lemma 5.3) Now we prove that the payments assigned by Procedure ExactNash pay for e. Denote by $\mathcal{T}_p^e$ the part of the path between s and the lonely terminal t_p, which is located in $\mathcal{T}^e$. We consider two cases and show in each case how to build a better network than $\mathcal{T}^*$ if Lemma 5.3 is violated.

Case 1: Suppose there is an edge $f \in A_h \cap \mathcal{T}_p^e$. Then for a vertex v incident to f, A_h includes all edges from $\mathcal{T}^h \cap \mathcal{T}_v$, in particular the vertex where $\mathcal{T}_p^e$ joins $\mathcal{T}_h$. If player h deviates to A_h and player p sticks to her payments, this yields a feasible network with cost less than or equal $c(A_h) + c(A_p) < c(\mathcal{T}^*)$.

Case 2: Assume that A_h excludes all edges from $\mathcal{T}_p^e$. Then A_p departs from $\mathcal{T}_p^e$ at some vertex d. However, as e is the first edge, which cannot be paid for, it is optimal for player h use the contribution to $\mathcal{T}^d$ to connect all her terminals located in $\mathcal{T}^d$ to d. A_h can be transformed into a network including $\mathcal{T}^d$ without cost increase. By assumption d is not part of A_h, so the repaired network $\mathcal{T}^d$ might be (part of) a component, which is not connected to the source s anymore. This connection is then established by A_p. Figure 5.2(b) depicts this constellation. The structure of A_h ensures that the other terminals of h outside $\mathcal{T}^d$ are still connected either to s or to $\mathcal{T}^d$. Note that if A_p uses other edges of $\mathcal{T}^e$ outside A_h, we must ensure that h contributes to these edges, because otherwise the cost for p could be increased. However, we can

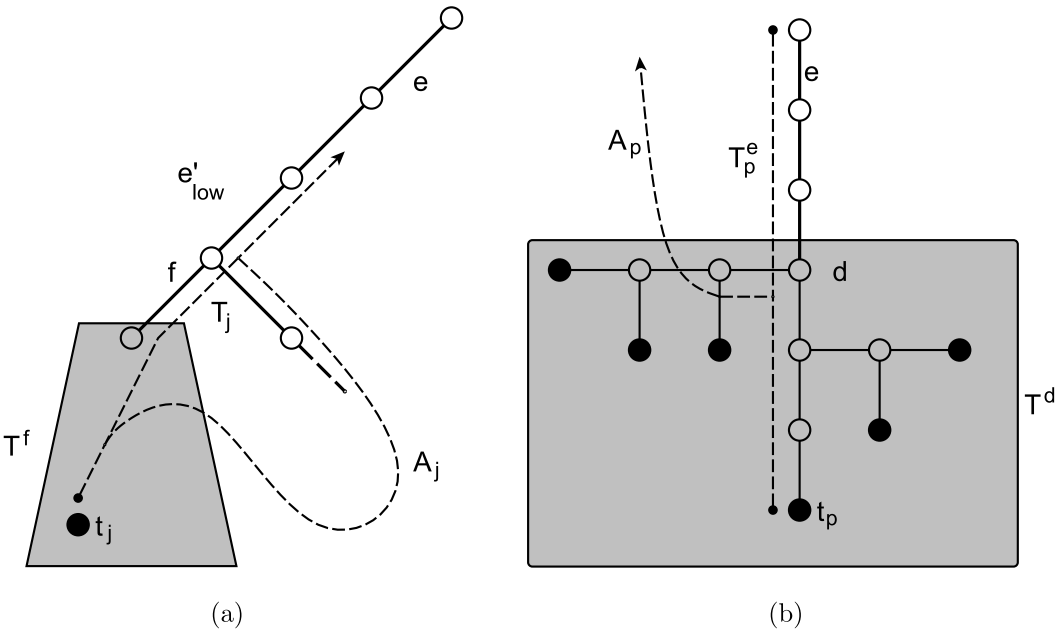


Figure 5.2: (a) A violation of the structure of Lemma 5.4, as A^j leaves and returns to $\mathcal{T}^e$ below e. As $\mathcal{T}^f + f$ is the cheapest way to connect the terminals of h in $\mathcal{T}^f$ to $\mathcal{T}^* \backslash \mathcal{T}^f$, it is possible to replace A_j by $\mathcal{T}_j$ for every terminal $t_j \in \mathcal{T}^f$ of h. This generates a feasible network that is not more expensive than A_h under c_e^h. (b) Improvement step for $\mathcal{T}^*$ if it cannot be paid for. $\mathcal{T}^d$ is optimal for h to connect all her terminals of $\mathcal{T}^d$ to d. Player p also sticks to her contributions in $\mathcal{T}^d$ and establishes a connection to s with A_p, which ensures feasibility.

again apply the same arguments to transform A_h such that a feasible network without cost increase is created.

Hence, if the costs $c_p^e(A_p)$ and $c_h^e(A_h)$ do not allow to pay for $\mathcal{T}^e + e$, there is a cheaper network than $\mathcal{T}^*$ that can be constructed in one of the two ways described. This is a contradiction and proves Lemma 5.3. □

The previous lemmas show that no player has a possibility to lower her contributions and that the cost of $\mathcal{T}^*$ is paid for. This proves Lemma 5.1. □

As Lemma 5.1 captures the crucial part of our induction, Theorem 5.2 follows. This proves that the price of stability in the PTCG is 1. □

If the Algorithm 4 uses Procedure ExactNash with $\mathcal{T}^*$, then it computes an optimum NE. For classes of PTCGs posing efficiently solvable Steiner tree problems (e.g. for constant k [DW72]), an optimal NE can be computed in polynomial time. Unfortunately, these advantageous properties are restricted to the PTCG. In a SSG

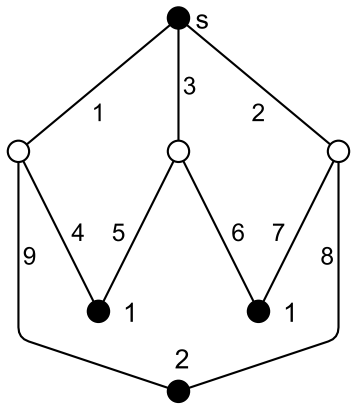


Figure 5.3: A SSG without a NE. Vertex labels indicate player ownership, edge labels are used for identification in the analysis given in the text.

with at most three terminals per player it is NP-hard to decide whether the game has a NE or not. Furthermore, the price of stability is at least $k-2$. We first show that there is a SSG in which for every (α, β)-approximate NE $\alpha > 1.0719$. This means that in every solution there must be a player who can reduce her contribution by at least a factor of 1.0719. In particular, this game has no exact NE. In addition, we provide an initial lower bound for the stability ratio of approximate NE in which $\mathcal{T}^*$ is purchased. Our bound of $\alpha > 1.1835$ is, however, significantly lower than the bound of $\alpha \geq 1.5$ shown in [ADTW03] for general connection games.

Lemma 5.5 *There is a SSG in which for every (α, β)-approximate NE $\alpha > 1.0719$. For $(\alpha, 1)$-approximate NE the bound increases to $\alpha > 1.1835$.*

Proof. Consider the game in Figure 5.3. Note that the topology results from translating the vertex cover game of Figure 4.1(a) into a facility location game and then into a SSG game as described in Sections 4.6.1 and 4.6.2. However, we suppose that $c(e_j) = 1$ for $j = 3, \ldots, 9$. Furthermore, we set $c(e_1) = c(e_2) = x$ and try to adjust the value x such that the minimum achievable stability ratio is maximized.

Consider any (α, β)-approximate NE. Clearly, the network purchased must be connected and include all terminals and the source s. Once players purchase a network with cycles, they can drop edges until the network becomes a tree. In addition, they only need edges on paths between s and a terminal t. Dropping unnecessary edges from the network only decreases the payments of players and increases the cost of possible deviations. Such a transformation decreases the stability ratio. Thus, the minimum stability ratio is obtained for a state that distributes the cost of a tree network with only terminals as leaves. The following case analysis derives bound for the stability ratio in each class of trees including some subset of the edges $\{e_1, e_2, e_3\}$.

Case 1.1 : Suppose only e_1 is included in the tree (the case with only e_2 is symmetric). We set up inequalities to place a general upper bound on certain deviation possibilities. These upper bounds can only be strengthened if certain parts of the network are purchased and the player has to contribute less to the respective edges. $s_p(e)$ denotes the contribution of player p to edge e.

$$\begin{aligned} |s_1| &\leq \alpha(s_1(e_2)+s_1(e_7)+s_1(e_1)+s_1(e_4)) &&\leq \alpha(2+x+s_1(e_1)) \\ |s_2| &\leq \alpha(s_2(e_1)+s_2(e_9)) &&\leq \alpha(1+s_2(e_1)) \end{aligned}$$

The cheapest tree in this case has cost $4+x$, hence adding the inequalities yields the best possible bound of $4+x \leq \alpha(3+2x)$, and thus $\alpha \geq 1+\frac{1-x}{3+2x}$.

Case 1.2: Suppose only edge e_3 is bought. Any such network has cost at least 5. In this case

$$\begin{aligned} |s_1| &\leq \alpha(s_1(e_2)+s_1(e_5)+s_1(e_6)) &&\leq 3\alpha \\ |s_2| &\leq \alpha(s_2(e_1)+s_2(e_9)) &&\leq \alpha(1+x), \end{aligned}$$

so the stability ratio in these networks is at least $\alpha \geq 1+\frac{1}{4+x}$. This is dominated by the bound of Case 2.2.

Case 2.1: Suppose e_1 and e_3 are purchased (the case with e_2 and e_3 is symmetric). It is possible to set up inequalities representing meaningful deviations as in Case 1.1 and to get a lower bound of $\alpha \geq 1+\frac{1-x}{3+2x}$.

Case 2.2: Suppose e_1 and e_2 are purchased. We bound some deviations by

$$\begin{aligned} |s_1| &\leq 3\alpha \\ |s_2| &\leq \alpha(1+s_2(e_1)) \\ |s_2| &\leq \alpha(1+s_2(e_2)) \end{aligned}$$

It is easy to see that the minimum case is achieved by setting $y := s_2(e_1) = s_2(e_2)$. This yields

$$\begin{aligned} 2+2x-2y &\leq 3\alpha \\ 1+2y &\leq \alpha(1+y) \end{aligned}$$

For a stability ratio $\alpha > 1$ we need $x > 0.5$. Then the bound of Case 1.2 is dominated by the one of Case 2.1. Hence, finding the optimum x poses the optimization problem

$$\max_{x\in(0.5,1)} \max_{y\in[0,x]} \min\left(\frac{4+x}{3+2x}, \frac{2+2x-2y}{3}, \frac{1+2y}{1+y}\right).$$

In the optimum case all inner terms are equal. This results in an optimum

$$y^* = \frac{x - 3 + \sqrt{x^2 - 2x + 7}}{2}.$$

Substitution yields the problem

$$\max_{x \in (0.5,1)} \min \left(\frac{4+x}{3+2x}, \frac{5 + x - \sqrt{x^2 - 2x + 7}}{3} \right).$$

Again, the inner bounds must be equal to yield the optimum. It gives $x \approx 0.68548$ and a lower bound on $\alpha > 1.07195$.

Case 3: Finally, suppose all three e_1, e_2 and e_3 are purchased. Then the network includes three additional edges. In this case the network allows the same deviation bounds as the network of Case 2.2, but it is more costly. The minimum stability ratio is not obtained in this case.

Finally, for the case, in which $\mathcal{T}^*$ must be bought, let x be arbitrarily close to 1. Then $\mathcal{T}^*$ includes both edges e_1 and e_2 of cost x, and the corresponding bound of Case 2.2 approaches

$$\lim_{x \to 1} \frac{5 + x - \sqrt{x^2 - 2x + 7}}{3} = 2 - \sqrt{\frac{2}{3}} > 1.1835$$

This proves the lemma. □

We can embed the game of Figure 5.3 into the construction for the price of anarchy. The resulting game does not have a cheap NE.

Theorem 5.3 *The price of stability in the SSG is at least* $k - 2$.

Proof. We combine the game of Figure 5.3 with the game that maximizes the price of anarchy. We use $c(e_1) = c(e_2) = 0.75$ for simplicity and note that the analysis of Lemma 5.5 can be repeated to see that the corresponding game has no NE. Consider the resulting game depicted in Figure 5.4.

Suppose there is a NE without the edge of cost $k - 2 - \epsilon$. Then players $3, \ldots, k$ must be connected with the edge of cost 1, so they all have a direct connection to the source. They do not contribute anything to the cost of the remaining edges of cost $O(\epsilon)$. Thus, the game for players 1 and 2 reduces to the one of Figure 5.3. Lemma 5.5 shows, however, that in this case there can be no NE. Hence, in any NE the players must purchase the edge of cost $k - 2 - \epsilon$. Consider the following strategies: Players $3, \ldots, k$ purchase the costly edge and the additional edge of cost ϵ to s in equal shares; player 1 purchases two edges of cost ϵ; player 2 purchases

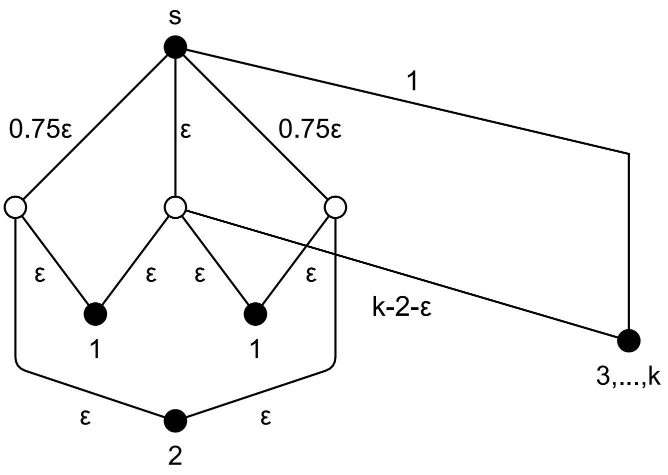


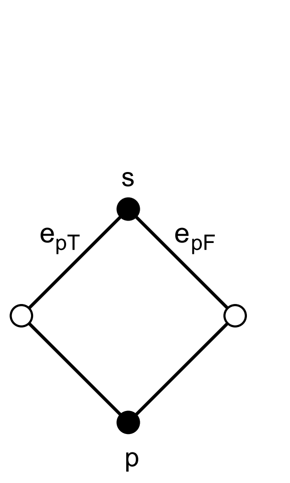
Figure 5.4: A SSG with price of stability arbitrarily close to $k-2$. Vertex labels indicate player ownership, edge labels indicate cost.

one edge of cost ϵ and one of cost 0.75ϵ. There is such a strategy combination that forms a NE. Hence, as ϵ tends to 0, the price of stability becomes arbitrarily close to $k-2$. □

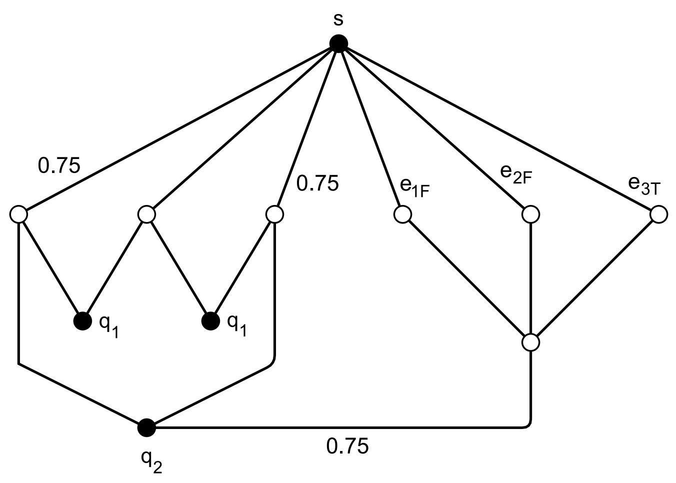
Theorem 5.4 *In the SSG it is* NP*-hard to decide whether a game has a NE.*

Proof. We briefly outline the proof, which uses a reduction from 3SAT (Problem 4.4 defined in Section 4.3) similar to the one shown for vertex cover games in the proof of Theorem 4.5. For an instance of 3SAT a SSG is constructed as follows. For each variable x_p we introduce a *variable player* p and a gadget depicted in Figure 5.5(a). It consists of a single terminal of p and two connections to the source s. A player has a *true* path to s including an edge e_{pT} and a *false* path including an edge e_{pF}. For each clause C_q we introduce two *clause players* q_1 and q_2 and a gadget depicted in Figure 5.5(b). It consists of a game of Figure 5.3 and an alternative connection to the source by a side gadget. This side gadget includes three edges e_{pT} or e_{pF}. These edges are the ones from the true or false path of the corresponding gadgets for the variables appearing in C_q. For instance the gadget for a clause $(\overline{x_1} \vee \overline{x_2} \vee x_3)$ includes edges e_{1F}, e_{2F} and e_{3T} from the variable gadgets of variables x_1, x_2 and x_3 (c.f. Figure 5.5(b)). There is only one edge e_{pT} and one edge e_{pF} for each variable x_p in the whole graph.

Suppose the 3SAT instance has a satisfying assignment. Then we assign the variable players to purchase the edges of the true or false path corresponding to their assignment. This results in a total cost of 2 for each of them. Then, in each



(a) Gadget for a variable.



(b) Gadget for a clause $(\overline{x_1} \vee \overline{x_2} \vee x_3)$.

Figure 5.5: Variable and clause gadgets. Vertex labels indicate player ownership. Numeric edge labels indicate cost, all unlabeled edges have cost 1.

clause gadget one of the e_{pT} or e_{pF} edges is bought. For a clause C_q we assign player q_1 to purchase the three edges of cost 1 directly connecting her terminals to s. Player q_2 is assigned to purchase two edges of the side gadget of total cost 1.75, which connect her terminal to one edge bought by a variable player. It is easy to note that in this case no player has a possibility to connect her terminals with a lower cost by choosing a different strategy.

On the other hand suppose there is a NE. If for a clause gadget player q_2 does not use edges of the side gadget to connect her terminal, an analysis similar to the proof of Lemma 5.5 tells us that q_1 and q_2 do not agree upon a set of edges to purchase. Hence, player q_2 does contribute only to edges of her side gadget. In addition, this implies that no clause player contributes to the cost of edges e_{pT} or e_{pF}. Thus, these edges must be fully bought by the variable players. However, to ensure a connection through the side gadgets for each clause player q_2, there must be for each clause gadget at least one edge e_{pT} or e_{pF} purchased by the variable players. As each variable player purchases exactly one of her true or false paths, this directly yields the desired satisfying assignment for the 3SAT instance.

This proves that in general deciding the existence of a NE is NP-hard. Our constructed games, however, involve only players with at most three terminals each. For such a player the optimum Steiner tree can be found in polynomial time [DW72], and thus it is possible to recognize a NE in polynomial time. Hence, for this special case the problem is NP-complete. □

5.4 Approximate Nash Equilibria

Similar to the covering and facility location games in the last chapter NE for TCGs can be hard to find or very expensive. This section studies the existence and computability of cheap approximate NE providing a trade-off between efficiency and stability. In contrast to the Algorithms 1 and 3, the algorithms here do not rely on LP-duality. Instead, they use the concept of *connection sets* [ADTW03] which are special sets of edges of $\mathcal{T}^*$.

Definition 5.4 *A connection set* S *of player* p *is a subset of edges of* $\mathcal{T}_p$*, such that for each connected component* C *in* $\mathcal{T}^* \setminus S$ *either*

1. *there is a terminal of* p *in* C*, or*
2. *any player that has a terminal in* C *has all of her terminals in* C*.*

For any non-terminal vertex of degree 2 in $\mathcal{T}^*$ the incident edges belong to the same connection sets, so for convenience we assume that $\mathcal{T}^*$ has no such vertices. Suppose we assign a player to purchase exactly a single connection set. If she switches to a cheaper set of edges connecting her terminals, this keeps the purchased network feasible and improves upon the cost of $\mathcal{T}^*$. Hence, the cost of a connection set for a player lower bounds any of her deviation cost. Thus, if $\mathcal{T}^*$ is purchased by assigning every player to pay for at most α connection sets, the state forms an $(\alpha, 1)$-approximate NE. Note that a subset of a connection set also is a connection set.

5.4.1 An Algorithm for PTCGs

In connection games with two terminals per player the edges of $\mathcal{T}^*$ can be partitioned into equivalence classes S_Q such that e and e' belong to the same class iff $Q = \{q : e \in \mathcal{T}_q\} = \{q : e' \in \mathcal{T}_q\}$.

Lemma 5.6 *In connection games with two terminals per player each* S_Q *forms a connection set for all players* $q \in Q$ *which is maximal under the subset relation. In the PTCG connection sets* S_Q *form a contiguous path.*

The lemma can be proved rather easily by assuming the contrary and deriving contradictions to the definition of connection sets and tree connection requirements.

We call a connection set S_Q *needed* by Q. For the rest of this section we silently consider only maximal connection sets and not explicitly mention a player, as this information is given implicitly by the player subtrees the set is located in. For deriving approximate NE we again use the framework Algorithm 4. In line 4, however, we use a different procedure to assign the costs. For a leaf player p we pick two connection sets and assign p to purchase them. If p is not a leaf player, $s_p = 0$.

We must carefully choose the connection sets that are assigned to a leaf player p. For instance, there might be two connection sets, which are needed by player sets differing only by p. If p is assigned to purchase none of these sets, then after the player is removed in line 7 of Algorithm 4, the two connection sets will be needed by the same player set. As aruged above, however, this identifies them as one connection set. Hence, in this case two distinct connection sets would be considered as one connection set after removal of p. Naturally, this would destroy our argumentation if this connection set is assigned to another player considered in later iterations. In addition, a connection set needed only by p must be assigned to p, because otherwise it will remain unpurchased. Avoiding these problems provides candidate connection sets for the assignment to p.

Definition 5.5 *A connection set is called an* endangered set *for player* p *if*

1. *it is needed only by* p*. We call such a connection set a* personal set*.*
2. *it is needed by the set of players* $Q \cup \{p\}$*, and there is another connection set (called a* forcing set*) needed by the set* Q*, with* $Q \neq \emptyset$ *and* $p \notin Q$*. We call such a connection set a* community set*.*

Indeed, for any leaf player there are at most two endangered sets.

Lemma 5.7 *For any leaf player in a PTCG there are at most two endangered sets.*

Proof. As there is only one personal set, we must show that there is at most one community set. Assume for contradiction that for a leaf player p there are several community sets. Arbitrarily pick two distinct forcing sets S'_1 and S'_2 with player sets Q_1 and Q_2, respectively. The corresponding community sets are denoted S_1 and S_2. We denote $Q = Q_1 \cup Q_2$.

Consider the tree $\mathcal{T}^*$. Upon removal of S_1 and S_2 three components evolve. Two of them (denoted C_1 and C_2) each contain one terminal of p. As the subtree $\mathcal{T}_p$ for each player is a path, the third component (denoted C_3) contains a terminal of each player in $(Q_1 \cup Q_2) - (Q_1 \cap Q_2)$. For the first two cases we suppose there is no forcing set in C_3.

Case (a): S'_1 and S'_2 are located in C_1 and C_2, each in a different component. Hence, after removal of S'_1 and S'_2 components C_4 and C_5 evolve (see Figure 5.6(a)). Now all terminals of players in Q are distributed to C_3, C_4 and C_5. If the underlying graph structure allows it, we can reconnect these components into a component and C_1 and C_2 into a second component. This would yield a disconnected graph that satisfies the connection requirements. This is a contradiction to the presence of tree connection requirements, no matter whether such a connection is actually possible with the edges from the underlying graph or not.

Case (b): S'_1 and S'_2 are located in C_1 and C_2, both in the same component. Hence the other component holds a terminal of each of the players in one set, w.l.o.g. we assume C_1 a terminal from each player in Q_1. As S'_1 is in C_2, all players of Q_1 need both S_1 and S_2, so $Q_1 \subset Q_2$. Hence, in C_3 there is one terminal of each player of $Q_2 - Q_1$. In C_2 there is only one terminal of each player in Q_1. If we remove S'_2, we split off a new component C_4 containing one terminal of each of the players in Q_2. S'_1 is, of course, located in C_4, because it is needed by a subset of the players. If we remove S'_1, we get a new component C_5 with a terminal of each of the players in Q_1 (see Figure 5.6(b)). In C_4 one terminal of each of the players in $Q_2 - Q_1$ remains. So if we connect C_4 and C_3 into a component, there is no need to connect this new component to the rest of the tree. This again violates the tree connection requirements.

Case (c): Suppose one forcing set (w.l.o.g. S'_1) is located in C_3. This means that $Q_1 \cap Q_2 = \emptyset$. The tree requirements ensure, however, that for each pair of players $q_1 \in Q_1$ and $q_2 \in Q_2$ there is a sequence of players that transitively require a connection between q_1 and q_2. Note that p cannot be part of this sequence as she is a leaf player. In particular, this means there is at least one player whose path includes either S_1 or S'_1. This is a contradiction to the definition of community and forcing sets.

