Approximation Algorithms for Distributed and Selfish Agents

by

Vahab S. Mirrokni

B.S., Sharif University of Technology, 2001

Submitted to the Department of Mathematics
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2005

Author . . . .
Department of Mathematics
April 25, 2005

Certified by . . . .
Michel X. Goemans
Professor of Applied Mathematics
Thesis Supervisor

Accepted by . . . .
Rodolfo Ruben Rosales
Chairman, Applied Mathematics Committee

Accepted by . . . .
Pavel I. Etingof
Chairman, Department Committee on Graduate Students

به نام دوست
و به یاد او که خواهد آمد.

تقدیم به پدر عزیز
و مادر مهربانم.

Approximation Algorithms for Distributed and Selfish Agents

by

Vahab S. Mirrokni

Submitted to the Department of Mathematics
on April 25, 2005, in partial fulfillment of the
requirements for the degree of
DOCTOR OF PHILOSOPHY

Abstract

Many real-world systems involve distributed and selfish agents who optimize their own objective function. In these systems, we need to design efficient mechanisms so that system-wide objective is optimized despite agents acting in their own self interest. In this thesis, we develop approximation algorithms and decentralized mechanisms for various combinatorial optimization problems in such systems.

First, we investigate the *distributed caching* and a general set of *assignment problems*. We develop an almost tight LP-based $1 - \frac{1}{e} - \epsilon$-approximation algorithm and a local search $\frac{1}{2} - \epsilon$-approximation algorithm for these problems. We also design efficient decentralized mechanisms for these problems and study the convergence of the corresponding games.

In the following chapters, we study the speed of convergence to high quality solutions on (random) best-response paths of players. First, we study the average social value on best response paths in basic-utility, market sharing, and cut games. Then, we introduce the *sink equilibrium* as a new equilibrium concept. We argue that, unlike Nash equilibria, the selfish behavior of players converges to sink equilibria and all strategic games have a sink equilibrium. To illustrate the use of this new concept, we study the social value of sink equilibria in weighted selfish routing (or weighted congestion) games and valid-utility (or submodular-utility) games. In these games, we bound the average social value on random best-response paths for sink equilibria..

Finally, we study cross-monotonic cost sharings and group-strategyproof mechanisms. We study the limitations imposed by the cross-monotonicity property on cost-sharing schemes for several combinatorial optimization games including set cover and metric facility location. We develop a novel technique based on the probabilistic method for proving upper bounds on the budget-balance factor of cross-monotonic cost sharing schemes, deriving tight or nearly-tight bounds for these games. At the end, we extend some of these results to group-strategyproof mechanisms.

Thesis Supervisor: Michel X. Goemans
Title: Professor of Applied Mathematics

Acknowledgments

This thesis and the growth in my knowledge owe a great deal to many teachers, colleagues, and friends. I would like to use this opportunity to thank them all, for their enthusiasm, encouragement and support, without which I could not have written this thesis.

First and foremost, I would like to thank my advisor, Michel Goemans. I benefited enormously from his insightful comments on my research and his perspective in combinatorial optimization. I am grateful to him for giving me the freedom in pursuing my broad interest in algorithm design, and helping me to clarify my ideas and explanation; and for his careful reading of this thesis. I am also grateful to David Karger for his support as an informal advisor. David's persistence in tackling problems and his enthusiasm in doing research will always be an inspiration for me. I would also like to thank the other members of my thesis committee, Lisa Fleischer, Daniel Kleitman, and Andreas Schulz, for taking the time and effort to read my thesis and giving me very useful comments.

Before coming to MIT, I had the good fortune to go to Shahid Soltani high school and Sharif university of technology in Iran. I am indebted to my high school teachers (and in particular, S. Fazli) for their encouragement that forms my interest in science. At Sharif university, I was lucky to work with many excellent advisors and students. I would like to thank S. Akbari, M. Ghodsi, M. Jamzad, and E. Mahmoodian for all their support and advice to build my career.

During my years at MIT, I visited Bell-Laboratories, IBM T. J. Watson research center and Microsoft Research several times. I have found, in these visits, many exciting problems and fantastic collaborators. I would like to thank Li E. Li, Bruce Shepherd and Marina Thottan of Bell-Labs; Lisa Fleischer, Tracey Kimbrel, Baruch Scheiber, and Maxim Sviridenko of IBM T.J Watson Research; and finally Victor Bahl, Christian Borgs, Jennifer Chayes, Kamal Jain and Lili Qiu of Microsoft Research.

The environment in computer science and artificial intelligence laboratory (CSAIL) at MIT is one of the most exciting environments in the world for doing research in computer science. I have been fortunate to work with several faculty members in the theory of computation group. I thank Nancy Lynch for her support and encouragement to continue working on power optimization in wireless networks. I thank Piotr Indyk for sharing his ideas and co-authoring a paper on approximate nearest neighbors with me. I am very grateful to Mike Sipser and Daniel Spielman for their support as academic advisors when I entered MIT. At CSAIL, I have had the chance to talk to several brilliant students in the TOC group -in particular Amit, Bobby, Brian, David, Eddie, Fumei, Jan, Maria, Mihai, Mohammad, MohammadTaghi, Nick, Nicole, Reina and Tasos. I also thank Linda Okun of the math department and the staff at CSAIL for their administrative support.

In my four years at MIT I have learned a lot, not only in computer science, but also in many aspects of life. All of this was not possible without the support and sympathy of many of my beloved friends whom I owe a lot. *Sepaas* to (besides the ones in CSAIL) Ali, Eaman, Fardad, Mana, Mehdi, Mohammad, MohammadTaghi, Mohsen, Mohsen, Nicole, Reza, and Saeed and others whose names are missed here. I thank them for their support and friendship.

I spent my years at MIT missing my brothers Vahid, Hadi and Mahdi, and my undergraduate friends. During this time, we have been trying to keep in touch and informed from each others' needs and success.

Finally and most importantly, I would like to thank my dear parents. I owe so much to them that cannot think of a proper way of thanking them. They have been supportive of me throughout my life and I see all my achievements a result of their efforts. This thesis is dedicated to them. *Madar va pedare aziz, az zahamaate shomaa sepaasgozaaram.*

Credits. Chapter 2 is based on a joint work with Fleischer, Goemans, and Sviridenko [23] and a joint work with Goemans, Li, and Thottan [29]. Chapter 3 is based a joint work with Vetta [58] and a joint work with Sidiropolous [56]. Chapter 4 is based on a joint work with Vetta [59]. Chapter 5 is based on a joint work with Immorlica and Mahdian [41].

Related Work. The thesis originally contained another chapter on power optimization for connectivity problems. I have removed this chapter based on a suggestion from the thesis committee (since that chapter was not closely related to the other topics of this thesis). In the removed chapter, we developed localized, distributed, and centralized approximation algorithms for power optimization in fault-tolerant network design in wireless multi-hop networks. In a given edge-weighted graph G, the power of each node is the maximum weight of an edge adjacent to this node. The power of the graph is the sum of powers of the nodes of this graph. We extended the (localized) cone-base topology control algorithm and achieve a localized algorithm to find the k-connected subgraph in unit disk graphs. We also designed distributed and centralized $O(k)$-approximation algorithms for geometric graphs. Finally, the main result of this chapter was a $\min(4k, O(\log^4(n)))$-approximation algorithm for the minimum power k-connected subgraph problem for most edge-weighted graphs. That chapter was based on a joint work with Bahramgiri and Hajiaghayi [5], a joint work with Hajiaghayi and Immorlica [35], and a joint work with Hajiaghayi, Kortsarz, and Nutov [37].

Among my recent papers, there are two other papers whose topics are closely related to the topic of this thesis. One paper in on spectrum sharing games [38] and is a joint work with Halpern, Halldorson, and Li. In this paper, we model the problem of spectrum sharing in wireless networks as a graph coloring problem in a game theoretic setting. We then study the performance and convergence in the resulting game. The other paper is on efficient cell-breathing in wireless LAN [4] and is a joint work with Bahl, Hajiaghayi, Qiu and Saberi. In this paper, we develop a distributed algorithm for power assignment of access points in a wireless local area network. Assuming that clients are selfish and go to the access point with the maximum power strength, we find a power assignment such that when each client gets connected to his favorite access point, none of the access points is overloaded. We use linear-programming duality to prove that if there is enough capacity in the network, there exists an efficient power assignment that satisfies all the clients.

During my years at MIT, I have worked on several other problems in network optimization and algorithm design whose topic is not closely related to this thesis. Among them are stochastic optimization [39, 42], approximate nearest neighbors and locally sensitive hashing [11], facility location problems with general cost function [36], cycle covers with short cycles [40], and other work in network optimization [19, 57, 54, 6].

My recent collaborators are Victor Bahl, Mohsen Bahramgiri, Randeep Bahtia, Mayur Datar, Joan Feigenbaum, Lisa Fleischer, MohammadTaghi Hajiaghayi, Magnus Halldorson, Joseph Halpern, Michel Goemans, Nicole Immorlica, Piotr Indyk, Kamal Jain, Tracy Kimbrel, David Karger, Guy Kortsarz, Li E. Li, Mohammad Mahdian, Maria Minkoff, Seffi Naor, Zeev Nutov, Lili Qiu, Amin Saberi, Rahul Sami, Baruch Scheiber, Anastasios Sidiropoulos, Andreas Schulz, Maxim Sviridenko, Marina Thottan, Adrian Vetta. I thank all of them for sharing their valuable time and insights with me.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Over the past fifty years, computer scientists have been trying to overcome different computational constraints in developing efficient algorithms. The main features traditionally characterizing efficient algorithms are their running time and space complexity. However, real-world systems require other important features such as the ability to run as a mechanism in the presence of selfish agents and the ability to run as a decentralized algorithm in a distributed setting. Complexity theory has taught us that designing efficient algorithms with these features is hard. Thus, it sometimes becomes necessary to settle for approximate solutions. In this thesis, we focus on developing approximation algorithms for various combinatorial optimization problems with these features.

With the Internet developing as the single most important arena for resource sharing among parties with diverse and selfish interests, traditional algorithmic and distributed systems approaches are insufficient. The Internet embodies a new paradigm in which distributed computation is performed by self-interested agents. Unable to control the algorithms and strategies employed by distributed users, we must instead design incentives to promote effective system-wide coordination. The goal of mechanism design is to implement some rules so that a system-wide objective is optimized despite agents acting in their own self interest. In this thesis, we consider various distributed settings in the Internet and wireless networks and design efficient approximation algorithms and decentralized mechanisms with a provable performance guarantee in terms of a desired objective function. In this chapter, we first give an overview of the results throughout the thesis. Then, we define some preliminaries that we need throughout the thesis. We suggest unfamiliar readers to first read Section 1.2 to learn the definitions.

1.1 Results and the Structure of the Thesis

The thesis is organized in six chapters. In this part of the introduction, we survey the main results of the other chapters (sometimes with informal definitions). Some formal preliminaries are also given at the end of the introduction in Section 1.2. Each chapter also contains an introduction and a set of preliminaries with formal definitions. We emphasize that in order to understand the details of the results of each chapter, the reader may need to read the formal definitions in the corresponding chapter.

In Chapter 2, we address the content distribution in service provider networks and design approximation algorithms and decentralized mechanisms for a *distributed caching problem* and a general set of assignment problems called the *separable assignment problems (*SAP*)*. Here, we give an informal definition of the distributed caching problem[1]. A service provider has a limited set of file caches. Each cache location has a storage capacity and a bandwidth limit. There are a set of requests, each of a particular file type, and a particular bandwidth. Given a set of requests for file types, where each file type has a size, the job of the service provider is to decide 1) which file types to store at each cache location, subject to storage capacity; and 2) which subset of requests to answer, subject to file selection at the cache, and available bandwidth. While the storage space for a particular type in a cache location is independent of the number of requests for that type, the bandwidth required to serve requests of the same type is the sum of bandwidth requirements for the individual requests. For each potential assignment of cache to request, there is an associated profit: the profit of assigning a request to a cache depends on the connection cost for the request to the cache. The goal is to maximize the profit of providing requests. DCP is a special case

[1]For formal definition, see Chapter 2.

of a general class of assignment problems called the *separable assignment problems.* In a *separable assignment problem*, we are given a set of n bins and a set of m items, and a value f_{ij} for assigning item j to bin i. We are also given a separate packing constraint for each bin i, i.e., a lower-ideal family $\mathcal{I}_i$ of feasible subsets of items for bin i. The goal in the SAP is to find an assignment of items to bins with the maximum aggregate value. We call this separate packing constraint for each bin, *the single-bin subproblem.*

In an instance of the *single-bin subproblem*, we are given a bin i, a set of items with value v_j for each item j, and a packing constraint for bin i, i.e., a lower-ideal family $\mathcal{I}_i$ of feasible subsets of items that can be packed in bin i. A β-approximation algorithm for $\beta \leq 1$ for the single-bin subproblem is an algorithm that outputs a feasible subset of items whose value is at least β times the value of the feasible packing of items in bin i with the maximum value.

Separable assignment problems include the distributed caching problems and the well-known *maximum generalized assignment problem (*GAP*)*: Given a set of bins and a set of items that have a different size and value for each bin, pack a maximum-valued subset of items into the bins.

Given a β-approximation algorithm for the single-bin subproblem, we design a polynomial-time LP-based $((1-\frac{1}{e})\beta)-\delta$-approximation algorithm and a local search combinatorial $\frac{\beta}{\beta+1}-\delta$-approximation algorithm for SAP, for any $\delta > 0$. This gives a $(1-\frac{1}{e}-\epsilon)$-approximation algorithm and a local search $\frac{1}{2}-\epsilon$-approximation algorithm for GAP and DCP, for any $\epsilon > 0$, as the single-bin subproblem for GAP and DCP admit a PTAS. This result is an improvement over the best previously known approximation algorithm for GAP (an LP-based $\frac{1}{2}$-approximation) by Shmoys and Tardos [82] and Chekuri and Khanna [10]. Our algorithm is based on rounding a new linear programming relaxation, with a provably better integrality gap.

At the end of Chapter 2, we also design decentralized mechanisms for the DCP (that can be generalized for problems in SAP). In this part, we assume that caches are selfish entities that want to maximize their own reward. This setting is particularly justified in the context of 3G cellular networks. We define the following rewarding mechanism for the distributed caching problem: Let cache locations decide on the set of file types and files that they can provide based on their constraints. We assume that the profit of each file request goes to the cache location that has the least connection cost to this file request and can provide it and prove that the price of anarchy of a (mixed) Nash equilibrium of the corresponding games is at most 2. We also study the convergence in the general DCP games and the complexity of finding Nash equilibria in these games. We show that pure Nash equilibria may not exist in this game. We prove existence of exponentially long best-response walks to equilibria using a reduction from a PLS-complete problem.

In Chapters 3 and 4, we study convergence in competitive games. We study the speed of convergence to approximate solutions in general classes of games including the cut games, basic-utility games, market sharing games, selfish routing games, and a general set of submodular-utility games (called the valid-utility games). We investigate the social value of states after a number of best responses of players as a measure of the cost of the lack of coordination in such games. This work deviates from other attempts to study the outcome of the selfish behavior of players in non-cooperative games in that we dispense with the insistence upon only evaluating Nash equilibria. Our basic model uses the underlying best-response graph induced by the selfish behavior of the players. In this model, we study the expected social value after a random sequence of best responses or the value of the social function after multiple rounds of best response behavior.

In Chapter 3, we study the speed of convergence in several subclasses of the potential games, i.e., games in which selfish behavior of players converge to a pure Nash equilibrium [2]. First, we study the convergence in the *cut game*. We prove fast convergence to constant-factor approximate solutions on random best-response walks in these games. Then we exhibit exponentially long fair best-response walks with poor social value in this game. In addition, we suggest a way to modify the game to enforce fast convergence to constant-factor solutions after one round of best responses of players in any order. In basic-utility games, we prove fast convergence to $\frac{1}{2} - \epsilon$-

[2]For the formal definitions, see Section 1.2.

approximate solutions after a polynomial-size random best-response walk. Finally, we prove that in market sharing games, after just one round of iterative selfish behavior, the social value is within $\Omega(\frac{1}{\log n})$ of the optimal social value.

In Chapter 4, we study convergence in games with cycles of best responses. In these games, a sequence of best responses may converge to a *sink equilibrium*. A sink equilibrium is a set of strategy profiles or a set of states that is closed under the best responses of players. We introduce this new equilibrium concept and define it formally and study the social value of sink equilibria in comparison to the optmail social value. In particular, we measure the social value of states after a random sequence of best responses of players and bound the expected social value of states in sink equilibria [3]. We study a weighted unsplittable selfish routing game and bound the expected social value of states in any sink equilibrium of this game. An implication of our result is that in weighted selfish routing games, if the delay functions of edges of the network are bounded-degree polynomial functions and we let players play their best responses in a random order, after a polynomial number of best-response moves, the total delay of players is a constant-factor approximation of the total delay of the optimal routing. This is in contrast to the negative result of Fabrikant et al. [15] that shows the existence of exponential best-response walks to Nash equilibria. Thus, even though the convergence to pure Nash equilibria might be poor, the convergence to approximately good solutions is fast. Finally, in Chapter 4, we show that even though the price of anarchy for Nash equilibria in valid-utility games is $\frac{1}{2}$, players may converge to a set of states with social value $\frac{1}{n}$ of that of the optimum. This shows that even in games with small price of anarchy for mixed Nash equilibria, selfish behavior of players may converge to a set of states with poor social value. In addition, using a reduction from a PLS-complete problem, we show existence of states that are exponentially far from any sink equilibrium in valid-utility games.

In Chapter 5, we study cross-monotone cost sharings and group-strategyproof mechanisms[4]. A cost sharing scheme is a set of rules defining how to share the

[3]For the formal definitions, see Chapter 4.

[4]For the formal definitions of these concepts, see Chapter 5.

cost of a service (often computed by solving a combinatorial optimization problem) amongst serviced customers. A cost sharing scheme is cross-monotonic if it satisfies the property that everyone is better off when the set of people who receive the service expands. A main application of cross-monotonic cost sharing schemes is in the design group-strategyproof mechanisms, i.e., mechanisms that are truthful for any coalition of players. An example of the use of cost sharings in mechanism design is in sharing the cost of multicast transmissions [44, 21]. We study the limitations imposed by the cross-monotonicity property on cost-sharing schemes for several combinatorial optimization games including edge cover, vertex cover, set cover, metric facility location, maximum flow, arborescence packing, and maximum matching. We develop a novel technique based on the probabilistic method for proving upper bounds on the budget-balance factor of cross-monotonic cost sharing schemes, deriving tight or nearly-tight bounds for each game that we study.

For the set cover game, that generalizes many of the above games, we show that no cross-monotonic cost sharing scheme can recover more than an $O(\frac{1}{n})$ fraction of the total cost, and thus we can not hope to use a set-cover cost sharing scheme as a black box for the cost sharing schemes of covering games. For the vertex cover game, we show no cross-monotonic cost sharing scheme can recover more than a $O(n^{-1/3})$, demonstrating that cross-monotonicity is strictly harder to achieve than the core property (vertex cover games have a solution in the core that is 1/2-budget balanced). For the facility location game, we show that there is no cross-monotonic cost sharing scheme that recovers more than a third of the total cost. This result together with a recent 1/3-budget-balanced cross-monotonic cost sharing scheme of Pal and Tardos [69] closes the gap for the facility location game.

1.2 Preliminaries

An α-approximation algorithm A for a maximization (minimization) problem is a polynomial-time algorithm that for every instance of the problem computes a solution whose cost is at least (at most) α times the cost of the optimal solution on that instance. We say that α is the approximation factor or the approximation ratio of algorithm A. In particular, $\alpha \geq 1$ for minimization problems and $\alpha \leq 1$ for maximization problems.

Now, we define game theoretic notations that we need in this thesis. For a more complete description of these concepts, we refer to the text books [68] and [25].

Strategic Game. A *strategic game* $\mathcal{G}$ is defined as a tuple $\mathcal{G}(U, \{F_i|i \in U\}, \{\alpha_i()|i \in U\})$ where (i) U is the set of n players or agents, (ii) F_i is a family of feasible *(pure) strategies* or *actions* for player i and (iii) $\alpha_i : \Pi_{i\in U} F_i \to \mathbb{R}^+ \cup \{0\}$ is the (private) *payoff* or *utility* function for agent i, given the set of strategies of all players. Player i's strategy is denoted by $s_i \in F_i$. A *strategy profile* or a (strategy) *state*, denoted by $S = (s_1, s_2, \ldots, s_n)$, is a vector of strategies of players. $\emptyset_i$ corresponds to a null or empty strategy for player i. Also let $S \oplus s'_i := (s_1, \ldots, s_{i-1}, s'_i, s_{i+1}, \ldots, s_k)$, i.e., the strategy profile obtained from S if agent i changes its strategy from s_i to s'_i. The vector of strategies of players except player i is denoted by by $S_{-i} := (s_1, \ldots, s_{i-1}, \cdot, s_{i+1}, \ldots, s_n)$. Also let $\mathcal{F} := \Pi_{i\in U} F_i$ be the set of all possible strategy profiles.

Non-cooperative Game. In a *non-cooperative* game, it is assumed that each agent wishes to maximize its own payoff. In other words, agents are selfish and given the strategy profile for other players, they play a strategy that maximizes their own utility function. For a strategy profile $S = (s_1, s_2, \ldots, s_n)$, an *improvement move* of player i is a strategy s'_i such that $\alpha_i(S \oplus s'_i) \geq \alpha_i(S)$. For a strategy profile $S = (s_1, s_2, \ldots, s_n)$, a *strict improvement move* of player i is a strategy s'_i such that $\alpha_i(S \oplus s'_i) > \alpha_i(S)$. Also, for a strategy profile $S = (s_1, s_2, \ldots, s_n)$ a *best response* of player i in S is a strategy $s^*_i \in F_i$ such that for any strategy $s_i \in F_i$, $\alpha_i(S \oplus s^*_i) \geq \alpha_i(S \oplus s_i)$. Note that a best response is an improvement move.

(Pure) Nash equilibria. A pure strategy Nash equilibrium (PSNE) of a strategic game is a strategy profile in which each player plays his best response. More formally, $S = (s_1, s_2, \ldots, s_n)$ is a pure strategy Nash equilibrium (PSNE) if for all $i \in U$ and for any strategy $s'_i \in F_i$, $\alpha_i(S) \geq \alpha_i(S \oplus s'_i)$.

Mixed Nash equilibria. A *mixed (randomized) strategy* for player i is a probability distribution $\pi_i : F_i \to \mathbb{R}^+ \cup \{0\}$ over pure strategies where $\sum_{s_i \in F_i} \pi_i(s_i) = 1$. A *mixed strategy profile*, denoted by $P = (\pi_1, \pi_2, \ldots, \pi_n)$, is a vector of mixed strategies of all players. For a mixed strategy profile P the probability of realizing a pure strategy profile $S = (s_1, \ldots, s_n)$ is $p^P(s_1, \ldots, s_n) := \Pi_{i \in U} \pi_i(s_i)$. The *expected payoff* of player i in the mixed strategy profile P is $\beta_i(P) := \sum_{s_1 \in F_1, \ldots, s_n \in F_n} p^P(s_1, \ldots, s_n) \alpha_i(s_1, s_2, \ldots, s_n)$. Similar to the above definition of pure Nash equilibrium, we can define a mixed Nash equilibrium. A mixed strategy Nash equilibrium is a mixed strategy profile in which each player does not have an incentive to play a different mixed strategy and improve her expected payoff. Nash [64] proved that any non-cooperative strategic game possesses a mixed Nash equilibrium. We denote Nash equilibrium by NE and the pure strategy Nash equilibrium by PSNE.

State Graph. In order to model the selfish behavior of players, we use the underlying *state graph*. Each vertex in the state graph represents a strategy profile or a state $S = (s_1, s_2, \ldots, s_n)$. The arcs in the state graph correspond to best-response moves by the players. Formally, the state graph, $\mathcal{D} = (\mathcal{F}, \mathcal{E})$, of a strategic game $\mathcal{G}$, is an arc-labeled directed graph, where the vertex set $\mathcal{F}$ corresponds to the set of strategy profiles or states in $\mathcal{G}$, and there is an arc from state S to state S' with label i if the only difference between S and S' is in the strategy of player i; and player i plays his best response or one of his best responses in strategy profile S'[5].

Best-Response Walk. Observe that the state graph will contain loops. A *best-response walk* is a directed walk in the state graph. We say that a player i plays in the best-response walk $\mathcal{P}$, if at least one of the edges of $\mathcal{P}$ is labeled i.

Social Value. The *social (utility) function* or *social value*, denoted by $\gamma : \Pi_{i \in U} F_i \to \mathbb{R}$, is defined for all strategy profiles in a strategic game. The *optimal strategy profile* is the strategy profile that optimizes the social value. The optimal strategy profile is called *the optimal solution*. Also, the social value of the optimal solution, denoted by OPT, is *the optimum*. Also let $\gamma'_{s'_i}(S) := \gamma(S \oplus s'_i) - \gamma(S)$, i.e., the increase in the

[5] In the definition of the best-response moves and the state graph, we only consider the myopic best responses. Other models can be considering nonmyopic players who can predict the responses of other players and play a strategy that gives more payoff in the long-term and not only at the current strategy profile.

social value if player i plays s'_i. The expected social value of a mixed Nash equilibrium $P = (\pi_1, \ldots, \pi_n)$ is $\gamma(P) := \sum_{s_1 \in F_1, \ldots, s_n \in F_n} p^P(s_1, \ldots, s_n)\gamma(s_1, \ldots, s_n)$.

Price of Anarchy. The major tool for analyzing the lack of coordination for some games is the notion of the *price of anarchy* in a game [53, 70]. Given a strategic game, $\mathcal{G}(U, \{F_i | i \in U\}, \{\alpha() | i \in U\})$, and a maximization social function $\gamma : \Pi_{i \in U} F_i \to \mathbb{R}$, the price of anarchy, denoted by $\mathsf{poa}(\mathcal{G}, \gamma)$, is the worst ratio between the optimum and the social value of a pure Nash equilibrium [70][6]. Formally, if $\mathcal{N}$ is the set of all pure Nash equilibria, then $\mathsf{poa}(\mathcal{G}, \gamma) := \frac{\mathrm{OPT}}{\min_{S \in \mathcal{N}} \gamma(S)}$. Similarly, if $\mathcal{P}$ is the set of all mixed Nash equilibria, the price of anarchy for mixed Nash equilibria is equal to $\frac{\mathrm{OPT}}{\min_{P \in \mathcal{P}} \gamma(P)}$. Also, the price of anarchy for a minimization social function γ is defined as $\frac{\max_{P \in \mathcal{P}} \gamma(P)}{\mathrm{OPT}}$.

Submodular Set Functions. A function f of the form $2^V \to \mathbb{R}^+ \cup \{0\}$ is called a set function on the ground set V. A set function $f : 2^V \to \mathbb{R}^+ \cup \{0\}$ is submodular if for any two sets $A, B \subseteq V$, $f(A) + f(B) \geq f(A \cap B) + f(A \cup B)$. A set function f, is non-decreasing if $f(X) \leq f(Y)$ for any $X \subseteq Y \subseteq V$.

Valid-Utility Game. One class of games that we consider in different chapters of this thesis is the class of valid-utility games introduced by Vetta [87]. We will use the definition of these games in Chapters 2, 3 and 4. Here, we describe an abstract definition of these games and refer the reader to the paper by Vetta [87] and Goemans et al. [29] for several examples of these games. In these games, for each player i, there exists a ground set V_i. Note that V_i's may intersect. We denote by V the union of ground sets of all players, i.e., $V = \cup_{i \in U} V_i$[7]. The feasible strategy set F_i of player i is a subset of the power set of V_i, 2^{V_i} that contains the empty set $\emptyset$ which corresponds to having no action. Thus, the strategy s_i of player i is a subset of V_i ($s_i \subseteq V_i$). In order to define these games, we need the following definitions.

Given a vector $S = (s_1, \ldots, s_n)$, where s_i is a subset of the ground set V_i ($s_i \subseteq V_i$), the set $\mathcal{H}_S = \{(i, j) : i \in U, j \in s_i\}$ is called *the pair set* for vector S. Note that S may or

[6] In this thesis, we use the term, price of anarchy, for pure Nash equilibria. For mixed Nash equilibria, we use the term, price of anarchy for mixed Nash equilibria. In some other places, the price of anarchy is defined for mixed Nash equilibria.

[7] We can define the valid-utility games by setting all ground sets $V_i = V$. But for later convenience, we let the ground set of player i be V_i.

may not be a feasible strategy profile. Given a function $f : \Pi_{i\in U}2^V \to \mathbb{R}^+ \cup \{0\}$, the *corresponding set function* f^s of f is a set function of the form $2^{\mathcal{H}} \to \mathbb{R}^+ \cup \{0\}$ where $\mathcal{H} = \{(i,j) : i \in U, j \in V\}$ and $f^s(\mathcal{H}_S) = f(S)$. In other words, for a set $A \subseteq \mathcal{H}$, $f^s(A) = f((a_1, a_2, \ldots, a_n))$ if $a_i = \{j : (i,j) \in A\}$. Here, we assume that the social function γ is of the form $\Pi_{i\in U}2^V \to \mathbb{R}^+ \cup \{0\}$ instead of the form $\Pi_{i\in U}F_i \to \mathbb{R}^+ \cup \{0\}$.

Let $\mathcal{G}(U, \{F_i | i \in U\}, \{\alpha_i() | i \in U\})$ be a non-cooperative strategic game where $F_i \subseteq 2^{V_i}$ is a family of feasible strategies for player i. Let $V = \cup_{i\in U}V_i$ and let the social function be $\gamma : \Pi_{i\in U}2^V \to \mathbb{R}^+ \cup \{0\}$. $\mathcal{G}$ is a *valid-utility game* if it satisfies the following properties:

1) **Submodular and Non-decreasing Social Function:** γ^s, the corresponding set function of γ over the set $\mathcal{H} = \{(i,j) : i \in U, j \in V\}$, is submodular and non-decreasing.

2) **Vickrey Condition:** The payoff of a player is at least the difference in the social function when the player participates versus when it does not participate, i.e., $\alpha_i(S) \geq \gamma'_{s_i}(S \oplus \emptyset_i)$. In *basic-utility* games, this is an equality, i.e., $\alpha_i(S) = \gamma'_{s_i}(S \oplus \emptyset_i)$.

3) **Cake Condition:** For any strategy profile, the sum of the payoffs of players should be less than or equal to the social function for that strategy profile, i.e., for any strategy profile S, $\sum_{i\in U} \alpha_i(S) \leq \gamma(S)$.

This framework encompasses a wide range of games including the facility location games, the traffic routing games, auctions [87], market sharing games (described in Chapter 2), and distributed caching games (described in Chapter 2). Vetta [87] proved that the price of anarchy for mixed Nash equilibria in valid-utility games is at most 2. While proving theorems about valid-utility and basic-utility games, we use the following notation: for two vectors $S = (s_1, \ldots, s_n)$ and $S' = (s'_1, \ldots, s'_n)$, we define $S \cup S' := (s_1 \cup s'_1, \ldots, s_n \cup s'_n)$. Also we define $S \cup s'_i := (s_1, s_2, \ldots, s_{i-1}, s_i \cup s'_i, s_{i+1}, \ldots, s_n)$.

Potential Game. *Potential games* are games in which any sequence of strict improvement moves by players converges to a pure Nash equilibrium. Equivalently, in potential games, there is no cycle of strict improvement moves of players. This is equivalent to the existence of a potential function $\mathsf{pot} : \Pi_{i \in U} F_i \to \mathbb{R}^+ \cup \{0\}$ with an upper bound that is strictly increasing after any strict improvement move. *Exact potential games* are a subclass of potential games for which there exists a potential function such that in any strict improvement move, the increase in the payoff of a player is equal to the increase in the potential function.

Congestion Game. Consider a set U of n players and a set V of elements; these elements are called factors in the original work of Rosenthal [76]. Let $F_i \subseteq 2^V$ be a family of subsets of elements. For each element $f \in V$, let $c_f : \mathbb{N} \to \mathbb{R}^+ \cup \{0\}$ be a congestion function. A strategic game $\mathcal{G}(U, \{F_i | i \in U\}, \{\alpha_i() | i \in U\})$ is called a congestion game if $F_i \subseteq 2^V$, and for a strategy profile, $S = (s_1, \ldots, s_n)$, $\alpha_i(S) = \sum_{f \in s_i} c_f(n_f(S))$ where $n_f(S)$ is the number of players that contains f in their strategy. Rosenthal [76] proved that congestion games are exact potential games. For the sake of completeness, we show the potential function for this game. The potential function is $\mathsf{pot}(S) = \sum_{f \in V} \sum_{t=1}^{n_f(S)} c_f(t)$. Let i be a player who changes his strategy from s_i in S to s'_i and increases his payoff. The change in the potential function is equal to $\mathsf{pot}(S \oplus s'_i) - \mathsf{pot}(S) = \sum_{f \in s'_i - s_i} c_f(S \oplus s'_i) - \sum_{f \in s_i - s'_i} c_f(S) = \alpha_i(S \oplus s'_i) - \alpha_i(S)$. Thus, if player i increases his payoff by changing his strategy, the potential function also increases by the same value. This shows that congestion games are exact potential games. Monderer and Shapley [60] proved that congestion games are equivalent to the class of exact potential games.

In the formulation of congestion games, instead of a congestion function, we may consider a delay function, $l_f : \mathbb{N} \to \mathbb{R}^+ \cup \{0\}$ for each element. In this case, given a strategy $S = (s_1, \ldots, s_n)$ of players, the delay of player i is $l_i(S) = \sum_{f \in s_i} l_f(n_f(S))$ where $n_f(S)$ is the number of players who has f in their strategy. The goal of each player is to minimize his delay (instead of maximizing his payoff). By setting $c_f(x) = -l_f(x)$ and $\alpha_i(S) = -l_i(S)$, we can show that these two formulations of congestion games are equivalent in that we can find a one-to-one correspondence ψ between the vertices of the state graphs of these two games such that if there exists an edge from S to S' with label i in the state graph of the first game, then there exists an edge between vertices $\psi(u)$ and $\psi(v)$ with the same label i. One of the well-known congestion games is the selfish routing game [78]. In this game[8], the elements of the game are edges of a directed network. Each player i wants to route a unit amount of flow from a source s_i to a destination, t_i. The set of feasible strategies of a player is the set of directed paths from s_i to t_i. The delay of an edge is a function of the total number of players that use this edge in their path. The delay of a player is the sum of the delay of the edges on its path.

Local Search Problems and the Class PLS. The class of Polynomial Local Search problems (*PLS*) is introduced by Johnson, Papadimitriou, and Yannakakis [47]. A local search problem is denoted by L and is given by a set of instances $\mathcal{I}(\mathsf{L})$. Each instance $I \in \mathcal{I}(\mathsf{L})$ is given by a tuple $I = (\mathcal{F}_I, \gamma : \mathcal{F}_I \rightarrow \mathbb{R}^+ \cup \{0\}, N_I : \mathcal{F}_I \rightarrow 2^{\mathcal{F}_I})$ where (i) $\mathcal{F}_I$ is the set of feasible solutions to instance I, (ii) $\gamma(F)$ is the value of solution $F \in \mathcal{F}_I$, and (iii) $N_I : \mathcal{F}_I \rightarrow 2^{\mathcal{F}_I}$ is a *neighborhood function*, that is, for any feasible solution $F \in \mathcal{F}_I$, it gives a set of feasible solutions that are in the neighborhood of F (in other words, this set of feasible solutions can be reached from F only by one local operation). The *local search problem* is to find a *local optimal solution* for any given instance of L, that is, given an instance $I \in \mathcal{I}(\mathsf{L})$, a feasible solution $F \in \mathcal{F}_I$ such that no feasible solution in the neighborhood of F has value more than $\gamma(F)$. The global optimization problem is to find a feasible solution $F \in \mathcal{F}_I$ with the maximum value $\gamma(F)$ for any given instance I of L. A local search problem L is in class PLS if for any instance $I \in \mathcal{I}(\mathsf{L})$, (i) a feasible solution in $\mathcal{F}_I$ is polynomially computable, (ii) for any solution $F \in \mathcal{F}_I$, $\gamma(F)$ can be computed in polynomial time, and (iii) a polynomial function is given that for any solution $F \in \mathcal{F}_I$ either determines that F is a local optimal solution, or returns $F' \in N_I(F)$ with $\gamma(F') > \gamma(F)$. Therefore, for any local search problem in PLS, given any instance I and any feasible solution $F \in \mathcal{F}_I$, it can be checked in polynomial time if F is a local optimal solution or not, and if it is not, a solution F' in the neighborhood of F ($F' \in N_I(F)$) can be found such that $\gamma(F') > \gamma(F)$. Many well-

[8]In fact, we describe the *atomic* version of the selfish routing game in which there are finite number of players. The other variant of the selfish routing game is the *nonatomic* variant in which there exist infinite number of infinitesimal players each with a small load.

known local search problems are in PLS; For example, the Max-SAT problem with the flip neighborhood, and the Max-Cut problem with the swapping neighborhood. The Max-Cut local search problem with swapping neighborhood is as follows: Given an edge-weighted graph G and a cut, the local operations are to switch one node from one side of the cut to the other side. The class PLS has its own type of reduction and its own complete problems. Formally, a local search problem L in PLS is *PLS-complete*, if using a polynomial-time algorithm to find a local optimal solution of any instance of L, we can design a polynomial-time algorithm to find the local optimal of any problem in PLS. Johnson, Papadimitriou, and Yannakakis [47] introduced this class of problems and proved that the Max-SAT problem with flip neighborhood is PLS-complete. They also introduced the concept of the PLS reduction by which one can prove that a local search problem L' is PLS-complete by reducing a PLS-complete problem to L'. It is not known if PLS-complete problems are hard to solve or not. In fact, Johnson et al. [47] proved that if a problem in PLS is NP-Hard, then NP= co-NP. As a result, it seems unlikely that these problems are NP-Hard, even though no polynomial-time algorithm is known for them. A problem A is PLS-hard if when we can solve A in polynomial time, then we can solve any instance of a PLS-complete problem in polynomial time. In other words, if we can solve a PLS-hard problem in polynomial time then we can solve any PLS problem in polynomial time.

Given any instance of a local search problem, the *neighborhood search algorithm* to find the local optimal solution is the algorithm that starts from a given solution of this instance, and does the following until it finds a local optimal solution: at each step, it finds a feasible solution in the neighborhood of the current solution with a better value, and moves to this solution. Papadimitriou et al. [71] and Schaffer and Yannakakis [79] have shown that for several PLS-complete problems, the neighborhood search algorithm takes exponential time. In fact, they proved that for some PLS-complete local search problems such as the Max-2SAT with flip neighborhood, and the Max-Cut problem with swap neighborhood, there are solutions that are exponentially far from any local optimal solution, i.e., it takes exponential number of local improvements to get to a local optimal solution.

Here, we formally define the PLS reduction. A PLS reduction from L to L' is given by a pair $(\mathcal{R}, \mathcal{M})$ where (i) $\mathcal{R} : \mathcal{I}(\mathsf{L}) \to \mathcal{I}(\mathsf{L}')$ is a polynomial-time computable function that maps any instance of L to an instance of L', and (ii) $\mathcal{M}$ is a polynomial-time computable function that maps any pair (F', I), where I is an instance of L and $F' \in \mathcal{F}_{\mathcal{R}(I)}$ is a solution of $\mathcal{R}(I)$, to a feasible solution F of I, and (iii) for any instance I of L and for any local optimal solution $F' \in \mathcal{F}_{\mathcal{R}(I)}$ of $\mathcal{R}(I)$, $\mathcal{M}(F', I)$ is a local optimal of the instance I of L.