This proves that there is only one community set, which yields at most two endangered sets for a leaf player. □

In line 4 of Algorithm 4 we thus simply assign a leaf player to purchase the endangered sets. This ensures that all connection sets of $\mathcal{T}^*$ are assigned, and the connection sets considered and assigned in later iterations correspond to original connection sets. Then the algorithm works correctly. It can be combined with recent approximation algorithms to yield the following theorem.

Theorem 5.5 *In a PTCG a $(2+\epsilon, 1.55)$-approximate NE can be computed in polynomial time, for any constant $\epsilon > 0$.*

Proof. The algorithm assigning endangered connection sets is still inefficient, because it requires $\mathcal{T}^*$ as input. For the translation into a polynomial time algorithm we use the idea presented in [ADTW03] and applied in detail in the proof of Theorem 4.11. It is possible to use a β-approximation algorithm for Steiner Tree to get an initial approximation $\mathcal{T}$. Assume this tree is optimal and assign connection sets to a leaf player. After the assignment consider each of her sets independently. In particular, for a connection set S_Q assume a cost of 0 for all $e \in \mathcal{T} \backslash S_Q$ and calculate the shortest path in G between the terminals of p. If S_Q is not optimal, then replace it with the cheapest path and output the improved network. In this way the network $\mathcal{T}$ can feasibly be improved, which yields a restart of Algorithm 4

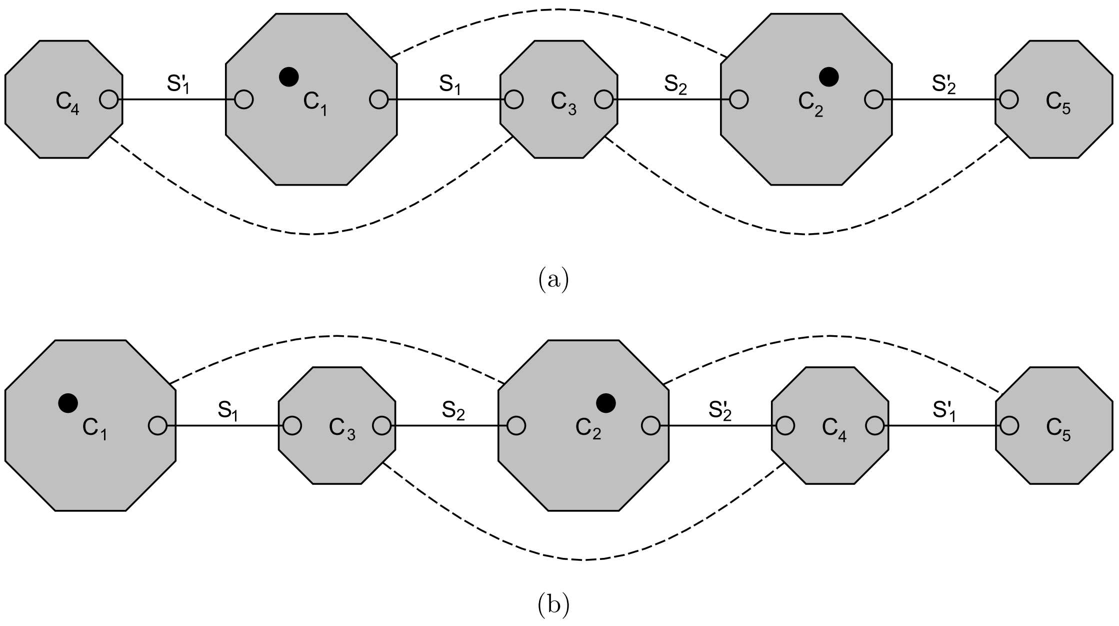


Figure 5.6: Component structures in the presence of more than two community sets for player p. Replacing connection sets S_1, S_2, S'_1, and S'_2 with dashed edges creates feasible, unconnected networks. Filled vertices are terminals of p.

on the improved network. To ensure a polynomial number of restarts, fix parameter $\kappa = \frac{\epsilon c(\mathcal{T})}{(1+\epsilon)n\beta}$ in the beginning (with ϵ small enough to ensure $\kappa < \min_{e \in E} c(e)$). For checking optimality of a connection set S_Q, temporarily reduce the cost of each edge in S_Q by κ. Then a cheaper path improves the cost of the tree by at least an amount of κ. This yields at most $\frac{(1+\epsilon)n\beta}{\epsilon}$ restarts of the framework. After the algorithm has run to completion, a last post-processing step is needed to restore the original costs of the edges. The remaining cost of at most κ is split between all players proportionally to the total contribution of each player to the cost of the tree. By repeating the analysis of Theorem 4.11 this yields at most an ϵ-factor deterioration in the stability ratio. The theorem follows with the recent 1.55-approximation algorithm for Steiner Tree [RZ05]. □

5.4.2 An Algorithm for TCGs

In this section we adjust the idea of assigning connection sets to get $(2, 1)$-approximate NE for TCGs with any number of terminals per player. Each player (denoted as parent player) is divided into a set of child players with two terminals per player. The child players have the same terminals as the parent player, and they are dis-

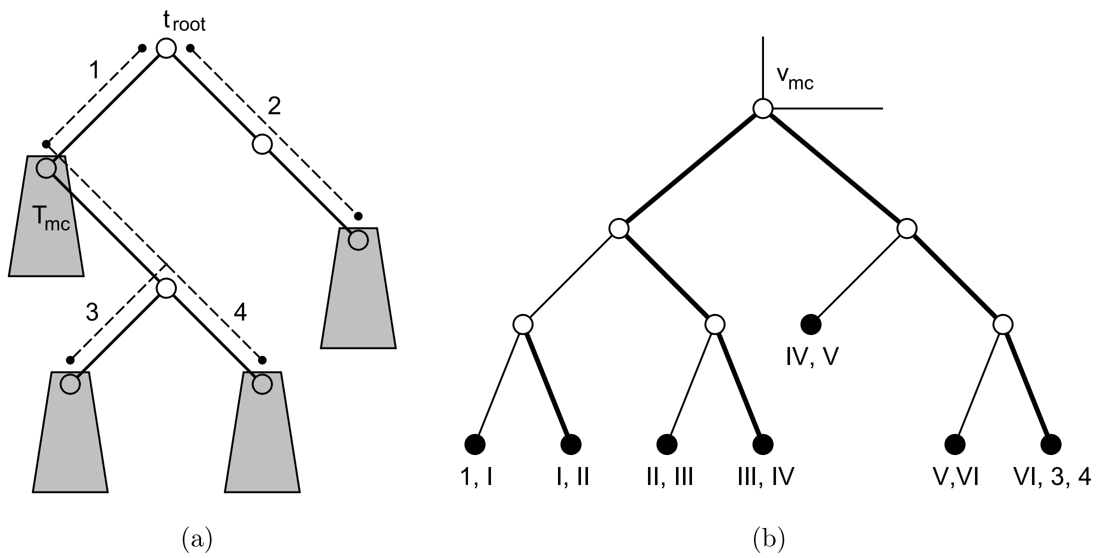


Figure 5.7: (a) Distribution of hierarchical players 1-4 for a parent player p; (b) distribution of personal players I-VI in the subtree marked $\mathcal{T}_{mc}$ in (a). Bold edges indicate personal sets of personal players after the assignment of the algorithm

tributed such that the child player game is a PTCG. Then, the algorithm assigning endangered sets can be used to assign $\mathcal{T}^*$ such that each child player purchases at most two connection sets. The union of these connection sets yields only two connection sets for the parent player.

Theorem 5.6 *For any social optimum tree $\mathcal{T}^*$ in a TCG there exists a $(2, 1)$-approximate NE in which $\mathcal{T}^*$ is exactly purchased.*

Proof. The following pattern is used to divide a leaf parent player p into *hierarchical* and *personal* child players. Then we process these child players such that the union of assigned connection sets forms two connection sets for p. This suffices to prove the theorem. For our division of player p it is possible to disregard all non-lonely terminals of p but one, as the corresponding connection requirements can be left for other players to satisfy. Denote this last remaining non-lonely terminal by t_{root}. If the player has only lonely terminals, we pick t_{root} arbitrarily. Then consider $\mathcal{T}^*$ rooted at t_{root} in BFS-order. For an edge e needed only by p, the tree connection requirements guarantee that the subtree below e is also needed only by p. Contract all such edges that are needed only by p. Denote this adjusted tree by $\mathcal{T}_{adj}$ and consider it again in BFS-order rooted at t_{root}. For each vertex $t \in T_p$ we introduce a new child player. She gets assigned t and the nearest ancestor vertex that is a terminal of p (see Figure 5.7(a)). These child players are termed *hierarchical players.*

Consider the portions of the tree that were contracted to form $\mathcal{T}_{adj}$. For each maximal connected subtree $\mathcal{T}_{mc} \subseteq \mathcal{T}_p$ that is needed only by p, let v_{mc} be the root vertex that represents $\mathcal{T}_{mc}$ in $\mathcal{T}_{adj}$. Let player q be the first hierarchical child player, who got assigned the root vertex v_{mc}. This player strives to connect upwards in $\mathcal{T}_{adj}$. Now we consider $\mathcal{T}_{mc}$ in DFS-order and consider the first encountered terminal of p. If this is not the root v_{mc}, we relocate child player q to this terminal. For each new terminal t_x encountered in the DFS order, we introduce a new child player and assign her terminals t_{x-1} and t_x. Except for the remaining hierarchical players at v_{mc} there is only one child player with a lonely terminal in $\mathcal{T}_{mc}$ at all times during this assignment. Finally, consider the last terminal t in the DFS-scan of $\mathcal{T}_{mc}$. We assign all hierarchical players connecting downward in $\mathcal{T}_{adj}$ to t instead of the root v_{mc}. Child players introduced in the DFS-scan of the components $\mathcal{T}_{mc}$ are called *personal players*, because they divide parts needed only by p (see Figure 5.7(b)).

After the division of a parent player the algorithm for PTCGs is used to assign connection sets. In any iteration a leaf child player is picked and assigned to purchase her endangered sets, however, we prefer to pick personal over hierarchical leaf players. Thus, the procedure works roughly bottom up to t_{root}. Finally, one connection set for the parent player p is formed by the union of all personal sets for the child players. The other connection set is the union of the community sets. Actually, a slightly stronger statement holds.

Lemma 5.8 *If the child players of a parent player p are created and eliminated in the described way, the removal of the child players' personal and community sets creates only components that contain terminals of p, respectively.*

To see the argument, we first have a closer look at the structure of endangered and forcing sets.

Lemma 5.9 *For any leaf player in a PTCG the personal, community, and forcing sets share a common vertex if they exist.*

Proof. If there is no forcing set, there is no community set and the lemma follows trivially. So let there be a forcing set S_Q needed by player set Q. Suppose for a leaf player p the sets do not share a vertex. Remove the community set, and let C_1, C_2 be the components with and without the lonely terminal of p, respectively.

Case (a): Suppose S_Q is in C_2 and remove it. This splits C_2 into two components. We denote by C_2' the remaining component including the terminal of p. The other component is denoted by C_3, and it contains one terminal of each player in Q. Now remove all edges that connect to the lonely terminal t_p of p in C_1. Connect all resulting components except for t to C_3. Then connect the vertex v to C_2' (see Figure 5.8(a)). All connection requirements are met, but there is a solution with two components. This contradicts the presence of tree connection requirements. Hence, S_Q must be in C_1.

Case (b): Suppose S_Q is in C_1 and remove it. Similar to Case 1 we refer to C_1' and C_3 as resulting components after removal of S_Q. Now suppose there is another player q with a terminal located in C_1'. $\mathcal{T}_q$ can only include one of S_Q and S_{Q+p}, so $q \notin Q$, and she must have both her terminals in C_1'. Again isolate the lonely terminal t_p. Then construct two components, one consisting of t, C_2 and C_3; the other one consisting of all other components (see Figure 5.8(b)). This generates a feasible solution with two components, which is a contradiction of tree connection requirements. Once there are no terminals in C_1', there can be no connection sets in C_1', except for the personal set S_p needed only by p. It remains to show is that S_p must be located in C_1'.

Case (c): Suppose S_p is in C_2 and remove it. Again we denote C_2' the component with the terminal of p and C_3 the other one. Observe that C_3 must contain all terminals of Q. Then remove S_Q from C_1 generating C_1' and C_3'. We can isolate the components containing terminals of Q from the rest of the components (see Figure 5.8(c)). A feasible network with two components is possible, which contradicts the presence of tree connection requirements.

If S_p exists, it is in C_1', and the three connection sets share a Steiner vertex. Otherwise, the two connection sets meet at the lonely terminal of p. This concludes the proof of Lemma 5.9. □

Proof. (of Lemma 5.8) We use an inverse induction to show the lemma. Suppose the algorithm has assigned all edges to child players. We reverse the elimination order of child players and consider player and edge additions instead of player removals and edge contractions. The child player that was eliminated last is now the one that is inserted first. It is obvious that for the first inserted (i.e. last eliminated) child player the lemma holds. This is the base case of our induction. Now suppose the property holds after we have inserted in the reverse order a given number of child players from one or more parent players and their assigned edges. Then consider the insertion of an additional child player q of a parent player p. This can be either a hierarchical or personal child player. Recall the elimination order outlined above. Player q can have a personal set. Consider the union of all personal sets of child players of p eliminated later (i.e. inserted already). By the induction hypothesis after removal of this union every evolving component contains a terminal of p. If in addition to that the personal set of q is removed, the only additional component that evolves is the lonely terminal of q. This proves the induction for the personal sets.

For the community sets the case is more complicated. Consider the hierarchical players. Their community sets are always needed by at least one additional player, which is not a child player of p. If the inserted player q is a hierarchical player, consider her terminal in the lower of the two subtrees $\mathcal{T}_{mc}$ containing her terminals.

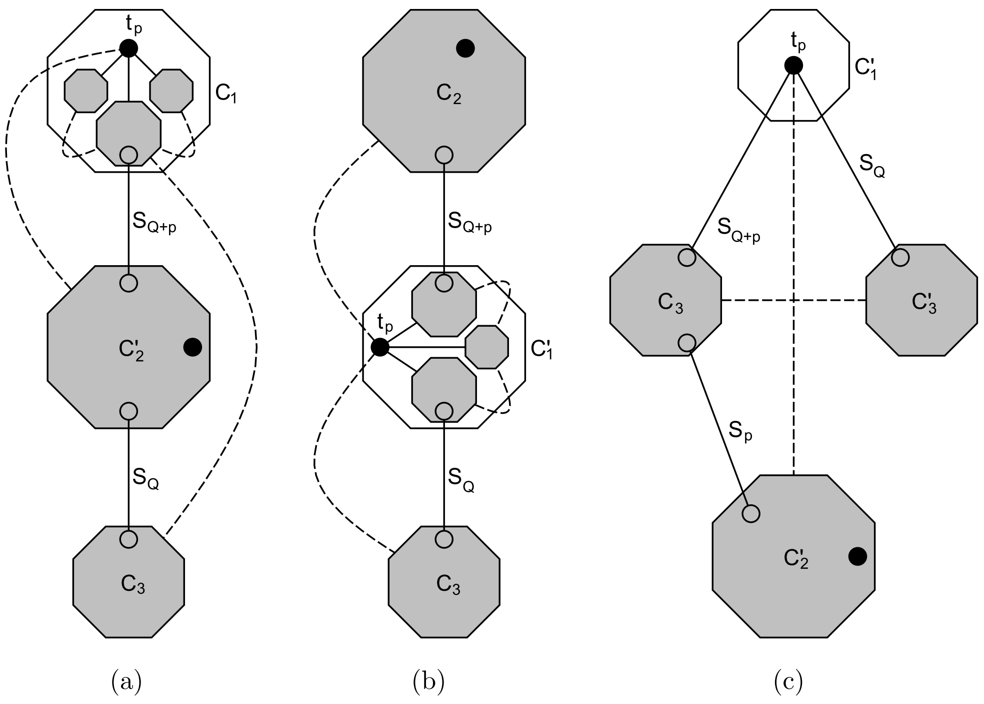


Figure 5.8: Component structures when endangered sets do not share a vertex. Replacing solid with dashed edges creates feasible, unconnected networks. Filled vertices are terminals of p.

This lower terminal is in the component newly created by the removal of her community set. This is ensured by the hierarchical structuring and the fact that other players are also present on the community set.

If q is a personal player, we consider the subtree $\mathcal{T}_{mc}$ that she is located in. Recall that $\mathcal{T}_{mc}$ is needed only by p, and note that the root vertex v_{mc} of $\mathcal{T}_{mc}$ does not have to be a terminal of p. In addition, v_{mc} might be incident to community sets of hierarchical players. The complete subtree $\mathcal{T}_{mc}$ must be purchased by p, hence it consists completely of personal and community sets of child players of p. The lemma is proven if we can show that every vertex in $\mathcal{T}_{mc}$ is either a terminal of p or connected by personal sets to a terminal of p. In addition, the root v_{mc} must be connected by personal sets to the terminal of the hierarchical players connecting downward in the tree (if any). This serves to keep the above given argument for hierarchical players feasible. Now consider as an additional invariant that for every connected subtree $\mathcal{T}' \subseteq \mathcal{T}_{mc}$ the vertex v' closest to t_{root} is connected by personal sets to the terminal considered last during the DFS-scan of $\mathcal{T}'$. Due to the DFS-based construction of personal players we always eliminate first the player constructed last in $\mathcal{T}_{mc}$. Consider the subtrees q is located in. At the current time she is inserted last, so in turn of the players present she was eliminated first. Therefore, we know she was constructed last, and because of that she cannot have *only* a community set. So if q has a community set, she also has a personal set. Then, due to the properties shown in Lemma 5.2, there is a Steiner vertex v between these sets. Suppose now the endangered sets of q are contracted, then by construction the lonely terminal of the second-last introduced child player or a hierarchical player is joined with v. Using the induction hypothesis and the fact that v is connected with a personal set to the lonely terminal of q, we see that the invariant holds in $\mathcal{T}_{mc}$. In particular, every vertex stays connected by a path of personal sets to a terminal of p, and hence no component without a terminal of p can evolve once all community sets of p are removed from $\mathcal{T}^*$. For illustration see Figure 5.7(b), in which the bold lines indicate the personal sets of the child players I-VI. This proves the induction hypothesis for community sets of hierarchical and personal players, and Lemma 5.8 follows. □

The splitting for a leaf parent player p creates two edge sets, which upon removal yield only components including terminals of p. If such a set is removed, all resulting components must be reconnected to form a feasible network. Hence, these edge sets are connection sets for p. It also ensures that in our induction we can add the community (personal) sets of q to the sets of community (personal) sets of other child players of p. This completes the proof of Theorem 5.6. □

The next lemma ensures that the assignment of personal and community sets for a leaf parent player p does not depend on the splitting of the other parent players. In an iteration of our framework we can thus assign edge costs to a parent player p assuming an arbitrary splitting of other parent players. This ensures that the algorithm can find the correct personal and community sets for child players in polynomial time.

Lemma 5.10 *The endangered sets of child players of a leaf parent player* p *are independent of the division of other parent players.*

Proof. Again we use an inductive argument based on a single child player. Suppose we have a child player q of parent player p, who is removed from the game. Consider an arbitrary splitting of the other parent players into child players obeying the tree connection requirements. The personal set of q is independent of the splitting of the other parent players, which proves the lemma for the personal set.

Suppose q has a community S_c set needed by $Q \cup \{q\}$ and a forcing set S_f needed by Q. We denote by v the vertex that both sets connect to. Consider a different player p' with a child player in Q and $Q \cup \{q\}$. $\mathcal{T}^{p'}$ cannot have a terminal at v and must not include any other edges incident at v than the two edges in S_c and S_f. Otherwise the tree connection requirements would require a different set of child players of p' on S_c and S_f, which would contradict the assumption that a community set for q is present. Now consider a different splitting of p', which results in different player sets needing S_c and S_f. There must be at least one child player of p' needing each of these sets, because they are both in $\mathcal{T}^{p'}$. However, as there is no terminal or alternate connection at v that is in $\mathcal{T}^{p'}$, any child player of p' needing S_c also needs S_f. Hence, S_c remains the community set for q. This argument can easily be adjusted for more players. □

Theorem 5.7 *In a TCG a* $(3.1+\epsilon, 1.55)$*-approximate NE can be computed in polynomial time, for any constant* $\epsilon > 0$.

Proof. Procedure ApproxNash sketches and summarizes the described steps to compute approximate NE for general TCGs. The complete algorithm again uses Algorithm 4, and in line 4 it calls Procedure ApproxNash to assign some cost of $\mathcal{T}^*$ to the parent player p. Note that for TCGs we can use the improvement steps on connection sets to obtain a polynomial time algorithm using the same scaling ideas as in Theorem 5.5. As each connection set is now a tree, we use the 1.55-approximation algorithm [RZ05] for Steiner Tree not only to compute a starting solution, but also to compute improvements for connection sets. This yields a stability ratio of $3.1 + \epsilon$. The theorem follows. □

Procedure ApproxNash($\mathcal{T}, c, p, \kappa$) for computing $(3.1+\epsilon, 1.55)$-approximate NE for TCGs

```
    Input: A feasible tree $\mathcal{T}$, a cost function $c$, a selected player $p$, a constant $\kappa$
    Output: A payment function $s_p$ or a tree $\mathcal{T}^+$
27  Pick a non-lonely terminal as $t_{root}$
28  Disregard all non-lonely terminals of $p$ except for $t_{root}$ from her set $T_p$
29  Generate $\mathcal{T}_{adj}$ by contracting subtrees $\mathcal{T}_{mc}$ only needed by $p$
30  Create hierarchical players on $\mathcal{T}_{adj}$
31  Expand $\mathcal{T}_{adj}$ and create child players on each $\mathcal{T}_{mc}$
32  Run algorithm for child players of $p$; prefer choice of personal over
    hierarchical leaf players
33  Assign $p$ to purchase connection sets assigned to her child players
34  for each of the two connection sets $S$ do
35      Create $c_S$ by $c_S(e) \leftarrow c(e) - \kappa$ for $e \in S$ and $c_S(e) \leftarrow c(e)$ otherwise
36      Create $G_S$ by contracting all edges of $\mathcal{T} \backslash S$.
37      Run 1.55-approximation algorithm on $G_S$ and $c_S$ for terminals of $p$
38      if returned solution $S'$ is cheaper than $S$ under $c_S$ then
39          return $\mathcal{T} - S + S'$
40  return $s_p$
```

5.5 Discussion

5.5.1 Tightness of the Bounds

Our Procedure ApproxNash and ADTW proposed in [ADTW03] both rely on the concept of connection sets. In this section we will argue that with respect to connection games the analytic power of connection sets is limited. In particular, algorithms that approach the problem of finding good approximate NE relying on connection sets cannot achieve a significantly lower stability ratio.

The difference between our algorithm and ADTW is that the assignment procedure used by ADTW does not employ the structural information of our child player splitting. With the structure of TCGs a splitting of parent players and a hierarchical elimination order are possible. This avoids the matching step ADTW uses to assign edge costs to players. This is crucial for achieving a guarantee of two connection sets.

Similar to our algorithm ADTW does not employ cost sharing of edges. The next theorem shows that no deterministic algorithm using only $\mathcal{T}^*$ as input can improve the guarantees even if it uses cost sharing. Thus, ADTW for general connection games and our algorithm for TCGs yield optimal stability ratios with respect to this class of algorithms. Our algorithm provides a better guarantee on TCGs, because

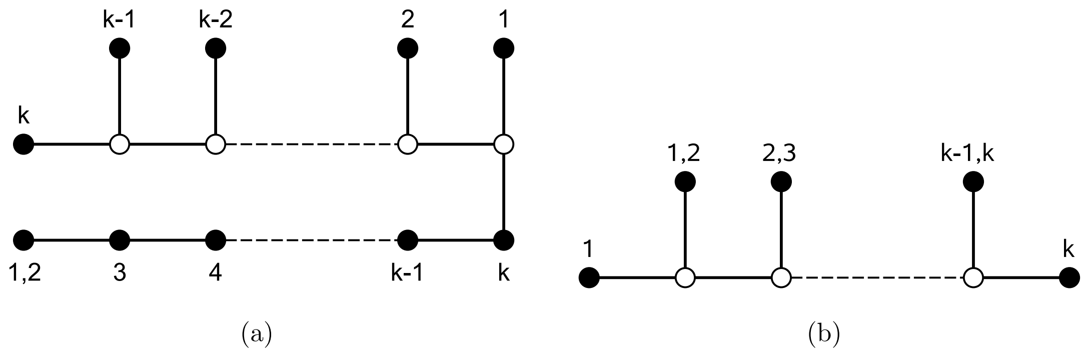


Figure 5.9: Optimal trees $\mathcal{T}^*$ for games yielding tightness of the analysis of the algorithms. One player can have a cheap additional edge of cost $1 + \epsilon$ connecting her terminals. This is unknown to the algorithms and thus gives a lower bound on the stability ratio of (a) $3 - \epsilon$ for general connection games and (b) $2 - \epsilon$ for TCGs.

ADTW can assign three connection sets to a player of a TCG with a cheap alternate path.

Theorem 5.8 *For any $\epsilon > 0$ there is a connection game such that any deterministic algorithm using only $\mathcal{T}^*$ as input constructs a state with stability ratio at least $3 - \epsilon$.*

Proof. Consider a game with two terminals per player and an optimal solution $\mathcal{T}^*$ of cost $3k - 3$ shown in Figure 5.9(a). All edges of $\mathcal{T}^*$ have cost 1. There is at least one player that pays a cost of $3 - \frac{3}{k}$. One player can have an alternative path of cost $(1 + \epsilon)$ outside $\mathcal{T}^*$. As this path is not known to the algorithm, the best approach is to equilibrate payments between players. It assigns each player p to pay a cost of $3 - \frac{3}{k}$ for parts inside her path $\mathcal{T}_p$. As k approaches infinity, the stability ratio becomes arbitrarily close to 3. □

Theorem 5.9 *For any $\epsilon > 0$ there is a PTCG such that any deterministic algorithm using only the optimum solution $\mathcal{T}^*$ as input constructs a state with stability ratio at least $2 - \epsilon$.*

Proof. An argument similar to the proof of the previous theorem for the game in Figure 5.9(b) shows that it is optimal to assign each player to contribute a cost of $2 - \frac{1}{k}$ within her subtree. So as a deterministic algorithm working only with $\mathcal{T}^*$ our algorithm delivers the optimum worst-case guarantee. □

Theorem 5.10 *For any $\epsilon > 0$ there is a PTCG for which ADTW constructs a $(3 - \epsilon, 1)$-approximate NE.*

Proof. Consider the game in Figure 5.9(b). ADTW proceeds as follows. At first each player is assigned to purchase her personal set. Afterwards, it picks two terminals of a player and assigns the path to be purchased by players that include parts of the path in their subtree. In particular, the terminals are assigned to purchase edges of the path. In the following iterations the paths to the purchasing terminals are assigned to other terminals and so on. Finally, a player is assigned to purchase all edges that were assigned to her terminals. The distribution of edges to terminals is done in a matching step. This is optimal for the general case, but for TCGs it might yield an unlucky assignment. If in our example the assignment starts by picking player 2, the matching can assign player 1 to purchase two connection sets. With the personal set this results in three connection sets. If all edges have cost 1 and there is an alternative path of cost arbitrarily close to 1, player 1 yields a stability ratio of arbitrarily close to 3. □

These tightness results cannot be easily strengthened for the polynomial time variants of the algorithms, because they heavily depend on tightness results and solution properties of the underlying approximation algorithms for Steiner Tree and Steiner Forest. It is, for instance, not known, whether the performance ratio of the recent 1.55-approximation algorithm [RZ05] is tight.

5.5.2 Backbone Games

In this section we discuss the possibility of extending the algorithm design techniques used in this thesis to more complicated network creation games. It is easy to observe that the notion of connection sets is closely tied to properties of the Steiner Tree and Steiner Forest problems [Hoe04]. The framework of our algorithms, however, can be used in other contexts. As an exposition we outline the *backbone game*, an extension of the connection game to *groups* of terminals. A group of terminals is simply a set of terminals, but we use the name group for easy reference. In the backbone game each of the k players has a set of groups of terminals. The player strives to connect at least one terminal from each of her groups into a connected network. The rest of the game is similar to the connection game, i.e. each player picks a payment function as strategy and can use bought edges for free. The backbone game becomes the connection game for singleton groups. Thus, some important results translate directly by restriction. The price of anarchy is k, and the price of stability at least $k-2$. It is NP-hard to decide whether a given game has a NE, and there are games, for which any $(\alpha, 1)$-approximate NE the stability ratio $\alpha \geq \frac{3}{2}$ (c.f. [ADTW03]). Finding the optimum network for a single player is the network design problem of the Group Steiner Tree (GSTP) [RW89, GKR00].The problem of finding a social optimum network $\mathcal{T}^*$ of the backbone game generalizes the GSTP in terms of forest connection requirements, so we term this Group Steiner Forest

(GSFP). There are polylogarithmic approximation algorithms for the GSTP, but we are not aware of any such results for the GSFP. In the following we thus concentrate on algorithms for games, in which the solution is guaranteed to be connected. The general case represents an interesting field for future work.

In a single source backbone game (SBG) each player p has one group of g_p terminals and there is a single source vertex s. The connection requirement of player p is satisfied, if there is a connection of bought edges from s to least one terminal of her group. Note that the price of anarchy is still exactly k as the arguments and the example from the beginning of Section 5.3 translate directly. It is possible to use techniques of this and previous chapters to show that the price of stability is 1 and cheap approximate equilibria can be found in polynomial time.

Theorem 5.11 *For any social optimum forest $\mathcal{T}^*$ in a SBG, there exists a NE in which $\mathcal{T}^*$ is exactly purchased. The price of stability in the SBG is 1.*

Proof. Consider $\mathcal{T}^*$ in BFS order from the source s. We use a reduction to connection games. Construct a new graph $G' = (V', E')$ by adding an artificial vertex u_p for each player p. Let $U = \{u_1, \ldots, u_k\}$ and $V' = V \cup U$. Connect u_p to all terminals of player p with an artificial edge of prohibitively high cost $c(E)$. The single source game in G', in which each player strives to connect u_p to s, is called the *corresponding connection game (CCG)*. The social optimum $\mathcal{T}^*$ for the backbone game corresponds to a social optimum tree for the CCG and vice versa, as the degree of every u_p in the optimum tree is 1. Note that a cheapest deviation for a player in both games consists of a path. Applying the algorithm ADTW-SS in the CCG every player p gets assigned exactly one artificial edge, and every reasonable deviation for player p in G' also includes only one such edge incident to u_p. This yields a correspondence of deviation paths, and the calculated NE for the CCG is also a NE for the backbone game. Furthermore, if an edge of $\mathcal{T}^*$ cannot be purchased in the CCG, there is a possible network improvement step, as in each improved network of the CCG all vertices u_p have degree 1. Hence, the improvement step of ADTW-SS in the CCG yields a better feasible network for the backbone game. To obtain an optimal NE it suffices to simply apply ADTW-SS in the CCG. □

Theorem 5.12 *In a SBG a $(1 + \epsilon,\ O(\log n \log k \log(\max_p g_p)))$-approximate NE can be computed in polynomial time, for any constant $\epsilon > 0$.*

Proof. Suppose we are given an β-approximate network $\mathcal{T}$. Again, the argument uses the scaling ideas as presented in the proofs of Theorems 4.11 and 5.5. We reduce the cost of every edge by a cost of $\kappa = \frac{\epsilon c(\mathcal{T})}{(1+\epsilon)\alpha n}$ with $\epsilon > 0$. As ADTW-SS offers an improvement step, we can iteratively improve the network $\mathcal{T}$ in polynomial time. This yields a $(1 + \epsilon, \beta)$-approximate NE in polynomial time. We only point out two important observations that crucially aid the adjustment. First, both $\mathcal{T}$ and possible optimal deviations include exactly one artificial edge in the CCG. Hence, rather than constructing the CCG we consider the cheapest deviation path for a player p from any of p's terminals to s in the original game. This avoids to include the costly artificial edges into consideration, which for instance prevents us to translate approximation results for Steiner Tree to the GSTP. Second, to ensure a constant $1+\epsilon$ and polynomial running time, for each game a *non-asymptotical* upper bound on β is needed, which must be polynomial in n and k. We cannot hope for a constant, as GSTP generalizes set cover. Using an algorithm [BHRZ97] to solve the GSTP with $\beta = (1+\ln\frac{k}{2})\sqrt{k}$ we can compare its solution against more recent efficient methods for the GSTP with asymptotical polylogarithmic performance guarantees [GKR00, FRT04]. In this way the solution network is an $O(\log n \log k \log(\max_p g_p))$-approximation, but never more than a $((1+\ln\frac{k}{2})\sqrt{k})$-approximation of $\mathcal{T}^*$. □

Interestingly the stability ratio from the connection game translates to the backbone game – in contrast to approximation ratio. The results extend to SBGs on directed graphs and games, in which each player p has a threshold τ_p on her maximum contribution and rather stays unconnected if her assigned share exceeds τ_p. These results translate from ADTW-SS and connection games. In a backbone game with a single source group consider edges between the terminals of this group. No player includes them into her best deviations. Furthermore, they do not appear in the optimum forest $\mathcal{T}^*$. Hence, we can construct an equivalent SBG with a source vertex by introducing and contracting edges between all terminals of the source group. Thus, the previous results also extend to this case.

Chapter 6

Extensions

The model of investment games offers a number of interesting extensions. Of particular practical interest is the following *wholesale* variant, in which resource cost increases due to the number of players that use the resource for constraint satisfaction. This cost increase allows to specify the extent to which players can deviate and use resources bought by other players for free. This is the key property that causes a high price of anarchy in the investment game.

A strategy in a wholesale investment game is again a function s_p specifying the payments as before. For each resource $r \in R$ there is a *fixed cost* $c(r)$. We consider only single unit resources. In regular investment games a resource is available to all players for constraint satisfaction if the fixed costs are paid for. In the wholesale game we instead assume that a resource r is available to a set of players Q if the sum of payments by these players exceed a *bundle cost* $c(r, |Q|)$. The bundle costs are specified by $c(r, i) = \mathsf{cap}(i)c(r)$ using the main ingredient, the function cap. This function captures how much capacity must be installed for a certain number of players, or how large the resource units must be that a certain number of players can jointly use them. Naturally, cap is assumed to be non-decreasing, hence a larger set Q results in more capacity and cost. However, cap is assumed to be concave, which results in an *economy of scale* with a wholesale aspect: the more players strive to make r available for constraint satisfaction, the smaller is the cost increase for an additional player. This is a standard modeling assumption in economics (see e.g. [Man03, pp. 281f]), and it has been used by computer scientists to study e.g. network design problems with capacity cost [AA97, And04, CHKMS06]. Again, the foremost interest of a player is to pick s_p such that the resources available to her suffice to satisfy her constraint. If there are several strategies that would do so, she picks the one that minimizes her total investment represented by s_p. A formal definition can be given as follows.

Definition 6.1 *A* wholesale investment game $([k], R, \mathcal{S}, \mathtt{util}, c, \mathtt{cap}, \mathtt{valid})$ *is given by*

- *a set* $[k] = \{1, \ldots, k\}$ *of* k *players, and a set of resources* R*,*
- *the state space* $\mathcal{S} = \mathcal{S}_1 \times \ldots \times \mathcal{S}_k$*, and for each player* $p \in [k]$ *a set of* strategies $\mathcal{S}_p$*, for which a strategy is a function* $s_p : R \to \mathbb{R}_{\geq 0}$*,*
- *the* utility functions $\mathtt{util}_p$ *defined as*

$$\mathtt{util}_p(s_p, s_{-p}) = \begin{cases} -|s_p|, & \textit{if } \mathtt{bought}(s,p) \in \mathtt{valid}_p \\ -\infty & \textit{otherwise} \end{cases}$$

  with $|s_p| = \sum_{r \in R} s_p(r)$*, and*

- $\mathtt{cap} : \mathbb{N} \to \mathbb{R}$ *is a non-decreasing, concave function with* $\mathtt{cap}(0) = 0$ *and* $\mathtt{cap}(1) = 1$*,*
- $c : R \to \mathbb{R}_{\geq 0}$ *specifies for each resource* $r \in R$ *a non-negative cost, and* $c : R \times \mathbb{N} \to \mathbb{R}$ *yields the adjusted cost* $c(r, i) = \mathtt{cap}(i) c(r)$*,*
- $\mathtt{bought} : \mathcal{S} \times [k] \to 2^R$ *specifies for each state* s *and each player* p *the set of resources available to the player*

$$\mathtt{bought}_r(s,p) = \max_{Q \subseteq [k]-p} \left\lfloor \frac{s_p + \sum_{q \in Q} s_q(r)}{c(r, |Q|+1)} \right\rfloor \quad \textit{for each resource } r \in R,$$

  and

- $\mathtt{valid}_p \subseteq 2^R$ *is a constraint function for player* $p \in [k]$*.*

Note that we assume that a resource is available to player p if there is *any* subset of players, with which p can jointly purchase the bundle cost. For every NE and every social optimum solution, the concavity of $\mathtt{cap}$ ensures that for each resource r there is always a unique maximal set of players to which r is available. We denote this set by Q_r

For the problem of optimizing the social cost in regular investment games one can find the optimum solution to the underlying optimization problem, i.e. a set $\mathcal{R}$ of items that satisfies the constraints $\mathtt{valid}_p$ at minimum total cost. Instead, for the wholesale game it is necessary to fix the sets of players using a resource. Hence, in the wholesale game $\mathcal{R} = (Q_r)_{r \in R}$ is the vector of maximal player subsets Q_r. This specifies the set of resources available to each player and suffices to determine feasibility and total bundle cost.

Note that by concavity and monotonicity $\mathsf{cap}(k) \in [1, k]$ for all $k \geq 1$. In the case $\mathsf{cap}(k) = 1$ for all $k \geq 1$, the game becomes the original investment game, as there is no cost increase due to the availability to more than one player. In case $\mathsf{cap}(k) = k$ the game exhibits a decomposition property that allows no suboptimal NE. For intermediate functions cap we consider the price of anarchy. Let $\mathcal{R}^*$ be the solution purchased by a social optimum state s^*, and $\mathcal{R}^*_p$ a the set of resources available to p. W.l.o.g. we assume that $\mathcal{R}^*_p$ is minimum in the sense that no resource can be removed without hurting feasibility of valid_p. We extend the cost function c to the vector $c(\mathcal{R}) = \sum_{r \in R} \mathsf{cap}(|Q_r|)c(v)$ and note (c.f. Equation (3.1))

$$\mathsf{cost}(s^*) = -\sum_{p=1}^{k} \mathsf{util}_p(s^*_p, s^*_{-p}) = \sum_{p=1}^{k} |s^*_p| = c(\mathcal{R}^*).$$

The price of anarchy can be bounded as follows.

Theorem 6.1 *The price of anarchy in the wholesale investment game is exactly* $k/\mathsf{cap}(k)$.

Proof. First, we prove the lower bound. Consider a vertex cover game with a star network, in which every player owns a single edge and each vertex v has cost $c(v) = 1$. If every player purchases the leaf node incident to her edge, a NE of cost k evolves. The optimum solution, however, consists of the center vertex v and has cost $\mathsf{cap}(k)c(v) = \mathsf{cap}(k)$. This proves that the price of anarchy is at least $k/\mathsf{cap}(k)$.

For the upper bound consider any NE s. In addition, let $\mathcal{R}^-_p$ be a minimum cost cover for player p in case she is alone in the game. In general it could be $\mathcal{R}^*_p \neq \mathcal{R}^-_p$, because $\mathcal{R}^*_p$ can be cheaper for p due to the presence of other players. Thus,

$$\sum_{r \in \mathcal{R}^-_p} c(r) \leq \sum_{r \in \mathcal{R}^*_p} c(r). \tag{6.1}$$

The concavity of cap ensures that with other players purchasing parts of $\mathcal{R}^-_p$ it becomes even more attractive for p as deviation, because p must only purchase the additional cost increase. As s is a NE, the cost of $\mathcal{R}^-_p$ must therefore be an upper bound on the contribution of p to the cost of $\mathcal{R}_p$:

$$\sum_{r \in \mathcal{R}_p} s_p(r) \leq \sum_{r \in \mathcal{R}^-_p} c(r).$$

Since s is a NE, all bundle costs of the purchased resources must be fully paid for. Using the bound from (6.1) we get

$$\sum_{p=1}^{k} \sum_{r \in \mathcal{R}_p} s_p(r) \leq \sum_{p=1}^{k} \sum_{r \in \mathcal{R}^-_p} c(r) \leq \sum_{p=1}^{k} \sum_{r \in \mathcal{R}^*_p} c(r). \tag{6.2}$$

For convenience we define function $\Delta^{cap}(i) = \mathsf{cap}(i) - \mathsf{cap}(i-1)$. Consider the following procedure of constructing a lower bound on the cost of the social optimum solution. Iteratively add players and the cost of their choices $\mathcal{R}_p^*$ to the solution. The presence of the i-th player on her cover $\mathcal{R}_i^*$ adds at least a cost $\Delta^{cap}(i)\sum_{r\in\mathcal{R}_i^*} c(r)$ to the cost of $\mathcal{R}^*$. As cap is concave, Δ^{cap} is monotonic decreasing. Hence, we can lower bound $\mathsf{cost}(s^*) = c(\mathcal{R}^*)$ by

$$\sum_{i=1}^{k} \Delta^{cap}(i) \sum_{r\in\mathcal{R}_i^*} c(r) \leq c(\mathcal{R}^*). \tag{6.3}$$

We now choose k orderings, in which to construct the social optimum solution, such that each player appears in each position exactly once (e.g. by making k cyclic rotations of the initial ordering of players). Then for each ordering the above inequality (6.3) holds. By summing all these inequalities we get

$$\sum_{p=1}^{k}\sum_{i=1}^{k} \Delta^{cap}(i) \sum_{r\in\mathcal{R}_p^*} c(r) = \mathsf{cap}(k) \sum_{p=1}^{k}\sum_{r\in\mathcal{R}_p^*} c(r) \leq kc(\mathcal{R}^*).$$

Together with (6.2) and $\mathsf{cost}(s^*) = c(\mathcal{R}^*)$ this yields

$$\mathsf{cost}(s) = \sum_{p=1}^{k}\sum_{r\in\mathcal{R}_p} s_p(r) \leq \sum_{p=1}^{k}\sum_{r\in\mathcal{R}_p^*} c(r) \leq \frac{k}{\mathsf{cap}(k)}\mathsf{cost}(s^*),$$

which proves the theorem. □

The theorem implies that once $\mathsf{cap}(k) = k$ every NE is socially optimal. In fact, a similar argumentation shows that in this case there is always an optimal NE. On the other hand, we have very recently obtained the following results [Hoe07a], which extensively use the insights of Chapter 4. The same properties are conjectured to hold for all games considered in the previous chapters.

Theorem 6.2 *For wholesale covering games the following results hold.*

- *The price of stability in the wholesale vertex cover game is* $\Theta\left(\frac{k}{\mathsf{cap}(k)}\right)$.
- *For every* cap *with* $\mathsf{cap}(k) < k$*, there is a class of vertex cover games with* k *players such that it is* NP*-hard to decide whether a game has a NE.*
- *An adaptation of Algorithm 1 computes* (f, f)*-approximate NE for any wholesale set cover game.*

- *For singleton wholesale set multi-cover games the price of stability is 1, and* $(1+\epsilon, \beta)$*-approximate NE can be obtained in polynomial time from any* β*-approximate solution to the underlying optimization problem. In addition, an exact NE can be computed in polynomial time.*

The consideration of economies of scale generates a trade-off for the optimality of NE. However, recall from the lower bound construction for the price of anarchy that the structure of bad NE is not disrupted. Instead, the improvement results from an increase in the optimum solution cost due to the increase in resource cost. If $\mathsf{cap}(k) = k$, a social optimum solution is a combination of personal optimum solutions for the players. In general, the cost increase tends to align personal objectives with social objectives by reducing the possibility that myopic strategy switches (negatively) influence other players preferences.

We only considered single unit resources in this model, because the amount offered to a resource r is used to determine availability and the player set Q_r. If the amount is used for both determination of the number of units bought and the availability to players, then there are disambiguities, in which a larger number of units are available to a smaller set of players and less units to a larger set of players. It is an interesting open problem how to formulate a sensible adaptation to such a scenario.

Part II

Clustering Games

Chapter 7

Formal Framework

Computing and characterizing a meaningful partition of the vertices of a graph is a general problem, which has long been studied in graph theory and computer science. Recently, there have been many new variants and applications in various domains such as social networks [GD03, GN04], biological networks [PDFV05, GA05], and the Internet [FLGC02, AA03]. In addition, there have been efforts to understand the effects of social relationships as a game or as a parameter for economic decision making [DJ03, Now06]. In the second part of the thesis we study games for graph clustering as a model, in which selfish agents group themselves into clusters or communities in a given networked environment. The underlying problem of graph clustering is to find for a simple undirected graph G a clustering $\mathcal{C}^*$ maximizing a revenue function (called clustering *index*). The index captures preferences and trade-offs for evaluating the quality of a given partition and thus can be quite different based on the application. The optimization is sometimes subject to further restrictions, e.g. there might be an exact requirement or an upper bound on the number of clusters.

In our clustering games each player is considered to be a vertex in a graph $\mathsf{G} = (\mathsf{V}, \mathsf{E})$. So in all our games $\mathsf{V} = [k]$, and thus $n = k$. Therefore we use the same notation for a player as for her vertex in G. She plays a strategic two player game with every other player. We refer to these games as *bilateral* games. The payoffs of the bilateral games depend on how the players are located within the graph. The strategy a player picks is assumed to be her cluster, and thus she can choose it only once and must stick to it in all bilateral games she plays. A state s corresponds to a clustering $\mathcal{C}$ of G. We exploit that some natural and recently popular clustering indices can be interpreted as $\mathsf{welfare}(s)$ for clustering games with natural and intuitive payoffs.

All games considered in this part are potential games, hence NE represent an intuitive equilibrium concept with guaranteed existence. The following chapters will focus on characterizing NE. Mixed NE in these games can also be a reasonable concept when probabilistic clustering models are considered, in which each vertex is allowed to partially belong to several clusters. This is an interesting direction for future work.

Clustering games are special classes of polymatrix games. Polymatrix games were proposed by Yanovskaya [Yan68] and are a well-studied class of games. Howson [How72] proposed a recursive algorithm to find a mixed NE. This algorithm was later refined and extended to more general games [Eav73, MZ91]. Characterizations for the structure of the set of mixed NE were provided by Ouintas [Qui89]. Recently, Papadimitriou showed how to compute a correlated equilibrium [Aum74] in succinctly represented polymatrix games in polynomial time, while computing the best correlated equilibrium with respect to sum of player utilities is NP-hard [Pap05].

Directly related to clustering games are *graphical games* [KLS01], a different prominent class of strategic games associated with graphs. Similar to clustering games, each player is represented by a vertex in a graph. However, this graph serves to specify the influence of utility changes. In contrast to clustering games, the utility of a player in graphical games depends only on the strategy choices of her neighbors. While a correlated equilibrium can be computed in polynomial time [Pap05], computing the best correlated equilibrium with respect to sum of player utilities is NP-hard [PR05]. Elkind et al. [EGG06] showed that if the game is based on a 3-regular graph, it can be PPAD-complete [Pap94] to find a mixed NE, however, for paths it can be done in polynomial time. Very recently, Elkind et al. [EGG07] provided ideas to obtain mixed NE with maximum social welfare for certain subclasses of graphical games in polynomial time.

Graphical representation of influences in games is, in fact, a popular idea in recent research. Some other notions and frameworks exploiting this idea are for instance *multi-agent influence diagrams* [KM03], for which games with a graph specifying utility influence can be constructed. Another concept are *local-effect* [LBT03] and *action graph games* [BLB04], in which there is a graph with a vertex for each strategy of each player. The edges of the graph again represent a mutual influence on utility values. In contrast to the results in this thesis, however, none of these works connects and applies game-theoretic ideas to graph clustering.

Definition and Initial Observations

The clustering games we consider in this part are a subclass of *polymatrix games.* A polymatrix game [Yan68] can be defined as follows.

Definition 7.1 *A* polymatrix game *is a strategic game* $([k], \mathcal{S}, \mathrm{util}, \Gamma^{pq})$*, which is given by*

- *a set* $[k] = \{1, \ldots, k\}$ *of* k *players,*
- *the state space* $\mathcal{S} = \mathcal{S}_1 \times \ldots \times \mathcal{S}_k$*, and for each player* $p \in [k]$ *a set of* strategies $\mathcal{S}_p$*,*
- *for every set* $\{p, q\} \subseteq [k]$ *of players a strategic game* $\Gamma^{pq} = (\{p, q\}, \mathcal{S}_p \times \mathcal{S}_q, \mathrm{util}^{pq})$*, and*
- *for each player* $p \in [k]$ *a utility function*

$$\mathrm{util}_p(s_p, s_{-p}) = \sum_{q \neq p} \mathrm{util}_p^{pq}(s_p, s_q).$$

In a polymatrix game each pair of players $\{p, q\}$ plays a bilateral game Γ^{pq}. The strategy for a player, however, is the same in all bilateral games she plays. The utility for a strategy choice is the sum of utilities she obtains in the bilateral games. For all games considered in this part of the thesis we use the social welfare function $\mathrm{welfare}$ to measure the quality of states. Furthermore, for all games considered in this part we use potential games with exact potentials as bilateral games. This implies that the corresponding polymatrix games are exact potential games as the following simple observation shows:

Consider the game Γ^{pq} and its exact potential function Φ^{pq}. W.l.o.g. we scale the potentials Φ^{pq} such that the corresponding constants are $w_p = w_q = 1$ in all potentials. Since the utility util_p is the sum of utilities util_p^{pq}, the potential Φ for the polymatrix game can be constructed as the sum of potentials for the bilateral games,

$$\Phi(s) = \sum_{p \in [k]} \sum_{q \in [k], q > p} \Phi^{pq}(s).$$

Note that the utility change for a single player equals the sum of utility changes in her bilateral games, which are given by the respective differences of the bilateral potentials. If a player now switches her strategy, these potentials are the only ones that change their value and contribute exactly this difference to $\Phi(s)$. Thus, $\Phi(s)$ correctly captures the utility change for the strategy switch of a single player.