Chapter 2

General Assignment and the Distributed Caching Problems

2.1 Introduction

The growth of the Internet, the World Wide Web and wide-area wireless networks allow increasing number of users to access vast amount of information in different geographic areas. Content delivery is one of the most important tasks of a service provider in these systems. It is well known that content delivery can be done by caching popular items in cache locations close to the users. In this chapter, we address this issue by formalizing different variants of *distributed caching problems* as a set of assignment problems and develop approximation algorithms and decentralized mechanisms for these problems. The general distributed caching problem that we formalize here is denoted by CapIBDC:

CapIBDC: Let U be a set of n cache locations with given available capacities A_i and given available bandwidths B_i for each cache location i. There are k request types; [1] each request type t has a size a_t ($1 \leq t \leq k$). Let H be a set of m requests with a reward R_j, a required bandwidth b_j, a request type t_j for each request j, and a connection cost c_{ij} for each cache location i to each request j. The profit of providing request j by cache location i is $f_{ij} = R_j - c_{ij}$. A cache location i can service a set of

[1] Request type can be thought as different files that should be delivered to clients.

requests S_i, if it satisfies the *bandwidth constraint*: $\sum_{j \in S_i} b_j \leq B_i$, and the *capacity constraint*: $\sum_{t \in \{t_j | j \in S_i\}} a_t \leq A_i$ (this means that the sum of the sizes of the request types of the requests in cache location i should be less than or equal to the available capacity of cache location i). We say that a set S_i of requests is feasible for cache location i if it satisfies both bandwidth and capacity constraints for bin i. The goal of the CapIBDC problem is to find a feasible assignment of requests to cache locations to maximize the total profit; i.e., the total reward of requests that are provided minus the connection costs of these requests.

We also consider the following special cases of the CapIBDC problem:

CapDC: The CapDC problem is a special case of CapIBDC problem without bandwidth constraint. In other words, in the CapDC problem we assume that for each cache location i, B_i is sufficiently large.

IBDC: The IBDC problem is a special case of CapIBDC problem without capacity constraint. In other words, in the IBDC problem we assume that for each cache location i, A_i is sufficiently large. From this definition, one can see that IBDC is a special case of CapDC where there exists exactly one request of each request type, the available capacity of cache locations in CapDC is equal to the available bandwidth of cache locations in IBDC, and the size of request types in CapDC corresponds to the bandwidth requirement of requests in IBDC.

uniform CapDC: The uniform CapDC problem is a special case of the CapDC problem where the size of all request types is the same, i.e., $a_t = a$ for all $1 \leq t \leq k$.

uniform IBDC: The uniform IBDC problem is a special case of the IBDC problem where the bandwidth requirement of all requests is the same, i.e., $b_j = b$ for all $j \in H$.

We refer to all variants of the distributed caching problems as DCP.

In this chapter, we develop approximation algorithms and decentralized mechanisms for DCP, and a general class of assignment problems, called the *separable assignment problems (*SAP*)*:

SAP: In a separable assignment problem, we are given a set U of n bins and a set H of m items, and a value f_{ij} for assigning item j to bin i. We are also given a separate packing constraint for each bin i, i.e., a *lower-ideal*[2] family $\mathcal{I}_i$ of feasible subsets of items for bin i. We call this separate packing constraint for each bin, *the single-bin subproblem*. The goal is to find an assignment of items to bins with the maximum aggregate value.

Separable assignment problems include all variants of the distributed caching problems described above. To see this we note that the family of feasible subsets of requests for a cache location i in the CapIBDC problem is all subsets of requests that satisfy two packing constraints: the capacity constraint and the bandwidth constraint. Thus, it is clear that the family of feasible subsets for a cache location is lower-ideal. Moreover, the profit of each request j to each cache location i is $f_{ij} = R_j - c_{ij}$ and corresponds to the value of item j to bin i in SAP. It follows that CapIBDC, and thus all variants of DCP are special cases of SAP. SAP also includes the well-known *maximum generalized assignment problem (*GAP*)*:

GAP: Given a set of bins and a set of items that have a (possibly) different size and value for each bin, pack a maximum-valued subset of items into the bins.

Picture 2-1 depicts the problems that we consider and their relation to each other.

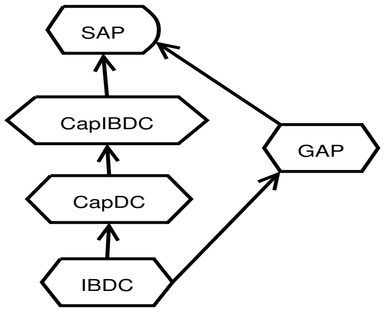


Figure 2-1: Assignment Problems. There is an arrow from problem A to problem B, if A is a special case of B.

We design LP-based and local search approximation algorithms for SAP. Our results depend on an algorithm to solve the single-bin subproblem in SAP. In an instance of the *single-bin subproblem*, we are given a bin i, a set of items with value

[2] A family $\mathcal{I} \subseteq 2^H$ of subsets of a set H is lower-ideal, if $\emptyset \in H$, and for each two subsets S and R such that $R \subseteq S$, if $S \in \mathcal{I}$ then $R \in \mathcal{I}$.

v_j for each item j, and a packing constraint for bin i, i.e., a lower-ideal family $\mathcal{I}_i$ of feasible subsets of items that can be packed in bin i. A β-approximation algorithm for $\beta \leq 1$ for the single-bin subproblem is an algorithm that outputs a subset of items $S \in \mathcal{I}_i$ such that for any other subset $S' \in \mathcal{I}_i$ of items, $\sum_{j \in S} v_j \geq \beta \sum_{j \in S'} v_j$.

In Section 2.2, given a β-approximation algorithm for the single-bin subproblem, we design a polynomial-time LP-rounding based $((1 - \frac{1}{e})\beta - \delta)$-approximation algorithm [3]. In Section 2.3, given a β-approximation algorithm for the single-bin subproblem, we design a simple, polynomial-time local search $(\frac{\beta}{\beta+1} - \epsilon)$-approximation algorithm [4]. For GAP and all variants of DCP, there exists an approximation scheme for the single-bin subproblem, thus we obtain an LP-based algorithm with $(1 - \frac{1}{e} - \epsilon)$ approximation and a local search algorithm with $\frac{1}{2} - \epsilon$ approximation guarantee. To complement these results, in Section 2.5, we show that SAP and DCP cannot be approximated within a factor better than $1 - \frac{1}{e}$ unless NP$\subseteq$ DTIME($n^{O(\log \log n)}$), even if there exists a polynomial-time exact algorithm for the single-bin subproblem.

We generalize the local search algorithm to yield an approximation algorithm for the maximization version of the k-median problem with hard capacity constraints (KMed):

KMed: Given a set U of n bins, a set H of m items with a value f_{ij} for each item j and each bin i, and also a single-bin subproblem for each bin i, i.e., a lower-ideal family $\mathcal{I}_i$ of subsets for bin i, choose at most K bins and pack a set of items in each selected bin to maximize the total value packed.

In Section 2.4, for any $\epsilon > 0$, given a β-approximation algorithm for the single-bin subproblem, we design an $(\frac{\beta}{\beta+1} - \epsilon)$-approximation algorithm for KMed. If the single-bin subproblem is a knapsack problem, this yields a $(\frac{1}{2} - \epsilon)$-approximation algorithm.

We discuss decentralized mechanisms for all variants of the distributed caching problem in Section 2.6. For the decentralized mechanisms, we may assume that

[3] We can set δ to be exponentially small, i.e., $\delta = \frac{1}{2^{cn}}$ for any constant $c > 0$. Throughout the chapter, we say "for any $\delta > 0$" to illustrate this.

[4] For the local search algorithm, we can set ϵ to be exponentially small, i.e., $\epsilon = \frac{1}{2^{cn}}$ for any constant $c > 0$.

caches are selfish entities that want to maximize their own reward. This setting is particularly justified in the context of 3G cellular networks [29] and this is formalized in Goemans, Li, Mirrokni, and Thottan [29]. In this setting, *resident* subscribers are cache locations that can provide content to *clients*. Clients ask for different requests and the service provider wants to cache these requests at resident subscribers to provide them in a distributed manner. This helps the service provider to use the capacity of the wireless network instead of relying solely on its own bandwidth. Goemans et al. [29] formalize the protocol and the system architecture of such mechanisms. In Section 2.6, we suggest a mechanism to induce selfish cache locations to serve the set of requests with a good performance guarantee. All mechanisms extend to all separable assignment problems, but we state the results for the distributed caching problems since the main motivation of this setting comes from this problem.

Previous Work. For the special case of GAP, Shmoys and Tardos [82] present an LP-rounding $\frac{1}{2}$-approximation algorithm for the minimization problem. Chekuri and Khanna [10] observed that a $\frac{1}{2}$-approximation for GAP is implicit in the paper by Shmoys and Tardos [82]. Their method is based on an LP formulation with an integrality gap of at least 2. Thus the LP we introduce here is provably stronger. Chekuri and Khanna [10] develop PTAS's for a special case of this problem called the multiple knapsack problem. In this problem, each item has the same size and the same profit for all cache locations. They also classify the APX-hard special cases of GAP.

Nemhauser, Fisher, and Wolsey, previously look at maximizing submodular functions with and without cardinality constraints [65, 66]. They give a greedy algorithm with approximation guarantee of $1 - \frac{1}{e}$ for maximizing submodular functions, and $\frac{1}{2}$-approximate local search algorithm for maximizing submodular functions with cardinality constraint. The latter paper actually looks more generally at restrictions to matroids. These results do not extend to handle knapsack constraints, since feasible sets for knapsack do not form a matroid. Sviridenko shows that the greedy algorithm gives a $1 - \frac{1}{e}$-approximation for maximizing a submodular function subject to one knapsack constraint [85], but does not consider assignment type problems with sets of packing constraints. Indeed the simple greedy algorithm does worse than $1 - \frac{1}{e}$ even for the multiple knapsack problem. Using LP techniques [1, 83], approximation algorithms with the guarantee of $1 - \frac{1}{e}$ for some maximum coverage problems are known. These techniques are different from ours and cannot handle SAP as the packing constraints in SAP are more general.

Gomes, Regis, and Shmoys [33] use a exponential-size LP and a rounding scheme similar to the one we use here but to obtain a $1-\frac{1}{e}$-approximation algorithm to solve the partial Latin square extension problem. In particular, their LP does not capture knapsack packing constraints.

Baeve and Rajaraman [3] study a problem of data placement in networks. They formalize a minimization version of our problem in which they need to place objects in caches to minimize the total connection costs. They give a constant-factor approximation for this problem, which is improved to factor 10 by Swamy [86]. In the conclusion of [3], they suggest considering the problem with bandwidth as an important extension.

2.2 LP-Based Approximation Algorithms

In this section, for any $\delta > 0$, we give a $((1 - \frac{1}{e})\beta - \delta)$-approximation for SAP and its variants where β is the approximation factor of the algorithm for the single-bin subproblem. The general approach is to formulate an (exponential-size) integer program, solve the linear program relaxation approximately, round the solution to the linear program, and prove that the rounded solution has this guarantee. There are two main issues here: proving the quality of the rounded solution, and obtaining a good solution to the large linear program in polynomial time. We first discuss the approach in the context of SAP and then discuss some extensions.

2.2.1 Separable Assignment Problems

Formulation. We give an exponential-size integer programming formulation for SAP. Let $\mathcal{I}_i$ for $i \in U$ be the set of all feasible assignments of items to bin i; these are the feasible solutions to the single-bin subproblem for bin i. For a set $S \in \mathcal{I}_i$, let X_S^i be the indicator variable that indicates if we choose S as the subset of items for bin i. The first constraint is that we cannot assign more than one set to a bin i, thus for all $i \in U$, $\sum_{S \in \mathcal{I}_i} X_S^i = 1$. Moreover, we cannot assign each item to more than one bin: $\sum_{i, S \in \mathcal{I}_i : j \in S} X_i^S \leq 1$. Our objective is to find an assignment of items to bins to maximize the sum of profits, i.e., $\sum_{i, S \in \mathcal{I}_i} f_i^S X_i^S$ where $f_i^S = \sum_{j \in S} f_{ij}$. By relaxing the 0-1 variables to nonnegative real variables, we obtain the following linear programming relaxation:

$$\begin{array}{lll} \max & \sum_{i \in U, S \in \mathcal{I}_i} f_i^S X_i^S & \\ \text{s.t.} & \sum_{i \in U, S \in \mathcal{I}_i : j \in S} X_i^S \leq 1 & \forall j \in H \\ & \sum_{S \in \mathcal{I}_i} X_i^S = 1 & \forall i \in U \\ & X_i^S \geq 0 & \forall i \in U, \forall S \in \mathcal{I}_i \end{array} \tag{2.1}$$

Let LP(SAP) denote the objective function value of this LP.

Rounding the Fractional Solution. Given a solution to the linear program (2.1), independently for each bin i, assign set S to i with probability X_S^i. In the resulting solution, some item j may be contained in the sets assigned to more than one bin. In this case, item j is assigned to the bin among these bins with the maximum f_{ij}-value. Note that the resulting assignment after this step is feasible, since the family $\mathcal{I}_i$ for each bin i is lower-ideal.

Theorem 2.2.1 *The expected value of the rounded solution is at least least* $(1 - \frac{1}{e})$LP*(*SAP*).*

Proof. For item j, sort the bins i for which $Y_i = \sum_{S \in \mathcal{I}_i : j \in S} X_i^S$ is nonzero in the non-increasing order of f_{ij}. Without loss of generality assume that these bins are

$1, 2, \ldots, l$ and $f_{1j} \geq f_{2j} \ldots \geq f_{lj} \geq 0$. With probability Y_1 the set that is assigned to bin 1 contains j, thus item j is assigned to bin 1. In this case, the value of item j is f_{1j}. With probability $(1 - Y_1)Y_2$, bin 1 does not have item j in its subset and bin 2 has item j in its set. In this case, the value of item j is f_{2j}. Proceeding similarly, we obtain that the expected value for request j in the rounded solution is $f_{1j}Y_1 + f_{2j}(1 - Y_1)Y_2 + \ldots + f_{lj}(\Pi_{i=1}^{l-1}(1 - Y_i))Y_l$. The contribution of item j in the value of the fractional solution is $\sum_{1 \leq i \leq l} f_{ij}Y_i$. This in conjunction with Lemma 2.2.2 below yields the result. □

Lemma 2.2.2 $f_{1j}Y_1 + f_{2j}(1 - Y_1)Y_2 + \ldots + f_{lj}(\Pi_{i=1}^{l-1}(1 - Y_i))Y_l \geq (1 - \frac{1}{e}) \sum_{1 \leq i \leq l} f_{ij}Y_i$ *whenever* $Y_i \geq 0$ *for all* i *and* $\sum_i Y_i \leq 1$ *and* $f_{1j} \geq f_{2j} \geq \ldots \geq f_{lj} \geq 0$.

Proof. From the arithmetic/geometric mean inequality, one can derive (see Lemma 3.1 in [31]):

$$1 - (\prod_{i=1}^{k}(1 - Y_i)) = Y_1 + (1 - Y_1)Y_2 + \cdots + (\prod_{i=1}^{k-1}(1 - Y_i))Y_k \geq (1 - \frac{1}{e}) \sum_{i=1}^{k} Y_i,$$

for any k. Multiplying this inequality by $f_{kj} - f_{k+1,j} \geq 0$ where $f_{l+1,j} = 0$ and summing over k, we derive the lemma. □

Solving the LP. The number of variables in the linear program (2.1) is exponential. To solve this LP, we first solve its dual (2.2) given below.

$$\begin{aligned} \min \quad & \textstyle\sum_{i \in U} q_i + \sum_{j \in H} \lambda_j & & \qquad (2.2) \\ \text{s.t.} \quad & q_i + \textstyle\sum_{j \in S} \lambda_j \geq f_i^S & & \forall i \in U, \forall S \in \mathcal{I}_i \\ & \lambda_j \geq 0 & & \forall j \in H. \end{aligned}$$

The dual linear program (2.2) has a polynomial number of variables, but exponentially many constraints. We rewrite it as a fractional covering problem as follows:

$$\begin{array}{lll} \min & \sum_{i \in U} q_i + \sum_{j \in H} \lambda_j & \\ \text{s.t.} & (q_i, \lambda) \in \mathcal{P}_i & \forall i \in U, \forall S \in \mathcal{I}_i \\ & \lambda_j \geq 0 & \forall j \in H. \end{array} \tag{2.3}$$

Here, $\mathcal{P}_i$ is the polytope defined by constraints of the form $q_i \geq \sum_{j \in S}(f_{ij} - \lambda_j)$ for all $S \in \mathcal{I}_i$. To solve the LP, we will need a separation algorithm for $\mathcal{P}_i$. We define an *β-approximate separation algorithm* for polytope $\mathcal{P}_i$ to be an algorithm that given a point $(q_i, \lambda_j | j \in H)$ returns either a violated constraint, or guarantees that $(\frac{q_i}{\beta}, \lambda_j | j \in H)$ is feasible for $\mathcal{P}_i$. We let $\mathsf{LP}(\mathsf{Dual\ SAP})$ denote the objective function value of the linear program (2.2).

Lemma 2.2.3 *For any $\delta > 0$, given a polynomial-time β-approximate separation algorithm for $\mathcal{P}_i$, we can design a polynomial-time $(\beta - \delta)$-approximation algorithm to solve the linear program (2.2) and hence the linear program (2.1).*

Proof. We run the ellipsoid algorithm on the linear program (2.2) using a β-approximate separation algorithm. More precisely, we move the objective into the set of constraints by adding the constraint $\sum_{i \in U} q_i^* + \sum_{j \in H} \lambda_j^* \leq v^*$ to the current linear program. For a given v^*, we use the ellipsoid algorithm to determine if this LP is feasible; and use binary search to find the smallest feasible value v^*. Using the ellipsoid algorithm in this binary search framework with a β-approximate separation algorithm, suppose that the process of the algorithm terminates with a solution $(q_i^*, \lambda_j^* | j \in H)$ such that $v^* = \sum_{i \in U} q_i^* + \sum_{j \in H} \lambda_j^*$. Thus, we know that the linear program (2.2) with the new constraint is infeasible for $v^* - \delta'$ where δ' depends on the precision of the binary search[5]. Thus, the optimal solution to the LP (2.2) is at least $v^* - \delta'$. Since we use a β-approximate separation algorithm, we are not guaranteed that this solution is feasible. However, we know that $(\frac{q_i^*}{\beta}, \lambda_j^* | j \in H)$ is feasible. Thus,

[5] We can set δ' to be $\frac{1}{2^{cn}}$ for any constant $c > 0$.

the optimal solution to the LP (2.2) is at most $\frac{v^*}{\beta}$. Thus the optimal solution to (2.2) is between $v^* - \delta'$ and $\frac{v^*}{\beta}$.

In the execution of the ellipsoid algorithm for $v^* - \delta'$, we check a polynomial number of constraints. This set of constraints is enough to show that the value of the dual is greater than $v^* - \delta'$. The dual of this restricted LP is equivalent to the linear program (2.1) restricted to the variables corresponding to this set of constraints (by setting all other variables to zero). By LP-duality, the cost of the solution to this program is at least $v^* - \delta'$. Thus, the solution to this polynomial-sized LP has value at least $v^* - \delta'$. By setting δ' sufficiently small, this is an $(\beta - \delta)$-approximation algorithm for the primal linear program, since $\mathsf{LP}(\mathsf{SAP}) \leq \mathsf{LP}(\mathsf{Dual\ SAP}) \leq \frac{v^*}{\beta}$. $\square$

The fact that, for a class of packing-covering linear programs, an approximate separation oracle for the dual implies an approximate solution for the primal is also observed by Carr and Vempala [9] and by Jain, Mahdian and Salavatipour [43]. An approximate solution to the linear programs (2.1) and (2.2) can also be obtained via Lagrangian LP algorithms [73, 88, 27, 89].

We can use a β-approximation algorithm for the single-bin subproblem for bin i to design a β-approximate separation algorithm for $\mathcal{P}_i$. The β-approximate separation algorithm for $\mathcal{P}_i$ asks, given $(q_i, \lambda_j | j \in H)$, find a set $S \in \mathcal{I}_i$ such that $q_i < \sum_{j \in S}(f_{ij} - \lambda_j)$. It is of course sufficient to find the set $S \in \mathcal{I}_i$ that maximizes $\sum_{j \in S}(f_{ij} - \lambda_j)$. Since $\mathcal{I}_i$ is lower-ideal, we know that if $q_i < 0$ then the set $S = \emptyset$ violates the above inequality. Moreover, we can consider only items j for which $f_{ij} - \lambda_j$ is positive. In fact, we can set $\max(0, f_{ij} - \lambda_j)$ as the value of item j in the single-bin subproblem and use a β-approximation algorithm for the single-bin subproblem, to find a subset $S^* \in \mathcal{I}_i$ with value q_i^* such that for any set $S' \in \mathcal{I}_i$, $q_i^* = \sum_{j \in S^*}(f_{ij} - \lambda_j) \geq \beta \sum_{j \in S'}(f_{ij} - \lambda_j)$. We know that either $q_i^* > q_i$ in which case we find a violated constraint, or $q_i^* \leq q_i$. In the later case, we know that for any subset $S' \in \mathcal{I}_i$, $\sum_{j \in S'}(f_{ij} - \lambda_j) \leq \frac{q_i^*}{\beta} \leq \frac{q_i}{\beta}$. Therefore, in this case $(\frac{q_i}{\beta}, \lambda_j | j \in U)$ is feasible for $\mathcal{P}_i$. Hence, a β-approximation algorithm for the single-bin subproblem is a β-approximate separation algorithm for $\mathcal{P}_i$. The above and previous discussion yield the following general result.

Theorem 2.2.4 *For any constant $c > 0$, and $\delta = \frac{1}{2^{nc}}$, given a polynomial-time β-approximation algorithm for the single-bin subproblem, there exists a polynomial-time $((1 - \frac{1}{e})\beta - \delta)$-approximation algorithm for* SAP.

2.2.2 Approximation Algorithms for DCP and GAP

In this section, we show that the single-bin subproblem for each problem class in SAP discussed in the introduction has an approximation scheme. Thus, for all problem classes, this yields polynomial-time $1 - \frac{1}{e} - \epsilon$-approximation algorithms.

GAP: The single-bin subproblem for GAP is a knapsack problem, for which an efficient FPTAS is well-known.

CapDC: The items in the single-bin subproblem for CapDC correspond to request types and the value of item t is equal to the sum of the profit of requests of request type t in the CapDC instance. The size of item t in the subproblem is the size of request type t in the CapDC instance. The size of bin i is the available capacity of cache location i. Therefore, the single-bin subproblem in CapDC is to pack request types into bin i (respecting the bin capacity) to maximize the value of the items that can be assigned to the bin. Thus, the single-bin subproblem is a knapsack problem, and has an FPTAS.

CapIBDC: The single-bin subproblem for CapIBDC is the following general 2-dimensional knapsack problem: Bin i has A_i available space and B_i available bandwidth. Item $j \in S$ has value v_j, type t_j, and bandwidth consumption b_j. Each type t has size a_t. A feasible packing of items into bin i, satisfies the bandwidth constraint, i.e., total bandwidth of items in bin i is at most B_i, and the capacity constraint, i.e., the total size of types of these items is at most A_i. The goal is to maximize the total value of items packed in this bin. Shachnai and Tamir [81] formalize this general 2-dimensional knapsack problem and describe a PTAS for it.

As a result of the above discussion, we have:

Theorem 2.2.5 *For any $\epsilon > 0$, there exists a polynomial-time $(1-\frac{1}{e}-\epsilon)$-approximation algorithm for* GAP, CapDC, *and* CapIBDC.

2.3 Local Search Algorithms

In this section, for any $\epsilon > 0$, we give a simple local search $(\frac{\beta}{\beta+1} - \epsilon)$-approximation algorithm for separable assignment problems given an β-approximation algorithm for the single-bin subproblem. This, in turn, gives the first combinatorial $(\frac{1}{2} - \epsilon)$-approximation algorithms for GAP and all variants of DCP. We then show how to extend this algorithm to give a $(\frac{1}{2} - \epsilon)$-approximation algorithm for the k-median problem with hard capacities and packing constraints.

We first give a naive local search $\frac{\beta}{\beta+1}$-approximation algorithm whose running time might be exponential. Then, we refine the algorithm and change it to a polynomial-time algorithm. Let $\mathcal{S} = (S_1, \ldots, S_n)$ be an assignment of items to bins, where S_i is the set of items in bin i. For an assignment $\mathcal{S} = (S_1, \ldots, S_n)$ of items to bins, we denote the value of this assignment by $v(\mathcal{S})$. Also, let $\alpha_i(\mathcal{S})$ be the total value of items satisfied by bin i in $\mathcal{S}$. For an item j, let $v_j(\mathcal{S})$ the value of item j in $\mathcal{S}$.

The naive algorithm repeatedly iterates over the bins. For bin i, it runs procedure $\mathsf{Local}(i)$. $\mathsf{Local}(i)$, given current solution $\mathcal{S}$, finds a repacking S'_i of bin i. If replacing S_i with S'_i improves the solution then this replacement is made. When no further improvements can be made on any bin, the algorithm halts. We call the result an *β-approximate local optimal* solution.

Specifically, $\mathsf{Local}(i)$ does the following:

1. For each item j, let $\mathsf{value}_j(\mathcal{S})$ be equal to $f_{i'j}$ if j is assigned to a bin $i' \neq i$ in $\mathcal{S}$, and be equal to zero if j is unassigned or is assigned to bin i in $\mathcal{S}$.

2. For each item j, let the *marginal value* of j be $w_j = f_{ij} - \mathsf{value}_j(\mathcal{S})$.

3. Use the β-approximation algorithm for the single-bin subproblem for bin i to pack a subset of items in bin i with the maximum *marginal value.*

Lemma 2.3.1 *Let $\mathcal{S} = (S_1, \ldots, S_n)$ be a β-approximate local optimal solution and $\Omega = (\omega_1, \ldots, \omega_n)$ be the optimal assignment. Then $v(\mathcal{S}) \geq \frac{\beta v(\Omega)}{\beta+1}$.*

Proof. Let R be the set of items that are better served in the optimal solution than in $\mathcal{S}$ and L be the rest of the items. Let R_i be the set of items in R served by bin

i in the optimum Ω. For any set T of items, let $o(T)$ be the value of the items in T in Ω and $l(T)$ be the value of items of T in $\mathcal{S}$. For all items $j \in R_i$, the marginal value for bin i is positive ($w_j > 0$), since the value of j for bin i is greater than the current value of j in $\mathcal{S}$. For each item $j \in R_i$, the marginal value of j for solution $\mathcal{S}$ is $f_{ij} - \mathsf{value}_j(\mathcal{S}) \geq f_{ij} - v_j(\mathcal{S})$. Thus, the total marginal value of items in set R_i for bin i is at least $\sum_{j \in R_i}(f_{ij} - v_j(\mathcal{S})) = o(R_i) - l(R_i)$. Since $\mathcal{S}$ is a β-approximate local solution, the operation $\mathsf{Local}(i)$ cannot find a solution with marginal value greater than $\alpha_i(\mathcal{S})$, otherwise this operation could increase the total value. Since we use a β-approximation algorithm for the single-bin subproblem and there exists a solution with marginal value $o(R_i)-l(R_i)$, it follows that $\alpha_i(\mathcal{S}) \geq \beta(o(R_i)-l(R_i))$. Therefore, $o(R_i) \leq l(R_i) + \frac{1}{\beta}\alpha_i(\mathcal{S})$. Furthermore, for items in set L, $o(L) \leq l(L) \leq v(\mathcal{S})$ by definition. Therefore,

$$\begin{aligned}
\mathrm{OPT} &= v(\Omega) \\
&= o(L) + o(R) \\
&= o(L) + \sum_{i \in U} o(R_i) \\
&\leq l(L) + \sum_{i \in U}(l(R_i) + \frac{1}{\beta}\alpha_i(\mathcal{S})) \\
&\leq (1 + \frac{1}{\beta})v(\mathcal{S}).
\end{aligned}$$

Thus, $v(\mathcal{S}) \geq \frac{\beta}{\beta+1} v(\Omega)$. □

Lemma 2.3.1 shows that if we can find an β-approximate local solution then we have a $\frac{\beta}{\beta+1}$-approximation algorithm. We prove it is PLS-hard to find a local solution. The proof of this fact is very similar to the proof of Theorem 2.6.6. An implication of the PLS-hardness of this problem is that there exists a set of instances for which the above local search algorithm may take exponential time to converge to a local optimal solution.

Below, we modify the naive algorithm to get a polynomial-time $(\frac{\beta}{\beta+1} - \epsilon)$-approximation algorithm. The analysis of this algorithm uses the following fact (which we prove in the proof of Theorem 2.3.3). Using this fact, we show that after a polynomial number of local improvements the value of the solution is a good approximate solution.

Fact 2.3.2 *If $v(\mathcal{S}) \leq \frac{\beta}{\beta+1}\mathrm{OPT}$ then there is a bin i for which* $\mathsf{Local}(i)$ *finds a packing with marginal value at least $\frac{\beta}{n}\mathrm{OPT} - \frac{1+\beta}{n}v(\mathcal{S})$.*

Local Search Algorithm.

1. Start with the empty solution, i.e., $\mathcal{S} = (S_1, \ldots, S_n)$ and $S_i = \emptyset$ for all $i \in U$.
2. For an appropriate $\epsilon' > 0$, run the following loop for $\frac{1}{\beta} n \ln(\frac{1}{\epsilon'})$ times:
   (a) Let the current assignment be $\mathcal{S} = (S_1, \ldots, S_n)$.
   (b) For each bin i, run $\mathsf{Local}(i)$. Let the marginal value of this solution for bin i be W_i and let S'_i be the set of items with marginal value W_i.
   (c) For each bin i, let $\Delta_i = W_i - \alpha_i(\mathcal{S})$.
   (d) Let bin i^* be the bin with the maximum Δ_i, i.e., $\Delta_{i^*} \geq \Delta_i$ for any bin i.
   (e) If $\Delta_{i^*} > 0$, change the set S_i of items for bin i to S'_i.

Theorem 2.3.3 *For any $\epsilon > 0$, the above local search algorithm is a polynomial-time $(\frac{\beta}{\beta+1} - \epsilon)$-approximation algorithm for* SAP.

Proof. Let Ω be an optimal assignment, and let $\mathcal{S}$ be an intermediate assignment obtained in the local search algorithm. Let R be the set of items that are better served in Ω than in $\mathcal{S}$ and L be the rest of the items. Let R_i be the set of items in R satisfied by bin i in Ω. For any set T of items, let $o(T)$ be the value of the items in T in assignment Ω and $l(T)$ be the value of items of T in assignment $\mathcal{S}$. Thus, we have $\mathrm{OPT} = o(R) + o(L)$ and $v(\mathcal{S}) = l(R) + l(L)$. For each item $j \in R_i$, the marginal value of j is $f_{ij} - \mathsf{value}_j(\mathcal{S}) \geq f_{ij} - v_j(\mathcal{S})$. Thus, the total marginal value of items in set R_i for bin i is at least $\sum_{j \in R_i}(f_{ij} - v_j(\mathcal{S})) = o(R_i) - l(R_i)$ and R_i is a feasible solution for bin i. Since we use a β-approximation algorithm to find W_i, $W_i \geq \beta(o(R_i) - l(R_i))$.

Therefore, $\sum_{i \in U} W_i \geq \beta \sum_{i \in U}(o(R_i) - l(R_i)) = \beta(o(R) - l(R))$. Since $o(L) \leq l(L)$, $\sum_{i \in U} W_i \geq \beta(o(R) - l(R) + o(L) - l(L)) = \beta(\mathrm{OPT} - v(\mathcal{S}))$. Thus,

$$\begin{aligned}
\sum_{i \in U} \Delta_i &= \sum_{i \in U} W_i - \sum_{i \in U} \alpha_i(S) \\
&\geq \beta(\mathrm{OPT} - v(\mathcal{S})) - v(\mathcal{S}) \\
&= \beta\mathrm{OPT} - (1+\beta)v(\mathcal{S}).
\end{aligned}$$

In particular, $\Delta_{i^*} \geq \frac{\beta}{n}\mathrm{OPT} - \frac{1+\beta}{n}v(\mathcal{S})$. Let $\mathcal{S}'$ be the assignment after changing the set of items of i^* to S'_{i^*}, i.e., $\mathcal{S}' = (S_1, S_2, \ldots, S_{i^*-1}, s'_{i^*}, S_{i^*+1}, \ldots, S_n)$. As a result,

$$\begin{aligned}
v(\mathcal{S}') &= v(\mathcal{S}) + \Delta_{i^*} \\
&\geq v(\mathcal{S}) + \frac{\beta}{n}\mathrm{OPT} - \frac{1+\beta}{n}v(\mathcal{S}) \\
&= v(\mathcal{S})(1 - \frac{1+\beta}{n}) + \frac{\beta}{n}\mathrm{OPT}.
\end{aligned}$$

Let y_k be the total value of the assignment after the k^{th} execution of Step 2. From the above discussion, $y_k \geq (1 - \frac{1+\beta}{n})y_{k-1} + \frac{\beta}{n}\mathrm{OPT}$ and $y_0 = 0$. Using induction, we get that for any $1 \leq i \leq k$:

$$\begin{aligned}
y_k &\geq (1 - \frac{1+\beta}{n})^i y_{k-i} + \frac{\beta}{n}\mathrm{OPT}(\sum_{l=0}^{i-1}(1 - \frac{1+\beta}{n})^l) \\
&= (1 - \frac{1+\beta}{n})^i y_{k-i} + \frac{\beta}{n}\mathrm{OPT}(\frac{1 - (1 - \frac{1+\beta}{n})^i}{\frac{\beta+1}{n}}) \\
&= (1 - \frac{1+\beta}{n})^i y_{k-i} + \frac{\beta}{\beta+1}\mathrm{OPT}(1 - (1 - \frac{1+\beta}{n})^i) \\
&\geq 0 + \frac{\beta}{1+\beta}(1 - (1 - \frac{\beta+1}{n})^k)\mathrm{OPT}
\end{aligned}$$

Therefore, by solving this recurrence relation, we get $y_k \geq \frac{\beta}{1+\beta}(1 - (1 - \frac{\beta+1}{n})^k)\mathrm{OPT}$. By setting $k = \frac{n\ln(\frac{1}{\epsilon'})}{\beta+1}$, we get $y_k \geq \frac{\beta}{1+\beta}(1 - \frac{1}{e^{\ln(\frac{1}{\epsilon'})}})\mathrm{OPT} = \frac{\beta}{1+\beta}(1-\epsilon')\mathrm{OPT}$. Therefore, for $\epsilon' = \epsilon\frac{1+\beta}{\beta}$, the value of the output of the above algorithm is at least $(\frac{\beta}{1+\beta} - \epsilon)\mathrm{OPT}$ as desired. $\square$

2.4 k-median with hard capacities

We can extend the local search algorithm for SAP to the k-median problem with hard capacities and packing constraints(KMed). Recall that KMed is as follows: Given a set U of n bins, a set H of m items with a value f_{ij} for each item j and each bin i, and also a single-bin subproblem for each bin i, i.e., a lower-ideal family $\mathcal{I}_i$ of subsets for bin i, choose at most K bins and pack a set of items in each selected bin to maximize the total value packed. To the best of our knowledge, this is the first constant-factor approximation algorithm for the the k-median problem with hard capacity constraints.

The local search algorithm for the KMed problem is very similar to the local search algorithm for SAP. At each step of the algorithm, we try to unpack a used bin, and pack a (possibly different) bin to increase the total value. The formal description of the algorithm is as follows:

Local Search Algorithm for KMed.

1. Start with the empty solution, i.e., $\mathcal{S} = (S_1, \ldots, S_n)$ and $S_i = \emptyset$ for all $i \in U$.

2. Let $P = \{1, 2, \ldots, K\}$.

3. For an appropriate ϵ', run the following loop for $\frac{1}{\beta}K\ln(\frac{1}{\epsilon'})$ times:

   (a) Let the current assignment be $\mathcal{S} = (S_1, \ldots, S_n)$ where $S_i = \emptyset$ for $i \notin P$.

   (b) For each bin $i_1 \in P$ and bin $i_2 \in (U - P) \cup \{i_1\}$ do

      i. For each item j, let $\mathsf{value}_j(\mathcal{S})$ be $f_{i'j}$ if j is assigned to a bin $i' \neq i_1$ in $\mathcal{S}$, and be equal to zero if j is unassigned or is assigned to bin i_1 in $\mathcal{S}$.

      ii. For each item j, let the *marginal value* of j (with respect to bins i_1 and i_2) be $w_j = f_{i_2j} - \mathsf{value}_j(\mathcal{S})$.

      iii. Use the β-approximation algorithm for the single-bin subproblem for bin i_2 to pack a subset of items in bin i_2 with the maximum *marginal value*. Let the marginal value of this solution for bin i_2 be $W_{i_1i_2}$ and let $S'_{i_1i_2}$ be the set of items with marginal value $W_{i_1i_2}$.

(c) For every two bins $i_1 \in P$ and $i_2 \in (U-P)\cup\{i_1\}$, let $\Delta_{i_1 i_2} = W_{i_1 i_2} - \alpha_{i_1}(\mathcal{S})$.

(d) Let bins i_1^* and i_2^* be the bins with the maximum $\Delta_{i_1 i_2}$, i.e., $\Delta_{i_1^* i_2^*} \geq \Delta_{i_1 i_2}$ for any bin i_1 and i_2.

(e) If $\Delta_{i_1^* i_2^*} > 0$, unpack bin i_1^* and pack the set $S'_{i_1^* i_2^*}$ of items in bin i_2^* (i.e., set $P = P - \{i_1^*\} \cup \{i_2^*\}$ and set $S_{i_1^*} = \emptyset$ and $S_{i_2^*} = S'_{i_1^* i_2^*}$).

Theorem 2.4.1 *For any $\epsilon > 0$, the above local search algorithm for* KMed *is a polynomial-time $(\frac{\beta}{\beta+1} - \epsilon)$-approximation algorithm for* SAP.

Proof. The output of the local search algorithm is a feasible solution of the KMed problem, since the number of nonempty bins in the output is at most K. Let Ω be an optimal assignment. Consider the current assignment $\mathcal{S}$ in the process of the algorithm. Let P_Ω and $P_\mathcal{S}$ be the set of used bins in Ω and $\mathcal{S}$ respectively. The set of used bins in Ω and $\mathcal{S}$ are not necessarily the same. Let $\pi : U \to U$ be a permutation function such that if $i \in U$ is used in both Ω and $\mathcal{S}$, then $\pi(i) = i$ and if i is used in Ω, but not in $\mathcal{S}$ then $\pi(i) = i'$ where i' is used in $\mathcal{S}$, but not in Ω. As the number of used bins in Ω is K, such a permutation exists. Let R be the set of items that are better served in Ω than in $\mathcal{S}$ and L be the rest of the items. Let R_i be the set of items in R satisfied by bin i in Ω. For any set T of items, let $o(T)$ be the value of the items in T in Ω and $l(T)$ be the value of items of T in $\mathcal{S}$. Thus, we have $\text{OPT} = o(R) + o(L)$ and $v(\mathcal{S}) = l(R) + l(L)$. For an item j, let $v_j(\mathcal{S})$ the value of item j in $\mathcal{S}$. Consider two bins $\pi(i)$ and $i \in P_\Omega$. For each item $j \in R_i$, the marginal value of j with respect to bins $\pi(i)$ and i is $f_{ij} - \mathsf{value}_j(\mathcal{S}) \geq f_{ij} - v_j(\mathcal{S})$. Thus, the marginal value of set R_i with respect to bins $\pi(i)$ and i is at least $\sum_{i \in R_i}(f_{ij} - v_j(\mathcal{S})) = o(R_i) - l(R_i)$ and R_i is a feasible solution for bin i. Since we use a β-approximation algorithm to find $W_{\pi(i)i}$, $W_{\pi(i)i} \geq \beta(o(R_i) - l(R_i))$. Therefore, $\sum_{i \in P_\Omega} W_{\pi(i)i} \geq \beta \sum_{i \in P_\Omega}(o(R_i) - l(R_i)) = \beta(o(R) - l(R))$. Since $o(L) \leq l(L)$, $\sum_{i \in P_\Omega} W_{\pi(i)i} \geq \beta(o(R) - l(R) + o(L) - l(L)) = \beta(\text{OPT} - v(\mathcal{S}))$. Thus, $\sum_{i \in P_\Omega} \Delta_{\pi(i)i} = \sum_{i \in P_\Omega} W_{\pi(i)i} - \sum_{i \in P_\mathcal{S}} \alpha_i(S) \geq \beta(\text{OPT} - v(\mathcal{S})) - v(\mathcal{S}) = \beta\text{OPT} - (1+\beta)v(\mathcal{S})$. In particular, $\Delta_{i_1^* i_2^*} \geq \frac{\beta}{K}\text{OPT} - \frac{1+\beta}{K}v(\mathcal{S})$. Let $\mathcal{S}'$ be the assignment after setting $S_{i_1^*}$ to $\emptyset$ and changing the set of items of i_2^* to $S'_{i_1^* i_2^*}$. As a result,

$$\begin{aligned} v(\mathcal{S}') &= v(\mathcal{S}) + \Delta_{i_1^* i_2^*} \\ &\geq v(\mathcal{S}) + \frac{\beta}{K}\mathrm{OPT} - \frac{1+\beta}{K} v(\mathcal{S}) \\ &= v(\mathcal{S})(1 - \frac{1+\beta}{K}) + \frac{\beta}{K}\mathrm{OPT}. \end{aligned}$$

Now, let y_k be the total value of the assignment after the k'th execution of Loop 3b. From above discussion, $y_k \geq (1 - \frac{1+\beta}{K})y_{k-1} + \frac{\beta}{K}\mathrm{OPT}$ and $y_0 = 0$. Similar to the proof of Theorem 2.3.3 and solving this recurrence relation, we get $y_k \geq \frac{\beta}{1+\beta}(1 - (1 - \frac{\beta+1}{K})^k)\mathrm{OPT}$. By setting $k = \frac{K \ln(\frac{1}{\epsilon'})}{\beta+1}$, we get $y_k \geq \frac{\beta}{1+\beta}(1 - \frac{1}{e^{\ln(\frac{1}{\epsilon'})}})\mathrm{OPT} = \frac{\beta}{1+\beta}(1 - \epsilon')\mathrm{OPT}$. Therefore, the value of the output of the above algorithm is at least $(\frac{\beta}{1+\beta} - \epsilon)\mathrm{OPT}$ as desired. $\square$

2.5 A Hardness Result

In this section, we show a hardness result for the CapDC problem and special cases of SAP. We prove that the CapDC problem is not approximable better than a factor of $1 - \frac{1}{e}$ unless NP $\subseteq$ DTIME$(n^{O(\log\log n)})$ showing that the $1 - \frac{1}{e} - \epsilon$-approximation algorithm for CapDC is almost tight. This hardness result uses a hardness result by Feige, Halldorson, Kortsarz, and Srinivasan [18] for the domatic number of graphs. The *domatic number* of a graph is the maximum number of disjoint dominating sets in the graph. A subset S of vertices of a graph $G(V, E)$ is a *dominating set* if for any vertex $v \notin S$, there exists a vertex $u \in S$ which is connected to v, i.e., $(u, v) \in E(G)$. We first define a set of problems that are used in the reduction and restate the result of Feige et al. [18].

The Max 3-colorability problem is as follows: Given a graph $G(V, E)$ color the vertices of G with 3 colors to maximize the number of legally colored edges (edges whose endpoints are colored differently). The Max 3-colorability-5 problem is the Max 3-colorability problem for 5-regular[6] graphs. Petrank [72] proved that the Max

[6]A graph is 5-regular if the degree of each vertex is five.

3-colorability problem is APX-hard. Using this proof, Feige et al. [18] proved that the Max 3-colorability-5 problem is APX-hard. Formally, they showed that for some number $\delta < 1$, it is NP-hard to distinguish between 5-regular graphs that have a legal 3-coloring, and 5-regular graphs in which every 3-coloring legally colors at most δ fraction of the edges. The following claim is implicit in the hardness result of Feige et al. [18]: Given an instance $G(V, E)$ of the Max 3-colorability-5 problem, we can construct an instance of a set cover problem with

$$m = O(|V(G)|^{O(\log\log|V(G)|)}|E(G)|^{O(\log\log|V(G)|)})$$

elements and

$$n = O(|V(G)|^{O(\log\log|V(G)|)}|E(G)|^{O(\log\log|V(G)|)})$$

sets of size $\frac{m}{p}$ where

$$\frac{m}{p} = O(|V(G)|^{O(\log\log|V(G)|)}|E(G)|^{O(\log\log|V(G)|)})$$

such that:

- If the vertices of graph G are (legally) 3-colorable, then there exist $\frac{n}{p}$ disjoint set covers, each with p sets[7] in the set cover instance.
- If any 3-coloring of G has less than $\delta|E(G)|$ legally colored edges then any collection of βp sets cover at most $(1-(1-\frac{1}{p})^{\beta p})m$ elements[8].

From an instance of the set cover problem with n sets and m elements, we construct an instance of CapDC with $\frac{n}{p}$ types, $m \cdot \frac{n}{p}$ requests, and n cache locations as follows:

[7]See Lemma 17 of [18]. In fact, Feige et al [18] present their result in terms of the dominating set and the domatic number problem. We restate their result for the set cover problem.

[8]The proof of this claim comes from the proof of Lemma 18 of [18]. For the proof of Lemma 18 of [18], the authors refer to the hardness result for the set cover problem by Feige [17] (e.g. see Proposition 4.3 of [17]). Essentially, the proof of our claim comes from the fact that in the construction of Feige [17], in this case, any collection of βp sets cover at most $(1-(1-\frac{1}{p})^{\beta p})m$ elements [16]. The reason is that the number of elements that βp sets cover is less than the expected number of elements that βp random sets of size p cover where a random set is a set in which each element is picked uniformly at random and independent of other elements. We also note that p is not a constant. In particular, as $|V(G)|$ tends to ∞, p also tends to ∞.

For each element j in the ground set of the set cover, we put $\frac{n}{p}$ requests $j_1, j_2, \ldots, j_{\frac{n}{p}}$ of different types in the CapDC instance. For each set i in the set cover instance, we put a cache location i in the CapDC instance. The capacity A_i for cache location i is $A_i = 1$ and the size of each request type is equal to 1. Thus, we can locate at most one type in each cache location. The profit of assigning request j_e to cache location i is 1 if the corresponding element j is in the corresponding set in the set cover instance. If the set cover instance has $\frac{n}{p}$ disjoint set covers then in the instance of the CapDC problem, we can satisfy all requests of a particular type using one set cover and thus, we can find a solution to the instance of the CapDC problem with a total profit of $\frac{mn}{p}$. Moreover, we claim that if any collection of βp sets in the set cover problem, cover at most $(1-(1-\frac{1}{p})^{\beta p})m$ of elements then the profit of any assignment to the CapDC problem is at most $(1-(1-\frac{1}{p})^{p})\frac{mn}{p}$. Assume that in the set cover instance, any collection of βp sets cover at most $(1-(1-\frac{1}{p})^{\beta p})m$ of the elements. Consider a solution $\mathcal{S}$ with the maximum profit for the CapDC problem. For $1 \le t \le \frac{n}{p}$, let $\alpha_t p$ be the number of cache locations that keep the request type t in solution $\mathcal{S}$. We know that $\sum_{t=1}^{\frac{n}{p}} \alpha_t p = n$. Also from the inequality $(1-(1-\frac{1}{p})^{xp}) + (1-(1-\frac{1}{p})^{tp}) \le (1-(1-\frac{1}{p})^{yp}) + (1-(1-\frac{1}{p})^{zp})$ where $x \le y \le z \le t$ and $x+t = y+z$, it follows that the profit of $\mathcal{S}$ maximizes when all α_t for $1 \le t \le \frac{n}{p}$ are the same, i.e., $\alpha_t = 1$. By setting $\alpha_t = 1$, we have that the profit of $\mathcal{S}$ is at most $\sum_{t=1}^{\frac{n}{p}}(1-(1-\frac{1}{p})^{\alpha_t p})m \le \sum_{t=1}^{\frac{n}{p}}(1-(1-\frac{1}{p})^{p})m \le (1-(1-\frac{1}{p})^{p})\frac{mn}{p}$.

Therefore, if we apply the Feige et al. reduction from Max 3-colorobality-5 to the set cover and the above reduction from the set cover to the CapDC problem, we have the following: Given an instance of Max 3-colorability-5 problem, we can construct an instance of the CapDC problem with $\frac{n}{p}$ types, $m \cdot \frac{n}{p}$ requests, and n cache locations such that:

- If vertices of graph G are (legally) 3-colorable, then there exists a solution with profit $\frac{mn}{p}$ for the CapDC instance.

- If any 3-coloring of G has less than $\delta|E(G)|$ legally colored edges, the maximum possible profit of the CapDC instance is at most $1-(1-\frac{1}{p})^{p}$ of the number of requests, i.e., $(1-(1-\frac{1}{p})^p)\frac{mn}{p}$.

Note that $(1-(1-\frac{1}{p})^p)$ tends to $1-\frac{1}{e}$ as p tends to ∞. Therefore, for any $\epsilon > 0$, there exists a sufficiently large p such that $(1-(1-\frac{1}{p})^p) \leq (1-\frac{1}{e}) + \epsilon$. Hence, for any $\epsilon > 0$, for any sufficiently large instance of Max 3-colorability-5 problem, we can construct an instance of the CapDC problem with $\frac{n}{p}$ types, $m \cdot \frac{n}{p}$ requests, and n cache locations such that:

- If vertices of graph G are (legally) 3-colorable, then there exists a solution with profit $\frac{mn}{p}$ for the CapDC instance.
- If any 3-coloring of G has less than $\delta|E(G)|$ legally colored edges, the maximum possible profit of the CapDC instance is at most $1-\frac{1}{e}+\epsilon$ of the number of requests, i.e., $(1-\frac{1}{e}+\epsilon)\frac{mn}{p}$.

This shows that for any $\epsilon > 0$, if we can approximate the CapDC problem within a factor better than $1-\frac{1}{e}+\epsilon$ then we can distinguish between the aforementioned cases of the sufficiently large instances of the Max 3-colorability-5 problem in time $O(V(G)^{O(\log\log V(G))}E(G)^{O(\log\log V(G))})$. Since distinguishing between these two cases of the Max 3-colorability-5 problem is NP-hard, if we can approximate the CapDC problem in polynomial time within a factor of $1-\frac{1}{e}+\epsilon'$ for $\epsilon' < \epsilon$, then NP$\subseteq$ DTIME($n^{O(\log\log n)}$). Note that in the above reduction, we only used instances of the CapDC problem with uniform sizes and uniform capacities, this shows that even the uniform CapDC problem is not approximable within a factor better than $1-\frac{1}{e}$ unless NP $\subseteq$ DTIME($n^{O(\log\log n)}$). In particular, it means that there are instances of SAP in which the single-bin subproblem is solvable in polynomial time, but the multiple-bin SAP problem is not approximable within a factor better than $1-\frac{1}{e}$ unless NP $\subseteq$ DTIME($n^{O(\log\log n)}$). Therefore, we get the following theorem:

Theorem 2.5.1 *There are instances of the* SAP *problem in which the subproblem is polynomially solvable that are not approximable better than a factor* $1-\frac{1}{e}$ *unless NP* $\subseteq$ *DTIME(*$n^{O(\log\log n)}$*). In particular, the uniform* CapDC *problem is not approximable better than a factor* $1-\frac{1}{e}$ *unless NP* $\subseteq$ *DTIME(*$n^{O(\log\log n)}$*).*

2.6 Decentralized Mechanisms

In this section, we explore methods to obtain decentralized algorithms for all variants of the DCP problem with a good performance. Before stating the formal definitions and results, we give an introduction on the applicability of the decentralized mechanisms in distributed caching in cellular networks.

The 3G subscriber market can be categorized into groups with shared interest in location-based services, e.g., the preview of movies in a theater or the scene of the beach nearby. Since the 3G radio resources are limited, it is expensive to repeatedly transmit large quantities of data over the air interface from the base station (BS). It is more economical for the service provider to offload such repeated requests on to the ad-hoc network comprised of its subscribers where some of them recently acquired a copy of the data. In this scenario the goal for the service provider is to give incentives for peer subscribers in the system to cache and forward the data to the requesting subscribers. Since each data item is large in size and transit subscribers are mobile, we assume that the data transfer occurs in a close range of a few hops.

In this system, we envision a system consisting of two groups of subscribers: resident and transit subscribers. Resident subscribers are less mobile and mostly confined to a certain geographical area. Resident subscribers have incentives to cache data items that are specific to this geographical region since the service provider gives monetary rewards for satisfying the queries of transit subscribers. Transit subscribers request their favorite data items when they visit a particular region. Since the service provider does not have knowledge of the spatial and temporal distribution of requests, it is difficult if not impossible for the provider to stipulate which subscriber should cache which set of data items. Therefore, the decision of what to cache is left to each individual subscriber. The realization of this content distribution system depends on two main issues. First, since subscribers are selfish agents, they may act to increase their individual payoff and decrease the performance of the system. Here, we provide a framework for which we can prove that in an equilibrium situation of this framework, we use the performance of the system efficiently (below, we will describe the efficiency formally in terms of the price of anarchy of a game). The second issue is that the payoff of each request for each agent must be a function the set of agents that have this request in their strategy, since these agents compete on this request and the profit of this request should be divided among these agents in an appropriate way. Therefore, each selfish agent may change the set of items it cached in response to the set of items cached by others. This leads to a non-cooperative caching scenario which we model as the distributed caching game. Motivated from service provider cellular networks, we assume that cache locations are selfish agents (resident subscribers) who want to maximize their own profit.

Consider a distributed caching setting in which selfish cache locations decide which types and which requests to provide, based on their limited space and bandwidth. The service provider let cache locations decide on the set of requests they want to satisfy, and run the following rewarding scheme: the profit of each request will go to the cache location that provides this request with the maximum profit. The *payoff* of a cache location is the total profit of the requests that are assigned to this cache location. Cache locations compete with each other on getting more profit from satisfying these requests. This defines a game - the CapIBDC game which is formally defined below. The service provider is interested in maximizing the total profit, i.e., the social function is the total profit of the cache locations.

We define the *distributed caching game* in the setting of CapIBDC - with both capacity and bandwidth constraints. Given an instance of the CapIBDC problem, we define a strategic game $\mathcal{G}(U, \{F_i | i \in U\}, \{\alpha_i | i \in U\})$ as follows. The set of players U is the set of cache locations. The family of feasible strategies F_i of a cache location i is the family of subsets s_i of requests such that $\sum_{j \in s_i} b_j \leq B_i$ and $\sum_{t \in \{t_j | j \in s_i\}} a_t \leq A_i$. Given a vector $S = (s_1, s_2, \ldots, s_n)$ of strategies of cache locations, the *favorite* cache locations for request j, denoted by $\mathsf{FAV}(j)$, is the set of cache locations i such that $j \in s_i$ and f_{ij} has the maximum profit among the cache locations that have request j in their strategy set, i.e., $f_{ij} \geq f_{i'j}$ for any i' such that $j \in s_{i'}$. For a strategy profile $S = (s_1, \ldots, s_n)$ $\alpha_i(S) = \sum_{j : i \in \mathsf{FAV}(j)} \frac{f_{ij}}{|\mathsf{FAV}(j)|}$. Intuitively, the above definition implies that the profit of each request goes to the cache locations with the minimum connection cost (or equivalently with the maximum profit) among the set of cache locations that provide this request. If more than one cache location have the maximum profit (or minimum connection cost) for a request j, the profit of this request is divided equally between these cache locations. The payoff of a cache location is the sum of profits from the requests it actually serves. We say that a player i serves a request j if $i \in \mathsf{FAV}(j)$. The social value of strategy profile S, denoted by $\gamma(S)$, is the sum of profits of all players. This value $\gamma(S)$ is a measure of the efficiency of the assignment of requests and request types to cache locations.

We similarly define games for the other versions of the distributed caching problem. For problems, such as CapDC and uniform CapDC, that just have capacity constraints, the strategy of player i is simply a subset of requests s_i and a strategy s_i is a feasible strategy for player i if $\sum_{t \in \{t_j | j \in s_i\}} a_t \leq A_i$. For IBDC, the strategy of player i is a subset of requests s_i and the set s_i is a feasible strategy for player i if $\sum_{j \in s_i} b_j \leq B_i$.

In this section, we bound the price of anarchy for these games (Section 2.6.1), study the existence of pure equilibria (Section 2.6.2) and demonstrate convergence results for the more general versions of the game (Section 2.6.3). Picture 2-2 depicts the games that we consider in this chapter and their relation to each other. Some of these games are defined later in the chapter.

2.6.1 CapIBDC Game: Price of Anarchy

Since the CapIBDC game is a strategic game, it has mixed Nash equilibria [64]. We prove that in a mixed Nash equilibrium of this game, the expected social value is at least $\frac{1}{2}$ of the optimal social value. We can prove this by showing that the CapIBDC game is a valid-utility game. First, we give a direct proof of this fact.

Theorem 2.6.1 *The price of anarchy of the* CapIBDC *game for a mixed Nash equilibrium is at most* 2.

Proof. In a mixed Nash equilibrium, each player chooses a probability distribution over its feasible pure strategies. We prove that the expected social value of this

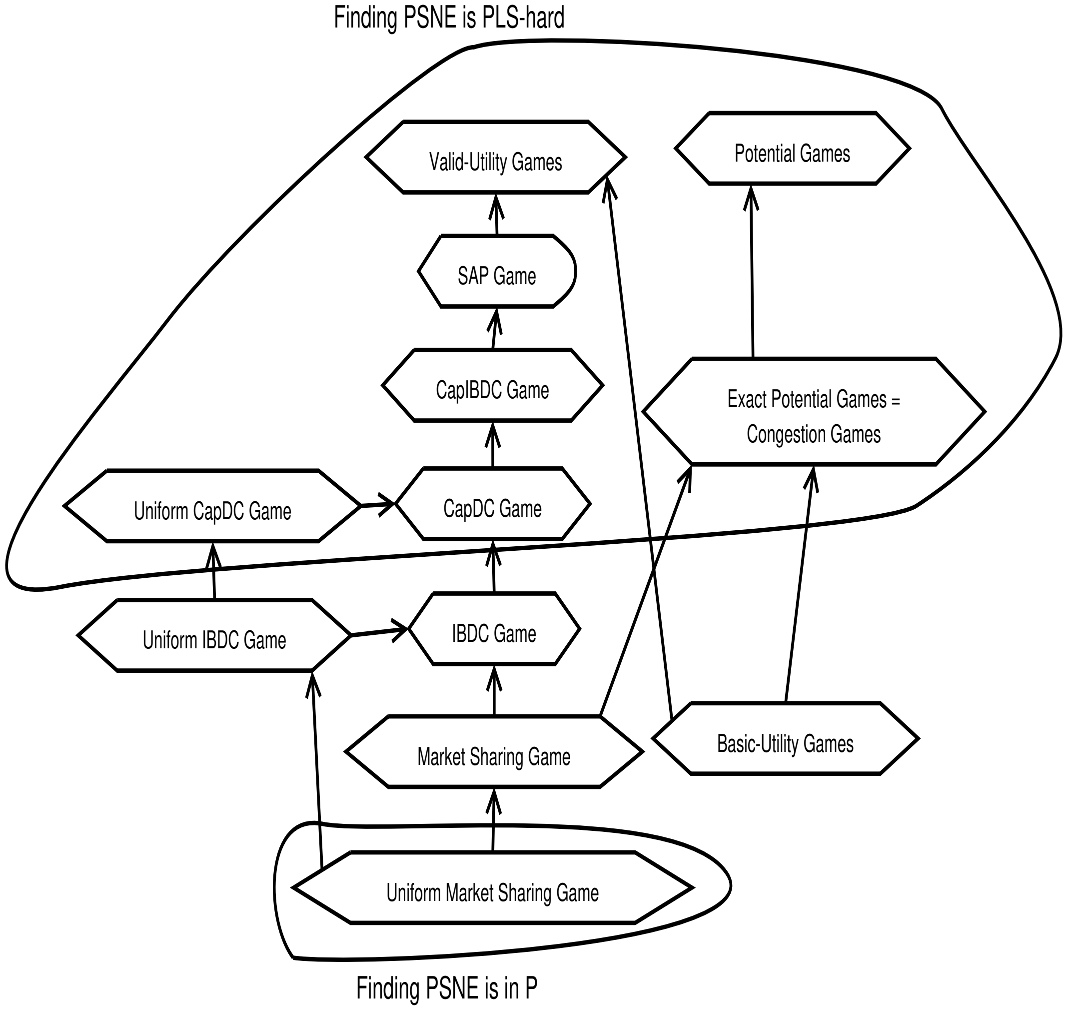


Figure 2-2: Different classes of Games. There is an arrow from game A to game B, if A is a special case of B.

equilibrium is at least $\frac{1}{2}$ of the optimum solution. Consider the optimum solution $\Omega = (\sigma_1, \ldots, \sigma_n)$. Let the profit of request j in Ω be ρ_j. Consider a mixed Nash equilibrium $\mathcal{P}$. Let the expected payoff of player i in $\mathcal{P}$ be $\alpha_i(\mathcal{P})$. For any request j, let p_j be the probability that j's profit in $\mathcal{P}$ is at least ρ_j. It is easy to see that $\mathrm{E}[\gamma(\mathcal{P})] \geq \sum_{j \in H} p_j \rho_j$, since with probability p_j the profit of j is at least ρ_j. We know that the expected social value of $\mathcal{P}$ is equal to $\mathrm{E}[\gamma(\mathcal{P})] = \sum_{i \in U} \alpha_i(\mathcal{P})$. Let player i change his strategy in $\mathcal{P}$ to the pure strategy σ_i. Consider a request $j \in \sigma_i$. With probability $1 - p_j$ the profit of request j is less than ρ_j. When player i provides this request, the profit is ρ_j. Thus, if player i changes his strategy to σ_i, his expected payoff is at least $\sum_{j\in\sigma_i}(1-p_j)\rho_j$. $\mathcal{P}$ is a mixed Nash equilibrium, thus player i's expected payoff in $\mathcal{P}$ is at least his expected payoff when he plays σ_i in $\mathcal{P}$. Therefore, $\alpha_i(\mathcal{P}) \geq \sum_{j\in\sigma_i}(1-p_j)\rho_j$. Therefore,

$$\begin{aligned}
\text{OPT} &= \gamma(\Omega) \\
&= \sum_{j\in H}\rho_j \\
&= \sum_{j\in H}p_j\rho_j + \sum_{j\in H}(1-p_j)\rho_j \\
&\leq \text{E}[\gamma(\mathcal{P})] + \sum_{i\in U}\sum_{j\in\sigma_i}(1-p_j)\rho_j \\
&\leq \text{E}[\gamma(\mathcal{P})] + \sum_{i\in U}\alpha_i(\mathcal{P}) \\
&= 2\text{E}[\gamma(\mathcal{P})]
\end{aligned}$$

as desired. □

In the following, we show that the CapIBDC game is a valid-utility game. By a result of Vetta [87], this gives an alternative proof for the price of anarchy for mixed Nash equilibria in CapIBDC game.

Theorem 2.6.2 *The* CapIBDC *game is a valid-utility game.*

Proof. We need to show the following three properties:

1) Nondecreasing and Submodular Social Function: First, it is clear that γ^s is non-decreasing. To show its submodularity, we use an equivalent definition of submodular functions: A set function f is submodular if for any two subsets A and B such that $A \subset B$ and for any element $i \notin B$, $f(A\cup\{i\}) - f(A) \geq f(B\cup\{i\}) - f(B)$ [26]. Thus, in order to prove that γ^s is submodular, it is enough to prove that for two (possibly infeasible) strategy profiles $S = (s_1, \ldots, s_n)$ and $S' = (s'_1, \ldots, s'_n)$ such that $s_i \subseteq s'_i$ for all $i \in U$, by adding a new request j to the strategy set of any player i the increase in γ^s for S is not less than the increase for S'. Let v_j and v'_j be the profit of request j in S and S', respectively ($v_j = 0$ if $j \notin \cup_{i \in U} s_i$ and $v'_j = 0$ if $j \notin \cup_{i \in U} s'_i$). As a result,

$$\begin{aligned} v'_j &= \max_{i:j \in s'_i} f_{ij} \\ &\geq \max_{i:j \in s_i} f_{ij} \\ &= v_j. \end{aligned}$$

Adding j to a strategy s_h and s'_h increases γ^s for S and S' by $\max(0, f_{hj} - v_j)$ and $\max(0, f_{hj} - v'_j)$, respectively. From $v'_j \geq v_j$, it follows that $\max(0, f_{h_j} - v_j) \geq \max(0, f_{hj} - v'_j)$. Hence, the increase in γ^s for S is greater than or equal to the increase for S'. Thus in any case, the increase for S is not less than the increase for S'.

2) Vickrey Condition: The difference in the social function when i plays s_i or empty (does not play at all) is equal to $\sum_{j:i \in \mathsf{FAV}(j), |\mathsf{FAV}(j)|=1} f_{ij}$ and this is indeed less than or equal to $\alpha_i(S) = \sum_{j:i \in \mathsf{FAV}(j)} \frac{f_{ij}}{|\mathsf{FAV}(j)|}$.

3) Cake Condition: By the definition of the social function, we have $\sum_{i \in U} \alpha_i(S) = \gamma(S)$ and therefore the third property is satisfied as well.

□

The above theorems extend to the general setting for all separable assignment problems. We can define the SAP game similar to the CapIBDC game: bins are selfish agents and their strategy is to keep a feasible subset of items according to the lower-ideal $\mathcal{I}_i$. The profit of an item j goes to a player i that have this item in his strategy and has the maximum profit for item j among all bins that have this item in their strategy. The above proofs show that the SAP game is a valid-utility game and thus, the price of anarchy for a mixed NE for this game is at most 2.

2.6.2 DCP Games: Pure Nash Equilibria

In this section, we address various problems related to pure strategies in the DCP games. We show that on the one hand, there are instances of CapDC and IBDC that have no pure Nash equilibria; while the uniform IBDC games in which there are no ties among the profit of the requests to cache locations have pure Nash equilibria. For the uniform CapDC game, we demonstrate a cycle of strict best responses. A *strict best-response move* for a player i is a best-response move in which player i strictly increases his payoff.

Theorem 2.6.3 *There are instances of the* IBDC *game that have no pure Nash equilibrium.*

Proof. Consider the following instance with 2 players and 4 requests. The bandwidth consumption of requests are 8, 3, 5 and 6 respectively. The profit of requests to player 1 are 10, 2, 5.5 and 5.5 respectively. The profit of requests to player 2 are 2, 3, 3 and 5.6 respectively. The available bandwidth of player 1 and 2 are 11 and 8 respectively. The example is depicted in Table 2.1. The only possible best responses of player 1 to any strategy of player 2 are sets $\{1, 2\}$ and $\{3, 4\}$. The only possible best responses of player 2 are subsets $\{4\}$ and $\{2, 3\}$. None of the 4 pairs of best responses is a pure Nash equilibrium, thus this game does not have a pure Nash equilibrium.

	The Profit of the Requests				
	1	2	3	4	Available Bandwidth
Cache Loc. 1	10	2	5.5	5.5	11
Cache Loc. 2	2	3	3	5.6	8

Table 2.1: The profit of request and available bandwidth for cache locations in the IBDC game without pure Nash equilibria.

□

Since, IBDC is a special case of CapDC, the above theorem implies that there are instances of the CapDC game that have no pure Nash equilibrium. In the above example the bandwidth consumption of requests are not uniform, and this was essential in finding the example. In the following, we study the uniform variant of these games. Note that we can easily change the above example to an example of the IBDC game without tie among the profit of the requests and with no PSNE. In the following, we prove that the uniform IBDC game in which there are no ties among the profit of requests does not contain any cycle of strict best-response moves. As a result, the subgraph of the state graph with strict best-response moves as arcs does not contain any cycle. Therefore, this graph has a vertex without any outgoing arcs, and this vertex is a PSNE.

Theorem 2.6.4 *Any instance of the uniform* IBDC *game in which there are no ties among the profit of the requests does not have any cycle of strict best-response moves and thus, has a PSNE.*

Proof. Given a strategy profile $S = (s_1, \ldots, s_n)$, and the resulting assignment of requests to cache locations, order the set of all pairs (i, j) of (cache i, request j) such that cache i serves a request j in non-increasing order of their profits. Consider the vector $p(S)$ of the profit of these pairs in the above order. We claim that this vector is lexicographically increasing as players play a strict best-response move.

To see this, consider a player i that plays a strict best-response move from s_i to s'_i. Consider the first pair (i^*, j^*) of cache i^* and request j^* that disappears from the vector of profits (i.e., $f_{i^*j^*}$ appears in vector $p(S)$ and not in vector $p(S \oplus s'_i)$). Thus, the profit of all pairs in $p(S)$ with a profit larger than $f_{i^*j^*}$ appear in the vector $p(S \oplus s'_i)$. Since request j^* is not served by i^* in $S \oplus s'_i$, either j^* is served by player i in $S \oplus s'_i$, or $i^* = i$ and j is not in s'_i.

In the first case, player i serves request j^* instead of player i^* and thus the profit of i for j^* should be more than $f_{i^*j^*}$. Therefore, there exists the number f_{ij^*} which is strictly greater than $f_{i^*j^*}$ in vector $p(S \oplus s'_i)$ instead of $f_{i^*j^*}$; thus the vector is lexicographically increasing. In the latter case, since $(i, j^*) = (i^*, j^*)$ is the pair with the largest profit whose profit disappears from the vector $p(S)$ and player i increased his payoff playing his best response from s_i to s'_i and the bandwidth requirement of all requests are the same, there exists another request $j' \in s'_i$ with profit $f_{ij'}$ such that

$f_{ij'}$ is greater than f_{ij^*}; and $f_{ij'}$ appears in vector $p(S \oplus s'_i)$ instead of $f_{i^*j^*}$. Therefore, in the latter case the vector is lexicographically increasing as well. $\square$

We note that the existence of pure Nash equilibria for the uniform IBDC game when there are no ties among profits can be derived from the existence of stable matchings in a general setting with social choice functions with substitutability property [22]. However, cycles of strict best-response moves are known even for stable matching games [77] [9]. Our proof for this variant of the uniform IBDC indicates that not only PSNE exists, but also any sequence of strict best-response moves of players converges to a PSNE.

For the uniform CapDC game, in the example below, we demonstrate a cycle of strict best-response moves. We do not know if the uniform CapDC game or the uniform IBDC game always have a PSNE or not.

Theorem 2.6.5 *There exist cycles of strict best-response moves in the uniform* CapDC *game.*

Proof. Consider an instance of the distributed caching game with three players

	Profit of the Requests to Players		
	Player 1	Player 2	Player 3
A_1	2	0	1
A_2	3	0	4
B_1	0	6	5
B_2	0	2	1
C_1	1	7	0
C_2	3	2	0

Table 2.2: The profit of the requests for the example of the CapDC game with a cycle of strict best responses.

and three request types, two requests from each type. The available capacity of each

[9] In stable matching games, each player has an arbitrary preference list for each request, and each request has an arbitrary preference list for the players. Players offer to a request in their preference list, and each request j goes to a player with the highest priority among the players that offer to j. Each player likes to get a request with a higher priority in his preference list. A cycle of strict best-response moves is known for this game [77]. The IBDC game in which each cache location can cache only one request can be formalized as a special case of the stable matching game.

player is one and the size of each request type is one. Let players be 1, 2, and 3. Let request types be A, B, and C. There are two requests of type A, i.e, A_1 and A_2, two requests of type B, i.e., B_1 and B_2, and two requests of type C, i.e., C_1 and C_2. In each strategy profile, each player can choose both requests of only one request type. The profit of all requests are depicted in Table 2.2. Since, each player can cache both requests of only one type, we can refer to the strategy of a player as one type. In the following, by a type as the strategy of a player, we mean the set of both requests of that type.

In this instance, starting from the strategy profile (C, C, B) for players, if we let players 2, 1, 3, 2, 1, and 3 play their best responses in this order, players will end up with the same configuration (C, C, B). The complete cycle is depicted in Table 2.3. In this table, we refer to the strategy of a player as a type and by that we mean both requests of that type.

Players	Player's Type (Player's Payoff)					
1	C(3)	C(4) →	A(5)	A(2)	A(2) →	C(3)
2	C(7) →	B(8)	B(8)	B(8) →	C(9)	C(7)
3	B(6)	B(0)	B(0) →	A(4)	A(4)	A(5) →

Table 2.3: A cycle of size 6 of best-responses in the uniform CapDC game. Each column represents the vector of request types that the players provide. The numbers in parenthesis in each column are the payoffs of players. The arrow (→) indicates that a player plays his best response and changes his strategy to the request type in the next column.

□

2.6.3 CapDC Game: Poor Convergence to Equilibria

In this section, we prove that there are instances of the uniform CapDC game in which finding a pure Nash equilibrium is PLS-hard [47] (See the definition of PLS-hard problems in Section 1.2).

We give a reduction from the Max-Cut local search problem with swapping neighborhood to the problem of finding a PSNE in some instances of the uniform CapDC game. In turn, this implies that there are states of this game from which any path of best response moves to the equilibrium has exponential length.

Recall that the Max-Cut local search problem with swapping neighborhood is as follows: Given an edge-weighted graph G and a cut, the local operations are to switch one node from one side of the cut to the other side if it can increase the value of the cut. In this section, WLOG we assume that the graph G is connected.

Theorem 2.6.6 *There are instances of the uniform* CapDC *game with pure Nash equilibria*[10] *for which finding a pure Nash equilibrium is PLS-hard.*

Proof. We give a reduction from the local search Max-Cut problem with swapping neighborhood to the uniform CapDC game.

Consider an instance $G(V, E)$ of the Max-Cut problem with weights $w : E(G) \rightarrow N$ on edges. We construct an instance $\mathcal{R}(G)$ of the CapDC game as follows: Each player corresponds to a vertex of graph G. There are two types of requests. The size of each request type is equal to one and the capacity of each cache location is one. For each edge, we define two requests. Edge uv has a request p_{uv} of type one and a request q_{uv} of type two. We assume that the connection costs of both requests on edge uv to either u or v is zero. The connection costs of these two requests to any other vertex is greater than $w(uv)$. The reward of each of these two requests is $w(uv)$.

Each player caches either requests of type one or requests of type two. If both u and v cache requests of the same type then they each get profit $\frac{w(uv)}{2}$ from the requests from edge (u, v). If they cache different types then they each get profit $w(uv)$ for this edge. Thus, given a strategy profile, the total profit obtained is exactly $w(E) + w(C)$ where C is the set of edges with one player caching type 1 and the other caching type 2. In other words, C is the cut set defined by the cut where all vertices that cache type 1 are on one side and all vertices that cache type 2 are on the other.

From the definition of the game $\mathcal{R}(G)$, if s'_i is a best-response move of player i in strategy profile S, we claim that either s'_i contains all requests of type 1 on edges

[10]We can also say that finding a *sink equilibrium* is PLS-hard. A sink equilibrium is a set of strategy profiles that is closed under best-response moves. A pure equilibrium is a sink equilibrium with exactly one profile. This equilibrium concept is formally defined in Chapter 4.

adjacent to i, or s'_i contains all requests of type 2 on edges adjacent to i. The reason is that if s'_i does not contain all requests of one type on edges adjacent to i then player i can strictly increase his payoff by including the rest of requests of the same type on edges adjacent to him in his strategy (here, we use the fact that G is connected.). We let $\mathcal{L}$ be the set of strategy profiles S of the instance $\mathcal{R}(G)$ in which each player i plays a set s_i such that s_i either contains all requests of type one on edges adjacent to player i, or s_i contains all requests of type two on edges adjacent to player i. From the above discussion, it follows that any PSNE of game $\mathcal{R}(G)$ is in $\mathcal{L}$.

From a feasible strategy profile $S \in \mathcal{L}$ of $\mathcal{R}(G)$, we construct a cut $\mathcal{M}(S, G)$ in the Max-Cut local search problem on G. For a player u in the game $\mathcal{R}(G)$ if u caches requests of type one, then we put u in side one of the cut $\mathcal{M}(S, G)$, and if u caches requests of type two, then we put u in side two of $\mathcal{M}(S, G)$.

In a strategy profile $S \in \mathcal{L}$ of game $\mathcal{R}(G)$, player u can strictly improve his payoff by playing switching from requests of type 1 to requests of type 2 (or vice-versa) if and only if the value of the cut set $\mathcal{M}(S, G)$ strictly improves by moving u to the other side of the cut. Thus, if a strategy profile $S \in \mathcal{L}$ is a pure Nash equilibrium in game $\mathcal{R}(G)$, then $\mathcal{M}(S, G)$ is a local optimal solution of the Max-Cut local search problem for graph G.

Thus, if we have a polynomial-time algorithm for finding a pure Nash equilibrium S (or a sink equilibrium) in any instance of the uniform CapDC game, since this pure Nash equilibrium is in $\mathcal{L}$, this implies a polynomial-time algorithm for finding a local optimum $\mathcal{M}(S, G)$ of any instance of the Max-Cut local search problem with swapping neighborhood. The PLS-hardness follows from a PLS-completeness result of Schaffer and Yannakakis [79]. $\square$

Using the above proof and a result of Schaffer and Yannakakis [71, 79], we can show that in some instances of the uniform CapDC game there are states from which all paths of best responses have exponential length.

Corollary 2.6.7 *There are instances of the uniform* CapDC *game that have pure Nash equilibria with states from which any sequence of best-response moves to any* *pure Nash equilibrium (or sink equilibrium) has an exponential length.*

Proof. In [79], it is shown that there exists a weighted graph G, and an initial cut C, such that the length of any sequence of local operations for the Max-Cut local search problem from C to any local optimal solution is exponential. Consider the CapDC game $\mathcal{R}(G)$ defined in the previous proof and the set $\mathcal{L}$ of strategy profiles. From the cut C of G, we can easily construct a strategy profile $S \in \mathcal{L}$ such that $\mathcal{M}(S, G) = C$ by letting player u play all requests of type 1 or 2 in S, if vertex u is in the side 1 or 2 of the cut C, respectively. We claim that any sequence of best responses from S to any PSNE in $\mathcal{R}(G)$ has exponential length.

Assume for contradiction that there is a sequence of strategy profiles $S_0 = S$, S_1, ..., S_t where $t \leq \text{poly}(n)$ and player u_i plays his best-response move from S_i to S_{i+1} and S_t is a PSNE. Since $S_0 = S \in \mathcal{L}$ and u_i plays his best-response move from S_i to S_{i+1}, by induction it follows that $S_i \in \mathcal{L}$ for all $0 \leq i \leq t$. It is easy to see that $C = \mathcal{M}(S_0, G)$, $C_1 = \mathcal{M}(S_1, G)$, $C_2 = \mathcal{M}(S_2, G)$, ..., $C_t = \mathcal{M}(S_t, G)$ is a sequence of cuts such that cut C_{i+1} results from the cut C_i by the local operation of vertex u_i and C_t is a local optimal solution. Therefore, $t \leq \text{poly}(n)$ implies that there exists a sequence of local operations of polynomial length from cut C to a local optimal solution. This contradicts the assumption that no such sequence of polynomial length from cut C exists. $\square$

2.6.4 Market Sharing Game: Price of Anarchy

In this section, we formalize a special case of non-cooperative content distribution in wireless networks as a *market sharing game* introduced in [29] and study these games. This class of games is a subclass of both congestion games and valid-utility games. Here, we define this set of games.