Observation 7.1 *If all games* Γ^{pq} *are exact potential games, then the polymatrix game* Γ *is an exact potential game and has at least one NE.*

As social value function we use $\mathrm{welfare}(s) = \sum_{p \in [k]} \mathrm{util}_p(s)$. A simple condition under which optimal NE exist in a polymatrix game is as follows. Suppose for each game Γ^{pq} each state yields payoff values, which are equal for both players. In this case social welfare becomes a potential for the game. Hence, the NE are exactly the local optima of $\mathsf{welfare}$ with respect to strategy changes of single players. In particular, the state maximum welfare is a NE. We refer to this subclass of games as *welfare optimizing* games.

Definition 7.2 *A polymatrix game is* welfare optimizing *if each state of every game* Γ^{pq} *yields the same payoff for both players* p *and* q.

Note that even for the special case of symmetric (2×2) bilateral games, in which the game becomes a potential game [Blu93, You98], a polymatrix game can have bad properties. In particular, prices of anarchy and stability might not be reasonable measures, because payoffs can have mixed signs, or unbounded as in the case of bilateral Prisoner's Dilemma games. For welfare optimizing games with $\mathsf{welfare}(s^*) \geq 0$, however, the price of stability is 1, and this will be the case for almost all clustering games considered in the following chapters. In terms of complexity, recall the MaxCut game presented in Section 2.3. This game can be cast as a welfare optimizing polymatrix game: set the payoffs of the bilateral game for an edge $e = \{u, v\}$ to $c(e)$ for both u and v when they play different strategies and 0 for both otherwise. Bilateral games for unconnected players have all payoffs 0. Using this adjustment it is obvious that in a succinctly represented polymatrix game finding a NE is PLS-hard, and finding a NE with optimal $\mathsf{welfare}$ is NP-hard.

Chapter 8

Simple Clustering Games

In this chapter we consider a basic model of clustering games. A *simple clustering game* is a polymatrix game, which has only two types of bilateral games: one for connected players and one for unconnected players. Furthermore, as the graph is undirected and vertices (i.e. players) carry no further attributes, it is natural to assume that the bilateral games are symmetric.

Definition 8.1 *A* simple clustering game *is a polymatrix game, in which players, strategies and bilateral games are specified by*

- *an undirected graph* $G = (V, E)$ *with the set of players* V,
- *the same set of strategies* $\mathcal{S}_v$ *for every player* $v \in V$,
- *two symmetric games* $\Gamma^c = (\{u, v\}, \mathcal{S}_u \times \mathcal{S}_v, \mathtt{util}^c)$ *and* $\Gamma^d = (\{u, v\}, \mathcal{S}_u \times \mathcal{S}_v, \mathtt{util}^d)$ *for two players, and*
- *for every* $\{u, v\} \subseteq V$ *a bilateral game* $\Gamma^{uv} = \Gamma^c$ *if* $\{u, v\} \in E$, *and* $\Gamma^{uv} = \Gamma^d$ *otherwise.*

For a succinct representation it is sufficient to encode the graph G and the payoff values for the games Γ^c and Γ^d.

The rest of this chapter presents results on several special cases of simple clustering games. In the next Section 8.1 we outline related work on the class of games we are considering. Section 8.2 treats games based on optimizing unweighted MAX-l-CUT and MAXAGREE problems. In both games finding a NE is possible in polynomial time, but finding optimal NE is NP-hard. However, while in Max-l-Cut games the price of anarchy drops with the number of possible strategies, in MaxAgree games it can stay $\Theta(\sqrt{n})$ for structural reasons (Theorem 8.1). In Section 8.3 we extend MaxCut and MaxAgree games with two strategies to clustering games with symmetric (2×2) bilateral games. As relative performance measures like the price of anarchy are unbounded or not well-defined for the general class of games, we concentrate on computational properties. In particular, such clustering games always allow finding a NE in polynomial time (Theorem 8.2). For a subclass of neighborhood independent games even the social optimum and the best/worst NE can be computed in polynomial time (Theorems 8.3 and 8.4). Finally, Section 8.4 provides some interesting directions for future work.

8.1 Previous and Related Work

Our model of a simple clustering game fits into a recent stream of works that study polymatrix games with (2×2) bilateral games with respect to an underlying graphical relationship of players. It is a generalization of a game considered by Bramoullé [Bra07], which concentrates on the subclass of symmetric (2×2) anti-coordination games. In these games the NE are the states in which players choose different strategies. Furthermore, there is no game played by unconnected players, and the analysis in [Bra07] considers properties and structure of NE in some particular classes of graphs.

While in our model the graph is fixed and specified in advance, there are several works on polymatrix games, in which the graph is endogenous. In particular, (2×2) anti-coordination games on endogenous graphs were studied in [BLPGVR04]. Much more work, see for instance [Blu93, Ell93, You98, BD01], has been done for network creation and (2×2) coordination games, in which the NE are the states (1,1) and (2,2). In contrast to our model, all these games allow only connected players to play a bilateral game. In addition, there has been no focus on social welfare and computation of NE and optimum states. Instead, properties of the network structure and payoff properties in NE were analyzed [BV06], or stochastically stable states were characterized [JW02, GVR05], which are observed frequently in an infinite random process of perturbed best response iteration.

MaxCut is a standard problem in graph theory and computer science and has been studied for decades. The MaxCut game has been used as an introductory example in Fabrikant et al. [FLM$^+$03]. It was studied by Christodoulou et al. [CMS06] in terms of convergence time to NE and social welfare of states obtained after a polynomial number of best response steps. They also considered MaxAgree games and studied similar aspects.

8.2 MaxCut and MaxAgree Games

Consider the following simple game in which players are actors in a conflict network, and each one has to pick one of two teams she wants to join. Her objective is to minimize the conflicts within her team, or equivalently, to maximize the number of neighbors in the opposite team. Formally, we can cast this as a simple clustering game of Definition 8.1 with the symmetric (2×2) bilateral games depicted in Figure 8.1.

Γ^c	1	2
1	0,0	1,1
2	1,1	0,0

(a) Connected Players

Γ^d	1	2
1	0,0	0,0
2	0,0	0,0

(b) Unconnected Players

Figure 8.1: Payoffs in the MaxCut game.

This game is the unweighted version of the MaxCut game we outlined in Section 2.3 and Chapter 7. The following set of results is well-known and easy to derive.

Lemma 8.1 *For the unweighted MaxCut game we have the following:*

- *The game is welfare optimizing, hence the price of stability is* 1*, and any NE represents a local optimum under the* FLIP *neighborhood (c.f. Section 2.3).*
- *The price of anarchy is exactly* 2*.*
- *Finding an optimal NE is* NP*-hard.*
- *Every best response iteration can take at most* $m \leq \frac{n(n-1)}{2}$ *steps before reaching a NE.*

Thus, an arbitrary NE can be obtained in polynomial time, but finding an optimal one is NP-hard.

For the bound on the price of anarchy note that the value of 2 is due to the restriction to two strategies. In fact, as each player plays a best response, she always avoids the cluster with the larger part of her neighbors. Therefore, for each player in NE at least half of her incident edges are counted towards welfare. In the optimum solution, however, at most the total degree is counted towards welfare, and thus any NE must have an approximation ratio of no worse than 2. This ratio is tight in a game on a cycle graph with four players.

Now suppose the game is played with $l \geq 2$ strategies. Connected players still receive a payoff of 1 each when choosing different strategies. This is equivalent to the unweighted MAX-l-CUT problem in graphs. Note that all results of Lemma 8.1

transfer to this game. For the price of anarchy, however, the following lemma provides more specific bounds.

Lemma 8.2 *The price of anarchy in the Max-l-Cut game is at most $1 + \frac{1}{l-1}$. This bound is tight for $l \leq \frac{n}{2}$.*

Proof. Note that in every NE of this game the players play the maximum number of possible different strategies. For the upper bound consider the decision each player faces. In a NE s she picks a strategy for which she has the least number of neighbors playing it. Thus, each player drops at most a fraction of $\frac{1}{l}$ of her incident edges out of the cut. Hence, we have $\mathsf{welfare}(s) \geq 2m\left(1 - \frac{1}{l}\right) = \frac{2m(l-1)}{l}$. With $\mathsf{welfare}(\mathcal{S}^*) \leq 2m$ the bound follows.

For a lower bound we consider a complete bipartite graph $K_{l,l}$. For a state s, in which players form l clusters of two connected players each, we can verify that it represents a NE s with $\mathsf{welfare}(s) = 2l(l-1)$ as there are $l(l-1)$ edges in the cut. Otherwise, in the optimum solution s^* we assign l—2 arbitrary players to play strategies $3, \ldots, l$. The remaining ones are assigned to play strategies 1 and 2 depending on the bipartition they are located in. This state achieves $\mathsf{welfare}(s) = 2l^2 = 2m$. This gives a lower bound of $\frac{l^2}{l(l-1)} = 1 + \frac{1}{l-1}$ on the price of anarchy. □

With an increasing number of possible strategies the price of anarchy drops. The upper bound is valid even if the number of players is allowed to grow simultaneously. It is tight for complete bipartite graphs of sufficient size. For $l > \frac{n}{2}$ the price of anarchy is likely to drop even faster than the upper bound suggests. In particular, for $l = n$ strategies the price of anarchy is 1.

Now consider the case, in which a player picks her strategy not only to be unconnected to players with a different strategy, but also to be connected to players with the same strategy. We call this the MaxAgree game, as the problem of maximizing social welfare is the corresponding version of the MaxAgree problem for correlation clustering [BBC04, GG06]. The payoffs for the MaxAgree game with $l = 2$ strategies are given in Figure 8.2, the extension to more strategies is straightforward.

For the MaxAgree game with any number of possible strategies it is easy to observe the following results.

Lemma 8.3 *For the MaxAgree game we have the following:*

- *The game is welfare optimizing, hence $\mathsf{welfare}$ is an exact potential, NE are local optima of $\mathsf{welfare}$, and the price of stability is 1.*

- *Finding an optimal NE is NP-hard [GG06].*

Γ^d	1	2
1	1,1	0,0
2	0,0	1,1

(a) Connected Players

Γ^c	1	2
1	0,0	1,1
2	1,1	0,0

(b) Unconnected Players

Figure 8.2: Payoffs in the MaxAgree game.

- *Every best response iteration can take at most* $\frac{n(n-1)}{2}$ *steps before reaching a NE.*

Obviously, for MaxAgree and Max-l-Cut games there are some identical properties. However, the price of anarchy behaves quite differently.

Theorem 8.1 *For the MaxAgree game the price of anarchy is at most* $\min(l, n)$. *It is* $\Omega(l)$ *for* $l \in O(\sqrt{n})$, *and at least* $\sqrt{n}/2$ *for* $l \geq \sqrt{n}$.

Proof. In a NE s for a game with l strategies, each player picks the one that maximizes her utility. Each of the strategies allows her to count a certain number of existing edges and non-existing edges towards her payoff. In any case, however, there is always one strategy that yields a payoff of at least $\mathsf{util}_u(s) \geq \frac{n-1}{l}$. Summing this for all players we get $\mathsf{welfare}(s) \geq \frac{1}{l}(n(n-1))$. The upper bound follows with $\mathsf{welfare}(s^*) \leq n(n-1)$ and the observation that no more than n different strategies can be played in any state.

For a lower bound with constant l we construct a graph for any $x \in \mathbb{N}$, $x \geq l$ as follows. We introduce l cliques of x vertices each. In addition, each vertex has exactly $x - 1$ connections into each of the $l - 1$ other clusters. Hence, the graph is $l(x - 1)$-regular. An example for $x = l = 4$ is shown in Figure 8.3. A bad NE s can be given as follows. For each clique let all players play the same strategy, which is different from the one of all other cliques. Then each player gets a payoff of $x - 1$ from the connections within the clique and a payoff of $l - 1$ for the remaining unconnected players in different cliques (see Figure 8.3(b)). If she changes to a strategy of a different clique, her payoff is $x - 1$ for the connections to the new clique. But within this clique there is one player that is not a neighbor. Thus, she gets only $l - 2$ for the remaining unconnected players with different strategies. This yields $x+l-3 < x+l-2$, and so the state s is a NE. Note that $\mathsf{welfare}(s) = xl(x+l-2)$, but the social optimum has at least $\mathsf{welfare}(s^*) \geq xl(l(x-1))$ for the state in which all players play the same strategy. We get a price of anarchy of $\frac{xl}{x+l-2}$. We have $n = xl$,

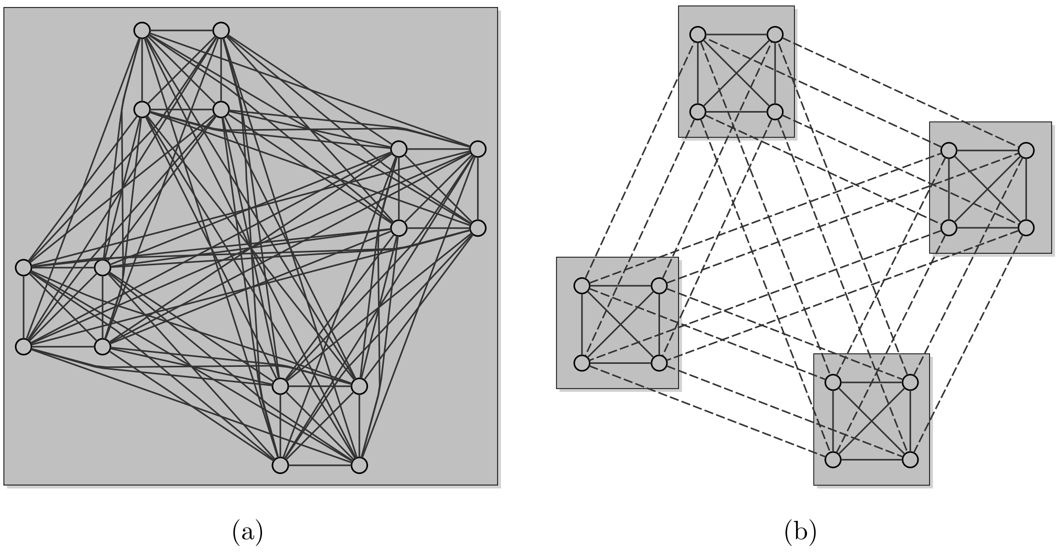


Figure 8.3: A MaxAgree game with a bad NE. (a) Optimum solution s^*; (b) a bad NE s. $\mathit{welfare}$ counts twice the displayed existing (solid) and non-existing edges (dashed).

hence if we obey $l = O(\sqrt{n})$, then for growing n we get $x = \Omega(\sqrt{n})$. In this case the price of anarchy asymptotically approaches l for growing n.

For $l \geq \sqrt{n}$ this construction yields a lower bound of $\sqrt{n}/2$ with $x \in \mathbb{N}$ cliques of x vertices each. Then $x = \sqrt{n} \leq l$, so s might not be a NE if a player can profit by picking a totally new strategy. Her payoff for such a choice, however, would be $x - 1 < 2x - 2$, hence s remains a NE. The same analysis as above delivers the bound of $\sqrt{n}/2$. $\square$

Best response iteration is equivalent to a local search algorithm in Max-l-Cut and MaxAgree games. Therefore, it is possible to obtain near-optimal NE in polynomial time. First obtain an initial state by using an approximation algorithm, and then allow players in a best response iteration to converge to a NE. This process can take only $\frac{n(n-1)}{2}$ iterations and can only improve the approximation ratio. For Max-l-Cut there are approximation algorithms based on semidefinite programming with performance ratio $1/\left(1 - \frac{1}{l} + 2\frac{\ln l}{l^2}\right)$ [FJ97, MR99]. For MaxAgree there is a PTAS yielding a $(1 + \epsilon)$-approximation in polynomial time, for any constant $\epsilon > 0$ [BBC04, GG06]. Using these algorithms one can obtain exact NE with the same approximation ratio in polynomial time. While for Max-l-Cut games this improves only slightly over the price of anarchy, the improvement for MaxAgree games is significant. In MaxAgree games uncoordinated dynamics can end up in

NE that are much worse than states obtained by centralized optimization.

8.3 Clustering Games with Two Strategies

Let us consider the class of simple clustering games with symmetric (2×2) bilateral games. We denote these games for short as *2-clustering games*. The notation for payoffs of the bilateral games is given in Figure 8.4.

Γ^c	1	2
1	a^c, a^c	d^c, c^c
2	c^c, d^c	b^c, b^c

(a) Connected Players

Γ^d	1	2
1	a^d, a^d	d^d, c^d
2	c^d, d^d	b^d, b^d

(b) Unconnected Players

Figure 8.4: Payoffs in 2-clustering games.

In order to derive expressions for welfare and potential functions for this game, it is possible to simplify the game by subtracting c^c and c^d from every entry of the connected and unconnected edge game, respectively. This does not alter the payoff differences for the players and preserves the incentives. In addition, the welfare of every state reduces by exactly $2m(c^c - c^d) + n(n-1)c^d$. Thus, w.l.o.g. we assume $c^c = c^d = 0$.

It is well-known [Blu93, You98] that every symmetric (2×2) game is a potential game. Assuming $c^c = c^d = 0$, the potentials Φ^c and Φ^d can be given by

$$\Phi^c = \begin{pmatrix} a^c & 0 \\ 0 & b^c - d^c \end{pmatrix} \qquad \Phi^d = \begin{pmatrix} a^d & 0 \\ 0 & b^d - d^d \end{pmatrix}. \tag{8.1}$$

For a state s the set of players playing strategy 1 is denoted V_1, their number $n_1 = |V_1|$, and for a player v the number $n_1(v) = |N(v) \cap V_1|$ denotes the number of neighbors playing 1. V_2, n_2 and $n_2(v)$ are defined analogously for strategy 2. We consider the size of the cut of a state s, which is the number of edges connecting players that play different strategies, and denote this number by m_{12}.

8.3.1 Welfare and Potential

If $d^c = d^d = 0$, then the game becomes welfare optimizing. If a 2-clustering game is not welfare optimizing, we can use two transformations into welfare optimizing games either with equivalent welfare or with equivalent potential. In particular, we outline transformations into a game of the type in Figure. 8.5.

Γ^c	1	2
1	A^c, A^c	0,0
2	0,0	B^c, B^c

(a) Connected Players

Γ^d	1	2
1	A^d, A^d	0,0
2	0,0	B^d, B^d

(b) Unconnected Players

Figure 8.5: Payoffs in transformed welfare optimizing games.

For $\mathit{welfare}$ it does not matter how the payoff of a bilateral game is distributed among the participating players. Hence, we can as well assume d^c in Γ^c is given in equal parts to both players when they play different strategies. Then, using previous arguments we can reduce every entry in the game by $\frac{d^c}{2}$. This removes md^c from the welfare of every state, but leaves the welfare differences of states unaltered. The same argument holds for Γ^d and d^d. With

$$A^c = a^c - \frac{d^c}{2} \qquad B^c = b^c - \frac{d^c}{2} \qquad A^d = a^d - \frac{d^d}{2} \qquad B^d = b^d - \frac{d^d}{2}$$

we get a welfare optimizing game Γ_w of the form of Figure 8.5 with an equivalent welfare function. This game is not equivalent in terms of incentives, as we change the payoff differences for a player in the states. Thus, it might have different NE.

Now consider the potentials in Equation (8.1). If we use

$$A^c = a^c \qquad B^c = b^c - d^c \qquad A^d = a^d \qquad B^d = b^d - d^d$$

the new welfare optimizing game exhibits the same potential for bilateral games, and hence the same potential as the original game. We get a welfare optimizing game Γ_p of the form of Figure 8.5 with the same potential function. This game is not equivalent in terms of social welfare, as we subtract different payoffs from different states. Thus, it might have a different state s^* of optimal welfare.

The appealing property in welfare optimizing games is that they allow to concentrate on one function describing both welfare and potential. Although by restriction to MaxCut games finding the best/worst NE for 2-clustering games is NP-hard, in some special cases it can be done in polynomial time. Let us analyze the underlying characteristic function of the welfare optimizing game of Figure 8.5 more closely.

We denote $\mathtt{S}^c = \mathtt{A}^c + \mathtt{B}^c$, $\mathtt{S}^d = \mathtt{A}^d + \mathtt{B}^d$ and $\mathtt{D} = \mathtt{A}^c - \mathtt{A}^d$. Potential and welfare functions of the welfare optimizing game in Figure 8.5 are

$$\begin{aligned}
\Phi(s) &= \mathit{welfare}(s) \\
&= \sum_{v\in V_1} n_1(v)\mathtt{A}^c + (n_1 - n_1(v) - 1)\mathtt{A}^d + \sum_{v\in V_2} n_2(v)\mathtt{B}^c + (n_2 - n_2(v) - 1)\mathtt{B}^d \\
&= n(n-1)\mathtt{B}^d + n_1^2\mathtt{S}^d - (2(n-1)\mathtt{B}^d + \mathtt{S}^d)n_1 + m_{12}(\mathtt{S}^d - \mathtt{S}^c) + \sum_{v\in V_1} \deg(v)\mathtt{D} \\
&\quad + \sum_{v\in V_2} \deg(v)(\mathtt{B}^c - \mathtt{B}^d) \\
&= n(n-1)\mathtt{B}^d + n_1^2\mathtt{S}^d - (2(n-1)\mathtt{B}^d + \mathtt{S}^d)n_1 + 2m(\mathtt{B}^c - \mathtt{B}^d) + (\mathtt{S}^d - \mathtt{S}^c)m_{12} \\
&\quad + \sum_{v\in V_1} \deg(v)(\mathtt{D} - \mathtt{B}^c + \mathtt{B}^d) \\
&= n(n-1)\mathtt{B}^d + 2m(\mathtt{B}^c - \mathtt{B}^d) + n_1^2\mathtt{S}^d - (2(n-1)\mathtt{B}^d + \mathtt{S}^d)n_1 + (\mathtt{S}^d - \mathtt{S}^c)m_{12} \\
&\quad +(\mathtt{S}^d - \mathtt{S}^c + 2\mathtt{D}) \sum_{v\in V_1} \deg(v)
\end{aligned}$$

It is possible to drop the constant terms $n(n-1)\mathtt{B}^d + 2m(\mathtt{B}^c - \mathtt{B}^d)$ from every state and derive a characteristic function $\Psi(s)$ given by

$$\Psi(s) = n_1^2\mathtt{S}^d - (2(n-1)\mathtt{B}^d + \mathtt{S}^d)n_1 + (\mathtt{S}^d - \mathtt{S}^c)m_{12} + (\mathtt{S}^d - \mathtt{S}^c + 2\mathtt{D}) \sum_{v\in V_1} \deg(v).$$

It captures both potential and welfare in welfare optimizing games. Hence, potential and welfare depend - in addition to the payoffs - only on three parameters of the game: the number n_1 of players playing strategy 1, their degrees $\sum_{v\in V_1} \deg(v)$ and the cut size m_{12}. Observe that the choice of payoffs for the MaxCut game in Figure 8.1 actually singles out the parameter m_{12} in $\Psi(s)$.

For each game that is not welfare optimizing, the results can be used to derive $\Psi_w(s)$ and $\Psi_p(s)$ characterizing welfare and potential. The functions result from plugging in the payoffs of the welfare optimizing games Γ_w and Γ_p into the parameters $\mathtt{A}^c$, $\mathtt{B}^c$, $\mathtt{A}^d$ and $\mathtt{B}^d$ as described above. In both cases $\mathtt{S}^c = \mathtt{a}^c + \mathtt{b}^c - \mathtt{d}^c$ and $\mathtt{S}^d = \mathtt{a}^d + \mathtt{b}^d - \mathtt{d}^d$. However, whereas $\mathtt{D}_p = \mathtt{a}^c - \mathtt{a}^d$, the value $\mathtt{D}_w = \mathtt{D}_p + \frac{\mathtt{d}^d - \mathtt{d}^c}{2}$. This yields an equivalent potential function $\Phi \equiv \Psi_p$ and an equivalent welfare function $\mathit{welfare} \equiv \Psi_w$ with

$$\begin{aligned}
\Psi_p(s) &= n_1^2\mathtt{S}^d - ((n-1)(2\mathtt{b}^d - 2\mathtt{d}^d) + \mathtt{S}^d)n_1 + (\mathtt{S}^d - \mathtt{S}^c)m_{12} \\
&\quad +(\mathtt{S}^d - \mathtt{S}^c + 2\mathtt{D}_p) \sum_{v\in V_1} \deg(v) \\
\Psi_w(s) &= \Psi_p(s) + (n-1)n_1\mathtt{d}^d + (\mathtt{d}^d - \mathtt{d}^c) \sum_{v\in V_1} \deg(v).
\end{aligned}$$

Now that we have a more insightful expression for potential and welfare at hand, the main result of this section is easy to see.