Market Sharing Game. Consider a set U of n agents and a set V of m markets. For each agent i, we are given a limited budget B_i and a subset V_i of markets that are of player i's interest (we write i is interested in market j, if $j \in V_i$). For each market $j \in V$, we are given a cost C_j and a value v_j; this value depends on the rate at which market i is requested per unit time. The strategy set, $F_i \subseteq 2^{V_i}$ of player i is a family of subsets of V_i such that the sum of the cost of markets is less than the budget of player i, i.e., $s_i \in F_i$ if $\sum_{j \in s_i} C_j \leq B_i$. The strategic game $\mathcal{G}(U, \{F_i | i \in U\}, \{\alpha_i() | i \in U\})$ is called a *market sharing game* if for a strategy profile $S = (s_1, \ldots, s_n)$ $\alpha_i(S) = \sum_{j \in s_i} \frac{v_j}{n_j(S)}$ where $n_j(S)$ is the number of agents that serve market j in S. We also consider the social utility function $\gamma(S) = \sum_{j \in U} \alpha_j(S)$ for this game. By definition, the market sharing game is a congestion game with the congestion function $c_j(x) = \frac{v_j}{x}$ and thus, it is an exact potential game.

Consider an instance $\mathcal{G}'$ of the IBDC game. Assume that the connection cost of each request j follows the following pattern: the connection cost of j to a subset of cache locations, denoted by Υ_j, is zero and the connection cost to the rest of cache locations is a large number. Thus, the profit of request j for any cache location in Υ_j is $f_{ij} = R_j$ and the profit of j for other cache locations is $f_{ij} = 0$. Therefore, given the strategy of players the profit of a request j is divided equally among the set of cache locations in Υ_j that serve this request. We demonstrate a correspondence between the game $\mathcal{G}'$ and a market sharing game $\mathcal{G}(U, \{F_i | i \in U\}, \{\alpha_i | i \in U\})$, where F_i is a family of subsets of markets in V_i, i.e., $F_i \subseteq 2^{V_i}$. The set of players U corresponds to the set of cache locations. The set of requests in IBDC corresponds to the set of markets, $V = \cup_{i \in U} V_i$ in the market sharing game. We let $V_i = \{j | i \in \Upsilon_j\}$. The available bandwidth B_i of cache location i in IBDC, corresponds to the budget B_i of agent i in the market sharing game. The reward R_j and bandwidth b_j of request j correspond to the value v_j and cost C_j of market j, respectively. If s_i is a strategy of cache location i in $\mathcal{G}'$, it is implied $\sum_{j \in s_i} b_j \leq B_i$. Equivalently, if agent i's strategy is s_i in the market sharing game, $\sum_{j \in s_i} C_j \leq B_i$. This correspondence shows that the market sharing game is a special case of the IBDC game, and thus, it is a valid-utility game, and the price of anarchy for mixed Nash equilibria of any special case of this game is at most 2. In this section, we study the market sharing game and prove tighter results on the price of anarchy for PSNE in some special cases of this game. In Section 2.6.5, we study the problem of finding a pure Nash equilibrium in these games.

As mentioned above, in the market sharing game, the value of market j corresponds to the reward R_j of request j. In the distributed caching setting, the reward of a request j depends on the rate at which clients ask for this request. Thus, v_j directly depends on the demand rate of clients for market j. It has been observed that in many practical situations, demand curves follow the power law (Zipf) distributions [8], namely $v_j = \frac{1}{j^\beta}$ for a parameter $0 < \beta \leq 1$. This motivates us to study the special case of the market sharing game in which the value of markets follows power law distributions. We prove that in a uniform market sharing game where the cost of all markets is the same, i.e., $C_j = C$ for all j, if all players are interested in all markets, the price of anarchy (for PSNE) is less than $1.45 + o(1)$ in the worst case, where $o(1)$ depends on n, i.e. $o(1)$ tends to 0 as $n \to \infty$. Furthermore, for cases in which $V_i = V$ for each player i or markets have different costs, we prove that the factor 2 for the price of anarchy is tight.

Theorem 2.6.8 *In the uniform market sharing game, if $V_i = V$ for each player i and values are from a Zipf distribution with parameter β, it is implied that*

- *The price of anarchy (for PSNE) is less than or equal to $\frac{1}{(1-\beta)^{1-\beta}} + o(1)$ for any $\beta < 1$. In particular, it is less than $e^{\frac{1}{e}} + o(1) < 1.45 + o(1)$ for any $\beta < 1$ and it tends to $1 + o(1)$ when $\beta \to 1$.*
- *For $\beta = 1$, the price of anarchy (for PSNE) is $\left(1 + \frac{\ln\ln(n)}{\ln(n)}(1 + o(1))\right)$.*

Proof. Consider a pure strategy Nash equilibrium and let p be the least index such that the players do not select p but select all markets 1 to $p-1$. No market beyond p can be selected by any player, since otherwise such a player would have an incentive to switch to market p, thus $\frac{v_j}{n_j} \geq v_p$ or $\frac{1}{n_j j^\beta} \geq \frac{1}{p^\beta}$ for $1 \leq j \leq p-1$. Summing over all markets $1 \leq j \leq p-1$, we get $\sum_{j=1}^{p-1} \frac{1}{j^\beta} \geq (\sum_{j=1}^{p-1} n_j)\frac{1}{p^\beta}$. Letting $V_\beta(k) = \sum_{j=1}^{k} \frac{1}{j^\beta}$, we get $p^\beta V_\beta(p-1) \geq n$. As OPT can at best serve all markets, we have that the price of anarchy is at most $\frac{V_\beta(n)}{V_\beta(p-1)}$.

We consider the two cases $\beta = 1$ and $\beta < 1$ separately. We start with $\beta = 1$.

We need to compute $\frac{V_1(n)}{V_1(p-1)}$. From $\ln(n) \le V_1(n) \le \ln(n)+1$, it is not hard to see that $p > \frac{n}{\ln(n)}$, for sufficiently large n. Therefore,

$$\frac{V_1(n)}{V_1(p-1)} \le \frac{\ln(n)+1}{\ln(\frac{n-\ln(n)}{\ln(n)})} \le \frac{\ln(n)+1}{\ln(n-\ln(n))-\ln\ln(n)} = 1 + \frac{\ln\ln(n)}{\ln(n)} + o(\frac{\ln\ln(n)}{\ln(n)}),$$

where the last step can be proved using L'Hopital's rule when $n \to \infty$.

We now consider the case $\beta < 1$. Let $L_\beta(k) = \frac{1}{1-\beta}(k^{1-\beta}-1)$, then it is easy to see $L_\beta(k) \le V_\beta(k) \le L_\beta(k)+1$. Using this fact, we can bound the ratio $\lim_{n\to\infty} \frac{V_\beta(n)}{V_\beta(p-1)}$. Observing the facts that $p \to \infty$ and $L_\beta(p-1) \to \infty$ and β is a positive constant less than 1, we can compute the bound as follows:

$$\begin{aligned}
\lim_{n\to\infty} \frac{V_\beta(n)}{V_\beta(p-1)} &\le \lim_{n\to\infty} \frac{L_\beta(n)+1}{L_\beta(p-1)} \\
&= \lim_{n\to\infty} \frac{n^{1-\beta}-1}{(p-1)^{1-\beta}-1} \\
&\le \lim_{n\to\infty} \frac{(p^\beta V_\beta(p-1))^{1-\beta}}{(p-1)^{1-\beta}-1} \\
&\le \lim_{p\to\infty} \frac{(p^\beta (L_\beta(p-1)+1))^{1-\beta}}{(p-1)^{1-\beta}-1} \\
&\le \lim_{p\to\infty} \frac{p^{\beta(1-\beta)} \frac{1}{(1-\beta)^{1-\beta}} (p^{1-\beta}-1)^{(1-\beta)}}{(p-1)^{1-\beta}-1} \\
&= \lim_{p\to\infty} \frac{p^{(1-\beta)\beta} \frac{1}{(1-\beta)^{1-\beta}}}{((p-1)^{1-\beta}-1)^\beta} \\
&= \lim_{p\to\infty} \frac{1}{(1-\beta)^{1-\beta}} \left(\frac{p^{1-\beta}}{(p-1)^{1-\beta}-1} \right)^\beta \\
&= \frac{1}{(1-\beta)^{1-\beta}}.
\end{aligned}$$

Now, one can observe that this bound is less than $e^{\frac{1}{e}}$ for any $\beta < 1$ and tends to 1 as β tends to 1. ☐

Theorem 2.6.9 *There are instances of the uniform market sharing games for power law (Zipf) distribution in which the price of anarchy for PSNE is arbitrarily close to 2. Moreover, the price of anarchy of some instances of the nonuniform market sharing game where $V_i = V$ for each player i is arbitrarily close to 2.*

Proof. We give an example with $v_j = \frac{1}{j}$. There are n^2+n markets and n^2 players. Players are partitioned in n groups of size n. Players in group k are interested in markets $k, kn+1, kn+2, \ldots, kn+n$. All budgets and costs are equal, i.e., for all $1 \le j \le n^2+n$ and $1 \le i \le n^2$, $B_i = C_j = c$. The strategy profile in which all players of group k provide market k is a a Nash equilibrium and the social value of this Nash equilibrium is $H_n = 1 + \frac{1}{2} + \cdots + \frac{1}{n}$. However if we assign a new market to each player, all markets are provided except n of them. The value of this assignment is $H_{n^2+n} - \sum_{j=1}^{n} \frac{1}{jn+n} = H_{n^2+n} - \frac{H_n}{n+1}$. Thus the ratio is $\frac{H_{n^2+n}}{H_n} - \frac{1}{n+1}$ which is equal to $\frac{\ln(n^2+n)}{\ln(n)} = 2$ as $n \to \infty$.

The proof that this bound is tight for general market sharing games where $V_i = V$ for all $i \in U$ is based on an example similar to the above one. Let $V_i = V$ for each player i. The budgets of players in group k are $n-k$. The cost of markets $k, kn+1, kn+2, \ldots, kn+n-1$ is also $n-k$ for $1 \le k \le n$. The cost of the market $kn+n$ is large. The value of market j is $v_j = \frac{1}{j}$. Similar to the previous example, if all players of group k provide market k, no player has incentive to change his strategy. It follows that the price of anarchy for this PSNE is arbitrarily close to 2. $\square$

2.6.5 Market Sharing Game: Finding a Nash Equilibrium

By definition, the market sharing game is a congestion game and thus, a potential game; and any sequence of strict improvement moves of players converges to a pure Nash equilibrium (See Section 1.2 for definitions).

In a market sharing game with one player, finding a pure strategy Nash equilibrium corresponds to solving optimally a knapsack problem. Thus, the problem of finding a Nash equilibrium in the market sharing game is NP-hard. However, a Nash equilibrium always exists. In this section, we give a polynomial-time algorithm to find a pure Nash equilibrium in a uniform market sharing game.

Recall that in the uniform market sharing game, we assume that $C_j = C$ for all markets $j \in V$. One main feature of the uniform variant is that it is easy for player i to determine its best response, given the set of strategies for other players. Indeed, player i only needs to solve an easy maximization problem corresponding to selecting the k_i most rewarding markets, where $k_i = \lfloor \frac{B_i}{C} \rfloor$. We could therefore let players repeatedly and optimally improve their strategy, but the main issue is to show that such a process converges to a Nash equilibrium in polynomial time. In fact, we will not analyze this algorithm. Instead we analyze an iterative algorithm in which each agent is restricted to a set of changes at each step. This proves that if players change according to these restrictions, they will converge to a Nash equilibrium in polynomially many steps. This implies that a Nash equilibrium can be found in polynomial time. An iterative algorithm to find a Nash equilibrium seeks a sequence of improvement moves[11] starting from an empty strategy profile.

Theorem 2.6.10 *For the uniform market sharing game, a pure strategy Nash equilibrium always exists and can be found in polynomial time. Furthermore, it can be obtained by a sequence of length m^2n of improvement moves of players.*

Proof. Our algorithm for finding the sequence of improvement moves and finding a pure strategy Nash equilibrium proceeds in rounds. The first round starts at the strategy profile $(\emptyset_1, \ldots, \emptyset_n)$ corresponding to the set of empty strategies. In each round, the first improvement move corresponds to a player, say i, switching from s_i to s'_i (with the maximum increase) where $s'_i = s_i \cup \{j\}$. In other words, player i only adds precisely one market which gives the maximum increase in the payoff to its strategy. We refer to this first improvement move as an *add* improvement move. After this first improvement move, subsequent improvement moves in a round are *change* improvement moves. These correspond to a player, say i, replacing s_i by $s_i \cup \{j\} \setminus \{k\}$, where $j \notin s_i$ and $k \in s_i$; player i exchanges market k for market j. Furthermore, given i and k, j is selected among all possible markets i of interest to i and not currently in s_i in order to maximize i's payoff. A round finishes when there are no *change* improvement moves from the current strategy profile. Subsequent rounds start at the strategy profile where the previous round finishes, unless this strategy profile has no *add* improvement move from it in which case this is the last round.

First, observe that when the last round finishes, the current strategy profile has

[11] See the definition of an improvement move in Section 1.2

no *add* or *change* improvement moves and therefore must be a pure Nash equilibrium. This implicitly uses the fact that we are dealing with a *uniform* market sharing game and therefore any maximal strategy for player i can be obtained from any other maximal strategy by exchanging in and out two markets at a time. Furthermore, at the end of each round, the current state has no *change* improvement moves outgoing from it, which implies that it corresponds to a pure Nash equilibrium if we suitably modify the budgets of each player (so that they cannot add markets).

As one player adds a market at the beginning of each round, the number of rounds cannot be greater than mn. We now show that each round ends after traversing at most $m-1$ *change* improvement moves (and one *add* improvement move).

Let us focus on one round and let n_j be the number of players servicing market j at *the beginning* of a round. For simplicity, we assume that the markets are sorted in such a way that $\frac{v_1}{n_1+1} \geq \frac{v_2}{n_2+1} \geq \cdots \geq \frac{v_m}{n_m+1}$. Consider any strategy profile in the round after the first *add* improvement move. Let s_i be the markets currently served by i and let T_i be the markets of interest to i not in s_i. For player i, let $m(i)$ denote $\min\{j \in T_i\}$. We show by induction that the following properties hold throughout the round:

1. For any player i, $m(i)$ does not decrease during the round.

2. Every market j is covered by n_j players, except one, denoted by p which is covered by $n_p + 1$.

3. For any player i, any market $j \in s_i$ and any market $k \in T_i$ we have $\frac{v_j}{n_j} \geq \frac{v_k}{n_k+1}$.

Properties 1 and 2 are obviously true after the first *add* improvement move corresponding to player, say l, adding market p. Property 3 is also true after the first *add* improvement move. Indeed, the condition reduces to the fact that the last round ended in a pure Nash equilibrium except for the case where $i = l$ and $j = p$ where it follows from the choice of p: $\frac{v_p}{n_p} > \frac{v_p}{n_p+1} \geq \frac{v_k}{n_{k+1}}$.

We see now what happens when we traverse a *change* improvement move corresponding to player l exchanging two markets. Condition 3 implies that l leaves market

p since all other markets do not increase l's payoff. Thus property 2 is maintained after the change (with a different value for p). Secondly, l will now serve market $m(l)$ by definition of $m(l)$. This implies that $\frac{v_p}{n_p+1} < \frac{v_{m(l)}}{n_{m(l)}+1}$, i.e. $m(l) < p$. This means that property 1 is also maintained. To verify that Property 3 is still maintained, we only need to consider the cases in which $i = l$ and either $k = p$ or $j = m(l)$. If $y = m(l)$, property 3 follows from the definition of $m(l)$: $\frac{v_{m(l)}}{n_{m(l)}} > \frac{v_{m(l)}}{n_{m(l)}+1} \geq \frac{v_k}{n_k+1}$. If $k = p$, it follows from $\frac{v_j}{n_j} \geq \frac{v_{m(l)}}{n_{m(l)}+1} > \frac{v_p}{n_p+1}$.

All three properties are maintained during the round. Furthermore, since player l replaces market p by market $m(l)$ and $m(l) < p$, we have that p decreases as we traverse *change* improvement moves. This implies that we have at most $m-1$ *change* improvement moves in a round. This proves our bound of nm^2 on the length of the number of improvement moves that we need to reach a pure Nash equilibrium.

□

In order to run the algorithm, we actually do not need to construct the entire state graph. We only need to be able to find the next improvement move to traverse, and this can be done in $O(m+n)$ time, resulting in a total running time of $O((m+n)m^2n)$.

2.7 Conclusions and Open Problems

In this chapter, we developed centralized $1 - \frac{1}{e} - \epsilon$-approximation algorithms, $\frac{1}{2}$-approximate decentralized mechanisms, and local search $\frac{1}{2} - \epsilon$-approximation algorithms for a broad class of maximizing assignment problems. As our main motivation, we focused on variants of a distributed caching problem, but all algorithms in this chapter work for separable assignment problems. We complement this result by proving that the uniform CapDC problem is not approximable better than a factor of $1 - \frac{1}{e}$, unless NP$\subseteq$ DTIME$(n^{O(\log\log n)})$. The most natural question is to improve the approximation factor for GAP.

In the decentralized mechanism, we show a good price of anarchy for mixed Nash equilibria, but we also show several negative results for the convergence of these games. One interesting open question is to prove fast convergence to constant-factor solutions in the CapIBDC game[12]. Another interesting open question is to see if the uniform CapDC game or the uniform IBDC game have a pure Nash equilibrium. In this chapter, we showed that a special case of IBDC game has a PSNE and the CapDC game contains a cycle of strict best-response moves.

[12] In Chapters 3 and 4, we see some examples of games for which we can prove fast convergence to constant-factor solutions.

Chapter 3

Convergence in Potential Games

Traditionally, research in operations research has focused upon finding a global optimum. Computer scientists have also long studied the effects of lack of different resources, mainly the lack of computational resources, in optimization. Recently, the *lack of coordination* inherent in many problems has become an important issue in computer science. A natural response to this has been to analyze Nash equilibria in these games. Of particular interest is the *price of anarchy* [1] in a game [70]. Clearly, a low price of anarchy may indicate that a system has no need for a single regulatory authority. Conversely, a high price of anarchy is indicative of a poorly functioning system in need of some regulation.

In this chapter, we move away from only measuring the social value of Nash equilibria to evaluate the performance of a game. There are several reasons for this. The first reason is that we are not guaranteed that the selfish behavior of players converges to a Nash equilibrium. Moreover, if a sequence of selfish behavior of players converges to a Nash equilibrium, the time it takes for this convergence even may be extremely long. So, from a practical viewpoint, in order to analyze the decentralized mechanism, it is important to evaluate the speed or rate of convergence of the corresponding game.

As is clear, these issues are particularly important in games in which the use of pure strategies and repeated moves are the norm, for example, auctions. For these

[1] Some of the definitions and notations that are used in this chapter can be found in Section 1.2.

games, then, it is not sufficient to just study the value of the social function at Nash equilibria. Instead, we must also investigate the speed of convergence (or non-convergence) to an equilibrium. Towards this goal, we will not restrict our attention to Nash equilibria but rather prove that after some number of improvements or best responses the value of the social function is within a factor of the optimal social value. We tackle this by modeling the behavior of players using the underlying state graph on the set of strategy states. We consider best-response walks in this graph and evaluate the social function at states along these walks. The rate of convergence to high quality solutions (or Nash equilibria) can then be measured by the length of the walk. We address these issues in two chapters of this thesis. In this chapter, we study potential games in which any sequence of strict improvement moves of players converges to a PSNE. In these games, we study the rate of convergence to approximate solutions. In Chapter 4, we examine games in which selfish behavior of players does not necessarily converge to a PSNE. In fact, these games may not possess any pure Nash equilibrium. We will define a new equilibrium concept for those games and measure the social value of this new equilibrium concept (See Chapter 4).

In this chapter, we study convergence for three classes of potential games: *Cut games* (See Section 3.2 for definitions and results), basic-utility games (Section 3.3) and market sharing games (Section 3.4).

3.1 Preliminaries

In this chapter, we use the definitions and notations from Section 1.2. Given a best-response walk starting from an arbitrary state in the state graph, we are most interested in the social value of the last state on the walk. Notice that if we do not allow every player to make a best response on a walk $\mathcal{P}$, then we may not be able to bound the social value of a state with respect to the optimal solution. This follows from the fact that the strategy of a single player may be very important for producing solutions of high social value. Hence, we consider the following models:

One-round walk: Consider an arbitrary ordering of all players $i_1, \ldots, i_n$. A walk $\mathcal{P}$

of length n in the state graph is a *one-round walk* if for each $j \in \{1, 2, \ldots, n\}$, the jth edge of P has label i_j.

Covering walk: A walk $\mathcal{P}$ in the state graph is a *covering walk* if for each player i, there exists an edge of P with label i.

k-Covering walk: A walk $\mathcal{P}$ in the state graph is a *k-covering walk* if there are k covering walks $\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_k$ such that $\mathcal{P} = (\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_k)$.

Random walk: A walk $\mathcal{P}$ in the state graph is a *random walk*, if at each step the next player is chosen uniformly at random and independently of the previous players.

Random one-round walk: Let σ be a ordering of players picked uniformly at random from the set of all possible orderings. Then, the one-round walk $\mathcal{P}$ corresponding to the ordering σ, is a *random one-round walk*.

Note that a one-round walk is a covering walk. Also in the one-round walk we let each player play his best response exactly one time, but in a covering walk we let each player play at least one time. For a non-cooperative game $\mathcal{G}$ with a social function γ, we are interested in the social value of states (especially the final state) along one-round, covering, k-covering, and random walks.

Related Work. Here, we give a brief overview of related work in this area. The consequences of the selfish behavior and the question of efficient computation of Nash equilibria have recently drawn much attention in computer science [70, 67]. Moreover, the use of the price of anarchy [70] as a measure of the cost of the lack of coordination in a game is now widespread, with a notable success in this realm being the selfish routing game [78]. A basic result of Rosenthal [76] defines congestion games for which pure strategy Nash equilibria exist. Monderer and Shapley [60] proved that congestion games are equivalent to the class of *exact potential games*[2]. Milchtaich [55] studied player-specific congestion games and the length of best-response walks in this set of

[2]See Section 1.2 for the definition.

games. Even-Dar et al. [14] considered the convergence time to Nash equilibria in variants of a load balancing game. They bound the number of required steps to reach a pure Nash equilibrium in these games. They consider a central policy that let agents move in a certain order, and analyze different policies to choose such an ordering. In contrast to this work, their interest is in the convergence time to a pure Nash equilibrium and not to good approximate solutions. Fabrikant et al. [15] studied the complexity of finding a pure strategy Nash equilibrium in general congestion games. Their PLS-completeness results show that in some congestion games (including selfish routing games), the length of a best-response walk in the state graph to a pure Nash equilibrium might be exponential.

A Simple Example. Here, we illustrate the use of the above definitions by studying covering walks in a simple load balancing game; The speed of convergence to Nash equilibria in this game has been considered by Even-Dar et al. [14]. Consider n jobs that can be scheduled on m machines. Assume that it takes p_i units of time for job i to run on any of the machines. Formally, we define the strategic game $\mathcal{G}(U, \{F_i | i \in U\}, \{\alpha_i | i \in U\})$ as follows: The set of players U is the set of jobs, and the strategy set F_i of a player i is the set of all machines $F_i = \{j | 1 \le j \le m\}$. Given the strategy profile $S = (s_1, \ldots, s_n)$, let $l_j(S)$ be the total processing time or the load of machine j. The payoff of player i is $\alpha_i(S) = \frac{1}{l_{s_i}(S)}$. Thus, each job wants to be scheduled on a machine with the minimum load. The social function is the maximum load over all machines, i.e., $\gamma(S) = \max_{1 \le j \le m} l_j(S)$.

This game is a potential game. The potential function is as follows: Given a strategy profile S, we sort the numbers $l_j(S)$ for all machines in a decreasing order. Consider the resulting vector. We claim that this vector is a potential function for this game. To see this, we can show that if a job moves from a machine to another machine and decreases its payoff, this potential function decreases lexicographically. Therefore, any sequence of strict improvement moves of players converges to a PSNE. First, we show that the price of anarchy for PSNE in this game is at most 2. Let OPT be the value of the optimal schedule. Consider a PSNE S of this game. As no job i has incentive to change machine and decreases the load of the machine that it is scheduled on, we have: for any job i and any machine j, $l_{s_i}(S) \leq l_j(S) + p_i$. Thus, for any machine j and any job i such that $s_i = j$, all machines are busy at all times before $l_j(S) - p_i$, thus, OPT $\geq l_j(S) - p_i$. Consider the machine j^* with the maximum load in S and a job i that is scheduled on j^*. Hence, OPT $\geq l_{j^*}(S) - p_i = \gamma(S) - p_i$. Clearly, OPT $\geq p_i$. Thus, $\gamma(S) \leq 2$OPT as desired.

In addition, from any state there is a walk of length at most n to some pure Nash equilibrium [80]. On an arbitrary best-response walk, it may, however, take more than n steps to converge to a pure Nash equilibrium. Our goal is to show that the social value of any state at the end of a covering walk is within a factor 2 of optimal. Consider a covering walk $\mathcal{P} = (S_1, S_2, \ldots, S_k)$. Let j^* be the machine with the largest load at state S_k. Consider the last job i^* that was scheduled on machine j^*, and let the strategy profile after scheduling i^* be S_t. Ignoring job i^*, at time t the load of all the machines is at least $l_{j^*}(S_k) - p_{i^*}$. If not, job i^* would not have been scheduled on machine j^*. Consequently, we have $\sum_{1\leq i\leq n} p_i \geq m(l_{j^*}(S_k) - p_{i^*})$. Thus, OPT $\geq \sum_{1\leq i \leq n} p_i/m \geq l_{j^*}(S_k) - p_{i^*}$. Clearly, OPT $\geq p_{i^*}$. Thus, $\gamma(S_k) = l_{j^*}(S_k) = l_{j^*}(S_k) - p_{i^*} + p_{i^*} \leq 2$OPT as desired.

3.2 The Max-Cut Game: Convergence

In this section, we study an illustrative potential game, called the *cut game* or the *Max-Cut game*. First, we define this game formally. We are given an undirected graph $G(V, E)$, with n vertices and edge weights $w : E(G) \to \mathbb{Q}^+$. In this section, we assume that G is connected, simple, and does not contain loops. For each $v \in V(G)$, let $\deg(v)$ be the degree of v, and let $\mathsf{Adj}(v)$ be the set of neighbors of v. Let also $w_v = \sum_{u \in \mathsf{Adj}(v)} w_{uv}$. A cut in G is a partition of $V(G)$ into two sets T and $\bar{T} = V(G) - T$, and is denoted by $(T, \bar{T})$. The value of a cut is the sum of edges between the two sets T and $\bar{T}$, i.e., $\sum_{v\in T, u \in \bar{T}} w_{uv}$.

The *Max-Cut game* or the *cut game* on a graph $G(V, E)$ is defined as follows:

each vertex $v \in V(G)$ is a player, and the strategy of v is to choose one side of the cut, i.e., v can choose $s_v = -1$ or $s_v = 1$. A strategy profile $S = (s_1, s_2, \dots, s_n)$, corresponds to a cut $(T, \bar{T})$ where $T = \{i | s_i = 1\}$. The payoff of player v in a strategy profile S, denoted by $\alpha_v(S)$, is equal to the contribution of v in the cut, i.e., $\alpha_v(S) = \sum_{i:s_i \neq s_v} w_{iv}$. It follows that the cut value is equal to $\frac{1}{2}\sum_{v \in V} \alpha_v(S)$. If S is clear from the context, we use α_v instead of $\alpha_v(S)$ to denote the payoff of v. We denote the maximum value of a cut in G, by $c(G)$. We consider the cut value as the social function.

The Max-Cut problem is a well-studied problem [30]. Simple greedy approximation algorithms are known for this problem with the performance guarantee of $\frac{1}{2}$. A greedy algorithm is as follows: start from an empty cut, add vertices of the graph one by one, and add each vertex to the side that maximizes the contribution of vertex v in the cut. The total weight of edges in the resulting cut is at least $\frac{1}{2}$ of the total weight of edges, thus, it is a $\frac{1}{2}$-approximation algorithm. Goemans and Williamson [30] gave a 0.878-approximation algorithm for the Max-Cut problem which is the best known approximation algorithm for this problem. Local search algorithms have been considered for this problem. At each step of the local search algorithm for the Max-Cut problem, we find a vertex such that if we move this vertex to the other side of the cut, the value of the cut increases. It is known that the cut value of a local optimal solution or equivalently the PSNE of the cut game of the cut game is at least $\frac{1}{2}$ of the optimal solution. The reason is that in a PSNE, the payoff of each vertex is at least $\frac{1}{2}$ of the total weight of the edges that are adjacent to this vertex. It follows that the sum of the payoffs of players in a PSNE of this game is at least the total weight of edges of the graph. Therefore, the cut value in a PSNE is at least $\frac{1}{2}$ of the maximum cut.

The cut game is a potential game. The potential function for this game is the value of the cut. As a result, selfish behavior of vertices will converge to a PSNE. But, it is well known that finding a local optimal solution of the Max-Cut local search problem or a PSNE of the cut game is PLS-complete [47, 79] and there are some configurations that are exponentially far from any local optimal solution. In other words, there are strategy profiles in the cut game in which the shortest best-response walks to any PSNE is exponentially long. On the positive side, Poljak [74] proved that for cubic graphs the convergence time is at most $O(n^2)$ steps.

3.2.1 Fast Convergence on Random Walks

First we prove positive results for the convergence to constant-factor approximate solutions with random walks. We show that the expected value of the cut after a random one-round walk is within a constant factor of the maximum cut.

Theorem 3.2.1 *In weighted graphs, the expected value of the cut at the end of a random one-round walk in the cut game is at least $\frac{1}{8}$ of the maximum cut.*

Proof. It suffices to show that after a random one-round walk, for every $v \in V(G)$, $\mathbf{E}[\alpha_v] \geq \frac{1}{8} w_v$.

Consider a vertex v. The probability that v occurs after exactly k of its neighbors is $\frac{1}{\deg(v)+1}$ for $k = 0, 1, \ldots, \deg(v)$. After v moves, the contribution of v in the cut is at least $\frac{w_v}{2}$. Conditioning on the fact that v occurs after exactly k neighbors, for each vertex u in the neighborhood of v, the probability that it occurs after v is $\frac{\deg(v)-k}{\deg(v)}$, and only in this case u can decrease the contribution of v in the cut by at most w_{uv}. Thus the expected contribution of v in the cut is at least $\max(0, w_v(\frac{1}{2} - \frac{\deg(v)-k}{\deg(v)}))$. Summing over all values of k, we obtain

$$\begin{aligned}
\mathbf{E}[\alpha_v] &\geq \sum_{k=0}^{\deg(v)} \frac{1}{\deg(v)+1} \max(0, w_v(\frac{1}{2} - \frac{\deg(v)-k}{\deg(v)})) \\
&\geq \frac{w_v}{\deg(v)+1} \sum_{k=\lfloor \frac{\deg(v)}{2} \rfloor + 1}^{\deg(v)} \frac{2k - \deg(v)}{2\deg(v)} \\
&\geq \frac{w_v}{8}.
\end{aligned}$$

The result follows by linearity of expectation. □

The next theorem studies a random walk of best responses that is not necessarily a one-round walk.

Theorem 3.2.2 *The expected value of the cut at the end of a random walk of length $3n \log n$ is at least a constant-factor of the maximum cut.*

Proof. Let $G(V, E)$ be a weighted graph, and let $X = x_1, x_2, \ldots, x_k$ be a sequence, where each x_i is chosen uniformly at random from $V(G)$. If $k = 3n \log n$, then X contains each element of $V(G)$ with probability $1 - \frac{1}{n^3}$. By the union bound, all vertices occur in X with probability $1 - \frac{1}{n^2}$. Thus, it is sufficient to prove the assertion conditioning on the fact that all vertices occur in X.

Assume now that X contains all the elements of $V(G)$, and for each $v \in V(G)$ let $t(v)$ be the largest i, with $1 \leq i \leq k$, such that $x_i = v$. Consider now the subsequence X' of X, such that X' contains only those elements x_i, such that $i = t(v)$, for some $v \in V(G)$. It is easy to see that X' induces a random one-round walk. Observe that for $x_{t(u)}, x_{t(v)} \in X'$, with $t(u) < t(v)$, we know that after vertex v plays, the contribution of v in the cut that is due to the edge $\{u, v\}$ cannot change. Therefore, by applying the same argument as in the proof of Theorem 3.2.1, the assertion follows. □

3.2.2 Poor Deterministic Convergence

We now give lower bounds for the convergence to approximate solutions for the cut social function. First, we give a simple example for which we need at least $\Omega(n)$ rounds of best responses to converge to a constant-factor cut. The construction resembles a result of Poljak [74].

Theorem 3.2.3 *There exists a weighted graph $G(V, E)$, with $|V(G)| = n$, and an ordering of vertices such that for any $k > 0$, the value of the cut after k rounds of letting players play in this ordering is at most $O(k/n)$ of the maximum cut.*

Proof. Consider a graph $G(V, E)$, with $V(G) = \{1, 2, \ldots, n\}$, and $E(G) = \{\{i, i+1\} | 1 \leq i \leq n-1\}$. For any i, with $1 \leq i < n$, the weight of the edge $\{i, i+1\}$, is $1 + (i-1)/n^2$. Since G is bipartite, the value of the maximum cut of G is $c(G) = \sum_{i=1}^{n-1}(1 + (i-1)/n^2) = \Omega(n)$. The graph G is depicted in Figure 3-1.

Let σ be an ordering of the vertices of G, with $\sigma(i) = i$. Consider the execution of the one-round walk for the ordering σ. At the beginning,, we have $T = V(G)$. It is easy to see that in any round $i \geq 1$, when vertex j plays, if $j \leq n - i$, j moves to the other part of the cut. Otherwise, if $j > n - i$, j remains in the same part of the cut. Thus, after round i, we have

$$T = \begin{cases} \{n, n-2, n-4, \ldots, n-i+1\} & \text{if i is odd} \\ \{1, 2, \ldots, n-i-1\} \cup \{n, n-2, n-4, \ldots, n-i\} & \text{if i is even} \end{cases}$$

It easily follows that the size of the cut after k rounds according to the ordering σ, is $\sum_{i=n-k}^{n-1} 1 + (i-1)/n^2 = O(k)$.

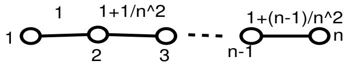


Figure 3-1: A path of length n on which k rounds of best responses of vertices result in a cut of value $\Omega(\frac{k}{n})$ of the maximum cut. The numbers on edges are the weight of the edges.

□

We next combine a modified version of the above construction with a result of Schaffer and Yannakakis for the Max-Cut local search problem [79] to obtain an exponentially-long walk with poor cut value.

Theorem 3.2.4 *There exists a weighted graph $G(V, E)$, with $|V(G)| = \Theta(n)$, and a k-covering walk $\mathcal{P}$ in the state graph, for some k exponentially large in n, such that the value of the cut at the end of $\mathcal{P}$, is at most $O(1/n)$ of the optimum cut.*

Proof. In [79], it is shown that there exists a weighted graph $G_0(V, E)$, and an initial cut $(T_0, \bar{T}_0)$, such that the length of *any* walk in the state graph, from $(T_0, \bar{T}_0)$ to a pure strategy Nash equilibrium, is exponentially long. Consider such a graph of size $\Theta(n)$, with $V(G_0) = \{v_0, v_1, \ldots, v_N\}$. Let $\mathcal{P}_0$ be an exponentially long walk from

$(T_0, \bar{T}_0)$ to a Nash equilibrium in which we let vertices $v_0, v_1, \ldots, v_N$ play in this order for an exponential number of rounds. Let $S_0, S_1, \ldots, S_{|\mathcal{P}_0|}$ be the sequence of states visited by $\mathcal{P}_0$ and let y_i be the vertex that plays his best response from state S_i to state S_{i+1}. The result of [79] guarantees that there exists a vertex, say v_0 that wants to change side (i.e., strategy) an exponential number of times along the walk $\mathcal{P}_0$ (since otherwise we can find a small walk to a pure Nash equilibrium). Let $t_0 = 0$, and for $i \geq 1$, let t_i be the time in which v_0 changes side for the i-th time along the walk $\mathcal{P}_0$. For $i \geq 1$, let $\mathcal{Q}_i$ be the sequence of vertices $y_{t_{i-1}+1}, \ldots, y_{t_i}$. Observe that each $\mathcal{Q}_i$ contains all of the vertices in G_0.

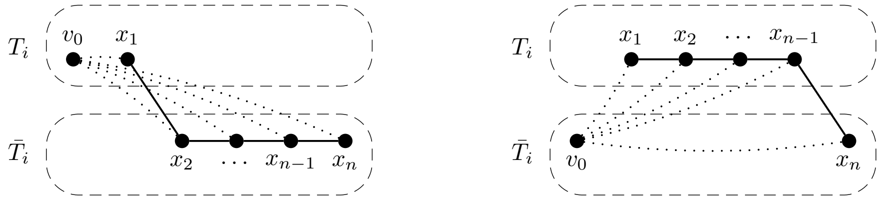
Consider now a graph G, which consists of a path $L = x_1, x_2, \ldots, x_n$, and a copy of G_0. For each $i \in \{1, \ldots, n-1\}$, the weight of the edge $\{x_i, x_{i+1}\}$ is 1. We scale the weights of G_0, such that the total weight of the edges of G_0 is less than 1. Finally, for each $i \in \{1, \ldots, n\}$, we add the edge $\{x_i, v_0\}$, of weight ϵ, for some sufficiently small ϵ. Intuitively, we can pick the value of ϵ, such that the moves made by the vertices in G_0, are independent of the positions of the vertices of the path L in the current cut.

For each $i \geq 1$, we consider an ordering $\mathcal{R}_i$ of the vertices of L, as follows: If i is odd, then $\mathcal{R}_i = x_1, x_2, \ldots, x_n$, and if i is even, then $\mathcal{R}_i = x_n, x_{n-1}, \ldots, x_1$.

We are now ready to describe the exponentially long path in the state graph. Assume w.l.o.g., that in the initial cut for G_0, we have $v_0 \in T_0$. The initial cut for G is $(T, \bar{T})$, with $T = \{x_1\} \cup T_0$, and $\bar{T} = \{x_2, \ldots, x_n\} \cup \bar{T}_0$. It is now straightforward to verify that there exists an exponentially large k, such that for any i, with $1 \leq i \leq k$, if we let the vertices of G play according to the sequence $\mathcal{Q}_1, \mathcal{R}_1, \mathcal{Q}_2, \mathcal{R}_2, \ldots, \mathcal{Q}_i, \mathcal{R}_i$, then we have (see Figure 3-2):

- If i is even, then $\{v_0, x_1\} \subset T$, and $\{x_2, \ldots, x_n\} \subset \bar{T}$.
- If i is odd, then $\{x_1, \ldots, x_{n-1}\} \subset T$, and $\{v_0, x_n\} \subset \bar{T}$.

It follows that for each i, with $1 \leq i \leq k$, the size of the cut is at most $O(1/n)$ times the value of the optimal cut. The result follows since each walk in the state graph induced by the sequence $\mathcal{Q}_i$ and $\mathcal{R}_i$ is a covering walk. $\square$



(a) i is even. (b) i is odd.

Figure 3-2: The cut $(T_i, \bar{T}_i)$ along the walk of the proof of Theorem 3.2.4 after playing $\mathcal{Q}_i$ and $\mathcal{R}_i$.

3.2.3 Mildly Greedy Players

By Theorem 3.2.1, it follows that for any graph, and starting from an arbitrary cut, there exists a walk of length at most n to an $\Omega(1)$-approximate cut. On the other hand, Theorems 3.2.3 and 3.2.4, show that there are cases where a deterministic ordering of players may result to very long walks that do not reach an approximately good cut.

We observe that if we change the game by assuming that a vertex changes side in the cut if his payoff is multiplied by at least a factor $1+\epsilon$, for a constant $\epsilon > 0$, then the convergence is faster. We call such vertices $(1+\epsilon)$-*greedy*. In the following, we prove that if all vertices are $(1+\epsilon)$-greedy for a constant $\epsilon > 0$, then the value of the cut after any one-round walk is within a constant factor of the optimum.

Theorem 3.2.5 *If all vertices are* $(1+\epsilon)$*-greedy the cut value at the end of any one-round walk is within a* $\min\{\frac{1}{4+2\epsilon}, \frac{\epsilon}{4+2\epsilon}\}$ *factor of the optimal cut.*

Proof. Consider a one-round walk $\mathcal{P}$. For each vertex v, let α'_v be the payoff of v right after its occurrence in $\mathcal{P}$, and let α_v be the payoff of v at the end of $\mathcal{P}$. Let V_1 be the set of vertices that did not change their side in the one-round walk and $V_2 = V(G) \setminus V_1$. For a vertex $v \in V_2$, let r_v be the total weight of the edges that are removed from the cut at the time that v moves to the other side. Since vertices are $1+\epsilon$-greedy, $\frac{w_v - r_v}{r_v} \geq 1+\epsilon$ otherwise v would not change side. Thus, $r_v \leq \frac{1}{2+\epsilon} w_v$. We claim that the difference between $\sum_{v \in V(G)} \alpha'_v$ and $\sum_{v \in V(G)} \alpha_v$ is at most $\sum_{v \in V_2} r_v$.

To see this, we observe that when a vertex $v \in V_2$ changes side, it can decrease the summation $\sum_{v\in V(G)} \alpha'_v$ by at most r_v. Moreover, if $v \in V_1$ then $\alpha'(v) \geq \frac{1}{2+\epsilon} w_v$, and if $v \in V_2$ then $\alpha'(v) \geq \frac{1+\epsilon}{2+\epsilon} w_v$. In the following, for a set $T \subseteq V(G)$, we let $W(T) = \sum_{v\in T} w_v$. Thus,

$$\begin{aligned}
\sum_{v\in V(G)} \alpha_v &\geq \sum_{v\in V(G)} \alpha'_v - \sum_{v\in V_2} r_v \\
&= \sum_{v\in V_1} \alpha'_v + \sum_{v\in V_2} \alpha'_v - \sum_{v\in V_2} r_v \\
&\geq \frac{1}{2+\epsilon} W(V_1) + \frac{1+\epsilon}{2+\epsilon} W(V_2) - \frac{1}{2+\epsilon} W(V_2) \\
&\geq \min\{\frac{1}{2+\epsilon}, \frac{\epsilon}{2+\epsilon}\} W(V(G)).
\end{aligned}$$

Thus, the value of the cut after this one-round walk is at least a $\min(\frac{1}{4+2\epsilon}, \frac{\epsilon}{4+2\epsilon})$-approximation. The best bound is obtained when $\epsilon = 1$, for which we obtain a $\frac{1}{6}$-approximate cut after one round. $\square$

3.2.4 Unweighted Graphs

In unweighted simple graphs, it is straight-forward to verify that the value of the cut at the end of an n^2-covering walk is at least $\frac{1}{2}$ of the optimum. The following theorem shows that in unweighted graphs, the value of the cut after any $\Omega(n)$-covering walk is a constant-factor approximation.

Theorem 3.2.6 *For unweighted graphs, the value of the cut after an $\Omega(n)$-covering walk is within a constant-factor of the maximum cut.*

Proof. Consider a k-covering walk $\mathcal{P} = (\mathcal{P}_1, \ldots, \mathcal{P}_k)$, where each $\mathcal{P}_i$ is a covering walk. Let $M_0 = 0$, and for any $i \geq 1$, let M_i be the size of the cut at the end of $\mathcal{P}_i$. Note that if $M_i - M_{i-1} \geq \frac{|E(G)|}{10n}$, for all i with $1 \leq i \leq k$, then clearly $M_k \geq k\frac{|E(G)|}{10n}$, and since the maximum size of a cut is at most $|E(G)|$, the lemma follows.

It remains to consider the case where there exists i with $1 \leq i \leq k$ such that $M_i - M_{i-1} < \frac{|E(G)|}{10n}$. Let V_1 be the set of vertices that change their side in the cut on the walk $\mathcal{P}_i$, and $V_2 = V(G) \setminus V_1$. Observe that when a vertex changes its side in the cut, the size of the cut increases by at least 1. Thus, $|V_1| < \frac{|E(G)|}{10n}$, and since the degree of each vertex is at most $n-1$, it follows that the number of edges that are incident to vertices in V_1, is less than $\frac{|E(G)|}{10}$.

On the other hand, if a vertex of degree d remains in the same part of the cut, then exactly after it plays, at least $\lceil d/2 \rceil$ of its adjacent edges are in the cut. Thus, at least half of the edges that are incident to at least one vertex in V_2, were in the cut, at some point during walk $\mathcal{P}_i$. At most $\frac{|E(G)|}{10}$ of these edges have an end-point in V_1, and thus at most that many of these edges may not appear in the cut at the end of $\mathcal{P}_i$. Thus, the total number of edges that remain in the cut at the end of walk $\mathcal{P}_i$, is at least $\frac{|E(G)|-|E(G)|/10}{2} - \frac{|E(G)|}{10} = \frac{7|E(G)|}{20}$. Since the maximum size of a cut is at most $|E(G)|$, we obtain that at the end of $\mathcal{P}_i$, the value of the cut is within a constant factor of the optimum. $\square$

We complement the upper bound of Theorem 3.2.6, by exhibiting an example that requires $\Omega(\sqrt{n})$ rounds of best responses to converge to a constant-factor cut.

Theorem 3.2.7 *There exists an unweighted graph $G(V,E)$ with $|V(G)| = n$ and an ordering of the vertices such that for any $k > 0$, the value of the cut after k rounds of letting players play in this ordering is at most $O(k/\sqrt{n})$ of the maximum cut.*

Proof. Let $V(G) = \{v_{i,j} | 1 \leq j \leq i \leq t\}$ and $E(G) = \{v_{i,j}v_{i+1,l} | 1 \leq j \leq i \leq t-1, 1 \leq l \leq i+1\}$. Clearly, G is bipartite, and thus the maximum cut value $c(G) = |E(G)| = \Omega(t^3) = \Omega(n^{3/2})$. The graph G is depicted in Figure 3-3. Vertex $v_{i,j}$ for any $1 \leq i,j \leq n$ is labeled by (i,j). Let the subset $\{v_{ij} | 1 \leq j \leq i\}$ of vertices be the layer i of vertices of this graph.

Consider now the ordering σ, such that for any i,j with $1 \leq j \leq i \leq t$, $\sigma(\frac{i(i-1)}{2} + j) = v_{i,j}$. We start from the empty cut. By an argument similar to the one used in the proof of Theorem 3.2.3, we obtain that after k rounds of letting players play according to the ordering σ, the size of the cut is at most $O(kt^2) = O(kn)$. In fact in the ith round of best responses of players, all vertices in layers $1, 2, \ldots, t-i$ change side to the other side of the cut.

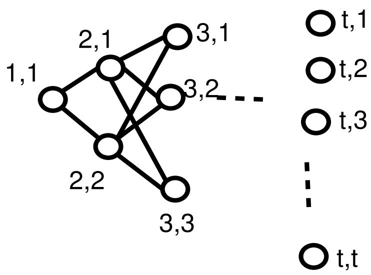


Figure 3-3: An unweighted graph G on which k rounds of best responses of vertices result in a cut of value $\Omega(\frac{k}{\sqrt{n}})$ of the maximum cut. The graph consists of t layers; layer i consist of i vertices, and all vertices of layer i are connected to vertices of layer $i+1$.

$\square$

3.3 Basic-utility Games: Convergence

In this section, we study the social value of random and deterministic best-response walks for basic-utility games. Recall that basic-utility games are a subclass of valid-utility games (See Section 1.2 for the definition of valid-utility and basic-utility games). Basic-utility games include a wide class of facility location games. These facility location games are introduced by Vetta [87]. Vetta [87] observed that basic-utility games are potential games. In fact, a potential function is the social utility function. To see this, consider a strategy profile $S = (s_1, \ldots, s_n)$ where $s_i \subseteq V_i$ and a player i who changes strategy from s_i to s'_i and increases her payoff. Therefore, $\alpha_i(S \oplus s'_i) > \alpha_i(S)$. The difference between the social value of $S \oplus s'_i$ and S is

$$\begin{aligned}
\gamma(S \oplus s'_i) - \gamma(S) &= (\gamma(S \oplus s'_i) - \gamma(S \oplus \emptyset_i)) - (\gamma(S) - \gamma(S \oplus \emptyset_i)) \\
&= \alpha_i(S \oplus s'_i) - \alpha_i(S) \\
&> 0.
\end{aligned}$$

Therefore, the social utility function is a potential function for basic-utility games. This shows that any sequence of strict improvement moves converges to a PSNE.

Vetta [87] proved that the price of anarchy for (mixed) Nash equilibria for basic-utility (and valid-utility) games is at most 2. This implies that in basic-utility games, any sequence of strict improvement moves converges to a state with social value at least $\frac{1}{2}$OPT. In the following, we study the rate of convergence to states with high social value on random walks.

Here, we prove that in basic-utility games, the expected social value of a state after $\Omega(n)$ random best responses is at least $\frac{1}{2} - \epsilon$ of the optimal social value, for any constant $\epsilon > 0$. In Chapter 4, we will show that this fast convergence does not hold for valid-utility games.

Theorem 3.3.1 *In basic-utility games, for any constant $\epsilon > 0$, there exists a constant c such that the expected social value of a state after $cn \log \frac{1}{\epsilon}$ random best responses is at least $\frac{1}{2} - \epsilon$ of the optimum. Moreover, for any constant $\epsilon' > 0$, there exist constants $\epsilon, c' > 0$ such that after $c'n \log n \log \frac{1}{\epsilon}$ random best responses, the social value is at least $\frac{1}{2} - \epsilon'$ of the optimum with high probability.*

Proof. Let $\Omega = (\sigma_1, \ldots, \sigma_n)$ denote an optimal state, and $T = (t_1, t_2, \ldots, t_n)$ be a strategy profile of agents. Let T^i be the strategy profile resulting from T after agent i plays its best response in T and let $\Omega^i = (\sigma_1, \ldots, \sigma_i, \emptyset_{i+1}, \ldots, \emptyset_n)$. Let $Y = \frac{1}{n} \sum_{i \in U} \gamma(T^i)$ be the expected social value of the state after a random agent plays its best response. Our goal is to lower bound Y.

To do so, using submodularity, basicness and the cake condition we get:

$$
\begin{aligned}
nY - n\gamma(T) &= \sum_{i\in U}(\gamma(T^i) - \gamma(T)) \\
&= \sum_{i\in U}(\gamma(T^i) - \gamma(T\oplus\emptyset_i)) - \sum_{i\in U}(\gamma(T) - \gamma(T\oplus\emptyset_i)) \\
&= \sum_{i\in U}\alpha_i(T^i) - \sum_{i\in U}\alpha_i(T) \quad \text{[by basicness]} \\
&\geq \sum_{i\in U}\alpha_i(T^i) - \gamma(T) \quad \text{[by cake condition]} \\
&\geq \sum_{i\in U}\alpha_i(T^i\oplus\sigma_i) - \gamma(T) \quad \text{[since } i \text{ plays his best response in } T^i\text{]} \\
&= \sum_{i\in U}\gamma'_{\sigma_i}(T\oplus\emptyset_i) - \gamma(T) \quad \text{[by basicness]} \\
&\geq \sum_{i\in U}(\gamma(T\oplus\sigma_i) - \gamma(T\oplus\emptyset_i)) - \gamma(T) \\
&\geq \sum_{i\in U}(\gamma(T\cup\Omega^i) - \gamma(T\cup\Omega^{i-1})) - \gamma(T) \quad \text{[by submodularity]} \\
&= \gamma(T\cup\Omega) - \gamma(T) - \gamma(T) \quad \text{[since it is a telescopic summation]} \\
&\geq \text{OPT} - 2\gamma(T) \quad \text{[since } \gamma \text{ is non-decreasing].}
\end{aligned}
$$

The above inequalities show that $Y \geq \frac{n-2}{n}\gamma(T) + \frac{1}{n}\text{OPT}$. Let Y_0 be the actual social value of the initial state. At each step, a random agent is picked and plays its best response. Thus, if Y_i is the social value of the state after step i, then $\mathbf{E}[Y_i|Y_{i-1} = y] \geq (\frac{n-2}{n})y + \frac{1}{n}\text{OPT}$. Let $p_{yy'}$ be the probability that $Y_{i-1} = y'$ given that $Y_{i-2} = y$. Thus, $\mathbf{E}[Y_{i-1}|Y_{i-2} = y] = \sum_{y'} p_{yy'}y'$. Therefore,

$$
\begin{aligned}
\mathbf{E}[Y_i|Y_{i-2} = y] &= \sum_{y'} p_{yy'}\mathbf{E}[Y_i|Y_{i-2} = y, Y_{i-1} = y'] \\
&\geq \sum_{y'} p_{yy'}((\frac{n-2}{n})y' + \frac{1}{n}\text{OPT}) \\
&= (\frac{n-2}{n})\mathbf{E}[Y_{i-1}|Y_{i-2} = y] + \frac{1}{n}\text{OPT} \\
&\geq (\frac{n-2}{n})((\frac{n-2}{n})y + \frac{1}{n}\text{OPT}) + \frac{1}{n}\text{OPT} \\
&= (\frac{n-2}{n})^2 y + \frac{1}{n}\text{OPT}(1 + (\frac{n-2}{n})).
\end{aligned}
$$

Thus, $\mathbf{E}[Y_i|Y_{i-2} = y] \geq (\frac{n-2}{n})^2 y + \frac{1}{n}\mathrm{OPT}(1 + (\frac{n-2}{n}))$. Similarly, we can prove that $\mathbf{E}[Y_i|Y_0 = y_0] \geq (\frac{n-2}{n})^i y_0 + \frac{1}{n}\mathrm{OPT}(1 + (\frac{n-2}{n}) + \ldots + (\frac{n-2}{n})^{i-1})$. Since $y_0 \geq 0$, $\mathbf{E}[Y_i] \geq \frac{\mathrm{OPT}}{2}(1 - (1 - \frac{2}{n})^i)$.

This proves that for a sufficiently large constant c and by setting $i = cn \log \frac{1}{\epsilon}$, the expected social value after $cn \log \frac{1}{\epsilon}$ best responses is at least $\frac{1}{2} - \epsilon$ of the optimum. Moreover, since in basic-utility games the social value is non-decreasing as agents play their best responses, we claim that for a sufficiently large $c' = cc'' > 0$ and a sufficiently small $\epsilon > 0$, after $c'n \log n \log \frac{1}{\epsilon}$ random best responses, with high probability the social value is at least $\frac{1}{2} - \epsilon'$ of the optimum. The reason is that we can partition the best response walk of length $c'n \log n \log \frac{1}{\epsilon}$ into $c'' \log n$ best-response walks of length $cn \log \frac{1}{\epsilon}$, and after each of these subwalks, the expected social value is at least $\frac{1}{2} - \epsilon$ of the optimum. Thus, by Markov inequality, with a constant probability after each of the subwalks of length $cn \log \frac{1}{\epsilon}$, the expected social value is at least $(\frac{1}{2} - \epsilon')$ of the optimum. Hence, after $c'n \log n \log \frac{1}{\epsilon}$ best responses, the social value is at least $\frac{1}{2} - \epsilon'$ of the optimum with high probability. □

3.4 Market Sharing Games: Convergence

In this section we consider the market sharing game. For the formal definition and an introduction to these games, see Chapter 2. Note that in this game, to find the best-response strategy, each player should solve a knapsack problem. Therefore, in order to model computationally constrained agents, we may assume that the agents apply λ-approximation algorithms to determine their best-response strategies. More precisely, when a player uses a λ-approximation algorithm for his best-response move, he changes his strategy if he does not decrease his payoff and his payoff after this move is at least λ times his payoff after any other move from this state. We then obtain the following theorems concerning the social value after one round of best responses moves. In the following we use the Harmonic number $H_n = 1 + \frac{1}{2} + \ldots + \frac{1}{n}$.

Theorem 3.4.1 *In market sharing games, the social value of a state at the end of a one-round walk is at least* $\frac{1}{2H_n+1}$ *of the optimal social value (or at least* $\frac{1}{\frac{\lambda+1}{\lambda}H_n+1}$ *if* *the agents use λ-approximation algorithms).*

Proof. Let $\Omega = (\sigma_1, \ldots, \sigma_n)$ denote an optimal state. Here $\sigma_i \subseteq V$ is the set of markets that player i services in this optimal solution; we may also assume that each market is provided by at most one player in Ω (we will use this fact later in the proof). Let $T = (t_1, \ldots, t_n)$ and $S = (s_1, \ldots, s_n)$ be the initial and final states on the one-round walk, respectively. We assume that the agents play best-response strategies in the order $1, 2, \ldots, n$. So in step r, using a λ-approximation algorithm, agent r changes its strategy from t_r to s_r; thus $T^r = (s_1, \ldots, s_r, t_{r+1}, \ldots, t_n)$ is an intermediate state in the one-round walk $\mathcal{P} = \{T = T^0, T^1, \ldots, T^n = S\}$. The social value of state $S = T^n$ is $\gamma(S) = \sum_{i \in U} \alpha_i(S)$. We need to show that $\sum_{i \in U} \alpha_i(S) \geq \frac{1}{1+\frac{\lambda+1}{\lambda}H_n}\mathrm{OPT}$. Towards this goal, we first show that $\gamma(S) = \sum_{i \in U} \alpha_i(T^n) \geq \frac{1}{H_n} \sum_{i \in U} \alpha_i(T^i)$.

We know that agent r does not change its strategy from s_r after step r. Therefore a market j has a nonzero contribution in $\gamma(S)$ if and only if market j has a nonzero contribution in the summation $\sum_{i \in U} \alpha_i(T^i)$. For any market j, if j appears in any of strategies in T^n then the contribution of j to $\gamma(S)$ is q_j. On the other hand, at most n players use market j in their strategies. The payoff of the first player who serves market j is at most q_j. The payoff of the second player who served market j is at most $\frac{q_j}{2}$, since when he plays, at least one other player serve market j. Similarly, the payoff of the ith player who serves market j is at most $\frac{q_j}{i}$, since when he plays, at least $i-1$ other players serve market j. Consequently, the contribution of market j in the summation $\sum_{i \in U} \alpha_i(T^i)$ is at most

$$(1 + \frac{1}{2} + \frac{1}{3} + \ldots + \frac{1}{n})q_j = H_n q_j.$$

It follows that $\sum_{i \in U} \alpha_i(T^n) \geq \frac{1}{H_n} \sum_{i \in U} \alpha_i(T^i)$, as required. We denote by $\mathcal{T}$ the summation $\sum_{i \in U} \alpha_i(T^i)$. Consider the optimal assignment Ω, and let Y_i be the set of markets that are serviced by agent i in σ_i but that are *not* serviced by any agent in S, that is, $Y_i = \sigma_i - \cup_{r \in U} s_r$. Now, $\gamma(S)$ is greater than the value of all the markets in $\cup_{r \in U}(\sigma_r - Y_r)$, since these markets are a subset of markets serviced in S. Hence, using the notation $q(Q) = \sum_{j \in Q} q_j$ to denote the sum of the value of a subset Q of the markets, we have $\gamma(S) \geq \sum_{r\in U} q(\sigma_r - Y_r)$; here we use the fact that in the optimum each market is served by at most one agent.

Next, we will prove that $\mathcal{T} \geq \lambda \sum_{r\in U} q(Y_r)$. Let Y_i' be the markets in Y_i that are not serviced in T^i, that is, $Y_i' = Y_i - (s_1 \cup \cdots \cup s_i \cup t_{i+1} \cup \cdots \cup t_n)$. Then Y_i' is a feasible strategy for agent i at step i. Thus, we have $\alpha_i(T^i) \geq \lambda q(Y_i')$, since player i uses a λ-approximation algorithm. Therefore, $\mathcal{T} \geq \lambda \sum_{r\in U} q(Y_r')$. Again, we use the fact that Y_i''s are disjoint.

Finally, we claim that $\mathcal{T} \geq \sum_{i\in U} q(Y_i'')$ where $Y_i'' = Y_i - Y_i'$. To see this, consider a market $j \in Y_i''$. Market j is not in the strategy set of any agent in T^n, but is in the strategy set of at least one player in T^i. Therefore, somewhere on the walk $\mathcal{P}$, after T^i, some player must change its strategy and discontinue servicing market j. Also note that if $j \in Y_i''$ then $j \notin Y_{i'}''$ for any $i' \neq i$, since each market is serviced by at most one player in Ω. Let b_j be the time step such that T^{b_j} is the first state amongst $T^{i+1}, \ldots, T^n$ that does not service market j. Let $M_i = \{j \in V | b_j = i\}$ be the set of markets for which $b_j = i$. It follows that $\cup_{r\in U} Y_r'' = \cup_{r\in U} M_r$. Notice that $M_i \subseteq t_i$ and no other agents service any market in M_i in T^{i-1}, since after player i changes his strategy any market in M_i is not serviced in T^i. Player i changes his strategy from t_i to s_i and does not service M_i in T^i, thus the payoff of player i in T^i is at least $\sum_{j\in M_i} q_j$ (since j is only served by i in T^{i-1}). As a result, $\alpha_i(T^i) \geq q(M_i)$ (since player i changes his strategy only if he can increase his payoff). Therefore, $\sum_{i\in U} q(Y_i'') = \sum_{i\in U} q(M_i) \leq \sum_{i\in U} \alpha_i(T^i) = \mathcal{T}$. Hence we have,

$$\begin{aligned}
\text{OPT} &= \sum_{i\in U} q(\sigma_i) \\
&\leq \sum_{i\in U} q(\sigma_i - Y_i) + \sum_{i\in U} q(Y_i) \\
&\leq \gamma(S) + \sum_{i\in U} q(Y_i') + \sum_{i\in U} q(Y_i'') \\
&\leq \gamma(S) + \frac{1}{\lambda}\mathcal{T} + \mathcal{T} \\
&\leq (1 + \frac{\lambda+1}{\lambda} H_n)\gamma(S).
\end{aligned}$$

□

Theorem 3.4.2 *In market sharing games, the social value of a state at the end of a one-round walk may be as bad as $\frac{1}{H_n}$ of the optimal social value.*

Proof. Consider the following instance of a market sharing game. There are $m = n$ markets, and the value of market j is $q_j = \frac{n}{j} - \epsilon$ for all $1 \leq j \leq n$ where ϵ is sufficiently small. The cost of market j is $C_j = 1 + (n - j)\epsilon$ for $2 \leq j \leq n$ and $C_1 = 1$. There are n players and the budget of player i is equal to $1 + (n - i)\epsilon$. As a result, player i can only serve markets 1 and $i, i + 1, \ldots, n$ and we assume that he is interested is serving all these markets. Consider the ordering $1, 2, \ldots, n$ and the one-round walk starting from the empty set of strategies and letting each player play once in this order. The resulting assignment after this one-round walk is that all players provide market number 1 and the social value of this assignment is $n - \epsilon$. However, in the optimal solution, agent i services market i. This gives an optimal social value of $nH_n - n\epsilon$. Thus, the ratio between the optimum and the value of the resulting assignment is H_n at the end of a one-round walk. □

3.5 Conclusion and Open Problems

In this chapter, we introduced a framework for studying speed of convergence to approximate solutions in potential games. We believe that in order to capture the computational issues of the performance of systems under lack of coordination, instead of just bounding the performance of a Nash equilibrium, it is necessary to bound the performance of the system along walks induced by a polynomial number of movements by players. We are especially interested in the performance of the system along fair walks (e.g., random and covering walks) to equilibria, since we may hope for better social functions along these walks. This, in turn, has implications in local search method in optimization. In order to use local search to optimize a function, we do not need to find the local optimum. We can find short walks to approximate solutions, e.g., by randomizing over the choice of the next local operation. Similar questions can be asked about different classes of games and local optimization problems. In Chapter 4, we will see another example of games in which the convergence to equilibria is exponential, but we can show fast convergence to approximate solutions on random walks.

Market sharing games are not yet well understood. In particular, it is not known whether exponentially long best-response paths exist. Bounding the social value of a state at the end of a k-covering path is another open question. In Chapter 2, we gave a polynomial-time algorithm to find the pure Nash equilibrium in uniform market sharing games. Finding such an equilibrium is NP-complete for the general case, but the question of obtaining approximate Nash equilibria is open.

Among the problems for the convergence in cut games, we do not know if the result of Theorem 3.2.1 for random one-round walks holds with high probability or not. The complexity of finding an approximate Nash equilibrium in the above cut game is not known to us. Another open problem is bounding the length of walks to Nash equilibria in cut games in which all players are $(1 + \epsilon)$-greedy.

Chapter 4

Sink Equilibria and Convergence

A standard approach in analyzing the performance of systems controlled by non-cooperative agents is by the examination of Nash equilibria. Of particular interest is the *price of anarchy* in a game [53]. This gives one measure of the cost to society of the inherent *lack of coordination* in a game. As mentioned in Chapter 3, there are several drawbacks in the use of Nash equilibria. For example, one issue relates to use of non-randomized (pure) and randomized (mixed) strategies. Often pure Nash equilibria may not exist, yet the use of a randomized (mixed) strategy is unrealistic in many games. This necessitates the need for an alternative solution concept in evaluating such games. Another issue arises from the observation that Nash equilibria represent "stable" points in a system. Therefore (even if pure Nash equilibria exist), they are a more acceptable solution concept if it is likely that the system does converge to such stable points. In particular, the use of Nash equilibria seems more valid in games in which Nash equilibria arise when players iteratively engage in selfish behavior. However, in many games it is not the case that repeated selfish behavior always leads to Nash equilibria. In these games it also seems that another measure of the cost of the lack of coordination would be useful. Observe that these issues are particularly important in games in which the use of pure strategies and repeated moves are the norm, for example, auctions. We remark that for most practical games these properties are the rule rather than the exception and this observation motivates much of the work in this work. In this chapter, we address this issue by introducing a new solution concept in a game, namely *sink equilibria* and studying the (expected) social value in these equilibria. We first formally define this equilibrium concept.

4.1 Sink Equilibria

We model the behavior of agents via a *state graph.* In many games with iterative moves, the evolution of game-play may then be naturally modeled by a path in the state graph. Such a path may or may not converge to a PSNE; observe that a PSNE is a vertex in the state graph for which the best response move of each agent corresponds to a self-loop. Clearly it may also be the case that there are no PSNE. We may ask what happens in such games. Specifically, does some concept of stability or equilibrium exist? The answer is yes, and we now describe such an "equilibrium".

Consider the strongly connected components of the state graph. If we contract the strongly connected components to singletons then we obtain an acyclic graph. The sink nodes in this graph (nodes with out-degree equal to zero) correspond to strongly connected components with no out-going arcs in the state graph. We call such a strongly connected component a *sink equilibrium* [1]. The reason for this terminology is clear: if a best-response walk ever reaches a node in a sink equilibrium then it will never leave that set of nodes. This justifies using the terms sink and equilibrium. In addition, a long enough random walk in the state graph will converge to a sink equilibrium with probability arbitrarily close to 1. (This model can be justified in extensive games with complete information and is used in the economics literature extensively in the context of studying convergence in these games.)

We denote by $\mathcal{Q}$ the set of sink equilibria in a game. We remark that the union of states in sink equilibria correspond to the set of recurrent states in a Markov chain that only has non-zero transitional probabilities on arcs in the state graph. In a random sequence of best responses of agents, we choose an agent uniformly at random at each step and independent of the previous players and let this agent play

[1] We can also call this equilibrium the *myopic equilibrium*, since we only consider myopic best responses in the state graph

his best response (if the agent has more than one best-response move, we may assume that the agent arbitrarily chooses a move from the collection of best-response moves). When this walk reaches a state in some sink we then have a *random walk* over the states in that sink. For a sink $Q \in \mathcal{Q}$, let $\pi_Q : Q \to \mathbb{R}^+ \cup \{0\}$ be the steady state distribution of the random walk over states in Q. Let $\gamma(S)$ measure the social value of a state S. The (expected) social value of a sink equilibrium $Q \in \mathcal{Q}$, denoted by $\Gamma(Q)$, is the expected social value of states given by the steady distribution of the random walk over the states of Q, i.e., $\Gamma(Q) = \sum_{S \in Q} \pi_Q(S)\gamma(S)$[2] We then define, the *price of sinking*[3] for a maximization social function as

$$\text{Price of Sinking} = \frac{\text{OPT}}{\min\limits_{Q \in \mathcal{Q}} \Gamma(Q)} = \frac{\text{OPT}}{\min\limits_{Q \in \mathcal{Q}} \sum_{S \in Q} \pi_Q(S)\gamma(S)}.$$

In other words, the price of sinking is the worst ratio between the expected social value of a sink equilibrium and the social value of the optimum. Similarly, the price of sinking for a minimization problem is $\frac{\max_{Q \in \mathcal{Q}} \Gamma(Q)}{\text{OPT}}$. Given that sink equilibria are stable solutions in such games, this is a more realistic measure of the cost of the lack of coordination than the price of anarchy.

We illustrate the use of the price of sinking in Section 4.2 where we present an n-player valid-utility game that always converges to states with social value a factor n worse than optimal. Indeed, the price of sinking for this game is n. However the price of anarchy is almost 1. Thus, the price of anarchy gives us a misleading confidence in the social quality of an outcome that will result from selfish behavior.

As well as being a more appropriate solution concept than PSNE in many games, the existence of sink equilibria has several nice implications. Since sink equilibria always exist, the price of sinking can always be calculated even in games without PSNE. Unlike PSNE, sink equilibria also possess natural convergence properties. The price of sinking also has a close relation to the convergence to approximate solutions (states

[2]It is possible to define $\Gamma(Q) = \min_{S \in Q} \gamma(S)$. In fact, proving positive results with this definition is harder and stronger. We chose the definition based on the expected social value on the random walk, since it is closely related to the rate of convergence to approximate solutions on random best-response walks which is the topic of Chapter 3 of this thesis as well.

[3]We may call this *the price of myopia* since we only consider myopic best responses of players.

whose expected social value is within an approximation factor of optimal) in games. In particular, our proof techniques to bound the price of sinking also imply bounds on the speed of convergence to approximate solutions. We study two examples in Section 4.3:

(1) *Unsplittable Selfish Routing (and Weighted Congestion Games).* We present instances of the weighted unsplittable flow version of the selfish routing problem that possess no PSNE (For the formal definition of the unsplittable selfish routing game, see Section 4.3.1). However, we show that, for bounded-degree polynomial latency functions of degree at most d, the price of sinking is at most $O(2^{2d}d^{2d+3})$. In addition, our proof technique implies fast convergence to high quality solutions. This may be compared to the negative result by Fabrikant, Papadimitriou, and Talwar [15] showing the existence of exponentially long best-response walks to PSNE (in the unweighted version of this game). For example, consider the case of linear latency functions. Here it is known that PSNE exist [24]; it may be the case that the number of best-response moves needed for convergence to a PSNE is exponential. Our results show that after a polynomial number of random best-response moves, the social value of the flow is within a constant factor of the optimal solution.

(2) *Valid-Utility Games.* Our second example concerns the class of valid-utility games; Here, we show that the price of sinking is at most $n + 1$; thus the worst case price of sinking in a valid-utility game is between n and $n + 1$.

We also present a hardness result concerning sink equilibria. In section 4.4 we show that in general it is a PLS-hard problem to find a sink equilibrium (or PSNE) in valid-utility games. This implies the existence of exponentially long best-response paths to any sink equilibrium in valid-utility games.

We conclude this introduction with a very brief discussion on related work. In order to deal with the stability and convergence problems of Nash equilibria, equilibrium concepts other than Nash equilibria have been studied in the economics literature. Among these concepts are stable equilibria [52], stochastic adjustment models [48], iterative elimination of dominated strategies, the set of undominated strategies etc. Convergence and strategic stability of equilibria in evolutionary game theory is a also central subject of study for many economists. However, in their studies the most important factor is typically the stability of equilibria, and not measurements of the social value of equilibria. In Chapter 3, we began our investigation into the importance of moving away from the use of Nash equilibria as the main solution concept for measuring performance in a game.

4.2 Price of Sinking vs. Price of Anarchy

In this section, we present an n-agent game in which the price of sinking and the price of anarchy give very different pictures as to the consequences of non-cooperative behavior. In particular, the price of anarchy will be close to 1, suggesting that no form of mechanism design is required to enforce socially good solutions. However, every possible outcome of the game will result in a solution whose value is a factor n smaller than that of the optimal social solution.

Here, we present an n-agent (valid-utility) game for which the price of sinking is $\frac{n+\epsilon}{1+\epsilon}$ but the price of anarchy is just $1 + \frac{\epsilon}{n}$ for an $\epsilon > 0$. The ground set V_i of agent i consists of $n+1$ elements $V_i = \{y_i, x_i^1, x_i^2, \ldots, x_i^n\}$. For motivation, we can think of strategy y_i as a socially responsible strategy for agent i. In contrast, all the strategies $\{x_i^1, x_i^2, \ldots, x_i^n\}$ can be viewed as socially irresponsible strategies. Moreover, we will see that in any situation one of these n irresponsible strategies provides a better payoff for agent i than acting responsibly. Consequently, there is an incentive for every agent to act anti-socially with extreme consequences for the social outcome. In contrast, the price of anarchy is oblivious to this incentive for anti-social behavior. The reason being that the payoffs to each agent are intrinsically linked to the behavior of the other agents. Any specific irresponsible strategy may be beneficial in certain circumstances but typically (given the other agents responses) that specific strategy has smaller payoff than the responsible strategy. Consequently, unlike randomized strategies, playing an irresponsible strategy is likely to lead to low private returns. Thus mixed strategy Nash equilibria will require that most agents behave responsibly, blissfully ignoring the fact that in *every* possible situation *each* agent has an incentive to behave irresponsible.

The family of feasible strategies F_i for each agent i is the set of singletons of his ground set and the empty set, i.e., $F_i = \{s \subseteq V_i : |s| \leq 1\}$. Let $X_i = \{x_i^1, x_i^2, \ldots, x_i^n\}$ and $X = \cup_i X_i$. Let $S = (s_1, s_2, \ldots, s_n)$ be a vector of subsets $s_i \subseteq V_i$ for all $i = 1, 2, \ldots, n$. For a vector $S = (s_1, \ldots, s_n)$, we let $S^{\mathsf{U}} = \cup_{i \in U} s_i$. We construct a non-decreasing, submodular social utility function γ on $\Pi_{i \in U} V_i$ in the following manner.

$$\gamma(S) = \begin{cases} |S^{\mathsf{U}} \backslash X| & \text{if } S^{\mathsf{U}} \cap X = \emptyset \\ |S^{\mathsf{U}} \backslash X| + 2 & \text{otherwise.} \end{cases}$$

With this social utility function, we construct a valid-utility game. To do this we need to specify the private utilities of each agent at any state. In order to define the payoff functions, we define a function $i^*(S)$ for each strategy profile S. We set $i^*(S) = \text{null}$ for any strategy profile S in which no player plays an irresponsible strategy. If in a strategy profile S, some players play irresponsibly, $i^*(S)$ is the index of a player who plays irresponsibly. In addition, $i^*(S)$ satisfies the following property: given the strategies of the other agents, any agent i can always choose some irresponsible strategy so that after i's playing $i^*(S) = i$. In the following, we give one example of a function i^* that satisfies these properties.

Let $\chi_{ij}(S)$ be the indicator variable for the event that agent i plays the irresponsible strategy x_i^j. That is

$$\chi_{ij}(S) = \begin{cases} 1 & \text{if } x_i^j \in S^{\mathsf{U}} \\ 0 & \text{otherwise.} \end{cases}$$

Next let

$$i^*(S) = \begin{cases} \text{null} & \text{if } S^{\mathsf{U}} \cap X = \emptyset \text{(no one plays irresponsibly)} \\ i_l & \text{if } S^{\mathsf{U}} \cap X_i \neq \emptyset \text{ for } i \in \{i_1, \ldots, i_k\} \\ & \text{and } l = [\sum_{i \in U}(\sum_{j=1}^{n} j \cdot \chi_{ij}(S)) \mod k] + 1 \end{cases}$$

Observe that if $i^*(S) = \text{null}$ then i can play the irresponsible strategy $s_i' = \{x_i^i\}$ and make $i^*(S \oplus s_i') = i$. Otherwise, there exists a strategy $s_i' = \{x_i^p\}$ such that if i plays $s_i' = \{x_i^p\}$ it makes $i^*(S \oplus s_i') = i$.

We are now ready to give a payoff function α_i for each agent i.

$$\alpha_i(S) = \begin{cases} 0 & \text{if } y_i \notin s_i \text{ and } i \neq i^*(S) \\ 1 & \text{if } y_i \in s_i \text{ and } i \neq i^*(S) \\ 2 & \text{if } y_i \notin s_i \text{ and } i = i^*(S) \\ 3 & \text{if } y_i \in s_i \text{ and } i = i^*(S). \end{cases}$$

So agent i gets utility 1 for playing the responsible strategy and another 2 units of utility if $i = i^*(S)$. We will see in Section 4.3.2 that this is a valid-utility game with a non-decreasing social utility function. Thus we may apply the following result from [87].

Theorem 4.2.1 *The price of anarchy in a valid-utility game with a non-decreasing social utility function is at most* 2. □

If fact, it is easy to see that the price of anarchy in this game actually tends to 1 as the number of agents increases. In particular, it is easy to see that a socially optimal solution has $n-1$ of the agents playing their responsible strategies while exactly one of the agents plays an irresponsible strategy. Such an outcome has value $n+1$. Moreover, note that by playing responsibly an agent can guarantee that they receive 1 unit of utility. Thus, it must be the case that in a Nash equilibrium[4] every agent has an expected payoff of at least 1. Since $\gamma(S) \geq \sum_{i \in U} \alpha_i(S)$ for any state S, we have that the expected social value of a Nash equilibrium is at least n. Thus the price of anarchy is at most $1 + \frac{1}{n}$.

Now we consider the price of sinking in this game. Given any strategy profile S, the best response of each agent is to play the specific irresponsible strategy that gives it a payoff of 2. To see this, note that agent i always has a move that sets $i^*(S') = i$. Thus a responsible strategy y_i is never a best-response strategy. In fact, the best

[4]One Nash equilibrium is the following. Each agent i plays strategy y_i with probability p and each bad strategy with probability $\frac{1-p}{n}$. It is easy to check that letting $p = \sqrt[n-1]{\frac{1}{2}(1 - \frac{1}{n-1})}$ gives a Nash equilibrium.

response of each player is to play an irresponsible strategy to get the payoff of 2, but each player makes the payoff of other players who are playing the irresponsible strategy 0. It follows that there is a unique sink equilibrium consisting of every strategy profile in which each agent plays an irresponsible strategy. Thus, every state in the sink has social value exactly two. Hence the price of sinking is exactly $\frac{n+1}{2}$. We remark that even if we start at an optimal solution, if each agent makes a single best-response move in turn then we end up with a solution of value 2! Moreover, we can then never leave this sink if players play their myopic best responses.