Theorem 8.2 *Finding a NE in a 2-clustering game can be done by best response iteration from an arbitrary starting state in* $O(nm^2)$ *iterations.*

Proof. Consider the potential Φ. The value of n_1 can range from 0 to n, which constitutes the factor n in our guarantee. Note that m_{12} and $\sum_{v \in V_1} \deg(v)$ can take at most $O(m)$ different values each. Hence, the total number of possible combinations for these parameters yields a total of $O(nm^2)$ different values for Φ. As a best response iteration must strictly increase Φ in each step, every such iteration takes $O(nm^2)$ steps to reach a local optimum. This proves the theorem. □

We define function Δ_v^{util}, which captures the incentive of a single player v to pick strategy 1 instead of 2 given the profile s_{-v}. In our formula the value $n_{1,-v}$ denotes the number of players other than v playing strategy 1.

$$\begin{aligned}
\Delta_v^{\mathrm{util}}(s_{-v}) &= \mathrm{util}_v(1, s_{-v}) - \mathrm{util}_v(2, s_{-v}) \\
&= \frac{1}{2}(\Phi(1, s_{-v}) - \Phi(2, s_{-v})) \\
&= S^d n_{1,-v} - (n-1)(b^d - d^d) + \frac{1}{2}(S^d - S^c)(n_2(v) - n_1(v)) \\
&\quad + \frac{1}{2}(S^d - S^c + 2D_p)\deg(v).
\end{aligned}$$

8.3.2 Neighborhood independency

By restriction to the MaxCut game it is NP-hard to obtain a NE optimizing $\mathrm{welfare}$ in 2-clustering games. In this section we show that the cut m_{12} is the key parameter that causes hardness of this problem. In particular, we consider *neighborhood independent* games defined as follows.

Definition 8.2 *A 2-clustering game is called* neighborhood independent *if* $S^c = S^d$.

Note that the definition does not postulate any requirements on the type of the involved games, i.e. the payoff structure of Γ^c and Γ^d can still be of any kind - e.g. as in (anti-)coordination games and/or Prisoner's Dilemma games. To provide a more intuitive understanding of the restriction consider the values $\Delta_v^{\mathrm{util}}(s_{-v})$, which are now given as

$$\Delta_v^{\mathrm{util}}(s_{-v}) = S^d n_{1,-v} - (n-1)(b^d - d^d) + D_p \deg(v).$$

The incentive to prefer strategy 1 for a player v depends only on $\deg(v)$ and the number $n_{1,-v}$ of other players playing 1. It is independent of the property, whether these players are direct neighbors of v or not. Hence, the strategy choices of players are independent of the neighborhood. Potential and welfare are cut independent and amount to

$$\begin{aligned}\Phi(s) &= S^d n_1^2 - ((n-1)(2b^d - 2d^d) + S^d)n_1 + 2D_p \sum_{v \in V_1} \deg(v) \\ \Psi_w(s) &= \Phi(s) + (n-1)d^d + (d^d - d^c) \sum_{v \in V_1} \deg(v).\end{aligned}$$

With this formulation we can argue that states of maximum welfare can be obtained in polynomial time.

Theorem 8.3 *States of maximum and minimum welfare in neighborhood independent games can be computed in* $O(n \log n)$ *time.*

Proof. If $2D_p + d^d - d^c = 0$, then the game Γ_p has the same bilateral games for existing and non-existing edges. Hence, in this game G can be assumed to be a complete or empty graph, in which the desired fraction of players playing 1 can be derived directly. As the welfare function Ψ_w depends only on n_1, either all players playing a single strategy is optimal, or we can use the derivative of Ψ_w to fractionally optimize and round n_1 directly to the optimal value n_1^*. Assigning any set of n_1^* players to 1 yields a state of maximum welfare. Finding the state of minimum welfare can be done similarly.

If $2D_p + d^d - d^c < 0$, then for any given number n_1 of players playing 1 the maximum welfare is obtained with $\sum_{v \in V_1} \deg(v)$ being minimal. Thus, to get a state of maximum welfare, we can simply start with $s_v = 2$ for all $v \in V$ and then iteratively switch players to 1 in non-decreasing order of their degrees. The state of maximum welfare encountered in this procedure is a social optimum. For the state of minimum welfare we can use the same procedure to switch players in non-increasing order of degrees. Ordering the players is the most time-consuming operation and yields the stated bound on the running time. If $2D_p + d^d - d^c > 0$, then all previous observations hold with inverted orderings. □

Note that for each game there are states of maximum and minimum welfare that exhibit a *degree separation property (DSP).*

Definition 8.3 *A state* s *of a 2-clustering game has the* degree separation property (DSP) *if it exhibits a threshold* y *such that either*

- $\deg(v) \leq y$ *if* $s_v = 1$ *and* $\deg(v) \geq y$ *otherwise, or*
- $\deg(v) \geq y$ *if* $s_v = 2$ *and* $\deg(v) \leq y$ *otherwise.*

Potential and welfare functions have the same structure, so the proof of Theorem 8.3 can be applied to show similar results for states of maximum and minimum potential.

Corollary 8.1 *There are states of maximum/minimum potential/welfare in neighborhood independent games that have the DSP and can be computed in* $O(n \log n)$ *time.*

This provides a characterization and a procedure to find states with optimal welfare or potential. The case for best and worst NE, however, is slightly more complicated, as in this case the welfare must be optimized over the local optima of the potential. The following lemma proves useful in later observations. It provides an interesting insight about the existence and potential values of NE with the DSP.

Lemma 8.4 *Suppose there is a NE* s *in a neighborhood independent game with a number of* n_1 *players playing* 1. *Then there is a NE* s' *with* n_1 *players playing* 1, *which has the DSP and* $\Phi(s') \geq \Phi(s)$.

Proof. For the proof we consider two cases. In the first case $D_p \geq 0$. Suppose s does not have the DSP. We consider the player $u \in V_2$ of largest degree and exchange her strategy choice with a player $v \in V_1$ of smaller degree. This is shown to yield a new NE s' with $\Phi(s') \geq \Phi(s)$. Therefore, iteratively a NE with the DSP and a larger potential value than s can be constructed. In particular, consider the a player $u \in V_2$ in s of largest degree. s does not have the DSP, so there is at least one player $v \in V_1$ with $\deg(v) < \deg(u)$. s is a NE and u plays 2, which means her incentive to switch to 1 is $\Delta_u^{util}(s_{-u}) \leq 0$. All players $w \in V_2$ have $\deg(w) \leq \deg(u)$ and $D_p \geq 0$. This yields $\Delta_w^{util}(s_{-u}) \leq \Delta_u^{util}(s_{-u}) \leq 0$. To create s', we exchange the strategy choices of players u and v. Observe in the definition of function Δ_w^{util} that if holding $n_{1,-w}$ fixed, we do not alter the strategy preferences of any of the players $w \in V - \{u, v\}$. In s' player u is assigned to play 1 along with all players of higher degree. Player $v \in V_2'$, who is assigned to play 2 in s', has lower degree. With $n'_{1,-v} = n_{1,-u}$ we know that $\Delta_v^{util}(s'_{-v}) \leq \Delta_u^{util}(s_{-u}) \leq 0$. Therefore, in s' player v sticks to 2. The case is similar for player $v \in V_1$ in s. She plays 1 in s, so the incentive to prefer 1 to 2 is $\Delta_v^{util}(s_{-v}) \geq 0$. Player $u \in V_2$ has larger degree, and as $n'_{1,-u} = n_{1,-v}$, we know that $\Delta_u^{util}(s'_{-u}) \geq \Delta_v^{util}(s_{-v}) \geq 0$. Hence, in s' player u sticks to 1. This proves that s' is a NE. The switch of u and v keeps n_1 fixed, but it increases $\sum_{v \in V_1} \deg(v)$. This yields $\Phi(s') \geq \Phi(s)$.

In the second case $D_p < 0$. It is possible to repeat the proof of the previous case with inverted orderings. In particular, pick player $u \in V_2$ to be of smallest, and player $v \in V_1$ of larger degree. Then the inverted degree ordering cancels out the negativity of D_p and yields an identical argumentation. This proves the lemma. □

The following theorem shows that finding optimal NE in neighborhood independent games can also be done in polynomial time.

Theorem 8.4 *A NE of maximum or minimum welfare in neighborhood independent games can be computed in* $O(n^2)$ *time.*

Proof. We consider two cases for the proof. For minimum welfare it is possible to apply the same arguments by exchanging cases and adjusting degree orderings. For convenience, we will adjust Δ_v^{util} to be a function taking as argument the number $n_{1,-v}$ of other players playing strategy 1.

In the first case $(2D_p + d^d - d^c)D_p \geq 0$. If $\sum_{v \in V_1} \deg(v)$ increases, then potential and welfare both increase or both decrease. Thus, for a fixed number n_1 of players with $s_v = 1$ the state with maximum potential also maximizes the welfare. Using Lemma 8.4 this is a state with the DSP, and this proves the theorem for this case.

In the second case $(2D_p + d^d - d^c)D_p < 0$. If $\sum_{v \in V_1} \deg(v)$ increases, then the potential increases and the welfare decreases, or vice versa. For a fixed number n_1 of players with $s_v = 1$ a state minimizing the potential maximizes the welfare. Instead of assigning strategy 1 to players with more favorable degrees as in the proof of Lemma 8.4, we must now assign strategy 1 to players with more unfavorable degrees. However, the potential must remain in a local maximum.

We fix an integer $x \in [0, n]$ and try to construct the best NE with a fixed number of $n_1 = x$ players for strategy 1. For a player v consider the value $\Delta_v^{util}(x-1)$ for any state which satisfies $n_{1,-v} = x - 1$ and the value $\Delta_v^{util}(x)$ for any state with $n_{1,-v} = x$. Clearly, for any NE with $n_1 = x$ all players v playing strategy 1 must have $\Delta_v^{util}(x-1) \geq 0$. In turn, every player v that satisfies this property is a candidate to play 1 in a NE with $n_1 = x$. Accordingly, every player with $\Delta_v^{util}(x) \leq 0$ is a candidate to play 2. If a player v has $\Delta_v^{util}(x-1) > 0$ and $\Delta_v^{util}(x) > 0$, she must play 1 in any NE with $n_1 = x$. We will say she is a *required candidate* for strategy 1. Similarly, if both values are strictly less than 0, she is a required candidate for strategy 2. Note that each player is either a candidate for both strategies, a required candidate for one strategy, or no candidate at all. Thus, if one of the following conditions occurs, there can be no NE with $n_1 = x$:

- A player is no candidate at all.
- There are strictly less than x candidates for strategy 1.
- There are strictly more than x required candidates for strategy 1.

Otherwise, there is at least one NE with $n_1 = x$. For the one with the best welfare we fix all required candidates to the corresponding strategies. If $D_p \geq 0$, the incentive to switch to 1 increases with increasing degree. In this case the required candidates for 1 are the players with highest degree (if any). If their number is smaller than x, we pick for the remaining players for strategy 1 the candidates for both strategies with *smallest* degree. This yields the NE with $n_1 = x$ and largest welfare, because in this case the welfare decreases with the sum of degrees. Similarly, if $\mathtt{D_p} < \mathtt{0}$, the required candidates for strategy 1 are the players of smallest degree. Here we choose for the remaining players the candidates for both strategies with *highest* degree. This shows that with an ordered representation of players we can obtain in linear time the best NE for a fixed number $\mathtt{x}$ of players playing strategy 1. The bound on the running time follows, and the theorem is proven. □

8.4 Open problems

In this chapter we have introduced the class of simple clustering games. We have scratched the surface by studying several interesting special cases, but naturally a lot of open problems remain. It would be interesting to adjust upper and lower bounds on the price of anarchy for MaxAgree games to provide a tight characterization of the efficiency of NE in this game. Another interesting question is whether the favorable properties observed for neighborhood independent games can be extended to a more general class of games. Finally, there are many other clustering indices for graphs (e.g. MinDisAgree [GG06]), which can form a natural basis for distributed clustering and affiliation decisions. They can be formulated and explored in a competitive environment within our framework.

Chapter 9

Modularity Games

In this chapter we consider welfare optimizing polymatrix games based on a clustering index called *modularity* [GN04], which is defined as follows. For a graph $G = (V, E)$ the modularity $q(\mathcal{C})$ of a clustering $\mathcal{C}$ is given by

$$q(\mathcal{C}) = \sum_{C \in \mathcal{C}} \left[\frac{|E(C)|}{m} - \left(\frac{|E(C)| + \sum_{C' \in \mathcal{C}} |E(C, C')|}{2m} \right)^2 \right] , \tag{9.1}$$

in which $E(C, C')$ denotes the set of edges between vertices in clusters C and C′, and $E(C) = E(C, C)$. Note that C′ ranges over all clusters, so that edges in $E(C)$ are counted twice in the squared expression. This is to adjust proportions, since edges in $E(C, C')$, $C \neq C'$, are counted twice as well, once for each order of the arguments. Rewriting Equation (9.1) into the more convenient form

$$q(\mathcal{C}) = \sum_{C \in \mathcal{C}} \left[\frac{|E(C)|}{m} - \left(\frac{\sum_{v \in C} \deg(v)}{2m} \right)^2 \right] \tag{9.2}$$

reveals an inherent trade-off: to maximize the first term, many edges should be contained in clusters, whereas minimization of the second term is achieved by splitting the graph into many clusters of small total degrees. The first term is also known as *edge coverage* [Gae05]. To translate modularity optimization into our framework, we employ a second formulation that depends on the edges included into the clustering. In particular, we can assign a value of

$$c(u, v) = \begin{cases} \dfrac{1}{2m} - \dfrac{\deg(u)\deg(v)}{4m^2}, & \text{if } \{u, v\} \in E \\[2ex] -\dfrac{\deg(u)\deg(v)}{4m^2}, & \text{if } \{u, v\} \notin E \end{cases}$$

to each edge possible in G. The modularity is then given by summing over the values of all existing and non-existing edges covered by the clusters:

$$q(\mathcal{C}) = \sum_{C \in \mathcal{C}} \sum_{u,v,\in C} c(u,v). \tag{9.3}$$

Modularity games are defined as follows. Each two player game Γ^{uv} is a game with n strategies. The payoff for a pair (u,v) of players playing different strategies is both 0. If they play the same strategy, the payoff is $c(u,v)$ for both of them.

Definition 9.1 *A* modularity game *is a polymatrix game, in which players, strategies, and bilateral games are specified by*

- *a simple undirected graph* $G = (V,E)$ *with the set of players* V,
- *the same set of* n *strategies* $\mathcal{S}_v$ *for every player* $v \in V$, *and*
- *for every* $\{u,v\} \subseteq V$ *a bilateral game* $\Gamma^{uv} = (\{u,v\}, \mathcal{S}_u \times \mathcal{S}_v, \mathtt{util}^{uv})$ *with*

$$\mathtt{util}^{uv}_u(s_u,s_v) = \mathtt{util}^{uv}_v(s_u,s_v) = \begin{cases} c(u,v) & \textit{if } s_u = s_v \\ 0 & \textit{otherwise.} \end{cases}$$

For a succinct representation it is sufficient to encode only the graph G. Modularity games are clearly welfare optimizing, and so the price of stability is 1. Note that $\mathtt{welfare}(s)$ of a state s does not completely correspond to the modularity value of the induced clustering. In Equation (9.3) we must additionally count one non-existing self-loop for each player v, i.e. the formula includes the value for a non-existing edge $\{u,u\}$ for each $u \in V$. Hence, for a state s and the corresponding clustering $\mathcal{C}$ we have

$$\mathtt{welfare}(s) = q(\mathcal{C}) + \sum_{v \in V} \frac{1}{\deg(v)^2}.$$

It is possible to adjust $\mathtt{welfare}$ to equal $q(\mathcal{C})$ by subtracting $(n-1)^{-1}\deg(v)^{-2}$ from every utility value for every player v in every of the $n-1$ bilateral games she is playing. This adjustment affects only the price of anarchy. All other properties of the game studied here remain the same. In particular, the new game has the same NE and the same optimal NE. Thus, for simplicity we stick to the above outlined unadjusted version of the game.

The rest of this chapter is organized as follows. In the following Section 9.1 we overview directly related previous work on the modularity clustering index. Section 9.2 contains results on the cost and complexity of NE of modularity games. The price of anarchy can be as large as $\Omega(n)$ (Theorem 9.1). While finding a NE can be done in polynomial time by best response iteration (Corollary 9.1), finding optimal NE and social optimum clusterings is NP-hard (Corollary 9.2). This result is a consequence from the NP-hardness of optimizing the modularity index (Theorem 9.4). Section 9.3 adjusts all results for the case in which the game allows only two strategies (Theorem 9.5 - Corollary 9.3). Finally, Section 9.4 outlines open problems that arise from the results of this chapter.

9.1 Previous and Related Work

The graph clustering problem has been studied by mathematicians and computer scientists since the early 1970s [KL70]. Recently, there is a growing interest in the complex systems literature due to applications in physics and biology. In this context modularity was proposed as a new clustering index [GN04]. It immediately prompted a number of follow-up studies concerning different applications and possible adjustments of the measure (see, e.g. [FB07, ZMW05, MRC05, FPP06]). Also, a wide range of algorithmic approaches has been considered, for example based on a greedy agglomeration [New04, CMN04], spectral division [New05, WS05], simulated annealing [GSPA04, RB06] and extremal optimization [DA05]. None of these algorithms, however, produces optimal partitions, as our results show that modularity optimization is NP-hard. This has been speculated before [New05], but no formal arguments or proofs were provided.

Optimizing modularity is connected to the equipartition problem for graphs. In this problem vertices are weighted, and the goal is to partition the graph into connected subgraphs such that the weights are distributed as equal as possible [Sim99]. In particular, the problem of optimizing modularity with a fixed number of clusters for trees can be cast as tree equipartition with degrees as vertex weights and partition differences penalized by the l_2-norm [Sch01].

The modularity index is similar to clustering indices recently proposed under the name correlation clustering [BBC04, GG06], cluster editing [SST04], or performance [Gae05]. These indices and modularity both count the number of edges covered by the clusters. While correlation clustering indices count non-existing edges between clusters, modularity relies on a term based on degree sums instead. In particular, the problem of optimizing these indices is equal to the problem of optimizing modularity in $\frac{n}{2}$-regular graphs.

9.2 Cost and Complexity of Nash Equilibria

The modularity game is welfare optimizing, so the price of stability is 1. The price of anarchy, however, is significantly higher.

Theorem 9.1 *The price of anarchy in the modularity game is at least* $\frac{n}{2} - 1$.

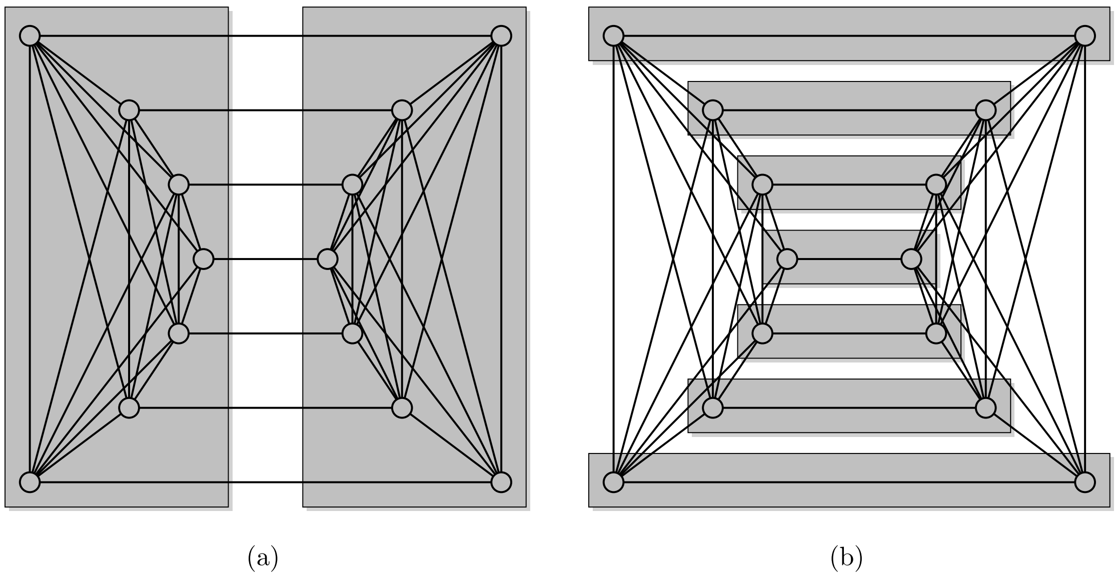


Figure 9.1: A modularity game with a bad NE. (a) Optimum solution s^*; (b) a bad NE s.

Proof. Consider the graph in Figure 9.2. It consists of two cliques of size $\frac{n}{2}$, which are connected by a matching of size $\frac{n}{2}$. The graph is $\frac{n}{2}$-regular, and has $m = \frac{n^2}{4}$ edges. The graph is regular, so each existing edge covered by the clustering has a value $\frac{1}{n^2} - \frac{n^2}{n^4} = \frac{1}{n^2}$ to the players. Each covered non-existing edge has value $-\frac{1}{n^2}$. It is easy to see that a state s^* with two strategies, one played by each clique (see Figure 9.1(a)), obtains a welfare value of

$$welfare(s) = n(\frac{n}{2} - 1)\frac{1}{n^2} = \frac{1}{2} - \frac{1}{n}.$$

The value approaches $\frac{1}{2}$ as n grows large. Now consider the state s with $\frac{n}{2}$ strategies, each one played by the vertices of exactly one matching edge (see Figure 9.1(b)). We first show that this state is a NE. Each player has a utility of $\frac{1}{n^2}$. Consider a player v that changes her strategy. She can either join a different cluster or create a new cluster by herself. If she joins a different cluster, she loses one edge and gains exactly one new edge, as each cluster contains exactly one player from the same clique as v. In addition, however, she must also include a non-existing edge to the second vertex present in the cluster, which results in a utility of 0. If she instead creates a new cluster by herself, she loses one edge and gets a utility of 0. Hence, no strategy switch increases her utility, so s is a NE. It has

$$welfare(s) = n\frac{1}{n^2} = \frac{1}{n},$$

which yields the desired result and proves the theorem. □

The next lemma bounds the range of possible modularity values for any clustering of a graph.

Lemma 9.1 *For any clustering $\mathcal{C}$ of any graph G the modularity value*

$$-\frac{1}{2} \leq q(\mathcal{C}) < 1.$$

Proof. The definition of modularity directly implies an upper bound of 1. A modularity value of exactly 1 requires total edge coverage. However, if an edge is covered, there is also a reduction by the quadratic term of the cluster it is located in. Therefore, a modularity value of 1 can never be achieved and the inequality is strict. Nevertheless, the bound is tight. For a sufficiently large graph with single edge components the value of the obvious component-wise clustering gets arbitrarily close to 1.

For the lower bound consider the contribution of a single cluster C to $q(\mathcal{C})$. We denote by $m_C = |E(C)|$ and $m_C^- = \sum_{C' \in \mathcal{C}, C' \neq C} |E(C, C')|$. This yields as the contribution

$$\frac{m_C}{m} - \left(\frac{m_C}{m} + \frac{m_C^-}{2m}\right)^2$$

Note that this expression is strictly decreasing in m_C^-. In addition, when varying m_C there is only a maximum point at $m_C = \frac{m - m_C^-}{2}$. Hence, the contribution of a cluster is minimized when $m_C = 0$ and m_C^- as large as possible. To minimize the modularity value it is thus necessary to pick clusters such that all edges run between them. Then

$$q(\mathcal{C}) = -\frac{1}{4m^2} \sum_{C \in \mathcal{C}} |\{(u, v) \in E \mid u \in C, v \notin C\}|^2.$$

As $(x + y)^2 \geq x^2 + y^2$ for all non-negative numbers x and y, this sum is minimized when edges are aggregated as much as possible. In the worst case there are exactly two clusters, and all edges run between them. Then $q(\mathcal{C}) = -\frac{1}{2}$, and the lemma is proven. □

Observe that finding an arbitrary NE can be done in polynomial time by best response iteration from any starting solution. The potential (and welfare) function of our game can only take as many values as there are for expressing modularity of a clustering. Lemma 9.1 shows that each clustering $\mathcal{C}$ has $-\frac{1}{2} \leq q(\mathcal{C}) < 1$. As $4m^2 \cdot q(\mathcal{C})$ is an integer, there are at most $6m^2$ modularity values. This restricts the values for our potential functions and yields the following direct corollary.

Corollary 9.1 *Finding a NE in a modularity game can be done by best response iteration from an arbitrary starting state in at most* $6m^2$ *iterations.*

When we try to obtain the best NE, or equivalently s^*, the problem gets more complicated. In particular, we show that the problem of finding the clustering of maximum modularity in a graph is NP-hard. This naturally extends to finding the best NE and the social optimum in our game, as the adjustment of $q(\mathcal{C})$ to $\mathsf{welfare}$ has no effects on the optimality of states. To tackle the proof, we start with a formal definition of the decision problem underlying modularity maximization.