Notice also that we could alter the payoffs in the game slightly so that the payoff resulting from the first irresponsible move is $1 + \delta$ rather than 2. Clearly the price of sinking is then $\frac{n+\delta}{1+\delta}$ which tends to n. Thus we have

Lemma 4.2.2 *There are valid-utility games, with non-decreasing social utility functions, for which the price of sinking is almost n while the price of anarchy is almost* 1. □

Thus the price of anarchy underestimates the social cost of the lack of coordination by a factor n. The reason for this is that the good strategy always gives a good return. Any bad strategy can give a high return but only in a small number of situations, thus any bad strategy performs badly against randomized strategies and players tend to play their good strategies in the mixed Nash equilibria. Hence the price of anarchy is good. This type of issue often arises in games, and explains why the price of anarchy will often significantly under-estimate the social cost of the lack of coordination in such games.

Finally, note that this game has no PSNE so focusing here upon sink equilibria is essential. Surprisingly, Lemma 4.2.2 is also almost tight; we will show in Section 4.3 that the price of sinking in a valid-utility game is at most $n + 1$.

4.3 Price of Sinking and Convergence

PSNE are special cases of sink equilibria. We have already seen that games in which agents repeatedly react to the other agent's strategies via the use of pure strategy best responses will converge to sink equilibria and not necessarily to PSNE. Moreover, many classes of games have instances for which no PSNE exists. In these games, we can still measure the cost to society of the lack of coordination using the price of sinking. Moreover, in bounding the price of sinking for sink equilibria we may obtain bounds on the expected social value of states after a random sequence of best responses.

4.3.1 Unsplittable Selfish Routing and Weighted Congestion Games

Consider the "unsplittable flow" version of the selfish routing game. We have a directed network $G = (V, E)$ with a flow dependent latency function $\ell_e : \mathbb{R} \rightarrow \mathbb{R}^+ \cup \{0\}$ on each arc $e \in E$. There is a set U of n agents; agent i wishes to route flow at a rate r_i from a source s_i to a sink t_i. Each agent aims to incur as small a latency as possible. In the unsplittable flow version, an agent may not split its flow. Hence each agent picks a unique $s_i - t_i$ path and routes all its flow along the path. The latency of an agent is equal to its traffic size multiplied by the sum of the latencies of arcs along the path that it chooses. The latency of an arc e is a non-decreasing and non-negative function of the total load on arc e. In this chapter, we consider bounded-degree polynomial latency functions. In particular, for an arc e, we let $\ell_e(x) = \sum_{0 \leq j \leq d} a_{e,j} x^j$ be a non-negative and non-decreasing delay function for arc e. For a strategy profile $\mathcal{P} = (P_1, P_2, \ldots, P_n)$ where P_i is a $s_i - t_i$ path, let the load of arc e be $f_e = \sum_{i: e \in P_i} r_i$. Then, the latency of agent i is $l_i(f) = r_i \sum_{e \in P_i} \ell_e(f_e)$ and the total latency of flow f is $l(f) = \sum_{i \in U} l_i(f) = \sum_{e \in E(G)} \ell_e(f_e) f_e$.

Before stating our results on weighted unsplittable selfish routing games, we note that all our results on these games extend to the general class of weighted congestion games. Weighted congestion games are the generalization of weighted unsplittable selfish routing game in which the family of feasible strategies of players are arbitrary family of subsets of arcs (and not necessarily paths from a source to a destination). This definition extends the definition of congestion games defined in Section 1.2. In none of the proofs for the price of sinking and convergence, we use the fact that the feasible strategy is a path. Therefore, all the results hold for general weighted congestion games.

Recently Awerbuch, Azar, and Epstein [2] proved that the price of anarchy in such games is exactly 2.618 for linear latency functions and is at most $O(2^d d^{d+1})$ for polynomial latency functions of degree at most d. They extended their results to mixed Nash equilibria, since the existence of pure Nash equilibria for these games with polynomial latency functions was not known. For linear latency function Fotakis, Kontogiannis, and Spirakis [24] proved that the game is a potential game. Here, we exhibit an instance of this game with quadratic latency functions that does not possess any PSNE. This, in turn, provides additional motivation for analyzing the price of sinking in these games. Our example is shown in Figure 4-1. It depicts a network with 4 vertices and 6 arcs. Arcs are labeled from 1 to 6. The latency functions of arcs are $\ell_1(x) = x + 33$, $\ell_2(x) = 13x$, $\ell_3(x) = 3x^2$, $\ell_4(x) = 6x^2$, $\ell_5(x) = x^2 + 44$, and $\ell_6(x) = 47x$. There are two agents with traffic $r_1 = 1$ and $r_2 = 2$. The source of both agents is vertex 1 ($s_1 = s_2 = 1$) and the destination of both agents is vertex 4 ($t_1 = t_2 = 4$). Consider four paths $P_1 = (6)$, $P_2 = (3, 5)$, $P_3 = (3, 4, 2)$, and $P_4 = (1, 2)$ where the numbers within the parentheses are the labels of arcs on the path. It is not hard to check that the only sink equilibrium of the weighted unsplittable selfish routing game on this network is the set of strategy profiles $\{(P_1, P_2), (P_3, P_2), (P_3, P_4), (P_1, P_4)\}$. To see this, one can check the following inequalities[5]:

$$
\begin{aligned}
l_1(P_1, P_2) = \ell_6(r_1) = 47 \quad &> \quad 46 = \ell_3(r_1 + r_2) + \ell_4(r_1) + \ell_2(r_1) = l_1(P_3, P_2) \\
l_2(P_3, P_2) = 2(\ell_3(r_1 + r_2) + \ell_5(r_2)) = 150 \quad &> \quad 148 = 2(\ell_1(r_2) + \ell_2(r_1 + r_2)) = l_2(P_3, P_4) \\
l_1(P_3, P_4) = \ell_3(r_1) + \ell_4(r_1) + \ell_2(r_1 + r_2) = 48 \quad &> \quad 47 = \ell_6(r_1) = l_1(P_1, P_4) \\
l_2(P_1, P_4) = 2(\ell_1(r_2) + \ell_2(r_2)) = 122 \quad &> \quad 120 = 2(\ell_3(r_2) + \ell_5(r_2)) = l_2(P_1, P_2)
\end{aligned}
$$

Using the above inequalities, we can show that $\{(P_1, P_2), (P_3, P_2), (P_3, P_4), (P_1, P_4)\}$ is a sink equilibrium. The only feasible strategies of players are paths P_1, P_2, P_3, P_4.

[5] In fact, we have found this example by solving a program to find a solution that satisfy these inequalities using CPLEX.

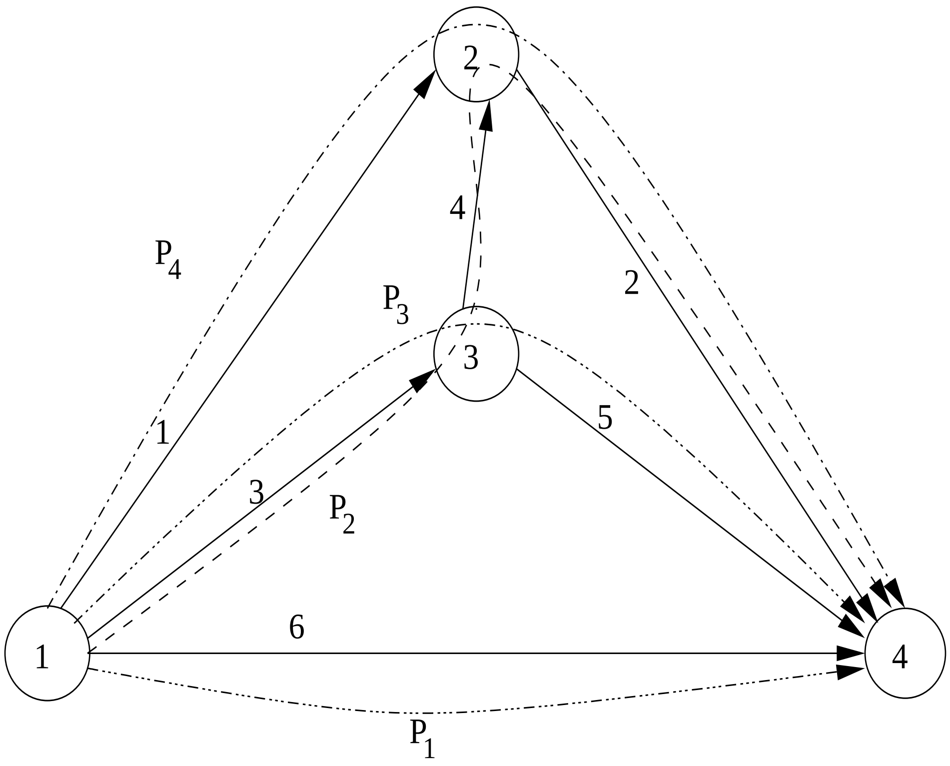


Figure 4-1: A weighted unsplittable selfish routing game without PSNE. Vertices are labeled from 1 to 4. Arcs are labeled from 1 to 6. Four paths (P_1, P_2, P_3, P_4) are highlighted from vertex 1 to vertex 4. Two players with traffic loads 1 and 2 send traffic from vertex 1 to vertex 4.

Moreover, if player 2 plays one of the paths, the best response of player 1 is one of the paths P_1 and P_3. Also, if player 1 plays one of the paths, the best response of player 2 is one of the paths P_2 and P_4.

Table 4.1 depicts the best-response move from any strategy profile in which players play one of the four paths. This table shows that no PSNE exists in this game and the only sink equilibrium of this game is $\{(P_1, P_2), (P_3, P_2), (P_3, P_4), (P_1, P_4)\}$.

The key to obtaining bounds on the price of sinking is that any agent making a best-response move cannot cause too much cumulative harm to the other agents. Consequently, if an agent can make a move that significantly increases its private welfare, then the overall social welfare must rise. This will be an important factor in allowing us to prove that we have a low price of sinking in these routing games.

Theorem 4.3.1 *The price of sinking for a weighted unsplittable selfish routing game*

	P_1	P_2	P_3	P_4
P_1	(2, P_2)	(1, P_3)	(2, P_2)	(2, P_2)
P_2	(2, P_4)	(1, P_3)	(2, P_4)	(1, P_1)
P_3	(2, P_4)	(2, P_4)	(2, P_4)	(1, P_1)
P_4	(2, P_4)	(1, P_3)	(1, P_1)	(1, P_1)

Table 4.1: The table corresponding to the weighted unsplittable selfish routing game without PSNE. Rows correspond to the strategy of the first player. Columns correspond to the strategy of the second player. The pair (i, P) in a cell of the table indicates that player i's best response in this strategy profile is P.

(or a weighted congestion game) is at most $O(2^{2d}d^{2d+3})$.

Proof. We need the following three lemmas for the proof.

Lemma 4.3.2 *Let f be the flow corresponding to the current strategy profile $\mathcal{P} = (P_1, \ldots, P_n)$. Suppose agent i changes its flow path from P_i to P_i', to give a new flow f_i'. Then $l(f_i') \leq l(f) + (d+1)l_i(f_i') - l_i(f)$. In particular, if agent i decreases its latency by changing to P_i', then $l(f_i') \leq l(f) + dl_i(f) \leq (d+1)l(f)$.*

Proof. The latency incurred by agent i is then

$$l_i(f_i') = r_i \sum_{e \in P_i'} \sum_{0 \leq j \leq d} a_{e,j}(f_{i,e}')^j = r_i \left(\sum_{e \in P_i' \cap P_i} \sum_{0 \leq j \leq d} a_{e,j} f_e^j + \sum_{e \in P_i' - P_i} \sum_{0 \leq j \leq d} a_{e,j}(f_e + r_i)^j \right).$$

Note that for $e \in P_i' - P_i$, we have $f_{i,e}' = f_e + r_i$. Moreover, we know that

$$l(f_i') \leq l(f) + (l_i(f_i') - l_i(f)) + \sum_{e \in P_i' - P_i} \left(\sum_{0 \leq j \leq d} (a_{e,j} {f_{i,e}'}^{j}) - (a_{e,j} f_e^j) \right) (f_{i,e}' - r_i),$$

the last term corresponding to the increase in latency for agents other than i due to the rerouting of agent i. We can get an upper bound on the increase in latencies faced by the other agents by noting that

$$\begin{aligned}
&\sum_{e\in P'_i-P_i}\left(\sum_{0\le j\le d}(a_{e,j}{f'_{i,e}}^{j})-(a_{e,j}f_e^j)\right)(f'_{i,e}-r_i)\\
&= \sum_{e\in P'_i-P_i}\sum_{0\le j\le d}(a_{e,j}({f'_{i,e}}^{j}-f_e^j)f_e)\\
&= \sum_{e\in P'_i-P_i}\left(\sum_{0\le j\le d}a_{e,j}(f'_{i,e}-f_e)\left(\sum_{1\le t\le j}{f'_{i,e}}^{j-t}f_e^{t-1}\right)f_e\right)\\
&< \sum_{e\in P'_i-P_i}\left(\sum_{0\le j\le d}a_{e,j}r_i\left(\sum_{1\le t\le j}(f_e+r_i)^{j-1}\right)(f_e+r_i)\right)\\
&\le r_i\sum_{e\in P'_i-P_i}\left(\sum_{0\le j\le d}ja_{e,j}(f_e+r_i)^j\right)\\
&\le dl_i(f'_i).
\end{aligned}$$

Thus, the total latency after agent i changes its strategy is at most $l(f)+(d+1)l_i(f'_i)-l_i(f)$. Since, $l_i(f'_i)\le l_i(f)$, this shows that $l(f'_i)\le l(f)+dl_i(f)\le(d+1)l(f)$. □

Lemma 4.3.3 *Let f be the flow corresponding to the current strategy profile. Consider the following random process: choose an agent i at random and let it play its best response. If f' is the new flow after this change, then* $\mathbf{E}[l(f')|f]\le(1+\frac{d}{n})l(f)$.

Proof. Let f'_i be the flow after agent i plays its best response to f. Then, using Lemma 4.3.2, we have:

$$\begin{aligned}
\mathbf{E}[l(f')|f] &= \frac{1}{n}\sum_{i\in U}l(f'_i)\\
&\le \frac{1}{n}\sum_{i\in U}(l(f)+dl_i(f))\\
&= \frac{1}{n}(nl(f)+dl(f))\\
&= (1+\frac{d}{n})l(f).
\end{aligned}$$

□

The third lemma we need is below. Its proof is inspired by the work of Azar et al. [2].

Lemma 4.3.4 *Let f be the flow corresponding to the current strategy profile. Consider the following random process: choose an agent i at random and let it play its best response. If f' is the new flow after this change, then either $\mathbf{E}[l(f')|f] \le (1-\frac{1}{2n})l(f)$, or $l(f) \le O(2^{2d}(d+1)^{2d+2})\mathrm{OPT}$.*

Proof. Assume that the best response of agent i is to switch from path P_i to P_i' resulting in the flow f_i'. Thus, $\mathbf{E}[l(f')|f] = \frac{1}{n}\sum_{i\in U} l(f_i')$. We consider the following two cases:

Case 1: $\sum_{i\in U} 2(d+1)l_i(f_i') \le \sum_{i\in U} l_i(f)$. In this case, by Lemma 4.3.2,

$$\begin{aligned}
\mathbf{E}[l(f')|f] &= \frac{1}{n}\sum_{i\in U} l(f_i') \\
&\le \frac{1}{n}\sum_{i\in U} (l(f) + (d+1)l_i(f_i') - l_i(f)) \\
&\le \frac{1}{n}\left(\sum_{i\in U} l(f) + \sum_{i\in U}\frac{1}{2}l_i(f) - \sum_{i\in U} l_i(f)\right) \\
&= \frac{1}{n}(nl(f) - \frac{1}{2}l(f)) \\
&= (1-\frac{1}{2n})l(f).
\end{aligned}$$

Thus, we obtain $\mathbf{E}[l(f')|f] \le (1-\frac{1}{2n})l(f)$.

Case 2: $\sum_{i\in U} 2(d+1)l_i(f_i') > \sum_{i\in U} l_i(f)$. Let $\mathcal{P}^* = (P_1^*, \ldots, P_n^*)$ be the optimal solution and let f^* be the flow corresponding to $\mathcal{P}^*$. Set $J^*(e) = \{i : e \in P_i^*\}$. Let f_i^* be the flow resulting from the switch of agent i from P_i to P_i^*. Since P_i' is i's best response, we have $l_i(f_i^*) \ge l_i(f_i')$. Thus, in this case, $\sum_{i\in U} 2(d+1)l_i(f_i^*) \ge$

$\sum_{i\in U} l_i(f) = l(f)$. Consequently,

$$\begin{aligned}
l(f) &\leq \sum_{i\in U} 2(d+1) l_i(f_i^*) \\
&\leq (2d+2)\sum_{i\in U} r_i \sum_{e\in P_i^*} \ell_e(f_e + r_i) \\
&= (2d+2)\sum_{i\in U} r_i \sum_{e\in P_i^*}\sum_{j=0}^{d} a_{e,j}(f_e + r_i)^j \\
&= (2d+2)\sum_{e}\sum_{j=0}^{d}\sum_{i\in J^*(e)} a_{e,j}(f_e + r_i)^j r_i.
\end{aligned}$$

The rest of the proof of this case is based on the proof of Lemmas A1, A2, and A3 in [2]. First, we use the following inequality from [2]: $(x+y)^d \leq cx^d + (y(\frac{d}{\ln c}+1))^d$ for any $c > 1$. Thus, we get:

$$\begin{aligned}
l(f) &\leq (2d+2)\sum_{e}\sum_{j=0}^{d}\sum_{i\in J^*(e)} a_{e,j}(f_e + r_i)^j r_i \\
&\leq (2d+2)\sum_{e}\sum_{j=0}^{d} a_{e,j} \sum_{i\in J^*(e)} \left(c f_e^j r_i + \left(\frac{j}{\ln c}+1\right)^j r_i^{j+1}\right) \\
&\leq (2d+2)\sum_{e}\sum_{j=0}^{d} a_{e,j} \left(c f_e^j f_e^* + \left(\frac{d}{\ln c}+1\right)^d f_e^{*j+1}\right) \\
&= c(2d+2)\sum_{e}\sum_{j=0}^{d} a_{e,j} f_e^j f_e^* + (2d+2)\left(\frac{d}{\ln c}+1\right)^d \sum_{e}\sum_{j=0}^{d} a_{e,j} f_e^{*j+1} \\
&= c(2d+2)\sum_{e}\sum_{j=0}^{d} a_{e,j} f_e^j f_e^* + (2d+2)\left(\frac{d}{\ln c}+1\right)^d \sum_{e} \ell_e(f_e^*) f_e^*
\end{aligned}$$

where the second inequality comes from the fact that $\sum_{i\in J^*(e)} r_i^d \leq f_e^{*d}$ and the function $f(x) = (\frac{x}{\ln c}+1)^x$ is an increasing function for $x \geq 0$. Holder's inequality states:

$$\sum_j a_j^{\alpha} b_j^{1-\alpha} \leq \left(\sum_j a_j\right)^{\alpha}\left(\sum_j b_j\right)^{1-\alpha}.$$

Applying this, with $a_j = a_{e,j} f_e^{j+1}$, $b_j = a_{e,j} f_e^{*j+1}$, $\alpha = \frac{j}{j+1}$, yields

$$
\begin{aligned}
l(f) \leq c(2d+2) \sum_e \sum_{j=0}^{d} a_{e,j} f_e^j f_e^* + (2d+2) \left(\frac{d}{\ln c} + 1 \right)^d \sum_e \ell_e(f_e^*) f_e^* \\
\leq \ & 2c(d+1) \sum_{j=0}^{d} \left(\sum_e a_{e,j} f_e^{j+1} \right)^{j/(j+1)} \left(\sum_e a_{e,j} f_e^{*j+1} \right)^{1/(j+1)} \\
& +2(d+1) \left(\frac{d}{\ln c} + 1 \right)^d \sum_e \ell_e(f_e^*) f_e^* \\
\leq \ & 2c(d+1) \sum_{j=0}^{d} \left(\sum_e \ell_e(f_e) f_e \right)^{j/(j+1)} \left(\sum_e \ell_e(f_e^*) f_e^* \right)^{1/(j+1)} \\
& +2(d+1) \left(\frac{d}{\ln c} + 1 \right)^d \sum_e \ell_e(f_e^*) f_e^* \\
\leq \ & 2c(d+1) \sum_{j=0}^{d} \left(\sum_e \ell_e(f_e) f_e \right)^{d/(d+1)} \left(\sum_e \ell_e(f_e^*) f_e^* \right)^{1/(d+1)} \\
& +2(d+1) \left(\frac{d}{\ln c} + 1 \right)^d \sum_e \ell_e(f_e^*) f_e^* \\
\leq \ & 2c(d+1)^2 \left(\sum_e \ell_e(f_e) f_e \right)^{d/(d+1)} \left(\sum_e \ell_e(f_e^*) f_e^* \right)^{1/(d+1)} \\
& +2(d+1) \left(\frac{d}{\ln c} + 1 \right)^d \sum_e \ell_e(f_e^*) f_e^*
\end{aligned}
$$

where the fourth inequality is from the inequality $x^\alpha y^{1-\alpha} \geq x^{\alpha'} y^{1-\alpha'}$ for $x \geq y > 0$ and $1 \geq \alpha \geq \alpha' \geq 0$ with $x = \sum_e \ell_e(f_e) f_e$ and $y = \sum_e \ell_e(f_e^*) f_e^*$. By letting

$$x = \frac{l(f)^{\frac{1}{d+1}}}{\mathrm{OPT}^{\frac{1}{d+1}}},$$

we get

$$x^{d+1} \leq 2c(d+1)^2 x^d + 2(d+1) \left(\frac{d}{\ln c} + 1 \right)^d.$$

After dividing both sides by x^d, we get:

$$x \leq 2c(d+1)^2 + 2(d+1) \left(\frac{\frac{d}{\ln c} + 1}{x} \right)^d.$$

We claim that if we set $c = 2 - \epsilon$ for $\epsilon = \frac{1}{d+1}\left(\frac{1}{2(d+1)}\right)^d$, then we have $x \le 4(d+1)^2$. Assume for contradiction that $x > 4(d+1)^2$. Then,

$$4(d+1)^2 < x \le 4(d+1)^2 - 2\epsilon(d+1)^2 + 2(d+1)\left(\frac{\frac{d}{\ln c}+1}{x}\right)^d.$$

Thus,

$$\begin{aligned}
(d+1)\epsilon &< \left(\frac{\frac{d}{\ln c}+1}{x}\right)^d \\
&\le \left(\frac{2d+1}{x}\right)^d \qquad \text{[since } \ln c > 0.5\text{]} \\
&< \left(\frac{2d+1}{4(d+1)^2}\right)^d \\
&< \left(\frac{1}{2(d+1)}\right)^d \\
&= (d+1)\epsilon
\end{aligned}$$

which is a contradiction. Therefore, by setting $c = 2-\epsilon$, we get $x \le 4(d+1)^2$. Hence, $l(f) = x^{d+1}\mathrm{OPT} \le O(2^{2d}(d+1)^{2d+2})\mathrm{OPT}$. □

From Lemma 4.3.4, we can bound the price of sinking as follows. Consider a sink Q. Let f_0 be a flow in Q. Consider a random walk starting from f_0 in which we let a random agent play his best response at each step. Let $f_0, f_1, f_2, \ldots, f_N$ be a sequence of observed flows in Q. Recall that the value for sink Q is equal to $\Gamma(Q) = \sum_{S\in Q} \pi_Q(S) l(f_S)$ where f_S is the flow corresponding to the state S and π_Q is the steady distribution for the random walk on Q. Since Q is strongly connected, this is equal to $\Gamma(Q) = \lim_{N\to\infty} \frac{\sum_{0\le j\le N} \mathbf{E}[l(f_j)]}{N}$. In order to upper bound this value, it is sufficient to upper bound $\mathbf{E}[l(f_j)]$ for each $0 \le j \le N$. Lemma 4.3.4 shows that there exists a state in any sink Q with total latency less than $O(2^{2d}(d+1)^{2d+2})\mathrm{OPT}$. Note that, as Q is strongly connected the value of the sink is independent of the choice of f_0. Therefore, we can set f_0 such that $l(f_0) \le c' 2^{2d}(d+1)^{2d+2}\mathrm{OPT}$. Let c_i be the coin toss of step i in the random walk. More precisely, we want to upper bound

$a_j = E_{c_1,c_2,\ldots,c_j}[l(f_j)]$. By Lemma 4.3.4 and Lemma 4.3.3, we have

- Either $E_{c_{j+1}}[l(f_{j+1})|f_j] \leq (1 - \frac{1}{2n})l(f_j)$ or $l(f_j) \leq c'2^{2d}(d+1)^{2d+2}\mathrm{OPT}$.
- $E_{c_{j+1}}[l(f_{j+1})|f_j] \leq (1 + \frac{d}{n})l(f_j)$

Let E_1 be the event that $l(f_j) \leq c'2^{2d}(d+1)^{2d+2}$ and E_2 be the event that $l(f_j) > c'2^{2d}(d+1)^{2d+2})\mathrm{OPT}$. Let p be the probability that event E_2 happens. Furthermore, let $Y = \mathbf{E}[l(f_j)|E_1] \leq c'2^{2d}(d+1)^{2d+2}$ and $X = \mathbf{E}[l(f_j)|E_2]$. Thus, $a_j = \mathbf{E}[l(f_j)] = pX + (1-p)Y$. Now,

$$\begin{aligned}
a_{j+1} &= \mathbf{E}[l(f_{j+1})] \\
&\leq p\left(1 - \frac{1}{2n}\right)X + (1-p)\left(1 + \frac{d}{n}\right)Y \\
&\leq \left(1 - \frac{1}{2n}\right)(pX + (1-p)Y) + \frac{2d+1}{2n}Y \\
&\leq \left(1 - \frac{1}{2n}\right)a_j + \frac{2d+1}{2n}Y \\
&\leq \left(1 - \frac{1}{2n}\right)a_j + \frac{2d+1}{2n}c'2^{2d}(d+1)^{2d+2}\mathrm{OPT}.
\end{aligned}$$

Combining the above recurrence relation and $a_0 \leq l(f_0) \leq 2c'2^{2d}(d+1)^{2d+3}\mathrm{OPT}$, we can prove $a_{j+1} \leq 2c'2^{2d}(d+1)^{2d+3}\mathrm{OPT}$ by induction. Thus, $E_{c_1,c_2,\ldots,c_j}[l(f_j)] \leq O(2^{2d}(d+1)^{2d+3}\mathrm{OPT})$. Hence, the price of sinking is at most $O(2^{2d}(d+1)^{2d+3})$ by the linearity of expectation. As $(d+1)^{2d+3} = O(d^{2d+3})$, we have the desired bound. □

We can also use the lemmas used in the proof of Theorem 4.3.1 to bound the rate of convergence to states with good social value in unsplittable (weighted) selfish routing games. We can prove that starting from a flow of latency C, after $O(n \log \frac{C}{\mathrm{OPT}})$ random best responses, the expected social value is less than 70 OPT for linear latency functions for any $\epsilon > 0$, and is less than $O(2^{2d}d^{2d+3})\mathrm{OPT}$ for polynomial latency functions of degree at most d. This is in contrast with the negative convergence result of Fabrikant, Papadimitriou, and Talwar [15], in which they exhibit exponentially long best-response paths to PSNE (or sink equilibria) in these games. Our bounds show that, even though convergence to PSNE (or sink equilibria) may be exponential, a random sequence of best responses of agents converges to a state with good social value after polynomial number of best responses. Here, we prove a tighter bound for convergence in the weighted unsplittable selfish routing game with linear latency functions. We assume that the latency function of arc e is a linear function. In particular, we let the latency function for arc $e \in E(G)$ be $\ell_e(x) = a_e x + b_e$ with $a_e, b_e \geq 0$.

Theorem 4.3.5 *In the weighted unsplittable selfish routing game with linear latency functions, starting from any state with total latency C the expected latency of the flow after $O(n \log \frac{C}{\mathrm{OPT}})$ random best responses is at most 70 OPT for any $\epsilon > 0$.*

Proof. Let f be the current flow, and suppose agent i changes its flow path from P_i to P'_i, to give a new flow f'_i. From Lemma 4.3.2, $l(f'_i) \leq l(f) + 2l_i(f'_i) - l_i(f)$. We will use the following refinement to Lemma 4.3.4.

Lemma 4.3.6 *Let f be the flow corresponding to the current strategy profile. Consider the following random process: choose an agent i at random and let it play its best response. If f' is the new flow after this change, then either $\mathbf{E}[l(f')|f] \leq (1 - \frac{1}{2n})l(f)$, or $l(f) \leq 23.32$ OPT.*

Proof. Assume that the best response of agent i is to switch from path P_i to P'_i resulting in the flow f'_i. Thus, $\mathbf{E}[l(f')|f] = \frac{1}{n}\sum_{i \in U} l(f'_i)$. We consider the following two cases:

Case 1: $\sum_{i \in U} 4l_i(f'_i) \leq \sum_{i \in U} l_i(f)$. In this case, similar to Case 1 of the proof of Lemma 4.3.4, it follows that $\mathbf{E}[l(f')|f] \leq (1 - \frac{1}{2n})l(f)$.

Case 2: $\sum_{i \in U} 4l_i(f'_i) > \sum_{i \in U} l_i(f)$. Let $\mathcal{P}^* = (P^*_1, \ldots, P^*_n)$ be the optimal solution and let f^* be the flow corresponding to $\mathcal{P}^*$. Set $J^*(e) = \{i : e \in P^*_i\}$. Let f^*_i be the flow resulting from the switch of agent i from P_i to P^*_i. Since P'_i is i's best response, we have $l_i(f^*_i) \geq l_i(f'_i)$. In this case, we can apply the method of Azar et al. [2] as follows:

$$\begin{aligned}
l(f) &= \sum_i (r_i \sum_{e\in P_i}(a_e f_e + b_e)) \\
&\le \sum_{i\in U} 4 l_i(f_i') \\
&\le \sum_{i\in U} 4 l_i(f_i^*) \qquad\qquad [\text{since player } i \text{ play his best response to } f_i'] \\
&\le \sum_{i\in U} 4 r_i \sum_{e\in P_i^*} (a_e(f_e + r_i) + b_e) \\
&= 4 \sum_{i\in U} \sum_{e\in P_i^*} \left[(a_e f_e + b_e) r_i + a_e r_i^2\right] \\
&\le 4 \sum_{e} \sum_{i:e\in P_i^*} \left[(a_e f_e + b_e) r_i + a_e r_i^2\right].
\end{aligned}$$

It follows that

$$\begin{aligned}
l(f) &\le 4 \sum_e f_e^*(a_e f_e + b_e) + 4 \sum_e a_e f_e^{*2} \\
&= 4 \sum_e f_e^* a_e f_e + 4 \sum_e (a_e f_e^* + b_e) f_e^* \\
&= 4 \sum_e f_e^* a_e f_e + 4 \text{ OPT} \\
&\le 4 \sqrt{\left(\sum_e (\sqrt{a_e} f_e)^2\right)\left(\sum_e (\sqrt{a_e} f_e^*)^2\right)} + 4 \text{ OPT}[\text{Cauchy-Schwartz inequality}] \\
&= 4 \sqrt{\left(\sum_e a_e f_e^2\right)\left(\sum_e a_e f_e^{*2}\right)} + 4 \text{ OPT} \\
&\le 4 \sqrt{\sum_e (a_e f_e + b_e) f_e \sum_e (a_e f_e^* + b_e) f_e^*} + 4 \text{ OPT} \\
&= 4 \sqrt{l(f)\text{OPT}} + 4 \text{ OPT}.
\end{aligned}$$

By setting $x = \frac{l(f)}{\text{OPT}}$, we have $x \le 4(\sqrt{x} + 1)$. This gives $x \le 23.32$. Hence, in this case, $l(f) \le 23.32$ OPT. □

Proof of Theorem 4.3.5. Let $a_0 = C$ be the social value of the initial flow. Assume that at each step we choose an agent at random and let it play its best response. Let

a_j be the expected latency of the flow after j's step. From Lemma 4.3.6, we have for any $j \geq 0$, $a_j \leq 23.32$ OPT or $a_{j+1} \leq a_j(1 - \frac{1}{2n})$. Moreover, from Lemma 4.3.3, $a_{j+1} \leq a_j(1 + \frac{1}{n})$ for any $j \geq 0$. Now, let p be the probability that $a_j > 23.32$ OPT. Let X be the expected value of a_j given that $a_j > 23.32\text{OPT}$ and Y be the expected value of a_j given that $a_j \leq 23.32\text{OPT}$. Thus,

$$\begin{aligned} a_{j+1} &\leq p\left(1 - \frac{1}{2n}\right) X + (1-p)(1+\frac{1}{n})Y \\ &\leq \left(1 - \frac{1}{2n}\right)(pX + (1-p)Y) + \frac{3}{2n}Y \\ &\leq \left(1 - \frac{1}{2n}\right) a_j + \frac{69.96}{2n}\text{OPT}. \end{aligned}$$

It follows that

$$a_j \leq a_{j-i}\left(1 - \frac{1}{2n}\right)^i + \frac{69.96}{2n}\text{OPT}\frac{\left(1 - \left(1 - \frac{1}{2n}\right)^i\right)}{\frac{1}{2n}}$$

for $i \leq j$. As a result, $a_j \leq a_0\left(1 - \frac{1}{2n}\right)^j + 69.96\ \left(1 - \left(1 - \frac{1}{2n}\right)^j\right)\text{OPT} \leq C\left(1 - \frac{1}{2n}\right)^j + 69.96$ OPT. Thus, for $j \geq n \log \frac{1}{\epsilon} \log \frac{C}{\text{OPT}}$, we get $a_j \leq (69.96 + \epsilon)$ OPT. Therefore, after $O(n \log \frac{C}{\text{OPT}})$ steps the expected value of a_j is at most 70 OPT. $\square$

Finally, we note that all our results on the price of sinking and convergence for weighted unsplittable selfish routing games extend to weighted congestion games, since we never used the fact that the strategy of players is a path from the source to the destination, and the feasible strategies could be any subset of arcs of the network. It follows that these results hold for general weighted congestion games.

4.3.2 Valid-Utility Games

Valid-utility games are formally defined in Section 1.2. This class of games are introduced by Vetta [87] for which he proved that the price of anarchy for mixed Nash equilibria is at most 2. Here we prove bounds on the worst-case price of sinking in valid-utility games. First, we show that our bad example in Section 4.2 is a valid-utility game. Thus the price of sinking in valid-utility games can be as bad as n. Then, we will prove that this lower bound for valid-utility games is almost tight. In particular, we will show that the price of sinking in a valid-utility game is at most $n+1$.

In order to prove that the bad example in Section 4.2 is a valid-utility game, we need to verify three conditions:

1) Non-decreasing and Submodular Social Function: First, it is clear that the corresponding set function of the social function γ^s is non-decreasing. To show its submodularity, we use an equivalent definition of submodular functions: A set function f is submodular if for any two subsets A and B such that $A \subset B$ and for any element $i \notin B$, $f(A \cup \{i\}) - f(A) \geq f(B \cup \{i\}) - f(B)$ [26]. Thus, in order to prove that γ^s is submodular, it is enough to prove that for two (possibly infeasible) strategy profiles $S = (s_1, \ldots, s_n)$ and $S' = (s'_1, \ldots, s'_n)$ such that $s_i \subseteq s'_i$ for all $i \in U$, by adding a new element j to the strategy of any player i the increase in γ^s for S is not less than the increase for S'. First, we consider the case that $j = x_i^t$. If $S'^{\cup} \cap X = \emptyset$ then $S^{\cup} \cap X = \emptyset$, and thus $\gamma'_{x_i^t}(S' \oplus \emptyset_i) = 2$ and $\gamma'_{x_i^t}(S \oplus \emptyset_i) = 2$. If $S'^{\cup} \cap X \neq \emptyset$ then $\gamma'_{x_i^t}(S' \oplus \emptyset_i) = 0 \leq \gamma'_{x_i^t}(S \oplus \emptyset_i)$. Hence if $j = x_i^t$, the desired condition for submodularity holds. Also, if $j = y_i$ it is implied that $\gamma'_{y_i}(S' \oplus \emptyset_i) = 1$ if and only if $S'^{\cup} \cap \{y_i\} = \emptyset$, otherwise $\gamma'_{y_i}(S' \oplus \emptyset_i) = 0$. It follows that $\gamma'_{y_i}(S' \oplus \emptyset_i) \leq \gamma'_{y_i}(S \oplus \emptyset)$. Therefore, γ^s is submodular.

2) Vickrey Condition: If player i plays y_i then she gets 1 and the social value changes by 1. If player i plays an element of X_i and increases the social value by 2, then she is the only player who plays an irresponsible strategy. Thus, $i = i^*(S)$ and so she receives those two utility units. Otherwise the playing of an element of X_i has no effect on the social value. Thus, the Vickrey condition is trivially satisfied.

3) Cake Condition: It is straightforward to check that $\sum_{i \in U} \alpha_i(S) = \gamma(S)$ and the cake condition holds.

Now, we prove that this bound is almost tight.

Lemma 4.3.7 *Given a strategy profile $T = (t_1, \ldots, t_n)$ in a valid-utility game, let the best response of agent i be s_i. Set $T^i = (t_1, \ldots, t_{i-1}, s_i, t_{i+1}, \ldots, t_n)$. Then $\sum_{i \in U} \alpha_i(T^i) \geq \mathrm{OPT} - \gamma(T)$.*

Proof. Let $\Omega = (\sigma_1, \ldots, \sigma_n)$ be the optimum state. Let

$$\Omega^i = (\sigma_1, \sigma_2, \ldots, \sigma_i, \emptyset_{i+1}, \emptyset_{i+2}, \ldots, \emptyset_n).$$

Given that s_i is a best-response strategy, we have $\alpha_i(T^i) \geq \gamma'_{\sigma_i}(T \oplus \emptyset_i)$. Combining this with the submodularity of γ, we obtain

$$\begin{aligned}
\sum_{i \in U} \alpha_i(T^i) &\geq \sum_{i \in U} \gamma'_{\sigma_i}(T \oplus \emptyset_i) \\
&= \sum_{i \in U} (\gamma(T \oplus \sigma_i) - \gamma(T \oplus \emptyset_i)) \\
&\geq \sum_{i \in U} (\gamma(T \cup \sigma_i) - \gamma(T)) \\
&\geq \sum_{i \in U} (\gamma(T \cup \Omega^i) - \gamma(T \cup \Omega^{i-1})) \\
&= \gamma(T \cup \Omega) - \gamma(T).
\end{aligned}$$

Since γ is non-decreasing, it follows that $\sum_{i \in U} \alpha_i(T^i) \geq \mathrm{OPT} - \gamma(T)$. □

Theorem 4.3.8 *The price of sinking in a valid-utility game is at most $n + 1$.*

Proof. Consider a sink equilibrium Q. Let $T = (t_1, \ldots, t_n)$ be a state in Q. Let the best response of agent i be s_i at state T, and set $T^i = (t_1, \ldots, t_{i-1}, s_i, t_{i+1}, \ldots, t_n)$. Let Y be the expected social value of the state after a random best-response move from T. By the cake property and Lemma 4.3.7, we have

$$
\begin{aligned}
Y &= \frac{1}{n}\sum_{i\in U}\gamma(T^i) \\
&\geq \frac{1}{n}\sum_{i\in U}\alpha_i(T^i) \\
&\geq \frac{1}{n}(\mathrm{OPT}-\gamma(T)).
\end{aligned}
$$

Observe that the price of sinking is equal to the expected social value on a sufficiently long random walk. Now take a long random walk $T_0, T_1, \ldots, T_k$. Let e_i be the expected value of $\gamma(T_i)$ where the expectation is over the random coin tosses of the random walk. We know that as i tends to ∞, $\Gamma(Q) = e_i$. We need to prove that $e_i \geq \frac{1}{n+1}\mathrm{OPT}$ as i tends to ∞. Let $p_{i,y}$ be the probability that $\gamma(T_i) = y$. Thus, $e_i = \sum_y p_{i,y}y$ and $e_{i+1} = \sum_y p_{i,y}\mathbf{E}[\gamma(T_{i+1})|\gamma(T_i) = y]$. The above inequality shows that $\mathbf{E}[\gamma(T_{i+1})|\gamma(T_i) = y] \geq \frac{1}{n}(\mathrm{OPT}-y)$. Therefore,

$$
\begin{aligned}
e_{i+1} &\geq \frac{1}{n}\sum_{y} p_{i,y}(\mathrm{OPT}-y) \\
&= \frac{1}{n}(\mathrm{OPT}-\sum_{y} p_{i,y}y) \\
&= \frac{1}{n}(\mathrm{OPT}-e_i).
\end{aligned}
$$

Hence, $e_{i+1} \geq \frac{1}{n}\mathrm{OPT} - \frac{e_i}{n}$. Since as i goes to ∞, $\Gamma(Q) = e_i = e_{i+1}$, we get $\Gamma(Q) \geq \frac{1}{n}\mathrm{OPT} - \frac{\Gamma(Q)}{n}$. Therefore, $\Gamma(Q) \geq \frac{1}{n+1}\mathrm{OPT}$ as desired. □

Thus the worst case price of sinking in a valid-utility game is between n and $n+1$.

4.4 A Hardness Result

In Section 2.6.3, we showed that finding a PSNE for some instances of the uniform CapDC game is PLS-hard (Theorem 2.6.6). Moreover, we proved that this result implies the existence of exponential best-response paths to equilibria in this game (Corollary 2.6.7). In Chapter 2, we proved that the CapIBDC games (and in particular the CapDC games) are special cases of valid-utility games (Theorem 2.6.2). These results from Chapter 2 imply the following corollary:

Corollary 4.4.1 *Finding a sink equilibrium (or a state in a sink equilibrium) is PLS-hard for some instances of valid-utility games. In addition, in some instances of valid-utility games, there exist states that are exponentially far from any sink equilibrium in the state graph.* □

Chapter 5

Cross-Monotonic Cost Sharing Methods and Group-Strategyproof Mechanisms

Consider a situation where a group of customers (which we call *agents*) wish to buy a service such as connectivity to a network. The total cost of this service is a function of the group of customers that are serviced: a group of customers in distant towns might incur a larger cost than a group of customers in the same town. The service provider must develop a pricing policy, or *cost-sharing scheme*, that, given any group of customers, divides the cost of the service amongst them. For example, one plausible cost-sharing scheme divides the cost of the service evenly amongst the customers. However, in the case of network connectivity, this scheme seems to undercharge distant customers with high connection costs and overcharge other customers. Developing a fair and economically viable cost-sharing scheme is a central problem in cooperative game theory. One commonly explored condition is that of *cross-monotonicity* [62, 63]. Intuitively, cross-monotonicity requires that the price charged to any individual in a group decreases as the group expands. Thus customers have an economic incentive to promote the service. In this chapter, we study this type of cost sharing schemes for different combinatorial optimization problems. Before stating the known results and the main contribution of this chapter, we state the formal definitions that are used in this chapter.

5.1 Preliminaries

Let $\mathscr{A}$ denote a set of n users who are interested in a service. The cost of providing service to a set $S \subseteq \mathscr{A}$ of users is denoted by $C(S)$. A *cost allocation* for a set $S \subseteq \mathscr{A}$ is a function $\psi : S \mapsto \mathbb{R}^+ \cup \{0\}$, that for each user $i \in S$, specifies the share $\psi(i)$ of i of the total cost of servicing S. A *cost-sharing scheme* is a collection of cost allocations for every $S \subseteq \mathscr{A}$. More formally, a cost sharing scheme is a function $\xi : \mathscr{A} \times 2^{\mathscr{A}} \mapsto \mathbb{R}^+ \cup \{0\}$, such that for every $S \subseteq \mathscr{A}$ and every $i \notin S$, $\xi(i, S) = 0$. Intuitively, we think of $\xi(i, S)$ as the share of i of the total cost if S is the set of agents receiving the service.

Ideally, we want cost sharing schemes (and cost allocations) to be *budget-balanced*, i.e., for every $S \subseteq \mathscr{A}$, $\sum_{i \in S} \xi(i, S) = C(S)$. However, it is not always possible to achieve budget balance in combination with other properties, or even if it is possible, it might be computationally hard to compute the cost shares. Therefore, we relax this notion to the notion of *α-budget balance* (for some $\alpha \leq 1$), which means that for every $S \subseteq \mathscr{A}$, $\alpha C(S) \leq \sum_{i \in S} \xi(i, S) \leq C(S)$.

In addition to budget balance, we usually require cost allocation and cost sharing schemes to satisfy additional properties. One property that is extensively studied in the cooperative game theory literature is the property of being in the *core* (see, for example, Bondareva [7] and Shapley [84]), which intuitively says that no subset of users should be overcharged for the service.

Definition 5.1.1 *A cost allocation ψ for a set $S \subseteq \mathscr{A}$ is in the α-*core *if and only if it is α-budget balanced and for every $T \subseteq S$, $\sum_{i \in T} \psi(i) \leq C(T)$. A cost sharing scheme ξ is in the α-core if and only if for every S, $\xi(., S)$ is in the α-core. We refer to* 1*-core as core.*

Another property, which was studied by Moulin [63] and Moulin and Shenker [62] in order to design group-strategyproof mechanisms (see the definition below), and has recently received considerable attention in the computer science literature (see, for example, [49, 46, 44, 69]), is *cross-monotonicity.* This property captures the notion that users should not be penalized as the serviced set grows. Namely,

Definition 5.1.2 *A cost sharing scheme ξ is* cross-monotone *if for all $S, T \subseteq \mathscr{A}$ and $i \in S$, $\xi(i, S) \geq \xi(i, S \cup T)$.*

It is a simple exercise to show that every α-budget-balanced cross-monotonic cost sharing scheme is in the α-core, but the converse need not hold. Therefore, cross-monotonicity is a strictly stronger requirement than being in the core.

The main application of cross-monotonic cost sharing schemes is in the design of cost sharing mechanisms, defined in the following setting: Each user i has a *willingness to pay* or a *true bid* $\overline{b}_i \in \mathbb{R}^+ \cup \{0\}$ for the service, i.e., she is willing to pay up to $\overline{b}_i$ dollars to get the service. We further assume that the happiness of user i is given by $\overline{b}_i q_i - x_i$, where q_i is an indicator variable which indicates whether she has received the service or not, and x_i is the amount she has to pay[1] . A *cost sharing mechanism* (also known as a *social choice function*) is an algorithm that elicits a bid $b_i \in \mathbb{R}^+ \cup \{0\}$ from each agent, and based on these bids, decides which agents should receive the service and how much each of them has to pay. More formally, a cost sharing mechanism is a function that associates to each vector b of non-negative bids a set $Q(b) \subseteq \mathscr{A}$ of agents to be serviced, and a vector $x(b) \in \mathbb{R}^n$ of non-negative payments. When there is no ambiguity, we write Q and x instead of $Q(b)$ and $x(b)$, respectively. Throughout this chapter, we assume that a mechanism does not charge an agent who does not receive the service (i.e., $x_i = 0$ for $i \notin Q$), does not charge an agent who receives the service more than her bid (i.e., $x_i \leq b_i$ for $i \in Q$), and for each agent i, there is some bid ∞_i such that if i bids ∞_i, she will get the service, no matter what others bid[2]. Furthermore, we would like the mechanisms to be approximately budget balanced. We call a mechanism α-budget balanced if the total amount the mechanism charges the agents is between $\alpha C(Q)$ and $C(Q)$ (i.e., $\alpha C(Q) \leq \sum_{i \in Q} x_i \leq C(Q)$).

[1] As noted by Moulin and Shenker [62], this assumption is without loss of generality

[2] For a discussion about these properties see Moulin [63] and Moulin and Shenker [62].

The main property that we want a mechanism to satisfy is incentive compatibility. We want our mechanism to encourage participants to submit their true willingness to pay as their bid. Agents should not be able to benefit from lying about the prices they are willing to pay. Ideally, not even a group of users should be able to benefit by cooperatively lying, thus discouraging complicated bidding strategies. More precisely, we look for mechanisms, called *group-strategyproof mechanisms* which satisfy the following additional property. Let $S \subseteq \mathscr{A}$ be a coalition of users, and $\overline{b}, b$ be two vectors of non-negative bids satisfying $\overline{b}_i = b_i$ for every $i \notin S$ (we think of $\overline{b}$ as the true willingness to pay of users, and b as a vector of strategically chosen bids). Let (Q, x) and (Q', x') denote the outputs of the mechanism when the bids are $\overline{b}$ and b, respectively. We say that the mechanism is *group strategyproof* if for every such $S, \overline{b}, b$, if the inequality $\overline{b}_i q'_i - x'_i \geq \overline{b}_i q_i - x_i$ holds for all $i \in S$, then it holds with equality for every $i \in S$.

We call the vector $(\bar{b}_1, \ldots, \bar{b}_n)$ of true bids *a scenario*. A coalition S with bid vector b of players is a *lying coalition* for a scenario $\bar{b}$ if when members of S announce b instead of $\overline{b}$ (their true willingness to pay) as their bids, every member of the coalition S is at least as happy as in the truthful scenario in which S announce their true bid $\bar{b}$, and at least one person is happier. Notice that we do not allow members of the coalition to sacrifice their own happiness to benefit the group's total happiness. For a group-strategyproof mechanism, any coalition of players with any bid vector is not a lying coalition for any scenario.

For the important class of services with submodular cost functions [3], various cross-monotonic cost-sharing schemes were studied by Moulin and Shenker [62] and further by Jain and Vazirani [46]. For submodular cost functions, there are cross-monotonic cost-sharing schemes that are budget-balanced. There are many other interesting classes of cost functions that arise from NP-hard optimization problems. For example, the cost of providing the service for a set of agents S could be expressed as the cost of building the cheapest rooted Steiner tree that covers a given root vertex r and all the elements of S, or the minimum cost of opening facilities and

[3] In a service with submodular cost function, $C(S)$ is a submodular set function.

connecting each member of S to an open facility[4]. These two games, and many others of practical importance, are instances of covering problems. For such problems, it is usually impossible for a cross-monotonic cost sharing scheme to be budget-balanced. Moreover, even if a budget-balanced cross-monotonic cost sharing scheme exists, it might be hard to compute. Therefore, it is natural to consider cost sharing schemes that are approximately budget balanced. Such schemes have been studied by Kent and Skorin-Kapov [49], Feigenbaum et al. [20], Jain and Vazirani [44], and Pal and Tardos [69].

Among the covering combinatorial optimization problems, a fundamental problem is the (fractional) set cover problem. Given a family of subsets $\mathcal{A} \subseteq 2^V$, a family $A \subseteq \mathcal{A}$ of subsets is a set cover for a set S, if any element $v \in S$ is contained in at least one set $T \in A$ ($v \in T$). The minimum set cover of a set S is a set cover of S with minimum number of sets. Given a family of subsets $\mathcal{F} \subseteq 2^V$, a fractional set cover for a subset $S \subseteq V$ is a collection of fractional numbers α_T for each $T \in \mathcal{F}$, such that (i) $0 \le \alpha_T \le 1$, and (ii) for each element $v \in S$, $\sum_{T \in \mathcal{F}: v \in T} \alpha_T \ge 1$. The minimum fractional set cover for a set S is a fractional set cover in which $\sum_{T \in \mathcal{F}} \alpha_T$ is minimized. We refer to $\sum_{T \in \mathcal{F}} \alpha_T$ as the cost of the fractional set cover. The corresponding set cover and fractional set cover games are defined as follow:

Definition 5.1.3 *Let $\mathcal{F} \subseteq 2^V$ be a collection of subsets of the universe V. The set of agents in the set cover game defined on $\mathcal{F}$ is the set of elements of the universe V. Given a subset $S \subseteq V$ of the agents , the cost of S is the minimum size of a set cover for S. In the fractional set cover game, the cost of a subset $S \subseteq V$ is the cost of the minimum fractional set cover for S.*

It is easy to show that if there is an α-budget balanced cross-monotonic cost-sharing scheme for the fractional set cover game, then for any special case of the set cover problem of integrality gap at most μ, there is an $\alpha\mu$-budget balanced cross-monotonic cost-sharing scheme. For example, if we could get a constant-factor for fractional set cover game, it would have implied a constant-factor for metric facility

[4] See the formal definition of the facility location game in Section5.2.3.

location and generalized Steiner tree games. Unfortunately, our result shows that no cross-monotonic cost-sharing scheme for fractional set cover game with a reasonable budget-balance factor exists, and thus this approach for designing cross-monotonic cost-sharing schemes fails to recover much of the cost. This raises the natural question of whether it is possible to design well budget-balanced schemes for these combinatorial optimization games.

We can derive simple bounds on the budget-balance factor of combinatorial optimization games using the integrality gaps of the "natural" LP-relaxations. The cross-monotonicity of a cost sharing scheme implies that for every set of agents the cost shares form an allocation in the core of the game. Therefore, the best budget-balance factor achievable by a cross-monotonic cost sharing scheme cannot be better than that of a cost sharing in the core. This, together with the folklore theorem that the best budget-balance factor for a cost sharing in the core of integer covering games is equal to the integrality gap of the "natural" LP-relaxation of the problem, gives us upper bounds for the best cross-monotonic cost sharing scheme for various combinatorial optimization problems. For example, this argument implies that cross-monotonic cost sharing schemes for metric facility location, vertex cover, and set cover games cannot recover more than a $\frac{1}{1.463}$, $\frac{1}{2}$, and $\frac{1}{\ln n}$ fraction of the total cost, respectively. Prior to this work, this was the only method known for upper bounding the cross-monotonic cost sharing schemes. In this chapter, we show stronger upper bounds for several combinatorial optimization problems using a novel technique based on the probabilistic method that will be explained in Section 5.2.1. In particular, we prove that the best budget-balance factor achievable for the facility location game is 1/3. This matches a lower bound recently given by Pal and Tardos [69]. Also, for the vertex cover and set cover games, we show that no cross-monotonic cost sharing scheme can recover more than an $O(n^{-1/3})$ and $O(\frac{1}{n})$ fraction of the total cost, respectively. Previously, Devanur et al. [13] give a strategyproof $\Omega(1/\log(n))$-budget balanced cost-sharing mechanism in the core for the set cover game, but their underlying cost-sharing scheme is not cross-monotonic. We also apply this technique to several other cost or profit sharing problems including edge cover, maximum flow, maximum matching, and arborescence packing.

Cross-monotonic cost sharing schemes are mainly used to obtain group-strategyproof mechanisms using a method developed by Moulin and Shenker [63, 62]. In fact, almost all known group-strategyproof mechanisms are Moulin-Shenker mechanisms. Therefore, one might hope that our negative results on cross-monotonic cost sharing schemes might imply similar negative results for group-strategyproof mechanisms. However, we observe that for almost any problem there are trivial group-strategyproof mechanisms that recover all the cost. These mechanisms completely ignore the structure of the problem and can therefore be unfair and inefficient. This suggests that new conditions should be added to the definition of group-strategyproofness to exclude such mechanisms. We study a few such conditions, and prove that with the extra assumptions of *no free riders* and *upper continuity*[5], group-strategyproof mechanisms give rise to cross-monotonic cost sharing schemes, and hence our upper bounds hold for group-strategyproof mechanisms with these extra assumptions. We also consider subsidy-freeness [61], which is a stronger fairness condition and prove the equivalence of budget-balanced cross-monotone cost sharing schemes and budget-balanced group-strategyproof mechanisms with this property.

The rest of this chapter is organized as follows. Section 5.2 contains a description of our upper bound techniques, our upper bounds for the covering game and the facility location game, and the statement of some other results that we have been able to prove using this technique. In Section 5.3 we present several trivial group-strategyproof mechanisms and study some of the axioms that can be added to the definition of group-strategyproof mechanisms to eliminate such trivial mechanisms.

5.2 Upper bounds for cross-monotonic cost sharing schemes

In this section we present the main idea behind our upper bound technique and prove several upper bounds for the games defined based on edge cover, vertex cover, and

[5] See the definitions in Section 5.3.

facility location. In Section 5.2.1 we explain the technique with a simple example of the edge cover game. Sections 5.2.2 and 5.2.3 contain the proofs of the upper bounds for the vertex cover and facility location games. Finally, in Section 5.2.4 we state (without proof) several other upper bounds that can be proved using our technique.

5.2.1 A simple example: the edge cover game

In this section, we explain our technique using the edge cover game as a guiding example. The edge cover cost function is defined as follows.

Definition 5.2.1 *Let $G(V, E)$ be a graph with no isolated vertices. The set of agents in the edge cover game on G is the set of vertices of G. Given a subset S of vertices, the cost of S is the minimum size of a set $F \subseteq E$ of edges such that for every $v \in S$, at least one of the edges incident to v is in F. Such a set F is called an* edge cover *for S.*

It is easy to see that for every set S, one can obtain a minimum edge cover of S by taking a maximum matching on S and adding one edge for every vertex that is not covered by the maximum matching (see [12]). Using this fact, we can give a cost-sharing scheme that is in the $\frac{2}{3}$-core of the game: charge each vertex that is covered by the maximum matching $\frac{1}{3}$, and other vertices $\frac{2}{3}$. Let α_v be the amount that vertex v is charged. Thus, for any two vertices u and v of G, if $\alpha_v = \alpha_u = \frac{2}{3}$, then there is no edge between u and v. Consider any subset S of vertices. Consider an optimal edge cover $T \subseteq E(G)$ of S. For any edge $\{u, v\} \in T$, we know that $\alpha_v + \alpha_u \leq 1$, since $\alpha_v \neq \frac{2}{3}$ or $\alpha_u \neq \frac{2}{3}$. Therefore, $|T| \geq \sum_{v \in S} \alpha_v$. Thus, this cost-sharing scheme satisfies the core property. Furthermore, it is easy to see that the sum of the cost shares is at least $\frac{2}{3}$ times the edge cover for S. Therefore, there is a cost-sharing scheme satisfying the core property with a budget-balance factor of $\frac{2}{3}$. In fact, Goemans [28] showed that for every graph there is a cost sharing scheme in the $\frac{3}{4}$-core. However, in the following, we show that no cross-monotonic cost-sharing scheme can achieve a budget-balance factor better than $\frac{1}{2}$.

Theorem 5.2.1 *For every $\epsilon > 0$, there is no $(\frac{1}{2}+\epsilon)$-budget balanced cross-monotonic cost sharing scheme for the edge cover problem.*

Here's the high-level idea of the proof: We assume, for contradiction, that there is a cross-monotonic cost sharing scheme that always recovers at least a $(\frac{1}{2}+\epsilon)$ fraction of the total cost. We explicitly construct a graph G (or in general the set of agents $\mathscr{A}$ and the structure based on which the cost function is defined), and look at the cost-sharing scheme on this graph. For edge cover, this graph is simply a complete bipartite graph $K_{n,n}$, with n large enough. Then, we need to argue that there is a set S of agents such that the total cost shares of the elements of S is less than $\frac{1}{2}+\epsilon$ times the size of the minimum edge cover for S. This is done using the probabilistic method: we pick a subset S at random from a certain distribution and show that in expectation, the ratio of the recovered cost to the cost of S is low. Therefore, there is a manifestation of S for which this ratio is low. In the edge cover example, we pick one vertex v of G uniformly at random and let S be the union of v and the set of vertices adjacent to v. We now need to bound the expected value of the sum of cost shares of the elements of S. We do this by using cross-monotonicity and bounding the cost share of each vertex $u \in S$ by the cost share of u in a substructure T_u of S. Bounding the expected cost share of u in T_u is done by showing that for every substructure T, every $u \in T$ has the same probability of occurring in a structure S in which $T_u = T$. This implies that the expected cost share of u in T_u (where the expectation is over the choice of S) is at most the cost of T_u divided by the number of agents in T_u. Summing up these values for all u gives us the desired contradiction.

Proof of Theorem 5.2.1. Assume that there is a $(\frac{1}{2}+\epsilon)$-budget-balanced cross-monotonic cost sharing scheme ξ. Let G be the complete bipartite graph $K_{n,n}$, where n will be fixed later, and consider ξ on G. For every $v \in V(G)$, we let S_v be the union of v and the set of vertices adjacent to v (i.e., vertices of the other part). We pick a set S of agents by picking v uniformly at random from $V(G)$ and letting $S = S_v$. By the definition of the edge cover problem,

$$C(S) = n \qquad \text{for every } S. \tag{5.1}$$

On the other hand,

$$\begin{aligned} \mathrm{E}_S\Big[\sum_{i\in S}\xi(i,S)\Big] &= \mathrm{E}_v\left[\xi(v,S_v)\right] + \mathrm{E}_v\Big[\sum_{u\in S_v\setminus\{v\}}\xi(u,S_v)\Big] \\ &\leq 1 + \mathrm{E}_v\Big[\sum_{u\in S_v\setminus\{v\}}\xi(u,\{u,v\})\Big], \end{aligned}$$

where the last inequality follows from the facts that for every vertex u and every set S, $\xi(u,S) \leq 1$, and that for every $v \in V(G)$ and $u \in S_v \setminus \{v\}$, $\xi(u,S_v) \leq \xi(u,\{u,v\})$. Both of these facts are consequences of the cross-monotonicity of ξ. By the definition of expected values, we have

$$\mathrm{E}_v\Big[\sum_{u\in S_v\setminus\{v\}}\xi(u,\{u,v\})\Big] = n\,\mathrm{E}_{v,u}\left[\xi(u,\{u,v\})\right], \tag{5.2}$$

where the second expectation is over the choice of v from $V(G)$ and u in $S_v \setminus \{v\}$. However, choosing a vertex v and then a neighbor u of v at random is equivalent to choosing a random edge e in G at random, and letting u be a random endpoint of e and v be the other one. By the budget-balance condition, the sum of the cost shares of the endpoints of e is at most one. Therefore, for every e, if u is a random endpoint of e and v is the other endpoint, $\mathrm{E}[\xi(u,\{u,v\})] \leq \frac{1}{2}$. Thus, the right-hand side of Equation 5.2 is at most $\frac{n}{2}$. Therefore, by Equations 5.1 and 5.2, we have

$$\mathrm{E}_S\left[\frac{\sum_{i\in S}\xi(i,S)}{C(S)}\right] \leq \frac{1+\frac{n}{2}}{n} < \frac{1}{2} + \epsilon$$

for $n > 1/\epsilon$. Therefore, there is a set S satisfying $\frac{\sum_{i\in S}\xi(i,S)}{C(S)} < \frac{1}{2} + \epsilon$, which is a contradiction with the assumption that ξ is $(\frac{1}{2}+\epsilon)$-budget balanced. □

It is not difficult to see that for any instance of the edger cover game, the cost-sharing scheme ξ satisfying $\xi(i,S) = \frac{1}{2}$ for every $i \in S$ is cross-monotonic and $\frac{1}{2}$-budget balanced. Therefore, the bound given in the above theorem is tight. Also, one can think of the edge cover problem as a special case of the set cover problem in which the size of each set is 2. It is not difficult to generalize the above result (using k-partite k-uniform complete hypergraphs) to the special case of set cover in which the size of each set is k, and prove that for k constant, no cross-monotonic cost-sharing scheme for this problem can recover more than a $\frac{1}{k}$ fraction of the cost. Similar argument shows that for the general case of the set cover game, no cross-monotonic cost-sharing scheme can recover more than a $O(\frac{1}{n})$ of the total cost. This result is also tight up to a constant multiple.

Theorem 5.2.2 *There is no cross-monotonic cost-sharing scheme ξ for the set cover game such that for every set $S \subseteq \mathscr{A}$, ξ recovers more than a $\frac{4}{|S|}$ fraction of the cost of S.*

Proof. Assume that there is such a cross-monotonic cost sharing scheme ξ. Consider the following instance of the set cover problem. Let $\mathscr{A}$ be a set of n^2 agents that can be partitioned as $\mathscr{A} = A_1 \cup A_2 \cup \cdots \cup A_n$, where A_i's are disjoint sets each of size n. Define $\mathcal{F}$ as the collection of all sets $S \subset \mathscr{A}$ such that $|S \cap A_i| = 1$ for every $i = 1, \ldots, n$. An alternative way to look at this is that $\mathscr{A}$ and $\mathcal{F}$ are sets of vertices and edges of an n-uniform n-partite complete hypergraph.

We pick a random set S of agents in the above instance as follows: Pick a random i from $\{1, \ldots, n\}$, and for every $j \neq i$, pick an agent a_j uniformly at random from A_j. Let $T = \{a_j : j \neq i\}$ and $S = A_i \cup T$. The cost of the optimal set cover solution on S is always at least n, since no set in $\mathcal{F}$ contains two distinct elements of A_i, and therefore each element of A_i must be covered with a distinct set in $\mathcal{F}$.

We now bound the average recovered cost over the random choice of S.

$$\begin{aligned}
\mathrm{E}_S\Big[\sum_{x \in S} \xi(x, S)\Big] &= \mathrm{E}\Big[\sum_{x \in A_i} \xi(x, S)\Big] + \mathrm{E}\Big[\sum_{j \neq i} \xi(a_j, S)\Big] \\
&\leq \mathrm{E}\Big[\sum_{x \in A_i} \xi(x, \{x\} \cup T)\Big] + \mathrm{E}\Big[\sum_{j \neq i} \xi(a_j, T)\Big]
\end{aligned}$$

Since all elements of T can be covered by one set, the second term in the above expression is at most 1. We write the first term as $n\mathrm{E}_{S,x}\left[\xi(x, \{x\} \cup T)\right]$ where the expectation is over the random choice of S and the random choice of x from A_i. As in the proof of Theorem 5.2.6, the expected value of $\xi(x, \{x\} \cup T)$ in this experiment is equal to the expected value of $\frac{1}{n}\sum_{j=1}^{n} \xi(a_j, \{a_1, \ldots, a_n\})$ in an experiment that consists of choosing an agent a_j from each A_j uniformly at random. By the budget-balance property, we always have $\sum_{j=1}^{n} \xi(a_j, \{a_1, \ldots, a_n\}) \leq C(\{a_1, \ldots, a_n\}) = 1$. Therefore, the first term in the right-hand side of the inequality (5.3) is at most one. This means that the expected total cost share recovered from the set S is at most two. Therefore, the ratio of recovered cost to total cost of S is at most $2/n < 4/|S|$.
$\square$

5.2.2 The vertex cover game

The vertex cover game is defined on a graph $G(V, E)$. The set of agents is the set of edges of G, and the cost of serving a set $S \subseteq E$ is equal to the minimum size of a set A of vertices such that for each $e \in S$, at least one of the endpoints of e is in A. Such a set is called a *vertex cover* for the set S. It is well-known that the integrality gap of the LP relaxation of vertex cover is 2, and therefore no allocation in the core can recover more than half the cost of the solution in the worst case. We show in the following theorem that if we require the cost-sharing scheme to be cross-monotonic then no constant-factor budget balanced scheme exists.

Theorem 5.2.3 *For every $\epsilon > 0$, there is no cross-monotonic cost sharing scheme for vertex cover that on every set S of n agents, recovers at least a $(2+\epsilon)n^{-1/3}$ fraction of the cost of S.*

Proof. Assume, for contradiction, that such a scheme ξ exists. We let G be a complete graph on $m + 2\ell$ vertices, where m and ℓ ($m < \ell$) are numbers that will be fixed later, and consider the cost-sharing scheme ξ on G. We show that there is some set S of edges of G for which ξ recovers at most a $|S|^{-1/3}$ fraction of the cost. We do this by picking S randomly from a distribution described below, and showing that

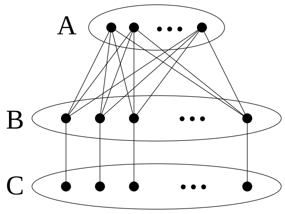


Figure 5-1: The structure S in the vertex cover game

the above statement holds in expectation, and therefore there should be a particular S satisfying the above statement.

Let π be a permutation of the $m + 2\ell$ vertices. Let A be the set of the first m vertices, B be the set of the next ℓ vertices, and C be the set of the remaining ℓ vertices. We denote the i'th vertices of B and C (based on the ordering given by π) by b_i and c_i. Let S_π denote the set of all $m\ell$ edges between A and B, union the set of edges b_ic_i for $i = 1, \ldots, \ell$. We pick S by picking the permutation π uniformly at random and letting $S = S_\pi$. See Figure 5-1 for an example.

If we denote the set of edges between A and B by T, we have

$$\mathrm{E}\Big[\sum_{e\in T}\xi(e,S)\Big] \leq \mathrm{E}\Big[\sum_{e\in T}\xi(e,T)\Big] \leq m, \tag{5.3}$$

where the first inequality follows from the cross-monotonicity of ξ and the second inequality is implied by the budget balance assumption and the fact that the cost of the minimum vertex cover in T is m. We also let T_i be the set of all $m+1$ edges in S that have b_i as an endpoint (see Figure 5-1). Equation 5.3 and the cross-monotonicity of ξ imply the following.

$$\begin{aligned}\mathrm{E}_S\Big[\sum_{i\in S}\xi(i,S)\Big] &= \mathrm{E}\Big[\sum_{e\in T}\xi(e,S)\Big] + \sum_{i=1}^{\ell}\mathrm{E}\left[\xi(b_ic_i,S)\right] \\ &\leq m + \sum_{i=1}^{\ell}\mathrm{E}\big[\xi(b_ic_i,T_i)\big]. \end{aligned} \tag{5.4}$$

We now need to analyze the expectation of $\xi(b_ic_i, T_i)$ over the random choice of π. Notice that the only elements of π that are important in $\xi(b_ic_i, T_i)$ are the first

m elements and the $m+i$'th and $m+\ell+i$'th elements (b_i and c_i). Therefore, the expectation of $\xi(b_i c_i, T_i)$ over the choice of π is equal to the expectation of $\xi(v_{m+2}v_{m+1}, \{v_1 v_{m+1}, v_2 v_{m+1}, \ldots, v_m v_{m+1}, v_{m+2} v_{m+1}\})$ over the random choice of an ordered list $v_1, v_2, \ldots, v_{m+2}$ of $m+2$ different vertices of G. However, in this experiment it is clear by symmetry that the expected cost share of $v_i v_{m+1}$ is the same for $i = 1, \ldots, m, m+2$, and therefore by the budget balance condition each of these expected cost shares is at most $\frac{1}{m+1}$. This, together with Equation 5.4 imply the following.

$$\mathrm{E}_S\left[\sum_{i\in S}\xi(i,S)\right] \le m + \frac{\ell}{m+1}. \tag{5.5}$$

On the other hand, the size of the minimum vertex cover in S is always ℓ. Therefore, the expected value of the ratio of $\sum_{i\in S}\xi(i,S)$ to $C(S)$ is at most $\frac{m}{\ell} + \frac{1}{m+1}$. Thus, there is a set S for which this ratio is at most $\frac{m}{\ell} + \frac{1}{m+1}$. Taking $m = \sqrt{\ell}$, we see that the allocation on S recovers at most a $\frac{2}{\sqrt{\ell}} < (2+\epsilon)|S|^{-1/3}$ fraction of the cost. $\square$

We can show the following positive result for cross-monotonic cost sharing schemes for the vertex cover. We do not know the right bound for the budget-balance factor of the vertex cover game.

Theorem 5.2.4 *For the vertex cover game, the cost sharing scheme that charges the edge uv in the set S an amount equal to* $\min(1/\deg_S(u), 1/\deg_S(v))$ *is cross-monotonic and $\frac{1}{2\sqrt{n}}$-budget balanced.*

Proof. It is clear that this scheme is cross-monotone. We only need to verify the budget-balance factor. Consider a set S of n agents (i.e., edges), and the graph $G[S]$ induced on this set of edges. First, we show that the total cost share of agents in S is at most the cost of vertex cover for S. Let $T \subseteq S$ be an optimal vertex cover of $G[S]$. For an edge $\{u,v\} \in E(G[S])$, $u \in T$ or $v \in T$ or both of u and v are in T. Let β_{uv} be $\frac{1}{\deg_S(v)+\deg_S(u)}$ if u and v are both in T; otherwise let β_{uv} be $\frac{1}{\deg_S(u)}$ or $\frac{1}{\deg_S(v)}$ if

$u \in T$ or $v \in T$. Then

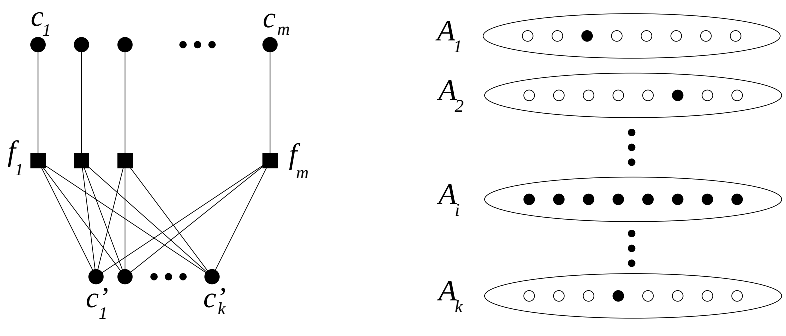
$$\begin{aligned}\sum_{\{u,v\}\in E(G[S])} \min(\frac{1}{\deg_S(u)}, \frac{1}{\deg_S(v)}) &\leq \sum_{\{u,v\}\in E(G[S])} \beta_{uv} \\ &= |T|.\end{aligned}$$

This shows that the sum of cost shares of edges in $G[S]$ is at most the cost of the minimum vertex cover of S.

Now, we prove that the total cost share of the agents in S is at least $\frac{1}{2\sqrt{n}}$ times the cost of a vertex cover for S. Divide the set of vertices into two subsets L and H, where L is the set vertices of degree less than $\sqrt{n}$ in $G[S]$ and H is the rest of vertices ($H = V(G) - L$). As a vertex cover solution, select H and both endpoints of all edges (u, v) such that $u, v \in L$. We show that the cost shares of the edges in S sum to at least a $\frac{1}{2\sqrt{n}}$ fraction of the cost of this solution. First consider any edge e between vertices in L. The cost share of e is at least $\frac{1}{\sqrt{n}}$, thus its cost share covers the cost of picking both its endpoints. Now consider the vertices in H. Since the degree of each vertex $v \in H$ is greater than or equal to $\sqrt{n}$, the sum of the cost shares of the edges adjacent to v is at least $\frac{1}{n}\sqrt{n} = \frac{1}{\sqrt{n}}$. Each edge is included in at most two such summations, and thus the sum of the cost shares of edges adjacent to vertices in H is at least a $\frac{1}{2\sqrt{n}}$ fraction of the cost of H. Therefore, the sum of the cost shares of the agents in S is at least $\frac{1}{2\sqrt{n}}$ times the cost of the optimal vertex cover for S. □

5.2.3 The metric facility location game

Given a set of cities, facilities with opening costs, and metric connection costs between cities and facilities, the facility location problem seeks to open a subset of facilities and connect each city to a facility in a manner that minimizes the total cost. In the facility location game, each city is an agent. The cost of a subset of agents is the cost of the minimum facility location solution for that subset; a cross-monotonic cost-sharing scheme tries to share this cost among the agents. In this section, we prove that any cross-monotonic cost-sharing scheme for facility location is at best



(a) (b)

Figure 5-2: Upper bound for facility location game

$\frac{1}{3}$-budget-balanced. This matches the budget-balance factor of the scheme given by Pal and Tardos [69].

We start by giving an example on which the scheme of Pal and Tardos [69] recovers only a third of the cost[6]. This example will be used as the randomly chosen structure in our proof.

Lemma 5.2.5 *Let $\mathcal{I}$ be an instance of the facility location problem consisting of $m + k$ cities $c_1, \ldots, c_m$, $c'_1, \ldots, c'_k$ and m facilities $f_1, \ldots, f_m$ each of opening cost 3. For every i and j, the connection costs between f_i and c_i and between f_i and c'_j are all 1, and other connection costs are obtained by the triangle inequality. See Figure 5-2(a). Then if $m = \omega(k)$ and k tends to infinity, the optimal solution for $\mathcal{I}$ has cost $3m + o(m)$.*

Proof. Let p be the number of opened facilities in the optimal solution. Then the facility opening cost of these facilities is $3p$, and at least $m - p$ cities among $c_1, \ldots, c_m$ should pay at least 3 for the connection cost. Thus, the total cost is $3p + 3(m - p) + p + k = 3m + k + p$. Since $p \geq 1$, the cost of the optimal solution is $3m + k + 1$ which is $3m + o(m)$ as $m = \omega(k)$. □

[6]This example also shows that the dual computed by the Jain-Vazirani facility location algorithm [45] can be a factor 3 away from the optimal dual.

Theorem 5.2.6 *Any cross-monotonic cost-sharing scheme for the facility location game is at most* $1/3$*-budget balanced.*

Proof. Consider the following instance of the facility location problem. There are k sets $A_1, \ldots, A_k$ of m cities each, where $m = \omega(k)$ and $k = \omega(1)$. For every subset B of cities containing exactly one city from each A_i ($|B \cap A_i| = 1$ for all i), there is a facility f_B with connection cost 1 to each city in B. The remaining connection costs are defined by extending the metric, i.e., the cost of connecting city i to facility f_B for $i \notin B$ is 3. The facility opening costs are all 3.

We pick a random set S of cities in the above instance as follows: Pick a random i from $\{1, \ldots, k\}$, and for every $j \neq i$, pick a city a_j uniformly at random from A_j. Let $T = \{a_j : j \neq i\}$ and $S = A_i \cup T$. See Figure 5-2(b) for an example. It is easy to see that the set S induces an instance of the facility location problem almost identical to the instance $\mathcal{I}$ in Lemma 5.2.5 (the only difference is that here we have more facilities, but it is easy to see that the only relevant facilities are the ones that are present in $\mathcal{I}$). Therefore, the cost of the optimal solution on S is $3m + o(m)$.

We show that for any cross-monotonic cost-sharing scheme ξ, the average recovered cost over the choice of S is at most $m + o(m)$ and thus conclude that there is some S whose recovered cost is $m + o(m)$. As in the previous proofs, we start bounding the expected total cost share by using the linearity of expectations and cross-monotonicity:

$$\begin{aligned}
\mathrm{E}_S\Big[\sum_{c\in S}\xi(c,S)\Big] &= \mathrm{E}\Big[\sum_{c\in A_i}\xi(c,S)\Big] + \mathrm{E}\Big[\sum_{j\neq i}\xi(a_j,S)\Big] \\
&\leq \mathrm{E}\Big[\sum_{c\in A_i}\xi(c,\{c\}\cup T)\Big] + \mathrm{E}\Big[\sum_{j\neq i}\xi(a_j,T)\Big].
\end{aligned}$$

Notice the set T has a facility location solution of cost $3 + k - 1$ and thus by the budget balance condition the second term in the above expression is at most $k + 2$. The first term in the above expression can be written as $m\mathrm{E}_{S,c}\left[\xi(c, \{c\} \cup T)\right]$ where the expectation is over the random choice of S and the random choice of c from A_i. However, it can be seen easily that this is equivalent to the following random experiment: From each A_j, pick a city a_j uniformly at random. Then pick i from $\{1,\ldots,k\}$ uniformly at random and let $c = a_i$ and $T = \{a_j : j \neq i\}$. From this description it is clear that the expected value of $\xi(c, \{c\} \cup T)$ is equal to $\frac{1}{k}\sum_{j=1}^{k} \xi(a_j, \{a_1,\ldots,a_k\})$. This, by the budget balance property and the fact that $\{a_1,\ldots,a_k\}$ has a solution of cost $k+3$ cannot be more than $\frac{k+3}{k}$. Therefore,

$$\mathrm{E}_S\Big[\sum_{c\in S}\xi(c,S)\Big] \leq m(\frac{k+3}{k}) + (k+2) = m + o(m), \tag{5.6}$$

when $m = \omega(k)$ and $k = \omega(1)$. Therefore, the expected value of the ratio of recovered cost to total cost tends to 1/3. □

5.2.4 Other combinatorial optimization problems

In this section we state upper bounds for three other combinatorial optimization games (in particular, the ones considered by Deng et al. [12]). These problems are maximization problems, therefore instead of cost sharing, we need to design *profit sharing* schemes. Definitions of profit sharing schemes and their properties are similar to the ones for cost sharing schemes (with $\alpha \geq 1$ and the direction of all inequalities reversed).