Problem 9.2 (Modularity) *Given a graph* G *and a number* K*, is there a clustering* $\mathcal{C}$ *of* G*, for which* $q(\mathcal{C}) \geq K$*?*

Note that we may ignore the fact that, in principle, K could be a real number in the range $[0, 1]$, because $4m^2 \cdot q(\mathcal{C})$ is integer for every partition $\mathcal{C}$ of G and polynomially bounded in the size of G.

The hardness result for Modularity is based on a transformation from the following decision problem.

Problem 9.3 (3-Partition) *Given a set* $A = \{a_1, \ldots, a_{3x}\}$ *or* $3x$ *numbers such that the sum* $\sum_{i=1}^{3x} a_i = xb$ *and* $b/4 < a_i < b/2$ *for an integer* b *and for any* $a_i \in A$*, is there a partition of these numbers into* x *sets, such that the numbers in each set sum up to* b*?*

We show that an instance $A = \{a_1, \ldots, a_{3x}\}$ of 3-Partition can be transformed into an instance $(G(A), K(A))$ of Modularity, such that $G(A)$ has a clustering with modularity at least $K(A)$, if and only if $a_1, \ldots, a_{3x}$ can be partitioned into x sets of sum b each.

It is crucial that 3-Partition is *strongly* NP-complete [GJ79], i.e. it remains NP-complete if the input is represented in unary coding. This implies that no algorithm can decide the problem in time polynomial even in the sum of the input values, unless $\mathsf{P} = \mathsf{NP}$. More importantly, it implies that our transformation need only be pseudo-polynomial.

The reduction is constructed as follows. From an instance A of 3-Partition, construct a graph $G(A)$ with x cliques $H_1, \ldots, H_x$ of size $a = \sum_{i=1}^{3x} a_i$ each. For each element $a_i \in A$ we introduce a single *element vertex*, and connect it to a_i vertices in each of the x cliques in such a way that each clique member is connected to exactly one element vertex. It is easy to see that each clique vertex then has degree a, and the element vertex corresponding to element $a_i \in A$ has degree xa_i. The number of edges in $G(A)$ is $m = \frac{x}{2}a(a+1)$. See Figure 9.2 for an example. The size of $G(A)$ is polynomial in the unary coding size of A, so that our transformation is indeed

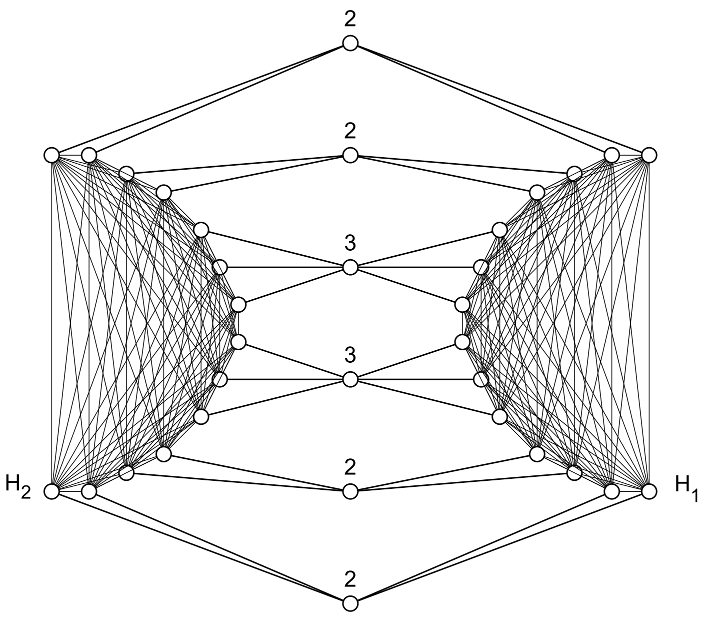


Figure 9.2: An example graph $G(A)$ for the instance $A = \{2, 2, 2, 2, 3, 3\}$ of 3-PARTITION. The resulting graph has $k = 2$ cliques H_1 and H_2. Vertex labels indicate the corresponding numbers $a_i \in A$.

pseudo-polynomial. Before specifying bound $K(A)$ for the instance of MODULARITY, we show three properties of maximum modularity clusterings of $G(A)$. Together these properties establish the desired characterization of solutions for 3-PARTITION by solutions for MODULARITY.

Lemma 9.2 *In a maximum modularity clustering of* $G(A)$ *none of the cliques* $H_1, \dots, H_x$ *is split.*

Proof. We consider a clustering $\mathcal{C}$ that splits a clique $H \in \{H_1, \dots, H_x\}$ into different clusters and then show how to obtain a clustering with strictly higher modularity. Suppose that $C_1, \dots, C_g \in \mathcal{C}$, $g > 1$, are the clusters that contain vertices of H. For $i = 1, \dots, g$ we denote by

- n_i the number of vertices of H contained in cluster C_i,
- $m_i = |E(C_i)|$ the number edges between vertices in C_i,
- f_i the number of edges between vertices of H in C_i and element vertices in C_i,
- d_i be the sum of degrees of all vertices in C_i.

The contribution of $C_1, \ldots, C_g$ to $q(\mathcal{C})$ is

$$\frac{1}{m}\sum_{i=1}^{g} m_i - \frac{1}{4m^2}\sum_{i=1}^{g} d_i^2 \, .$$

Now suppose we create a clustering $\mathcal{C}'$ by rearranging the vertices in $C_1, \ldots, C_g$ into clusters $C', C_1', \ldots, C_g'$, such that C' contains exactly the vertices of clique H, and each C_i', $1 \leq i \leq g$, the remaining elements of C_i (if any). In this new clustering the number of covered edges reduces by $\sum_{i=1}^{g} f_i$, because all vertices from H are removed from the clusters C_i'. The edges connecting the clique vertices to other non-clique vertices of C_i are now inter-cluster edges. For H itself there are $\sum_{i=1}^{g}\sum_{j=i+1}^{g} n_i n_j$ edges that are now additionally covered due to the creation of cluster C'. In terms of degrees the new cluster C' contains a vertices of degree a. The sums for the remaining clusters C_i' are reduced by the degrees of the clique vertices, as these vertices are now in C'. So the contribution of these clusters to $q(\mathcal{C}')$ is given by

$$\frac{1}{m}\sum_{i=1}^{g}\left(m_i + \sum_{j=i+1}^{g} n_i n_j - f_i\right) - \frac{1}{4m^2}\left(a^4 + \sum_{i=1}^{g}(d_i - n_i a)^2\right) \, .$$

We define $\Delta = 4m^2(q(\mathcal{C}') - q(\mathcal{C}))$ and find

$$\begin{aligned}
\Delta &= 4m\left(\sum_{i=1}^{g}\sum_{j=i+1}^{g} n_i n_j - f_i\right) + \left(\left(\sum_{i=1}^{g} 2d_i n_i a - n_i^2 a^2\right) - a^4\right) \\
&= \left(4m\sum_{i=1}^{g}\sum_{j=i+1}^{g} n_i n_j - 4m\sum_{i=1}^{g} f_i + \left(\sum_{i=1}^{g} n_i\left(2d_i a - n_i a^2\right)\right) - a^4\right)
\end{aligned}$$

Using the equalities $2\sum_{i=1}^{g}\sum_{j=i+1}^{g} n_i n_j = \sum_{i=1}^{g}\sum_{j\neq i} n_i n_j$ and $m = \frac{x}{2}a(a+1)$, and rearranging terms we get

$$\begin{aligned}
\Delta &= -a^4 - 2x(a^2+a)\sum_{i=1}^{g} f_i + a\sum_{i=1}^{g} n_i\left(2d_i - n_i a + x(a+1)\sum_{j\neq i} n_j\right) \\
&\geq -a^4 - 2x(a^2+a)\sum_{i=1}^{g} f_i + a\sum_{i=1}^{g} n_i\left(n_i a + 2xf_i + x(a+1)\sum_{j\neq i}^{g} n_j\right) .
\end{aligned}$$

For the inequality we use the fact that $d_i \geq n_i a + x f_i$, which holds because C_i contains at least the n_i vertices of degree a from the clique H. In addition, it contains both the clique and element vertices for each edge counted in f_i. For each such edge there are $k-1$ other edges connecting the element vertex to the $k-1$ other cliques. Hence, we get a contribution of xf_i in the degrees of the element vertices. We combine the terms n_i and one of the terms $\sum_{j\neq i} n_j$, use the fact that $\sum_{j=1}^{g} n_j = a$, and calculate

$$\begin{aligned}\Delta &\geq -2x(a^2+a)\sum_{i=1}^{g} f_i + a\sum_{i=1}^{g} n_i \left(2xf_i + ((x-1)a+x)\sum_{j\neq i}^{g} n_j\right)\\ &= a\sum_{i=1}^{g}\left(2xf_i(n_i-a-1)) + ((x-1)a+x)\sum_{j=1,j\neq i}^{g} n_in_j\right)\\ &\geq a\sum_{i=1}^{g}\left(2xn_i(n_i-a-1) + ((x-1)a+x)\sum_{j=1,j\neq i}^{g} n_in_j\right).\end{aligned}$$

For the last step note that $n_i \leq a-1$ and $n_i - a - 1 < 0$ for all $i = 1, \ldots, g$. So increasing f_i decreases the modularity difference. For each vertex of H there is at most one edge to a vertex not in H, and thus $f_i \leq n_i$. By rearranging and using the fact that $a \geq 3x$ we get

$$\begin{aligned}\Delta &\geq a\sum_{i=1}^{g} n_i\left(2x(n_i-a-1) + ((x-1)a+x)\sum_{j=1,j\neq i}^{g} n_j\right)\\ &\geq a((x-1)a-3x)\sum_{i=1}^{g}\sum_{j=1,j\neq i}^{g} n_in_j\\ &\geq 3x^2(3x-6)\sum_{i=1}^{g}\sum_{j=1,j\neq i}^{g} n_in_j > 0,\end{aligned}$$

as we can assume $x > 2$ for all relevant instances of 3-PARTITION. This shows that any clustering can be improved by merging each clique completely into a cluster. This proves the lemma. □

Next, we observe that the optimum clustering places at most one clique completely into a single cluster.

Lemma 9.3 *In a maximum modularity clustering of* $G(A)$ *every cluster contains at most one of the cliques* $H_1, \ldots, H_x$.

Proof. Consider a maximum modularity clustering. The previous lemma shows that each of the x cliques $H_1, \ldots, H_x$ is entirely contained in one cluster. Assume that there is a cluster C which contains at least two of the cliques. If C does not contain any element vertices, then the cliques form disconnected components in the cluster. In this case it is easy to see that the clustering can be improved by splitting

C into distinct clusters, one for each clique. In this way we keep the number of edges within clusters the same, however, we reduce the squared degree sums of clusters.

Otherwise, we assume C contains $y > 1$ cliques completely and in addition some element vertices of elements a_j with $j \in J \subseteq \{1, \dots, x\}$. Note that inside the y cliques $\frac{y}{2}a(a-1)$ edges are covered. In addition, for every element vertex corresponding to an element a_j there are ya_j edges included. The degree sum of the cluster is given by the ya clique vertices of degree a and some number of element vertices of degree xa_j. The contribution of C to q($\mathcal{C}$) is thus given by

$$\frac{1}{m}\left(\frac{y}{2}a(a-1) + y\sum_{j\in J} a_j\right) - \frac{1}{4m^2}\left(ya^2 + x\sum_{j\in J} a_j\right)^2.$$

Now suppose we create $\mathcal{C}'$ by splitting C into C_1' and C_2' such that C_1' completely contains a single clique H. This leaves the number of edges covered within the cliques the same, however, all edges from H to the included element vertices eventually drop out. The degree sum of C_1' is exactly a^2, and so the contribution of C_1' and C_2' to q($\mathcal{C}'$) is given by

$$\frac{1}{m}\left(\frac{y}{2}a(a-1) + (y-1)\sum_{j\in J} a_j\right) - \frac{1}{4m^2}\left(\left((y-1)a^2 + x\sum_{j\in J} a_j\right)^2 + a^4\right).$$

Considering the difference $\Delta = 4m^2(q(\mathcal{C}') - q(\mathcal{C}))$ we note that

$$\begin{aligned}
\Delta &= -4m\sum_{j\in J} a_j + \left((2y-1)a^4 + 2xa^2\sum_{j\in J} a_j - a^4\right)\\
&= 2(y-1)a^4 + 2xa^2\sum_{j\in J} a_j - 4m\sum_{j\in J} a_j\\
&= 2(y-1)a^4 - 2xa\sum_{j\in J} a_j\\
&\geq 18x^3(9x-1) > 0,
\end{aligned}$$

as $x > 0$ for all instances of 3-PARTITION.

Since the clustering is improved in each case, it is not optimal. This is a contradiction. □

The previous two lemmas show that any clustering can be strictly improved to a clustering that contains x *clique clusters*, such that each one completely contains one of the cliques $H_1, \dots, H_x$ (possibly plus some additional element vertices). In particular, this must hold for the optimum clustering as well.

Now that we know how the cliques are clustered we turn to the element vertices. As they are not directly connected, it is never optimal to create a cluster consisting only of element vertices. Splitting such a cluster into singleton clusters, one for each element vertex, reduces the squared degree sums but keeps the edge coverage at the same value. Hence, such a split yields a clustering with strictly higher modularity. The next lemma shows that we can further strictly improve the modularity of a clustering with a singleton cluster of an element vertex by joining it with one of the clique clusters.

Lemma 9.4 *In a maximum modularity clustering of* $G(A)$ *there is no cluster composed of element vertices only.*

Proof. Consider a clustering $\mathcal{C}$ of maximum modularity and suppose that there is an element vertex v_i corresponding to the element a_i, which is not part of any clique cluster. As argued above, we can improve such a clustering by creating a singleton cluster $C = \{v_i\}$. Suppose C_{min} is a clique cluster for which the sum of degrees is minimal. We know that C_{min} contains all vertices from a clique H and probably some other element vertices for elements a_j with $j \in J$ for some index set J. The cluster C_{min} covers all $\frac{a(a-1)}{2}$ edges within H and $\sum_{j \in J} a_j$ edges to element vertices. The degree sum is a^2 for clique vertices and $x \sum_{j \in J} a_j$ for element vertices. As C is a singleton cluster, it covers no edges, and the degree sum is xa_i. This yields a contribution of C and C_{min} to $q(\mathcal{C})$ of

$$\frac{1}{m}\left(\frac{a(a-1)}{2} + \sum_{j \in J} a_j\right) - \frac{1}{4m^2}\left(\left(a^2 + x\sum_{j \in J} a_j\right)^2 + x^2 a_i^2\right).$$

Again, we create a different clustering $\mathcal{C}'$ by joining C and C_{min} to a new cluster C'. This increases the edge coverage by a_i. The new cluster C' has the sum of degrees of both previous clusters. The contribution of C' to $q(\mathcal{C}')$ is given by

$$\frac{1}{m}\left(\frac{a(a-1)}{2} + a_i + \sum_{j \in J} a_j\right) - \frac{1}{4m^2}\left(a^2 + xa_i + x\sum_{j \in J} a_j\right)^2,$$

so that the difference $\Delta = 4m^2(q(\mathcal{C}') - q(\mathcal{C}))$ is given by

$$\begin{aligned}
\Delta &= 4ma_i - \left(2xa^2a_i + 2x^2a_i\sum_{j \in J} a_j\right) \\
&= \left(2xa(a+1)a_i - 2xa^2a_i - 2x^2a_i\sum_{j \in J} a_j\right) \\
&= a_i\left(2xa - 2x^2\sum_{j \in J} a_j\right).
\end{aligned}$$

At this point recall that C_{min} is the clique cluster with the minimum degree sum. For this cluster the elements corresponding to included element vertices can never sum to more than $\frac{1}{x}a$. In particular, as v_i is not part of any clique cluster, the elements of vertices in C_{min} can never sum to more than $\frac{1}{x}(a - a_i)$. Thus,

$$\sum_{j \in J} a_j \le \frac{1}{x}(a - a_i) < \frac{1}{x}a,$$

and so $\Delta > 0$. This contradicts the assumption that $\mathcal{C}$ is optimal. □

We have shown that for the graphs $G(A)$ the clustering of maximum modularity consists of exactly x clique clusters, and each element vertex belongs to exactly one of the clique clusters. Finally, we are ready to state the main result.

Theorem 9.4 MODULARITY *is strongly* NP*-complete.*

Proof. For a given clustering $\mathcal{C}$ of $G(A)$ we can check in polynomial time whether $q(\mathcal{C}) \ge K(A)$, so clearly the decision problem MODULARITY $\in \mathsf{NP}$.

For NP-completeness we transform an instance $A = \{a_1, \ldots, a_{3x}\}$ of 3-PARTITION into an instance $(G(A), K(A))$ of MODULARITY. We have already outlined the construction of the graph $G(A)$ above. For the correct parameter $K(A)$ we consider a clustering in $G(A)$ with the properties derived in the previous lemmas, i.e. a clustering with exactly x clique clusters. Any such clustering yields exactly $(x-1)a$ inter-cluster edges, so the edge coverage is given by

$$\sum_{C \in \mathcal{C}^*} \frac{|E(C)|}{m} = \frac{m - (x-1)a}{m} = 1 - \frac{2(x-1)a}{xa(a+1)} = 1 - \frac{2x-2}{x(a+1)} .$$

Hence, the clustering $\mathcal{C} = (C_1, \ldots, C_x)$ with maximum modularity must minimize

$$\deg(C_1)^2 + \deg(C_2)^2 + \ldots + \deg(C_x)^2 .$$

This requires to equilibrate the element vertices according to their degree as good as possible between the clusters. In the optimum case we can assign each cluster element vertices corresponding to elements that sum to $b = \frac{1}{x}a$. In this case the sum of degrees of element vertices in each clique cluster is equal to a. This yields $\deg(C_i) = a^2 + a$ for each clique cluster C_i, $i = 1, \ldots, x$, and gives

$$\deg(C_1)^2 + \ldots + \deg(C_x)^2 \ge x(a^2 + a)^2 = xa^2(a+1)^2 .$$

Equality holds only in the case, in which an assignment of b to each cluster is possible. Hence, if there is a clustering $\mathcal{C}$ with $q(\mathcal{C})$ of at least

$$K(A) = 1 - \frac{2x-2}{x(a+1)} - \frac{xa^2(a+1)^2}{x^2a^2(a+1)^2} = \frac{(x-1)(a-1)}{x(a+1)} ,$$

then we know that this clustering must split the element vertices perfectly to the x clique clusters. As each element vertex is contained in exactly one cluster, this yields a solution to the instance of 3-PARTITION. With this choice of $K(A)$ the instance $(G(A), K(A))$ of MODULARITY is satisfiable only if the instance A of 3-PARTITION is satisfiable.

Otherwise, suppose the instance for 3-PARTITION is satisfiable. Then there is a partition into x sets such that the sum over each set is $\frac{a}{x}$. If we cluster the corresponding graph by joining the element vertices of each set with a different clique, we get a clustering of modularity $K(A)$. This shows that the instance $(G(A), K(A))$ of MODULARITY is satisfiable if the instance A of 3-PARTITION is satisfiable. This completes the reduction and proves the theorem. □

As noted before the hardness of finding the social optimum and the best NE in the modularity game follows.

Corollary 9.2 *Finding the social optimum and the best NE for a modularity game is* NP*-hard.*

9.3 Modularity Games with Two Strategies

This section treats the special case of the modularity game, in which the strategy choice of each player is restricted to two strategies. We denote these games as *2-modularity games* and note that some results transfer directly from the previous section.

Theorem 9.5 *For the 2-modularity game we have the following:*

- *The price of stability is 1.*
- *The price of anarchy is at least* $\frac{n}{2} - 1$.
- *Every best response iteration can take at most* $6m^2$ *steps before reaching a NE.*

In particular, the lower bound for the price of anarchy can be obtained similarly as in the general case of Theorem 9.1. We simply join the clusters in the NE s described in the proof until two clusters of equal size are formed. It is straightforward to verify that this state is also a NE and has the same social welfare as s.

Finding optimal NE and the social optimum s^* is also NP-hard for 2-modularity games, but for the proof a different reduction is needed. Our proof is given for the problem of finding the clustering with maximum modularity that splits the graph into at most two clusters. It also works for the problem, in which the graph is to be split into exactly two clusters. The underlying decision problem is given as follows.

Problem 9.6 (2-Modularity) *Given a graph* G *and a number* K*, is there a clustering* $\mathcal{C}$ *of* G *into at most two clusters, for which* $q(\mathcal{C}) \geq \mathsf{K}$*?*

The following reduction is similar to the one given recently for showing hardness of the MinDisAgree problem of correlation clustering with exactly two clusters [GG06]. The problem used for the reduction is Minimum Bisection in Cubic Graphs (MB3).

Problem 9.7 (Minimum Bisection in Cubic Graphs) *Given a 3-regular graph* G *with* n *vertices and an integer* $\mathsf{K}_{\mathsf{cut}}$*, is there a clustering into two clusters of* $n/2$ *vertices each such that it cuts at most* $\mathsf{K}_{\mathsf{cut}}$ *edges?*

This problem is strongly NP-complete [BCLS87]. An instance $(\tilde{\mathsf{G}}(\mathsf{G}), \mathsf{K}(\mathsf{G}))$ of 2-Modularity is constructed from an instance of MB3 by creating a graph $\tilde{\mathsf{G}}(\mathsf{G})$ and a number $\mathsf{K}(\mathsf{G})$ as follows. First, set $\tilde{\mathsf{G}}(\mathsf{G}) = \mathsf{G}$. Then for each vertex $v \in V$ add $n-1$ new vertices to the graph and construct a clique. We denote these cliques as $\mathsf{H}(v)$ and refer to them as *vertex clique* for $v \in V$. Hence, in total n different new cliques are constructed, and after this transformation each vertex from the original graph has degree $n+2$ in $\tilde{\mathsf{G}}(\mathsf{G})$. Note that a cubic graph with n vertices has exactly $3n/2$ edges. Hence, after the adjustment there are exactly $\tilde{n} = n^2$ vertices and $\tilde{m} = \frac{n}{2}(n(n-1)+3)$ edges in $\tilde{\mathsf{G}}(\mathsf{G})$ (see Figure 9.3). In addition, we set

$$\mathsf{K}(\mathsf{G}) = \frac{1}{2} - \frac{\mathsf{K}_{\mathsf{cut}}}{\tilde{m}} .$$

In the following we show that the optimum clustering $\mathcal{C}^*$ of $\tilde{\mathsf{G}}(\mathsf{G})$ has exactly two clusters, and they induce a minimum bisection partition of G from the MB3 instance. In particular, $\mathsf{K}(\mathsf{G})$ is such that G has a bisection cut of size at most $\mathsf{K}_{\mathsf{cut}}$ if and only if $\tilde{\mathsf{G}}(\mathsf{G})$ has a 2-clustering of modularity at least K.

We begin by showing that there is always a clustering $\mathcal{C}$ with $q(\mathcal{C}) > 0$, and so $\mathcal{C}^*$ must have exactly two clusters. This serves to show that the hardness result holds for two versions of 2-Modularity, in which at most or exactly two clusters must be found.

Lemma 9.5 *The maximum modularity 2-clustering of* $\tilde{\mathsf{G}}(\mathsf{G})$ *has two clusters.*

Proof. It is straightforward to verify that a clustering with a single cluster yields modularity of 0. Consider the following clustering $\mathcal{C} = \{C_1, C_2\}$. We pick the vertices of $\mathsf{H}(v)$ for some $v \in V$ as C_1 and the remaining graph as C_2. Then

$$\begin{aligned} q(\{C_1, C_2\}) &= 1 - \frac{3}{\tilde{m}} - \frac{(n(n-1)+3)^2 + ((n-1)(n(n-1)+3))^2}{4\tilde{m}^2} \\ &= \frac{2n-2}{n^2} - \frac{3}{\tilde{m}} > 0, \end{aligned}$$

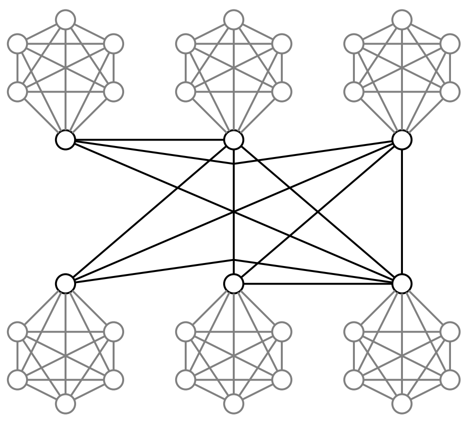


Figure 9.3: A graph $\tilde{G}(G)$ constructed from $G = K_{3,3}$ of an instance of MB3. Grey parts are added to the graph by the reduction. A minimum bisection and a maximum modularity 2-clustering are indicated by upper and lower layers. Note that a minimum bisection of $K_{3,3}$ is not the inherent bipartition.

as $n \geq 4$ for every cubic graph. As $q(\mathcal{C}^*) \geq q(\mathcal{C}) > 0$ and the lemma follows. □

Next, we show that for the constructed graph the optimum clustering splits none of the vertex cliques $H(v)$.

Lemma 9.6 *In a maximum modularity 2-clustering of* $\tilde{G}(G)$ *each vertex clique is contained completely in one of the clusters.*

Proof. For contradiction we assume a vertex clique $H(v)$ for some $v \in V$ is split between the two clusters. There are clusters C_1 and C_2, in which n_1 and $n - n_1$ vertices from the clique are located, for some $1 \leq n_1 \leq n-1$. We denote the sum of vertex degrees in both clusters excluding vertices from $H(v)$ by d_1 and d_2:

$$d_i = \sum_{u \in C_i, u \notin H(v)} \deg(u).$$

Denote by $\tilde{m}_+$ the number of edges covered by the clusters C_1 and C_2. W.l.o.g. we assume that $d_1 \geq d_2$.

In the following we consider two cases. In both cases we construct a new clustering $\mathcal{C}'$ by adding all vertices from $H(v)$ to C_2.

Case 1: In this case $v \in C_2$. Then

$$q(\mathcal{C}) = \frac{\tilde{m}_+}{\tilde{m}} - \frac{(d_1 + n_1(n-1))^2 + (d_2 + (n-n_1)(n-1)+3)^2}{4\tilde{m}^2},$$

and for the adjusted clustering

$$q(\mathcal{C}') = \frac{\tilde{m}_+ + n_1(n-n_1)}{\tilde{m}} - \frac{d_1^2 + (d_2 + n(n-1)+3)^2}{4\tilde{m}^2} .$$

Considering the difference $\Delta = 4\tilde{m}^2(q(\mathcal{C}') - q(\mathcal{C}))$ we find

$$\begin{aligned} \Delta &= 4\tilde{m}n_1(n-n_1) - d_1^2 + (d_2 + n(n-1)+3)^2 \\ &\quad +(d_1 + n_1(n-1))^2 + (d_2 + (n-n_1)(n-1)+3)^2 \\ &= 4\tilde{m}n_1(n-n_1) + (2n_1^2 - 2nn_1)(n-1)^2 - 6n_1(n-1) \\ &\quad +2(d_1 - d_2)n_1(n-1) \\ &\geq 4\tilde{m}n_1(n-n_1) - 2n_1(n-n_1)(n-1)^2 - 6n_1(n-1), \end{aligned}$$

as we assume $d_1 \geq d_2$. With $n-1 \geq n_1 \geq 1$ this yields

$$\begin{aligned} & n_1(n-n_1)(4\tilde{m} - 2(n-1)^2) - 6n_1(n-1) \\ \geq\ & n_1(n-n_1)(4\tilde{m} - 2(n-1)^2 - 6(n-1)). \end{aligned}$$

It remains to show

$$4\tilde{m} - 2(n-1)^2 - 6(n-1) = 2(n^3 - 2n^2 + 2n + 2) > 0.$$

This holds with $n^3 > 2n^2$ for all $n \geq 4$. Therefore, the modularity value strictly improves if we add all vertices from $H(v)$ to C_2. This proves the lemma for Case 1.

Case 2: In this case $v \in C_1$. Then

$$q(\mathcal{C}) = \frac{\tilde{m}_+}{\tilde{m}} - \frac{(d_1 + n_1(n-1)+3)^2 + (d_2 + (n-n_1)(n-1))^2}{4\tilde{m}^2} ,$$

and for the adjusted clustering

$$q(\mathcal{C}') = \frac{\tilde{m}_+ + n_1(n-n_1)}{\tilde{m}} - \frac{d_1^2 + (d_2 + n(n-1)+3)^2}{4\tilde{m}^2} .$$

Considering the difference $\Delta = 4\tilde{m}^2(q(\mathcal{C}') - q(\mathcal{C}))$ we find

$$\begin{aligned} \Delta &= 4\tilde{m}n_1(n-n_1) - d_1^2 + (d_2 + n(n-1)+3)^2 \\ &\quad +(d_1 + n_1(n-1)+3)^2 + (d_2 + (n-n_1)(n-1))^2 \\ &= 4\tilde{m}n_1(n-n_1) + (2n_1^2 - 2nn_1)(n-1)^2 - 6(n-n_1)(n-1) \\ &\quad +2(d_1 - d_2)(n_1(n-1)+3) \\ &\geq 4\tilde{m}n_1(n-n_1) - 2n_1(n-n_1)(n-1)^2 - 6(n-n_1)(n-1). \end{aligned}$$

Recall $1 \leq n_1 \leq n-1$, and thus it suffices to show

$$\begin{aligned} & 4\tilde{m}n_1 - 2n_1(n-1)^2 - 6(n-1) \\ = \; & 2n(n(n-1)+3)n_1 - 2n_1(n-1)^2 - 6(n-1) \\ = \; & 2n_1(n^2(n-1) - (n-1)^2) + 6nn_1 - 6(n-1) > 0. \end{aligned}$$

It holds for all $n_1 \geq 1$ and $n \geq 4$. This proves the lemma for Case 2. □

The previous lemma implies that each cluster $C \in \mathcal{C}^*$ consists of a number of complete vertex cliques. The next lemma shows that in the optimum case each cluster contains exactly $n/2$ of them.

Lemma 9.7 *In a maximum modularity 2-clustering of* $\tilde{G}(G)$ *each cluster contains* $n/2$ *complete vertex cliques.*

Proof. Suppose for contradiction that one cluster C_1 has $y_1 < n/2$ cliques. $\tilde{m}_+$ denotes the number of edges covered by the clusters. For the modularity

$$q(\mathcal{C}) = \frac{\tilde{m}_+}{\tilde{m}} - \frac{y_1^2(n(n-1)+3)^2 + (n-y_1)^2(n(n-1)+3)^2}{4\tilde{m}^2} .$$

Now create $\mathcal{C}'$ by transferring a complete vertex clique from C_2 to C_1. Since G is 3-regular and the clique is transfered completely, at most 3 edges are lost in coverage. For the modularity

$$q(\mathcal{C}') \geq \frac{\tilde{m}_+ - 3}{\tilde{m}} - \frac{(y_1+1)^2(n(n-1)+3)^2 + (n-y_1-1)^2(n(n-1)+3)^2}{4\tilde{m}^2} .$$

Considering the difference $\Delta = 4\tilde{m}^2(q(\mathcal{C}') - q(\mathcal{C}))$ we find

$$\begin{aligned} \Delta \; & \geq \; -12\tilde{m} + (y_1^2 + (n-y_1)^2 - (y_1+1)^2 - (n-y_1-1)^2)(n(n-1)+3)^2 \\ & = \; -12\tilde{m} + (2n - 4y_1 - 2)(n(n-1)+3)^2 \\ & \geq \; -12\tilde{m} + \frac{8(\tilde{m})^2}{n^2} \\ & = \; 4\tilde{m}^2\left(\frac{2}{n^2} - \frac{6}{n^3 - n^2 + 3n}\right) > 0 \end{aligned}$$

for all $n \geq 4$. The analysis uses the fact that we can assume n to be an even number, so $y_1 \leq \frac{n}{2} - 1$ and $4y_1 \leq 2n - 4$.

Thus, every clustering can be improved by balancing the number of complete vertex cliques in the clusters - independent of the loss in edge coverage. This proves the lemma. □

We are now ready to state the main result.

Theorem 9.8 *2-*MODULARITY *is strongly* NP*-complete.*

Proof. Each cluster in $\mathcal{C}^*$ contains exactly $n/2$ complete vertex cliques. Hence, the sum of degrees in the clusters is exactly $\tilde{m}$. If the optimum clustering has modularity at least

$$q(\mathcal{C}^*) \geq \frac{\tilde{m} - K_{\mathsf{cut}}}{\tilde{m}} - \frac{2\tilde{m}^2}{4\tilde{m}^2} = \frac{1}{2} - \frac{K_{\mathsf{cut}}}{\tilde{m}} = K(G),$$

then the number of edges between the clusters in $\mathcal{C}^*$ can be at most K_{cut}. As all vertex cliques are contained completely in the clusters, the clustering directly yields a bisection for G with cut size at most K_{cut}. Hence, if the modularity instance is satisfiable with $K(G)$, then the instance for MB3 is satisfiable with cut size K_{cut}. On the other hand, if the instance of MB3 is satisfiable and the graph G has a bisection with cut size less than K_{cut}, this directly yields a clustering of modularity at least $K(G)$. This proves the theorem. □

An interesting feature of the proof is that finding the social optimum is hard due to the hardness of minimizing squared degree sums, but in the case of two strategies due to the hardness of minimizing the edge cut. With the hardness of 2-MODULARITY we have also established the hardness of finding the social optimum and the best NE in 2-modularity games.

Corollary 9.3 *Finding the social optimum and the best NE for a 2-modularity game is* NP*-hard.*

9.4 Open problems

In this chapter we have considered the popular clustering index modularity as a game. The open question about the complexity status of modularity maximization was settled by showing NP-hardness. In addition, the restricted version with a bound of two on the number of strategies was shown to be NP-hard. An open problem is to present a matching upper bound on the price of anarchy for a tight characterization of the performance of NE in the game. Another interesting open problem is to obtain approximation algorithms for modularity maximization with provable performance ratios. Lemma 9.1 displays that modularity is a mixed sign measure and includes 0 as possible outcome. In [BDG$^+$07a] we used this property to argue with the instance of Figure 9.2 that a frequently applied greedy agglomeration algorithm achieves no finite performance ratio. A more promising approach is to adjust techniques developed for correlation clustering. Also, a reformulation of the measure to take only positive values could result in more meaningful relative performance ratios.

Chapter 10

Conclusion

In this thesis we have presented and analyzed two frameworks to study selfish behavior and incentives in combinatorial optimization problems. *Investment games* serve to model a cost sharing scenario, in which players have strategies that specify investments. The price of anarchy can be as large as k in all games we have considered. Furthermore, for all classes the price of stability is also close to k and some games do not have NE. From the perspective of a mechanism, who is interested in obtaining stability and can propose and/or influence player strategies, this suggests that the design of the game as a means of cost sharing has quite undesirable properties. Players might not be able to agree upon a solution, and even if they agree, the set of resources bought might be much more expensive than an optimal set. In some special subclasses of games, e.g. for singleton and integral set cover games, singleton and integral CRFL games, path TCGs, and single-source backbone games optimal NE always exist. Only for integral games, however, an efficient algorithm for computing such a cheap NE has been obtained. In most other subclasses the problem of computing exact NE remains unsolved. Hence, even if a mechanism has some means to influence the strategy choices of the agents, it remains unknown how to efficiently obtain a (good) NE. If we do not restrict to these subclasses of games, a mechanism is not even able to decide efficiently whether a game has a NE, because the problem is NP-hard in all the considered general classes of games.

On the other hand, to obtain best-response strategies player must - in most cases - optimize NP-hard problems. For a realistic picture we cannot assume that players will always find best responses. Instead, they must employ heuristics to find a reasonably good strategy. This means that in many cases players might not even be able to recognize that a better strategy exists. Thus, it is more reasonable to consider approximate NE and stability ratios in combination with approximation algorithms. A variety of games in this thesis allow cheap and stable approximate NE. In particular, vertex and set cover games have $(f, 1)$-approximate, UFL games $(3, 3)$-approximate, and TCGs $(2, 1)$-approximate NE. For set cover and UFL games it is possible to employ primal-dual approximation algorithms to get stable approximate NE in polynomial time. For the singleton classes of games it is possible to combine state-of-the-art approximation algorithms, a general scaling technique, and a local search routine to efficiently derive approximate NE that are almost stable. For TCGs the same idea can be employed to find approximate NE in polynomial time. Hence, if players have bounded computational powers, or a bounded ability or willingness to discriminate, the proposed cost sharing environment has quite favorable properties. It allows a mechanism to simultaneously and efficiently induce near-optimality and near-stability.

Clustering games serve to model affiliation and grouping decisions, in which players can be modeled as vertices of a graph. They are special classes of polymatrix games and allow to specify general payoffs for the players. Relative performance measures, like the price of anarchy, are not necessarily meaningful in this context. We have considered the price of anarchy only for sufficiently specialized subclasses of clustering games like Max-l-Cut, MaxAgree, and modularity games. In these games it can behave quite differently, i.e. $1 + \frac{1}{l-1}$ for Max-l-Cut games, $\Omega(\min(l, \sqrt{n}))$ in MaxAgree games, and $\Omega(n)$ in modularity games. Max-l-Cut and MaxAgree games have somewhat similar payoffs. For the standard optimization problems, however, there is a PTAS for MaxAgree [BBC04, GG06], and Max-l-Cut is APX-complete [KKLP97]. In light of these properties, it might be a bit surprising that the price of anarchy for MaxAgree games can be a factor of $\Omega(\sqrt{n})$ larger than for Max-l-Cut games.

All clustering games analyzed in this thesis are potential games, so NE are guaranteed to exist. Finding a NE is equivalent to finding a local optimum for the potential function. In all games studied in detail here the number of possible values for the potential is bounded by a polynomial in the size of the representation. Therefore, finding an arbitrary NE can be done with a local search algorithm using a polynomial number of best response steps from any starting solution. For neighborhood independent games even the NE of maximum and minimum welfare have a simple characterization and can be obtained in polynomial time. In all other games, i.e. Max-l-Cut, MaxAgree, and modularity games, it is NP-hard to obtain optimal NE. In the case of modularity games this confirms a recent conjecture about the complexity status of the underlying combinatorial optimization problem.

Table 10.1 presents an overview of the main results derived in this work. Column "NE" indicates results on the decision problem of NE existence. The next column "find NE/find best NE" shows whether the problem of finding a (best) NE is in P or is NP-hard. Note that if deciding NE existence is NP-hard, then trivially finding a NE can only be harder. The column "[PoA, PoS]" displays the range of the worst-case cost of NE, i.e. it shows the worst-case bounds on the prices of anarchy and stability we have found for the games. With a few exceptions (e.g. for the price of anarchy in MaxAgree and the modularity games) all these bounds are tight.

Finally, column “approximate NE” depicts the results on approximate NE that can be computed in polynomial time.

There are still some empty cells in this table, and they pose immediate open problems for further research. We have indicated throughout the chapters how some of these problems pose technical challenges to the tools and methods used here. For the investment game, characterizing the properties of the wholesale variant is an interesting direction with a practical motivation. Studying the properties of mixed NE seems a fundamental problem more of technical interest. In general, the complex nature of the Internet requires to advance the study of theoretical models to include crucial aspects of realistic scenarios. Hence, cost sharing in more complex network design and resource installation games should be analyzed.

Clustering games are a simple and straightforward model that allow to consider clustering indices relying on local accumulation of edge weights in the context of competitive selfish agents. We have provided first results concerning some well-known and recently popular measures. In general, however, the incentives of decision making for actors embedded in a network are still poorly understood. Our work has shed some light on the complexity of finding stable states in these games. As a next step, it is important to characterize equilibria and, in particular, characterize the influence of single players or player groups on equilibrium outcomes. These are important aspects that allow to obtain a deeper understanding of the properties and the dynamics of decision making by selfish agents in networked environments.

	NE	find NE / find best NE	[PoA, PoS]	apx. NE
Covering	see Set Cover			
Set Cover(SC)	NP-hard, Thm. 4.5 (36)		$[k, k-1]$, Thm. 4.2 (33), Thm. 4.3 (35)	(f, f), Thm. 4.7 (40)
Singleton SC	√	P Thm. 4.12 (51)/NP-hard [GJ79]	$[k, 1]$, Thm. 4.11 (48)	$(1+\epsilon, O(\log\|\mathcal{M}\|))$ Thm. 4.11 (48)
Integral SC	√	P Thm. 4.10 (46)	$[k, 1]$, Thm. 4.10 (46)	$(1, 1)$, Thm. 4.10 (46)
CRFL	see Metric UFL			
Metric UFL	NP-hard, Thm. 4.15 (56)		$[k, k-2]$, Thm. 4.14 (55)	$(3, 3)$, Thm. 4.17 (57)
Singleton CRFL	√		$[k, 1]$, Thm. 4.19 (61)	$(1+\epsilon, \beta)$, Cor. 4.3 (62)
Integral CRFL	√	P Thm. 4.18 (60)	$[k, 1]$, Thm. 4.18 (60)	$(1, 1)$, Thm. 4.18 (60)
TCG	NP-hard, Thm. 5.4 (80)		$[k, k-2]$, Thm. 5.3 (79)	$(3.1+\epsilon, 1.55)$, Thm. 5.7 (91)
PTCG	√		$[k, 1]$, Thm. 5.2 (70)	$(2+\epsilon, 1.55)$, Thm. 5.5 (84)
SBG	√		$[k, 1]$, Thm. 5.11 (95)	$(1+\epsilon, O(\log n \log k \log(\max_p g_p)))$ Thm. 5.12 (95)
Wholesale			$[k/cap(k), ?]$, Thm. 6.1 (99)	
Unweighted Max-l-Cut	√	P/NP-hard, [GJ79]	$[1+1/(l-1), 1]$, Lem. 8.2 (112)	$(1, 1/(1-1/l+(2\ln l)/l^2)$ [FJ97, MR99]
MaxAgree	√	P/NP-hard, [BBC04, GG06]	$[\Omega(\min(l, \sqrt{n})), 1]$ Thm. 8.1 (113)	$(1, 1+\epsilon)$, [BBC04, GG06]
2-Clustering	√	P, Thm. 8.2 (118) / NP-hard, [GJ79]	unbounded/undefined	
Neigh. Indep.	√	P, Thm. 8.2 (118), Thm. 8.4 (121)	unbounded/undefined	
Modularity	√	P, Cor. 9.1 (128) / NP-hard, Thm. 9.4 (134)	$[\Omega(n), 1]$, Thm. 9.1 (125)	
2-Modularity	√	P, Cor. 9.1 (128) / NP-hard, Thm. 9.8 (140)	$[\Omega(n), 1]$, Thm. 9.1 (125)	

Table 10.1: Main results

Bibliography

[AA97] B. Awerbuch and Y. Azar. Buy-at-bulk network design. In *Proc 38th Symp Foundations of Computer Science (FOCS'97)*, pages 542–547, 1997. 97

[AA03] L. Adamic and E. Adar. Friends and neighbors on the web. *Social Networks*, 25(3):211–230, 2003. 105

[AAE05] B. Awerbuch, Y. Azar, and A. Epstein. The price of routing unsplittable flow. In *Proc 37th Symp Theory of Computing (STOC'05)*, pages 331–337, 2005. 3

[ADK$^+$04] E. Anshelevich, A. Dasgupta, J. Kleinberg, T. Roughgarden, É. Tardos, and T. Wexler. The price of stability for network design with fair cost allocation. In *Proc 45th Symp Foundations of Computer Science (FOCS'04)*, pages 295–304, 2004. 3, 26, 68

[ADTW03] E. Anshelevich, A. Dasgupta, É. Tardos, and T. Wexler. Near-optimal network design with selfish agents. In *Proc 35th Symp Theory of Computing (STOC'03)*, pages 511–520, 2003. 3, 5, 25, 49, 54, 56, 65, 66, 68, 70, 72, 74, 75, 77, 82, 84, 92, 94

[AEED$^+$06] S. Albers, S. Eilts, E. Even-Dar, Y. Mansour, and L. Roditty. On Nash equilibria for a network creation game. In *Proc 17th Symp Discrete Algorithms (SODA'06)*, pages 89–98, 2006. 69

[AH02] R. Aumann and S. Hart, editors. *Handbook of Game Theory with Economic Applications*, volume 1–3. Elsevier/North-Holland Science Publishers, 1992–2002. 7, 159

[AKP$^+$02] A. Akella, R. Karp, C. Papadimitriou, S. Seshan, and S. Shenker. Selfish behavior and stability of the internet: A game-theoretic analysis of the TCP. In *Proc SIGCOMM 2002 Conf Applications, Technologies, Architectures, and Protocols for Computer Communication*, 2002. 3

[AKR95] A. Agrawal, P. Klein, and R. Ravi. When trees collide: An approximation algorithm for the generalized Steiner problem on networks. *SIAM J Comp*, 24(3):445–456, 1995. 63, 65

[AMS06] N. Alon, D. Moshkovitz, and D. Safra. Algorithmic construction of sets with k-restrictions. *ACM Transactions on Algorithms*, 2(2):153–177, 2006. 42

[And04] M. Andrews. Hardness of buy-at-bulk network design. In *Proc 45th Symp Foundations of Computer Science (FOCS'04)*, pages 115–124, 2004. 97

[ARB06] H. Ackermann, H. Rögelin, and B.Vöcking. On the impact of combinatorial structure on congestion games. In *Proc 47th Symp Foundations of Computer Science (FOCS'06)*, pages 613–622, 2006. 21

[Aum74] R. Aumann. Subjectivity and correlation in randomized strategies. *Journal of Mathematical Economics*, 1:67–96, 1974. 106

[Bak94] B. Baker. Approximation algorithms for NP-complete problems on planar graphs. *J ACM*, 41(1):153–180, 1994. 45

[Bal61] M. Balinski. Integer programming: Methods, uses, computation. *Management Science*, 12(3):253–313, 1961. 53

[BBC04] N. Bansal, A. Blum, and S. Chawla. Correlation clustering. *Machine Learning*, 56(1-3):89–113, 2004. 112, 114, 125, 142, 144

[BCLS87] T. Bui, S. Chaudhuri, F.T. Leighton, and M. Sipser. Graph bisection algorithms with good average case behavior. *Combinatorica*, 7(2):171–191, 1987. 136

[BD01] W. Brock and S. Durlauf. Discrete choice with social interactions. *Review of Economic Studies*, 68(2):235–260, 2001. 110

[BdFFM04] V. Biló, C. di Francescomarino, M. Flammini, and G. Melideo. Sharing the cost of multicast transmissions in wireless networks. In *Proc 16th Symp Parallelism in Algorithms and Architectures (SPAA'04)*, pages 180–187, 2004. 4

[BDG$^+$07a] U. Brandes, D. Delling, M. Gaertler, R. Görke, M. Hoefer, Z. Nikoloski, and D. Wagner. On finding graph clusterings with maximum modularity. In *Proc 33rd Intl Workshop Graph-Theoretic Concepts in Computer Science (WG'07)*, 2007. to appear. i, 140

[BDG$^+$07b] U. Brandes, D. Delling, M. Gaertler, R. Görke, M. Hoefer, Z. Nikoloski, and D. Wagner. On modularity clustering. *IEEE Transactions on Knowledge and Data Engineering*, 2007. to appear. i

[BHRZ97] C. D. Bateman, C. S. Helvig, G. Robins, and A. Zelikovsky. Provably good routing tree construction with multi-port terminals. In *Proc Intl Symp Physical Design (ISPD'97)*, pages 96–102, 1997. 96

[Bir76] C. Bird. On cost allocation for a spanning tree: a game theoretic approach. *Networks*, 6:335–350, 1976. 69

[BLB04] N. Bhat and K. Leyton-Brown. Computing Nash equilibria of action-graph games. In *Proc 20th Conf Uncertainty in Artificial Intelligence (UAI'04)*, pages 35–42, 2004. 106

[BLPGVR04] Y. Bramoullé, D. López-Pintado, S. Goyal, and F. Vega-Redondo. Network formation and anti-coordination games. *Intl J Game Theory*, 33(1):1–19, 2004. 110

[Blu93] L. Blume. The statistical mechanics of strategic interaction. *Games and Economic Behavior*, 5:387–424, 1993. 11, 108, 110, 115

[Bra07] Y. Bramoullé. Anti-coordination and social interactions. *Games and Economic Behavior*, 58(1):30–49, 2007. 110

[BV06] S. Berninghaus and B. Vogt. Network formation in symmetric 2×2 games. *Homo Oeconomicus*, 23(3/4):421–466, 2006. 110

[CC02] M. Chlebík and J. Chlebíková. Approximation hardness of the Steiner tree problem in graphs. In *Proc 8th Scandinavian Workshop on Algorithm Theory (SWAT)*, pages 170–179, 2002. 65

[CCLE$^+$06] C. Chekuri, J. Chuzhoy, L. Lewin-Eytan, J. Naor, and A. Orda. Non-cooperative multicast and facility location games. In *Proc 7th Conf Electronic Commerce (EC'06)*, pages 72–81, 2006. 69

[CD06] X. Chen and X. Deng. Settling the complexity of two-player Nash equilibrium. In *Proc 47th Symp Foundations of Computer Science (FOCS'06)*, pages 261–272, 2006. 3

[CDR03] R. Cole, Y. Dodis, and T. Roughgarden. Pricing network edges for heterogeneous selfish users. In *Proc 35th Symp Theory of Computing (STOC'03)*, pages 521–530, 2003. 3

[CH06] J. Cardinal and M. Hoefer. Selfish service installation in networks. In *Proc 2nd Workshop Internet & Network Economics (WINE'06)*, pages 174–185, 2006. i

[CHKMS06] C. Chekuri, M.T. Hajiaghayi, G. Kortarz, and M.-Salavatipour. Approximation algorithms for non-uniform buy-at-bulk network design. In *Proc 47th Symp Foundations of Computer Science (FOCS'06)*, pages 677–686, 2006. 97

[CK05a] G. Christodoulou and E. Koutsoupias. On the price of anarchy and stability of correlated equilibria of linear congestion games. In *Proc 13th European Symp Algorithms (ESA'05)*, pages 59–70, 2005. 3

[CK05b] G. Christodoulou and E. Koutsoupias. The price of anarchy in finite congestion games. In *Proc 37th Symp Theory of Computing (STOC'05)*, pages 67–73, 2005. 3

[CKV02] A. Czumaj, P. Krysta, and B. Vöcking. Selfish traffic allocation for server farms. In *Proc 34th Symp Theory of Computing (STOC'02)*, pages 287–296, 2002. 3

[CMN04] A. Clauset, C. Moore, and M. Newman. Finding community structure in very large networks. *Physical Review E*, 70(066111), 2004. 125

[CMS06] G. Christodoulou, V. Mirrokni, and A. Sidiropoulos. Convergence and approximation in potential games. In *Proc 23th Symp Theoretical Aspects of Computer Science (STACS'06)*, pages 349–360, 2006. 21, 110

[CP05] J. Corbo and D. Parkes. The price of selfish behavior in bilateral network formation. In *Proc 24th Symp Principles of Distributed Computing (PODC'05)*, 2005. 69

[CR06] H.-L. Chen and T. Roughgarden. Network design with weighted players. In *Proc 18th Symp Parallelism in Algorithms and Architectures (SPAA'06)*, pages 29–38, 2006. 26

[CV02] A. Czumaj and B. Vöcking. Tight bounds for worst-case equilibira. In *Proc 13th Symp Discrete Algorithms (SODA'02)*, pages 413–420, 2002. 3

[DA05] J. Duch and A. Arenas. Community Detection in Complex Networks using Extremal Optimization. *Physical Review E*, 72(027104), 2005. 125

[Dan63] G. Dantzig. *Linear Programming and Extensions.* Princeton University Press, 1963. 3

[DGH03] D. Dutta, A. Goel, and J. Heidemann. Oblivious AQM and Nash equilibrium. In *Proc 22nd Ann Joint Conf IEEE Computer and Communications Societies (INFOCOM)*, 2003. 3

[DGK$^+$05] N. Devanur, N. Garg, R. Khandekar, V. Pandit, A. Saberi, and V. Vazirani. Price of anarchy, locality gap, and a network service provider game. In *Proc 1st Workshop Internet & Network Economics (WINE'05)*, pages 1046–1055, 2005. 3, 53

[DGP06] C. Daskalakis, P. Goldberg, and C. Papadimitriou. The complexity of computing a Nash equilibrium. In *Proc 38th Symp Theory of Computing (STOC'06)*, pages 71–78, 2006. 3

[DIN97] X. Deng, T. Ibaraki, and H. Nagamochi. Combinatorial optimization games. In *Proc 8th Symp Discrete Algorithms (SODA'97)*, pages 720–729, 1997. 3, 32, 62, 69

[DJ03] B. Dutta and M. Jackson, editors. *Networks and Groups - Models of Strategic Formation.* Springer Verlag, 2003. 4, 105

[DMHZ07] E. Demaine, H. Mahini M.T. Hajiaghayi, and M. Zadimoghaddam. The price of anarchy in network creation games. In *Proc 26th Symp Principles of Distributed Computing (PODC'07)*, 2007. 69

[DMV05] N. Devanur, M. Mihail, and V. Vazirani. Strategyproof cost-sharing mechanisms for set cover and facility location problems. *Decision Support Systems*, 39(1):11–22, 2005. 31, 32

[dVV03] S. de Vries and R. Vohra. Combinatorial auctions: A survey. *INFORMS J Comp*, 15:284–309, 2003. 4

[DW72] S. E. Dreyfus and R. A. Wagner. The steiner problem in graphs. *Networks*, 1:195–207, 1972. 76, 81

[Eav73] B.C. Eaves. Polymatrix games with joint constraints. *SIAM J Appl Math*, 24:418–423, 1973. 106

[EGG06] E. Elkind, L. Goldberg, and P. Goldberg. Nash equilibria in graphical games on trees revisited. In *Proc 7th Conf Electronic Commerce (EC'06)*, pages 100–109, 2006. 106

[EGG07] E. Elkind, L. Goldberg, and P. Goldberg. Computing good Nash equilibria in graphical games. In *Proc 8th Conf Electronic Commerce (EC'07)*, 2007. to appear. 106

[Ell93] G. Ellison. Learning, local interaction, and coordination. *Econometrica*, 61(5):1047–1071, 1993. 110

[ELT93] H. Eiselt, G. Laporte, and J.-F. Thisse. Competitive location models: A framework and bibliography. *Transportation Science*, 27:44–54, 1993. 53

[Eve57] H. Everett. Recursive games. In H. Kuhn and A. Tucker, editors, *Contributions to the theory of games*, volume 39, pages 47–78. Princeton University Press, 1957. 17

[FB07] S. Fortunato and M. Barthelemy. Resolution Limit in Community Detection. *Proc National Academy of Sciences*, 104(1):36–41, 2007. 125

[FJ97] A. Frieze and M. Jerrum. Improved approximation algorithms for MAX k-CUT and MAX BISECTION. *Algorithmica 18*, 18:67–81, 1997. 114, 144

[FKL$^+$06] A. Fiat, H. Kaplan, M. Levy, S. Olonetzky, and R. Shabo. On the price of stability for designing undirected networks with fair cost allocations. In *Proc 33rd Intl Coll Automata, Languages and Programming (ICALP'06)*, volume 1, pages 608–618, 2006. 3, 68

[FLGC02] G. Flake, S. Lawrence, C. Lee Giles, and F. Coetzee. Self-organization and identification of web communities. *IEEE Computer*, 35(3):66–71, 2002. 105

[FLM$^+$03] A. Fabrikant, A. Luthera, E. Maneva, C. Papadimitriou, and S. Shenker. On a network creation game. In *Proc 22nd Symp Principles of Distributed Computing (PODC'03)*, pages 347–351, 2003. 69, 110

[FPP06] P. Fine, E. Di Paolo, and A. Philippides. Spatially Constrained Networks and the Evolution of Modular Control Systems. In *9th Intl. Conf. Simulation of Adaptive Behavior (SAB'06)*, pages 546–557, 2006. 125

[FPS01] J. Feigenbaum, C. Papadimitriou, and S. Shenker. Sharing the cost of multicast transmissions. *JCSS*, 63(1):21–41, 2001. 4

[FPSS02] J. Feigenbaum, C. Papadimitriou, R. Sami, and S. Shenker. A BGP-based mechanism for lowest-cost routing. In *Proc 21st Symp Principles of Distributed Computing (PODC'02)*, pages 173–182, 2002. 4

[FPT04] A. Fabrikant, C. Papadimitriou, and K. Talwar. The complexity of pure Nash equilibria. In *Proc 36th Symp Theory of Computing (STOC'04)*, pages 604–612, 2004. 21

[FRT04] J. Fakcharoenphol, S. Rao, and K. Talwar. A tight bound on approximating arbitrary metrics by tree metrics. *JCSS*, 69(3):485–497, 2004. 96

[GA05] R. Guimerá and L.A.N. Amaral. Functional cartography of complex metabolic networks. *Nature*, 433:895–900, 2005. 105

[Gae05] M. Gaertler. Clustering. In U. Brandes and T. Erlebach, editors, *Network Analysis: Methodological Foundations*, volume 3418 of *LNCS*, pages 178–215. Springer Verlag, 2005. 123, 125

[GD03] P. Gleiser and L. Danon. Community structure in jazz. *Advances in Complex Systems*, 6:565–573, 2003. 105

[GG06] I. Giotis and V. Guruswami. Correlation clustering with a fixed number of clusters. In *Proc 17th Symp Discrete Algorithms (SODA'06)*, pages 1167–1176, 2006. 112, 114, 122, 125, 136, 142, 144

[GH81] D. Granot and G. Huberman. On minimum cost spanning tree games. *Mathematical Programming*, 21:1–18, 1981. 69

[GJ79] M. Garey and D. Johnson. *Computers and Intractability: A guide to the theory of NP-completeness*. W.H. Freeman and Company, New York, 1979. 7, 14, 17, 35, 128, 144

[GK99] S. Guha and S. Khuller. Greedy strikes back: Improved facility location algorithms. *J Algorithms*, 31:228–248, 1999. 59

[GKPR07] A. Gupta, A. Kumar, M. Pál, and T. Roughgarden. Approximation via cost sharing: Simpler and better approximation algorithms for network design. *J ACM*, 54(3):11, 2007. 69

[GKR00] N. Garg, G. Konjevod, and R. Ravi. A polylogarithmic approximation algorithm for the Group Steiner tree problem. *J Algorithms*, 37:66–84, 2000. 94, 96

[GN04] M. Girvan and M. Newman. Finding and evaluating community structure in networks. *Physical Review E*, 69(026113), 2004. 6, 105, 123, 125

[GS04] M. Goemans and M. Skutella. Cooperative facility location games. *J Algorithms*, 50(2):194–214, 2004. 32, 60, 62, 69

[GSPA04] R. Guimerà, M. Sales-Pardo, and L. A. N. Amaral. Modularity from Fluctuations in Random Graphs and Complex Networks. *Physical Review E*, 70(025101), 2004. 125

[GVR05] S. Goyal and F. Vega-Redondo. Network formation and social coordination. *Games and Economic Behavior*, 50:178–207, 2005. 110

[GW95] M. Goemans and D. Williamson. A general approximation technique for constrained forest problems. *SIAM J Comp*, 24(2):296–317, 1995. 63, 65, 66

[HK05] M. Hoefer and P. Krysta. Geometric network design with selfish agents. In *Proc 11th Conf Computing and Combinatorics (COCOON)*, pages 167–178, 2005. 68

[HMU07] B. Heydenreich, R. Müller, and M. Uetz. Games and mechanism design in machine scheduling - an introduction. *Production and Operations Management*, 2007. to appear. 4

[Hoc96] D. Hochbaum, editor. *Approximation Algorithms for NP-Hard Problems.* PWS Publishing Company, Boston, 1996. 7, 49

[Hoe04] M. Hoefer. Network connnection games. Master's thesis, Technical University of Clausthal, September 2004. 94

[Hoe06a] M. Hoefer. Non-cooperative facility location and covering games. In *Proc 17th Intl Symp Algorithms and Computation (ISAAC)*, pages 369–378, 2006. i, 60

[Hoe06b] M. Hoefer. Non-cooperative tree creation. In *Proc 31st Intl Symp Math Foundations of Computer Science (MFCS'06)*, pages 517–527, 2006. i

[Hoe07a] M. Hoefer. Competitive investment with economies of scale. Manuscript, September 2007. 100

[Hoe07b] M. Hoefer. Non-cooperative tree creation. *Algorithmica*, 2007. to appear. i

[Hot29] H. Hotelling. Stability in competition. *Economic Journal*, 39:41–57, 1929. 53

[How72] J. Howson. Equilibria of polymatrix games. *Management Science*, 18(5):312–318, 1972. 106

[IMM05] N. Immorlica, M. Mahdian, and V. Mirrokni. Limitations of cross-monotonic cost sharing schemes. In *Proc 16th Symp Discrete Algorithms (SODA'05)*, pages 602–611, 2005. 32

[IMN$^+$05] S. Ieong, R. McGrew, E. Nudelman, Y. Shoham, and Q. Sun. Fast and compact: A simple class of congestion games. In *Proc 20th Conf Artificial Intelligence (AAAI'05)*, pages 489–494, 2005. 18

[Jac04] M. Jackson. A survey of models of network formation: Stability and efficiency. In G. Demange and M. Wooders, editors, *Group Formation in Economics; Networks, Clubs and Coalitions*, chapter 1. Cambridge University Press, Cambridge, 2004. 4, 69

[JMM$^+$03] K. Jain, M. Mahdian, E. Markakis, A. Saberi, and V. Vazirani. Greedy facility location algorithms analyzed using dual fitting with factor-revealing LP. *J ACM*, 50(6):795–824, 2003. 63, 64

[JPY88] D. Johnson, C. Papadimitriou, and M. Yannakakis. How easy is local search? *JCSS*, 37:79–100, 1988. 19, 20

[JT04] R. Johari and J. Tsitsiklis. Efficiency loss in a resource allocation game. *Mathematics of Operations Research*, 29(3):407–435, 2004. 26

[JT05] R. Johari and J. Tsitsiklis. Characterization theorems for smooth market-clearing mechanisms. submitted, 2005. 26

[JV01] K. Jain and V. Vazirani. Applications of approximation algorithms to cooperative games. In *Proc 33rd Symp Theory of Computing (STOC'01)*, pages 364–372, 2001. 4, 41, 45, 59, 63, 69

[JW02] M. Jackson and A. Watts. On the formation of interaction networks in social coordination games. *Games and Economic Behavior*, 41:265–291, 2002. 110

[Kar72] R.M. Karp. Reducibility among combinatorial problems. In R.E. Miller and J.W. Thatcher, editors, *Complexity of Computer Computations*. Plenum Press, New York, 1972. 65

[Kha79] L. Khachian. A polynomial algorithm in linear programming (in Russian). *Doklady Adademiia Nauk SSSR*, 244:1093–1096, 1979. English tranlation: Soviet Mathematics Doklady 20: 191–194. 3, 47

[Kho02] S. Khot. On the power of unique 2-prover 1-round games. In *Proc 34th Symp Theory of Computing (STOC'02)*, pages 767–775, 2002. 42

[KKLP97] V. Kann, S. Khanna, J. Lagergren, and A. Panconesi. Hardness of approximating MAX k-CUT and its dual. *Chicago Journal of Theoretical Computer Science*, 1997. 142

[KL70] B. Kerninghan and S. Lin. An efficient heuristic procedure for partitioning graphs. *Bell System Tech Journal*, 49:291–307, 1970. 125

[KLS01] M. Kearns, M. Littman, and S. Singh. Graphical models for game theory. In *Proc 17th Conf Uncertainty in Artificial Intelligence (UAI'01)*, pages 253–260, 2001. 106

[KLS05] J. Könemann, S. Leonardi, and G. Schäfer. A group-strategyproof mechanism for steiner forests. In *Proc 16th Symp Discrete Algorithms (SODA'05)*, pages 612–619, 2005. 69

[KM72] V. Klee and G. Minty. How good is the Simplex algorithm? In O. Shisha, editor, *Inequalities III*, pages 159–175. Academic Press, 1972. 3

[KM03] D. Koller and B. Milch. Multi-agent influence diagrams for representing and solving games. *Games and Economic Behavior*, 45(1):181–221, 2003. 106

[KP99] E. Koutsoupias and C. Papadimitriou. Worst-case equilibria. In *Proc 16th Symp Theoretical Aspects of Computer Science (STACS'99)*, pages 404–413, 1999. 15

[KR03] S. Khot and O. Regev. Vertex cover might be hard to approximate within 2-ϵ. In *Proc. 18th Conf Computational Complexity (CCC)*, page 379, 2003. 42

[Kre89] M. Krentel. Structure in locally optimal solutions. In *Proc 30th Symp Foundations of Computer Science (FOCS'89)*, pages 216–221, 1989. 20

[KT06] J. Kleinberg and E. Tardos. *Algorithm Design.* Pearson / Addison-Wesley, 2006. 47

[LBT03] K. Leyton-Brown and M. Tennenholtz. Local-effect games. In *Proc 18th Intl Joint Conf Artificial Intelligence (ICJAI'03)*, pages 772–780, 2003. 106

[LH64] C. Lemke and J. Howson. Equilibrium points of bimatrix games. *SIAM J Appl Math*, 12:413–423, 1964. 3

[LSW05] X. Li, Z. Sun, and W. Wang. Cost sharing and strategyproof mechanisms for set cover games. In *Proc 22nd Symp Theoretical Aspects of Computer Science (STACS'05)*, pages 218–230, 2005. 32

[Man03] N.G. Mankiw. *Principles of Economics*. South-Western College Publishers, third edition, 2003. 97

[Meg78] N. Megiddo. Cost allocation for Steiner trees. *Networks*, pages 1–6, 1978. 69

[MFT96] T. Miller, T. Friesz, and R. Tobin. *Equilibrium Facility Location in Networks*. Springer Verlag, 1996. 53

[MP03] R. Mettu and G. Plaxton. The online median problem. *SIAM J Comp*, 32(3):816–832, 2003. 58

[MR99] S. Mahajan and H. Ramesh. Derandomizing approximation algorithms based on semidefinite programming. *SIAM J Comp*, 28(5):1641–1663, 1999. 114, 144

[MRC05] S. Muff, F. Rao, and A. Caflisch. Local Modularity Measure for Network Clusterizations. *Physical Review E*, 72(056107), 2005. 125

[MS96] D. Monderer and L. Shapley. Potential games. *Games and Economic Behavior*, 14:1124–1143, 1996. 10, 11, 12

[MvN47] O. Morgenstern and J. von Neumann. *Theory of Games and Economic Behavior*. Princeton University Press, 1947. 3

[Mye77] R. Myerson. Graphs and cooperation in games. *Mathematics of Operations Research*, 2:225–229, 1977. 4

[MYZ02] M. Mahdian, Y. Ye, and J. Zhang. Improved approximation algorithms for metric facility location problems. In *Proc 5th Workshop Approximation Algorithms for Combinatorial Optimization Problems (APPROX'02)*, pages 229–242, 2002. 56, 63, 64

[MZ91] D. Miller and S. Zucker. Copositive-plus Lemke algorithm solves polymatrix games. *Operations Research Letters*, 10:285–290, 1991. 106

[Nas51] J. Nash. Non-cooperative games. *Annals of Mathematics*, 54(2):286–295, 1951. 2, 9, 12

[New04] M. Newman. Fast algorithm for detecting community structure in networks. *Physical Review E*, 69(066133), 2004. 125

[New05] M. Newman. Modularity and Community Structure in Networks. *Proc National Academy of Sciences*, 103(23):8577–8582, 2005. 125

[Now06] M. Nowak. Five rules for the evolution of cooperation. *Science*, 314:1560–1563, 2006. 105

[NR01] N. Nisan and A. Ronen. Algorithmic mechanism design. *Games and Economic Behavior*, 35:166–196, 2001. 4

[NTRV07] N. Nisan, É. Tardos, T. Roughgarden, and V. Vazirani, editors. *Algorithmic Game Theory.* Cambridge University Press, 2007. 4

[NW88] G. Nemhauser and L. Wolsey. *Integer and combinatorial optimization.* John Wiley & Sons. Inc., 1988. 3, 29

[OPS04] J. Orlin, A. Punnen, and A. Schulz. Approximate local search in combinatorial optimization. *SIAM J Comp*, 33(5):1201–1214, 2004. 50

[OR94] M. Osborne and A. Rubinstein. *A course in game theory.* MIT press, 1994. 7

[Pap94] C. Papadimitriou. On the complexity of the parity argument and other inefficient proofs of existence. *JCSS*, 48(3):498–532, 1994. 3, 106

[Pap01] C. Papadimitriou. Algorithms, games and the internet. In *Proc 33rd Symp Theory of Computing (STOC'01)*, pages 749–753, 2001. 3, 15

[Pap05] C. Papadimitriou. Computing correlated equilibria in multi-player games. In *Proc 37th Symp Theory of Computing (STOC'05)*, pages 49–56, 2005. 106

[Par01] D. Parkes. *Iterative Combinatorial Auctions: Achieving Economic and Computational Efficiency.* PhD thesis, Department of Computer and Information Science, University of Pennsylvania, May 2001. 4

[PDFV05] G. Palla, I. Derenyi, I. Farkas, and T. Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435(7043):814–818, 2005. 105

[Pol03] T. Polzin. *Algorithms for the Steiner problem in networks.* PhD thesis, Universität des Saarlandes, February 2003. 66

[PR05] C. Papadimitriou and T. Roughgarden. Computing equilibria in multiplayer games. In *Proc 16th Symp Discrete Algorithms (SODA'05)*, pages 82–91, 2005. 106

[PS03] B. Peleg and P. Sudhölter. *Introduction to the Theory of Cooperative Games.* Kluwer Academic Publishers, 2003. 32

[PT03] M. Pál and É. Tardos. Group strategyproof mechanisms via primal-dual algorithms. In *Proc 44th Symp Foundations of Computer Science (FOCS'03)*, pages 584–593, 2003. 58, 59, 63

[Qui89] L. Quintas. A note on polymatrix games. *Intl J Game Theory*, 18(3):261–272, 1989. 106

[RB06] J. Reichardt and S. Bornholdt. Statistical Mechanics of Community Detection. *Physical Review E*, 74(016110), 2006. 125

[RE02] T. Roughgarden and É.Tardos. How bad is selfish routing? *J ACM*, 49(2):236–259, 2002. 3

[Ros73] R.W. Rosenthal. A class of games possessing pure-strategy Nash equilibria. *Intl J Game Theory*, 2:65–67, 1973. 10, 11

[Rou03] T. Roughgarden. The price of anarchy is independent of the network topology. *JCSS*, 67(2):341–364, 2003. 3

[Rou04] T. Roughgarden. Stackelberg scheduling strategies. *SIAM J Comp*, 33(2):332–350, 2004. 3

[RW89] G. Reich and P. Widmayer. Beyond Steiner's problem: A VLSI oriented generalization. In *Proc 15th Intl Workshop Graph-Theoretic Concepts in Computer Science (WG'89)*, pages 196–210, 1989. 94

[RZ05] G. Robins and A. Zelikovsky. Tighter bounds for graph Steiner tree approximation. *SIAM J Disc Math*, 19(1):122–134, 2005. 85, 91, 94

[Sch01] M. Schröder. *Gebiete optimal aufteilen.* PhD thesis, School of Economics and Business Engineering, Universität Karlsruhe, 2001. 125

[She95] S. Shenker. Making greed work in networks: A game-theoretic analysis of switch service disciplines. *IEEE/ACM Transactions on Networking*, 3(6):819–831, 1995. 3

[Sim99] B. Simeone. Optimal connected partitions of graphs. DIMACS Tutorial, March 1999. 125

[SK95] D. Skorin-Karpov. On the core of the minimum cost Steiner tree game in networks. *Annals of Operations Research*, 57:233–249, 1995. 69

[SLWC05] Z. Sun, X. Li, W. Wang, and X. Chu. Mechanism design for set cover games when elements are agents. In *Proc 1st Intl Conf Algorithmic Applications in Management (AAIM)*, pages 360–369, 2005. 32

[SM04] A. Schulz and N. Stier Moses. Selfish routing in capacitated networks. *Mathematics of Operations Research*, 29(4):961–976, 2004. 3

[SST04] R. Shamir, R. Sharan, and D. Tsur. Cluster graph modification problems. *Disc Appl Math*, 144(1–2):379–390, 2004. 125

[SV07] A. Skopalik and B. Vöcking. Inapproximability of congestion games. Submitted, 2007. 50

[SvdN01] M. Slikker and A. van den Nouweland. *Social and Economic Networks in Cooperative Game Theory.* Kluwer Academic Publishers, 2001. 4

[SY91] A. Schäffer and M. Yannakakis. Simple local search problems that are hard to solve. *SIAM J Comp*, 20:56–87, 1991. 21

[Vaz00] V. Vazirani. *Approximation Algorithms.* Springer Verlag, Berlin, 2000. 5, 7, 14, 30, 40, 51, 63, 65

[vD91] E. van Damme. *Stability and Perfection of Nash Equilibria.* Springer Verlag, second edition, 1991. 7

[Vet02] A. Vetta. Nash equilibria in competitive societies with application to facility location, traffic routing and auctions. In *Proc 43th Symp Foundations of Computer Science (FOCS'02)*, page 416, 2002. 3, 53

[vSS06] B. von Stengel and R. Savani. Hard-to-solve bimatrix games. *Econometrica*, 74(2):397–429, 2006. 3

[WS05] S. White and P. Smyth. A Spectral Clustering Approach to Finding Communities in Graph. In *SIAM Data Mining Conference*, 2005. 125

[Yan68] E. B. Yanovskaya. Equilibrium points in polymatrix games. *(in Russian) Litovskii Matematicheskii Sbornik*, 8:381–384, 1968. (Math. Reviews 39 #3831). 106

[Yan97] M. Yannakakis. Computational complexity. In E. Aarts and J. Lenstra, editors, *Local Search in Combinatorial Optimization*, chapter 2, pages 19–55. Wiley-Interscience, 1997. 20

[You94] H. P. Young. Cost allocation. In Aumann and Hart [AH02], chapter 34, pages 1194–1235. 26

[You98] H. P. Young. *Individual Strategy and social structure.* Priceton University Press, 1998. 11, 108, 110, 115

[ZMW05] E. Ziv, M. Middendorf, and C. Wiggins. Information-Theoretic Approach to Network Modularity. *Physical Review E*, 71(046117), 2005. 125

List of Symbols

Index