The first example is the maximum flow game. In the maximum flow game, we are given a directed graph $G(V, E)$ with a source s and a sink t. Agents are directed edges of G. Given a subset of edges, S, the profit of S is the size of maximum flow from s to t on subgraph of G induced on the edges of S. It is known that the core of maximum flow game is nonempty [12]. The story is different for cross-monotonic profit sharing schemes.

Theorem 5.2.7 *There is no $o(n)$-budget balanced profit sharing scheme for the maximum flow game where n is the number of agents.*

Proof. Let G be a graph consisting of three nodes: s, u, and t. There are $n-1$ edges from s to u, and $n-1$ edges from u to t. Let E_{su} and E_{ut} denote the set of edges from s to u and from u to t, respectively. See Figure 5-3. We pick a random set S

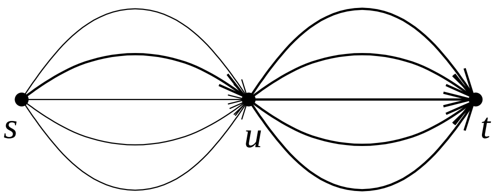


Figure 5-3: The graph G for the maximum flow game

of n agents as follows: With probability $1/2$, pick a random edge e from s to u, and let $S = \{e\} \cup E_{ut}$. With probability $1/2$, pick a random edge e from u to t, and let $S = \{e\} \cup E_{su}$. For example the set S could contain the thick edges in Figure 5-3.

Assume ξ is an $o(n)$-budget balanced cross-monotonic profit-sharing scheme for G. If edge e is picked uniformly at random from a set T, we write $e \stackrel{R}{\leftarrow} T$. We have that $E \equiv \mathrm{E}_S[\sum_{a \in S} \xi(a, S)]$ is

$$
\begin{aligned}
E \geq\ & \frac{1}{2}\mathrm{E}_{e \stackrel{R}{\leftarrow} E_{su}}\Big[\sum_{a \in E_{ut}} \xi(a, \{e\} \cup E_{ut})\Big] \\
& + \frac{1}{2}\mathrm{E}_{e \stackrel{R}{\leftarrow} E_{ut}}\Big[\sum_{a \in E_{su}} \xi(a, \{e\} \cup E_{su})\Big] \\
\geq\ & \frac{1}{2}\mathrm{E}_{e \stackrel{R}{\leftarrow} E_{su}}\Big[\sum_{a \in E_{ut}} \xi(a, \{a, e\})\Big] \\
& + \frac{1}{2}\mathrm{E}_{e \stackrel{R}{\leftarrow} E_{ut}}\Big[\sum_{a \in E_{su}} \xi(a, \{a, e\})\Big] \\
=\ & \frac{n-1}{2}\mathrm{E}_{a \stackrel{R}{\leftarrow} E_{su}, b \stackrel{R}{\leftarrow} E_{ut}}\Big[\xi(a, \{a, b\}) + \xi(b, \{a, b\})\Big] \\
\geq\ & \frac{n-1}{2}.
\end{aligned}
$$

On the the hand, the profit of every set S picked using the above procedure is one. Therefore, the expected ratio of the total profit shares to the profit of S is at least $(n-1)/2$. □

The second problem is the problem of packing the maximum number of arborescences in a digraph. An r-arborescence is a spanning tree rooted at r in which all edges are directed away from r. In the maximum r-arborescence game, we are given a directed graph $G(V, E)$ with a root r. Agents are directed edges of G. Given a subset of edges, S, the value of S is the maximum number of edge-disjoint r-arborescences on the subgraph induced by S. One can think of the profit of S as the maximum bandwidth for broadcasting messages from r to all vertices of the graph. It is known that the core of this game is nonempty [12].

Theorem 5.2.8 *There is no $o(n)$-budget balanced profit sharing scheme for maximum r-arborescence game.*

The same construction as the one used in the proof of Theorem 5.2.7 gives us a proof for Theorem 5.2.8.

Finally, we consider the maximum matching game, in which the agents are vertices of a graph G, and the profit of a subset of vertices S is the size of maximum matching in $G[S]$. One can show that there is a 2-budget balance profit allocation in the core of this game.

Theorem 5.2.9 *There is no $o(n)$-budget balanced profit sharing scheme for the maximum matching game.*

Proof. We use the same construction that was used in the proof of Theorem 5.2.1. Let G be a complete bipartite graph with $n-1$ vertices in each part (here we use $n-1$ instead of n so that the size of S becomes n), and pick S by picking a random vertex in G and all vertices in the other part. Using an argument essentially the same as the one used in the proof of Theorem 5.2.1, the expected total profit share of the elements of S is at least $(n-1)/2$. On the other hand, the profit of S is always one. Thus, there is an S on which the ratio between the total profit-share and the profit of S is at least $(n-1)/2$. $\square$

5.3 Group-strategyproof mechanisms

A main motivation behind cross-monotonic cost-sharing schemes is that they can be used to define group-strategyproof mechanisms [62]. In the previous section, we proved that every cross-monotonic cost sharing scheme for certain games is poorly budget balanced. A natural question to ask is whether all group-strategyproof mechanisms for these games are so poorly budget balanced. Towards this aim, one might hope to show that every group-strategyproof mechanism corresponds to a cross-monotonic cost sharing scheme. In fact, given a group-strategyproof mechanism $\mathcal{M}$, it is possible to define a corresponding cost-sharing scheme $\xi_{\mathcal{M}}$ by having the agents in a set S bid a sufficiently large value and others zero, and letting $\xi_{\mathcal{M}}(i, S)$ be the payment charged by the mechanism to the agent i in this scenario. Throughout this section, we refer to the sufficiently large bid value for agent i as ∞_i. Unfortunately, this scheme is not necessarily budget-balanced or cross-monotonic. In fact, the following simple example shows that for every cost function, there is a group-strategyproof mechanism recovering all the cost.

Example 1 *Single Payment Mechanism: Arbitrarily order the agents from* 1 *to* n*. Then, find the first agent* i *in this order whose bid is at least* $C(\{i, \ldots, n\})$*. The set that will receive the service is* $Q = \{i, \ldots, n\}$*, and the total cost of servicing this set is paid by the agent* i*. Other agents pay nothing.*

Clearly, the mechanism is truthful (or strategyproof), since in computing the payment of an agent, the mechanism does not look at the bid of that agent. For the vector of true bids, let k be the agent that pays $C(\{k, \ldots, n\})$, thus all agents $i < k$ were asked to pay $C(\{i, \ldots, n\})$, but did not afford to pay it. For any agent $i < k$, i cannot be in any lying coalition, since i does not have incentive to overbid (as his happiness may become negative) and if he underbids he does not change the output of the mechanism. Agent k cannot be in any lying coalition either, since the only lying strategy for him is to underbid and does not get the service; in this case, the happiness of other agents may only decrease. No agent $i > k$ can change the output of the mechanism by lying either. It follows that the above mechanism is group strategyproof.

Intuitively, the mechanism in Example 1 is neither fair nor efficient. It always place the burden of the entire service cost on a small subset of agents while servicing others for free. Agents who receive the service for free, called *free riders*, increase the cost of the solution but do not contribute any payment. We consider constraining our mechanism to rule out free riders. The main advantage of this restriction is that the budget-balance factor of a mechanism $\mathcal{M}$ without free rider and its derived cost sharing is the same, since when agents in set S bid ∞_i and other agents bid 0, the serviced set is S (because there is no free rider). As the following example shows, there exist group-strategyproof mechanisms without free rider for cost functions for which no cross-monotonic cost sharing exists.

Consider the following cost function for 3 bidders: $C(\{1\}) = C(\{2\}) = C(\{3\}) = C(\{1,2\}) = C(\{1,3\}) = C(\{2,3\}) = 1$ and $C(\{1,2,3\}) = 2$. It is not hard to check that no budget-balanced cross-monotonic cost sharing scheme exists for this cost function. We will prove that the following mechanism $\mathcal{M}$ is group strategyproof and budget balanced and does not have any free rider. The mechanism $\mathcal{M}$ for a bid vector (b_1, b_2, b_3) is as follows:

1. If $b_1 \geq 1$ then
   (a) If $\min(b_2, b_3) > \frac{1}{2}$ then charge $(1, \frac{1}{2}, \frac{1}{2})$,
   (b) Else if $\max(b_2, b_3) < \frac{1}{2}$ then charge $(1, 0, 0)$,
   (c) Else if $b_2 \geq b_3$ then charge $(\frac{1}{2}, \frac{1}{2}, 0)$,
   (d) Else (since $b_3 > b_2$) charge $(\frac{1}{2}, 0, \frac{1}{2})$.
2. Else if $\frac{1}{2} \leq b_1 < 1$ then
   (a) If $\min(b_2, b_3) > \frac{1}{2}$ then charge $(0, \frac{1}{2}, \frac{1}{2})$,
   (b) Else if $\max(b_2, b_3) < \frac{1}{2}$ then charge $(0, 0, 0)$,
   (c) Else if $b_2 \geq b_3$ then charge $(\frac{1}{2}, \frac{1}{2}, 0)$,
   (d) Else (since $b_3 > b_2$) charge $(\frac{1}{2}, 0, \frac{1}{2})$.
3. Else if $b_1 < \frac{1}{2}$ then
   (a) If $\min(b_2, b_3) \geq \frac{1}{2}$ then charge $(0, \frac{1}{2}, \frac{1}{2})$,

(b) Else if $b_2 \geq 1$ then charge $(0, 1, 0)$,

(c) Else if $b_3 \geq 1$ then charge $(0, 0, 1)$,

(d) Else charge $(0, 0, 0)$.

Table 5.1 depicts the payment vectors of the mechanism $\mathcal{M}$. In this table, b_i is the bid of agent i. In $\mathcal{M}$, we first look at b_1, and based on b_1, we check the bids of agents 2 and 3. The payment vectors of this mechanism is depicted in the second, third and fifth columns of Table 5.1. We refer to these three columns as the first, second and third column of the payment vectors of this table in the following discussion. Since $\mathcal{M}$ has no free rider, if the payment of an agent for a bid vector is zero, it means that this agent is not served for that bid vector.

	if $b_1 \geq 1$	oth. $\frac{1}{2} \leq b_1 < 1$	oth. if $b_1 < \frac{1}{2}$	
if $\min(b_2, b_3) > \frac{1}{2}$	$(1, \frac{1}{2}, \frac{1}{2})$	$(0, \frac{1}{2}, \frac{1}{2})$	if $\min(b_2, b_3) \geq \frac{1}{2}$	$(0, \frac{1}{2}, \frac{1}{2})$
oth. if $\max(b_2, b_3) < \frac{1}{2}$	$(1, 0, 0)$	$(0, 0, 0)$	oth. if $b_2 \geq 1$	$(0, 1, 0)$
oth. if $b_2 \geq b_3$	$(\frac{1}{2}, \frac{1}{2}, 0)$	$(\frac{1}{2}, \frac{1}{2}, 0)$	oth. if $b_3 \geq 1$	$(0, 0, 1)$
oth. if $b_2 < b_3$	$(\frac{1}{2}, 0, \frac{1}{2})$	$(\frac{1}{2}, 0, \frac{1}{2})$	oth.	$(0, 0, 0)$

Table 5.1: The budget-balanced group-strategyproof mechanism without free rider.

Let $p_i(b_1, b_2, b_3)$ denote the payment of agent i for the bid vector (b_1, b_2, b_3). Suppose, for contradiction, that $(\bar{b}_1, \bar{b}_2, \bar{b}_3)$ be the true bid vector (willingness to pay) of players for which there exists a lying coalition. Let S be the lying coalition and (b_1, b_2, b_3) be the lying bid vector. We first prove that S must include 1. If $\bar{b}_1 < \frac{1}{2}$, the cost sharing for players 2 and 3 in the third column of bid vectors in Table 5.1 is cross monotonic and the Moulin-Shenker mechanism [62] implies that the mechanism is group strategyproof. If $\bar{b}_1 \geq \frac{1}{2}$, players 2 and 3 can only lie and change the payment to another vector in the same column of the bid vectors in the Table 5.1. The only way that a player $i \in \{2, 3\}$ can benefit is when $\bar{b}_i > \frac{1}{2}$ and i does not get the service for true bids. But this does not happen in any of the first two columns of bid vectors in the table. Therefore $1 \in S$.

Now, we need to argue that 1 cannot be in any lying coalition either. We consider two cases:

Case 1: $\bar{b}_1 < \frac{1}{2}$. In this case, the payment of player 1 should be zero, even though $b_1 \geq \frac{1}{2}$. This happens only when $\min(b_2, b_3) > \frac{1}{2}$ or $\max(b_2, b_3) < \frac{1}{2}$. If $\max(b_2, b_3) < \frac{1}{2}$, then none of players 2 and 3 gets the service, thus this cannot be the lying strategy. If $\min(b_2, b_3) > \frac{1}{2}$, the payments of players 2 and 3 in the mechanism for $b_1 < \frac{1}{2}$ and $b_1 \geq \frac{1}{2}$ is the same. Therefore, players 2 and 3 cannot benefit from this lying coalition.

Case 2: $\bar{b}_1 \geq \frac{1}{2}$. In this case, agent $i \in \{2, 3\}$ can only benefit if $\bar{b}_i > \frac{1}{2}$ and i does not get the service for true bids. But for any bid vector $b_1 \geq \frac{1}{2}$ and $b_i > \frac{1}{2}$, the payment of i is $\frac{1}{2}$. Therefore, i cannot strictly benefit from any lying coalition in which $b_1 \geq \frac{1}{2}$. Agent 1 can only bid $b_1 < \frac{1}{2}$, if either $\bar{b}_1 = \frac{1}{2}$ and $p_1(\bar{b}_1, \bar{b}_2, \bar{b}_3) = \frac{1}{2}$, or $\bar{b}_1 = 1$ and $p_1(\bar{b}_1, \bar{b}_2, \bar{b}_3) = 1$. But in this case for each $i \in \{2, 3\}$, either $p_i(\bar{b}_1, \bar{b}_2, \bar{b}_3) = \frac{1}{2}$ or $\bar{b}_i \leq \frac{1}{2}$. Therefore, $i \neq 1$ cannot strictly benefit from any lying coalition.

Finally, we need to argue that 1 cannot strictly benefit from any lying coalition. In order for agent 1 to benefit, $\bar{b}_1 > \frac{1}{2}$ and $p_1(\bar{b}_1, \bar{b}_2, \bar{b}_3) \neq \frac{1}{2}$. In this case, agent 1 cannot benefit by bidding $b_1 < \frac{1}{2}$. Thus, $b_1 \geq \frac{1}{2}$. As a result, in order for agent 1 to strictly benefit from the lying coalition $\min(\bar{b}_2, \bar{b}_3) > \frac{1}{2}$ or $\max(\bar{b}_2, \bar{b}_3) < \frac{1}{2}$. If $\min(\bar{b}_2, \bar{b}_3) > \frac{1}{2}$, the happiness of 2 and 3 in bid vector $(b_1, \bar{b}_2, \bar{b}_3)$ is positive and one can check that in order for player 2 or 3 to change his bid such that 1 strictly benefits, 2 or 3's happiness strictly decreases (since one of them does not get the service). Therefore, in this case, 2 and 3 cannot be in any lying coalition. If $\max(\bar{b}_2, \bar{b}_3) < \frac{1}{2}$, 1 can only strictly benefit if 2 or 3 bid greater than or equal to $\frac{1}{2}$. Again, in order for 2 or 3 to strictly help 1, this player should pay at least $\frac{1}{2}$ and thus, his happiness will be negative. Thus, 2 or 3 cannot be in the lying coalition.

It follows that there is no lying coalition for $\mathcal{M}$ and thus, $\mathcal{M}$ is group strategyproof. The cost sharing scheme derived from $\mathcal{M}$ is not cross-monotonic though, e.g., $\xi_{\mathcal{M}}(1, \{1, 2, 3\}) > \xi_{\mathcal{M}}(1, \{1, 2\})$.

In the mechanism given above, there are scenarios in which an agent does not receive the service, but would receive service if she increased her bid by any positive amount, however small. For example, if $b_1 = b_2 = \frac{1}{2}$, the third bidder gets service for any bid greater than $\frac{1}{2}$, but not for $b_3 = \frac{1}{2}$. In fact, we can show that a cross-monotonic cost sharing scheme can be derived from any mechanism with no free riders for which such situations do not occur. More precisely, we call a mechanism $\mathcal{M}$ *upper continuous* if for every agent i, if i gets the service for every bid value greater than x holding other bids fixed, then i gets the service if she bids x. Upper continuity by itself is not difficult to satisfy. In fact, the mechanism in Example 1 is upper continuous. However, we prove that mechanisms satisfying upper continuity and no-free-rider conditions are as hard as cross-monotonic cost sharing schemes. To prove this statement, we prove a useful structural lemma (Lemma 5.3.1) on the cost sharing schemes derived from a group-strategyproof mechanism. We need the following definition to state this lemma.

Definition 5.3.1 *Let $\xi : \mathscr{A} \times 2^{\mathscr{A}} \mapsto \mathbb{R}^+ \cup \{0\}$ be a cost-sharing scheme, $S \subseteq \mathscr{A}$, and $i \in \mathscr{A} \setminus S$. We say i is* good *for set $S \cup \{i\}$ if for every $j \in S$, $\xi(j, S) \geq \xi(j, S \cup \{i\})$ and for at least one j a strict inequality holds; i is* bad *for $S \cup \{i\}$ if for every $j \in S$, $\xi(j, S) \leq \xi(j, S \cup \{i\})$ and for at least one j a strict inequality holds. If for all $j \in S$, $\xi(j, S) = \xi(j, S \cup \{i\})$, we say i is* neutral *for $S \cup \{i\}$.*

Lemma 5.3.1 *Let $\xi_{\mathcal{M}}$ be a cost-sharing scheme derived from a group-strategyproof mechanism $\mathcal{M}$ with no free rider. Then for every agent i and set S, $i \in S$, i is either good, bad, or neutral for S.*

Proof. For an agent i, let ∞_i be a large enough number such that if agent i bids ∞_i, he will get the service, independent of other agents' bids. Let $S = \{1, 2, \ldots, k\}$. It is enough to show that agent k is good, bad, or neutral for S. Consider the following 3 bid vectors:

1. $(\infty_1, \ldots, \infty_k, 0, \ldots, 0)$

2. $(\infty_1, \ldots, \infty_{k-1}, \xi_{\mathcal{M}}(k, S), 0, \ldots, 0)$

3. $(\infty_1, \ldots, \infty_{k-1}, 0, 0, \ldots, 0)$

Since the mechanism does not have a free rider, in the first bid vector, set S is served and in the third bid vector, $S - \{k\}$ is served. Consider two cases:

Case 1: k is served for bid vector 2. We claim that in this case, k's payment is equal to $\xi_{\mathcal{M}}(k, S)$. Otherwise k would lie in scenario 1 by announcing $\xi_{\mathcal{M}}(k, S)$ as his bid and consequently pay less. We claim that for every $j \neq k$, $\xi_{\mathcal{M}}(j, S) \leq \xi_{\mathcal{M}}(j, S - \{k\})$. Assume for contradiction that $\xi_{\mathcal{M}}(j, S) > \xi_{\mathcal{M}}(j, S - \{k\})$. Then j can convince k to bid 0 in scenario 2. Agent k's happiness remains the same (zero) and j pays less, and so $\{k, j\}$ form a lying coalition, contradicting the group strategyproofness. Here we used coalitions of size at most 2 in the proof.

Case 2: k is not served for bid vector 2. In this case, we claim that for every $j \neq k$, $\xi_{\mathcal{M}}(j, S) \geq \xi_{\mathcal{M}}(j, S - \{k\})$. Assume for contradiction that $\xi_{\mathcal{M}}(j, S) < \xi_{\mathcal{M}}(j, S - \{k\})$. Now, j can convince k to bid ∞_k in scenario 2. Agent k's happiness remains the same and j pays less, so again $\{k, j\}$ is a lying coalition (of size at most 2), contradicting group strategyproofness.

In case 1, k is good or neutral for S by definition and in case 2, k is bad or neutral by definition. □

Before proving the main theorem, we need to prove the following three lemmas:

Lemma 5.3.2 *Consider a group-strategyproof mechanism $\mathcal{M}$ without free riders. For the bid vector $B = (b_1, \ldots, b_n)$, if $b_i = 0$ for $i \notin S$ and $b_i > \xi_{\mathcal{M}}(i, S)$ for $i \in S$, then set S is serviced at their cost share, i.e., each $i \in S$ pays $\xi_{\mathcal{M}}(i, S)$.*

Proof. Order the bidders such that $S = \{1, \ldots, k\}$. Let ∞_j be a large enough number such that $\infty_j > \xi_{\mathcal{M}}(i, S)$ and if agent j bids ∞_j, he will get the service, independent of other bids. We will prove by induction that if the first i bidders in S bid ∞_i and the rest bid b_i, the set S is serviced at the cost share. First notice that if $b_j = \infty_j$ for all $j \in S$, S gets serviced at the cost share by the definition of the derived cost sharing $\xi_{\mathcal{M}}$. Consider the following bid vectors:

1. $(\infty_1, \ldots, \infty_{i-1}, \infty_i, b_{i+1}, \ldots, b_k, 0, \ldots, 0)$

2. $(\infty_1, \ldots, \infty_{i-1}, b_i, b_{i+1}, \ldots, b_k, 0, \ldots, 0)$

By inductive hypothesis, i gets serviced at his cost share $\xi_{\mathcal{M}}(i, S)$ for bid vector 1. If he gets serviced at less than his cost share for bid vector 2, when the true bids are as in scenario 1, i will lie and bid b_i. Similarly, if he gets serviced at more than his cost share for bid vector 2 or does not get the service, when the true bids are as in scenario 2, i will lie and bid ∞_i. Since there is no lying coalition, i gets serviced at his cost share for bid vector 2 and is thus indifferent between scenario 1 and 2. Therefore, no one can benefit between the two scenarios or else they could form a lying coalition with bidder i. Since all nonzero bidders get serviced at their cost share for bid vector 1, they should get the service at their cost share for bid vector 2 as well. □

Lemma 5.3.3 *Consider a group-strategyproof mechanism $\mathcal{M}$ with no free riders. For any bid vector $\mathcal{B} = (b_1, \ldots, b_n)$, if the serviced set is S, then payment of $i \in S$ is exactly equal to $\xi_{\mathcal{M}}(i, S)$.*

Proof. Let $S_1 = \{i \in S | b_i \leq \xi_{\mathcal{M}}(i, S)\}$. First we prove that all members of S_1 should pay exactly $\xi_{\mathcal{M}}(i, S)$, thus their bid should be equal to $\xi_{\mathcal{M}}(i, S)$. Let $S_2 = S - S_1$ be the members of S who bid strictly greater than their cost share in S and $S_3 = A - S$ be the remaining bidders. Let $\mathcal{P}$ be a bid vector at which every member i of S_1 bids ∞_i, every member i of S_2 bids b_i (his bid in $\mathcal{B}$) and all members of S_3 bid zero. From Lemma 5.3.2, in scenario $\mathcal{P}$, set S will get the service at the cost $\xi_{\mathcal{M}}(i, S)$. If any agent i in S_1 pays less than $\xi_{\mathcal{M}}(i, S)$ (his payment in $\mathcal{P}$) for bid vector $\mathcal{B}$, then in scenario $\mathcal{P}$, i can form a lying coalition with $S_1 \cup S_3$ and pretend the bid vector $\mathcal{B}$. As a result nobody in $S_1 \cup S_3$ pays more and at least agent i pays strictly less contradicting group strategyproofness.

Now consider agent $i \in S_2$. If he pays more than $\xi_{\mathcal{M}}(i, S)$ for bid vector $\mathcal{B}$, he would make the lying coalition $S_1 \cup S_3 \cup \{i\}$ in scenario $\mathcal{B}$ and pretend the bid vector $\mathcal{P}$ instead and pay less. Agent i strictly benefits from this lying strategy and no other agent pays more. Therefore, $S_1 \cup S_3 \cup \{i\}$ pretending $\mathcal{P}$ is a lying coalition for scenario $\mathcal{B}$ which contradicts group strategyproofness. Therefore, all members of S_2 also pay the same as $\xi_{\mathcal{M}}(i, S)$. □

Lemma 5.3.4 *Consider a bid vector $\mathcal{B}' = (b'_1, b'_2, \ldots, b'_n)$ in a group-strategyproof mechanism $\mathcal{M}$ with no free rider. If the set S gets the service for the bid vector $\mathcal{B}'$ and $i \in S$ is bad for S then $b'_i > \xi_{\mathcal{M}}(i, S)$.*

Proof. Assume for contradiction that $b'_i = \xi_{\mathcal{M}}(i, S)$. If the true bid of i is $\bar{b}_i = b'_i$, the happiness of agent i for bid vector $\mathcal{B}'$ is zero. Since the set S gets the service at the bid vector $\mathcal{B}'$, from Lemma 5.3.3, the payment of agent $j \in S$ is $\xi_{\mathcal{M}}(i, S)$ for bid vector $\mathcal{B}'$. Consider the bid vector $\mathcal{B} = (b_1, \ldots, b_n)$ where $b_j = \infty_j$ for $j \in S - \{i\}$; and $b_j = 0$ otherwise. By the definition of $\xi_{\mathcal{M}}$, the payment of $j \in S - \{i\}$ for $\mathcal{B}$ is $\xi_{\mathcal{M}}(j, S - \{j\})$. Since i is bad for S, $\xi_{\mathcal{M}}(j, S - \{i\}) \leq \xi_{\mathcal{M}}(j, S)$ for all $j \in S$ and the strict inequality holds at least for one agent $t \in S$. In addition, agent i is indifferent between bid vectors $\mathcal{B}$ and $\mathcal{B}'$, since in both cases his happiness is zero.

Thus, in scenario $\mathcal{B}'$, all agents can make a lying coalition and pretend $\mathcal{B}$: nobody is harmed by this lying strategy and agent t strictly benefits. Therefore, the fact that i gets service in $\mathcal{B}'$ at his cost share contradicts the group strategyproofness of $\mathcal{M}$. □

Using the above lemmas, we can prove the following theorem.

Theorem 5.3.5 *The cost function C has an upper-continuous α-budget-balanced group-strategyproof mechanism with no free riders if and only if it has an α-budget-balanced cross-monotonic cost-sharing scheme.*

Proof. Given an α-budget-balanced cross-monotonic cost-sharing scheme ξ, the Moulin-Shenker mechanism $\mathcal{M}(\xi)$ gives an α-budget-balanced group-strategyproof mechanism [62] with upper-continuity and without free riders.

Given an α-budget-balanced group-strategyproof mechanism, the upper continuity condition implies that all agents get the service at their cost share if all other agents bid ∞_i. Therefore, Lemma 5.3.4 implies that there is no bad agent for any set. Thus, from Lemma 5.3.1 all agents are good or neutral in $\mathcal{M}$. This proves that $\xi_{\mathcal{M}}$

is cross monotonic. Moreover, the budget-balance factor of $\xi_{\mathcal{M}}$ is the same as the budget-balance factor of $\mathcal{M}$, since there is no free rider. $\square$

In the proof of Theorem 5.3.5 we do not use coalitions of size greater than 2. Thus, with the assumptions of no-free-rider and upper-continuity, coalitions of size 2 are as strong as coalitions of arbitrary size. We do not know if this equivalence holds without these assumptions.

Finally, we consider a strong fairness property called the subsidy-freeness property[7] [61]. This condition says that the total charge of the mechanism to all users in a set S is at most the cost of the set $c(S)$[8]. In the following theorem, we show that the budget-balanced group-strategyproof mechanisms with this property are equivalent to the budget-balanced cross-monotonic cost sharing schemes.

Theorem 5.3.6 *There exists a budget-balanced group-strategyproof mechanism with no free rider satisfying the subsidy-freeness property for a cost function C if and only if this cost function has a budget-balanced cross-monotonic cost-sharing scheme.*

Proof. The "if" part follows from the Moulin-Shenker mechanism [62]. In order to prove the "only if" part, we first prove that given a budget-balanced group-strategyproof mechanism $\mathcal{M}$ satisfying subsidyfreeness, there is no bad agent for any set. Consider agent i who is bad for the set S. Then for any agent $j \in S - \{i\}$,

$$\xi_{\mathcal{M}}(j, S) \geq \xi_{\mathcal{M}}(j, S - \{i\}),$$

and strict inequality holds at least for one j. Thus,

$$\sum_{j \in S-\{i\}} \xi_{\mathcal{M}}(j, S) > \sum_{j \in S-\{i\}} \xi_{\mathcal{M}}(j, S - \{i\}),$$

contradicting the subsidy-freeness property. Therefore, all agents are good or neutral for any set S, and so the cost sharing $\xi_{\mathcal{M}}$ derived from $\mathcal{M}$ is cross monotonic. Since

[7]This property is considered by Moulin. Also see Devanur et al. [13] for a discussion of strategyproof (but not group-strategyproof) mechanisms satisfying subsidy-freeness for set cover and facility location problems.

[8]This is the same as the core property for each subset S.

the mechanism has no free rider, the budget-balance factor of the mechanism and the derived cost sharing is the same. □

We do not know if this theorem holds for budget-balance factors other than 1, and so the upper bounds for different problems in this chapter only imply that there is no budget-balanced group-strategyproof mechanism without free rider and with subsidy-freeness for these problems.

5.4 Conclusion

In this chapter, we studied upper bounds for the budget-balance factor of cross-monotonic cost-sharing schemes for a variety of combinatorial optimization games. Our techniques are quite general and may prove applicable to variety of other interesting games. For example, the facility location game restricted to a tree always has a budget-balanced cost sharing in the core [32], but we do not have a tight lower and upper bound for the budget-balance factor of a cross-monotonic cost sharing scheme. For the facility location on the line, we have the upper bound of $\frac{6}{7}$.[9] A more challenging open question is that of cross-monotonic cost-sharing schemes for the Steiner tree game. There are 1/2-budget-balanced scheme for the Steiner tree and Steiner forest [49, 50]. Recently, Konemann et. al [51], showed that this budget-balanced factor is tight for cross-monotone cost-sharing schemes for Steiner tree and Steiner forest. Their construction is based on the construction of our example for the metric facility location problem. However, the best known upper bound for the budget-balanced factor of the core property for both problems is 8/9. Closing the gap between the known lower and upper bounds ($\frac{8}{9}$ and $\frac{1}{2}$) for the budget-balanced factor of a cost sharing in the core for Steiner tree and Steiner forest is an interesting open problem.

A main motivation behind cross-monotonic cost-sharing schemes is the development of group-strategyproof mechanisms. As mentioned in this thesis, almost any

[9] The example is as follows: Consider 3 facilities with opening cost $2-\epsilon$, and 4 cities. The connection cost of facility 1 to cities 1 and 2 is 1. The connection cost of facility 2 to cities 2 and 3 is 1. The connection cost of facility 3 to cities 3 and 4 is 1. The other connection costs follow the triangle inequality.

cost function (including all those for which we derived upper bounds on cost sharing schemes) has a trivial group-strategyproof mechanism. Several different sets of axioms, some of which are explored in this thesis, can be added to the mechanisms to rule out these trivial ones. An interesting open question in this area is to extend the result of Theorem 5.3.6 and show that α-budget-balanced group-strategy mechanisms with subsidy-freeness are equivalent to α-budget-balanced cross-monotone cost-sharing schemes.

It is a standard economic result that a strategyproof mechanism can not be both efficient (i.e., return a solution that maximizes social welfare) and budget-balanced [34, 75]. It would be interesting to explore the possible budget-balance factor of group-strategyproof mechanisms that are in some sense close to efficient.

Chapter 6

Open Problems

In this thesis, we develop several new algorithms and decentralized mechanisms for different combinatorial optimization problems. In this chapter, we describe the main open problems corresponding to each chapter of the thesis.

1. The main open problem in Chapter 2 is to close the gap between the known approximation factor and the inapproximability result of GAP and IBDC. The best known approximation factor for these problems is $1 - \frac{1}{e} + \epsilon$ for any $\epsilon > 0$ and the only hardness result that is known for these problems is their APX-hardness. In Chapter 2, we proved that the factor $1 - \frac{1}{e}$ is almost tight for the CapDC problem.

   Another interesting open problem is to extend the local search or the LP-based algorithm to the maximization version of the facility location problem with packing constraints. The facility location problem with packing constraints can be viewed as the IBDC problem with an extra opening cost for each cache location and an installation cost for putting each request type in each cache location. No constant-factor approximation algorithm is known for these problems. In Chapter 2, we developed constant-factor approximation algorithms for all variants of the DCP problems and SAP and KMed.

2. The main open questions in Chapter 3 are as follows:

(a) A strategy profile in a strategic game is a *c-approximate Nash equilibrium* if no player can change his strategy and increase his payoff by more than a factor of c. The first open problem is to find a 2-approximate (or any constant-factor approximate) Nash equilibrium in the cut game in polynomial time or to prove that finding such an equilibrium is PLS-complete. The answer to this question may or may not imply a fast or poor convergence of $(1+\epsilon)$-greedy players (see the definition in Chapter 3) to Nash equilibria. In Chapter 3, we proved fast convergence of $(1+\epsilon)$-greedy players to approximate solutions, but not to pure Nash equilibria. The same set of questions can be asked about basic-utility and market sharing games.

(b) One open question is to bound the length of best-response walks to pure Nash equilibria in basic-utility games and market sharing games, or to prove that a best-response walk to a PSNE may be exponential.

(c) Another open question is the speed of convergence to constant-factor approximate solutions on polynomial-length random best-response walks in the market sharing game.

3. In Chapter 4, we developed a model for studying the outcome of a game as a result of a sequence of selfish behavior of players. Considering other types of walks in the state graph and studying other types of selfish behavior of players is an interesting area for future research. We can define the price of sinking for other types of selfish behavior and prove bounds on the performance of the outcome of the game by bounding the price of sinking in those models. Considering the model that we described in Chapter 4, the most interesting open problem is to bound the price of sinking for different variants of the distributed caching games (and in particular, the CapIBDC game). Characterizing the set of sink equilibria for different classes of games, e.g., the CapDC game is another area for future consideration.

4. Two main open questions related to Chapter 5 are as follows:

(a) Proving upper bounds or improving lower bounds for the budget-balanced factor of the "fair" group-strategyproof mechanisms is the most important open problem of this chapter. In particular, understanding the budget-balanced factor of the group-strategyproof mechanisms with no free rider and with the subsidy-freeness property is a challenging open problem.

(b) Another open problem is to close the gap between the lower and upper bounds of the budget-balanced factor of the cost allocations in the core for the rooted steiner tree game. The known lower and uppers bounds for this problem are $\frac{1}{2}$ and $\frac{8}{9}$, respectively. We conjecture that the right answer is $\frac{8}{9}$.

Bibliography

[1] Ageev and M. Sviridenko. Pipage rounding: A new method of constructing algorithms with proven performance guarantee. *J. of Combinatorial Optimization*, 8:307–328, 2004.

[2] B. Awerbuch, Y. Azar, and A. Epstein. The price of routing unsplittable flow. In *Proceedings on 37th Annual ACM Symposium on Theory of Computing (STOC)*, page To appear, 2005.

[3] R. Baev and R. Rajaraman. Approximation algorithms for data placement in arbitrary networks. In *Proceedings of the 12th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 661–670, 2001.

[4] V. Bahl, M. T. Hajiaghayi, V. S. Mirrokni, L. Qiu, and A. Saberi. Efficient power assignment in wireless LAN using duality. Submitted, 2005.

[5] M. Bahramgiri, M. Hajiaghayi, and V. S. Mirrokni. Fault-tolerant and 3-dimensional distributed topology control algorithms in wireless multi-hop networks. *ACM/Kluwer Wireless Networks*, To appear.

[6] R. Bahtia, N. Immorlica, T. Kimbrel, V. S. Mirrokni, S. Naor, and B. Scheiber. Confluent flow augmentation for data traffic engineering. In *17th ACM Symposium on Parallelism in Algorithms and Architectures(SPAA)*, 2005. To appear.

[7] O.N. Bondareva. Some applications of linear programming to cooperative games. *Problemy Kibernetiki*, 10:119–139, 1963.

[8] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker. Web caching and zipf-like distributions: Evidence and implications. In *Proceedings of 18th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, pages 126–134, 1999.

[9] B. Carr and S. Vempala. Randomize meta-rounding. *Random Structure and Algorithms*, 20(3):343–352, 2002.

[10] C. Chekuri and S. Khanna. A ptas for the multiple knapsack problem. In *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 213–222, 2000.

[11] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the 20th Annual ACM Symposium on Computational Geometry (SoCG)*, pages 253–262, 2004.

[12] X. Deng, T. Ibaraki, and H. Nagamochi. Algorithms and complexity in combinatorial optimization games. In *Proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 720–729, 1997.

[13] N. Devanur, M. Mihail, and V. Vazirani. Strategyproof cost sharing mechanisms for set cover and facility location problems. *Proceedings of ACM Conference on Electronic Commerce*, 2003.

[14] E. Even-dar, A. Kesselman, and Y. Mansour. Convergence time to Nash equilibria. In *Automata, Languages and Programming: 30th International Colloquium, ICALP*, pages 502–513, 2003.

[15] A. Fabrikant, C. Papadimitriou, and K. Talwar. On the complexity of pure equilibria. In *Proceedings on 36th Annual ACM Symposium on Theory of Computing (STOC)*, 2004.

[16] U. Feige and G. Kortsarz. Personal communications. 2005.

[17] Urie Feige. A threshold of ln n for approximating set cover. *Journal of the ACM*, 45(4):634 – 652, 1998.

[18] Uriel Feige, Magnus M. Halldorsson, Guy Kortsarz, and Aravind Srinivasan. Approximating the he domatic number. *SIAM Journal of Computing*, 32(1):172–195, 2002.

[19] J. Feigenbaum, D. Karger, V. S. Mirrokni, and R. Sami. Subjective cost policy routing. Submitted, 2005.

[20] J. Feigenbaum, C. Papadimitriou, and S. Shenker. Sharing the cost of multicast transmission. *Journal of Computer and System Sciences*, 63:21–41, 2001.

[21] J. Feigenbaum, C. Papadimitriou, and S. Shenker. Sharing the cost of multicast transmissions. *Journal of Computer and System Sciences*, 63:21–41, 2001.

[22] T. Fleiner. A fixed-point approach to stable matchings and some applications. *Mathematics of Operations Research*, 28(1):103–126, 2003.

[23] L. Fleischer, M. Goemans, V. S. Mirrokni, and M. Sviridenko. Distributed caching and the generalized assignment problem: New algorithms and mechanisms. Submitted, 2004.

[24] D. Fotakis, S. Kontogiannis, and P. Spirakis. Selfish unsplittable flow. *Theoretical Computer Science*, To appear.

[25] Drew Fudenberg and Jean Tirole. *Game Theory*. MIT Press, 1991.

[26] S. Fujishige. *Submodular Functions and Optimization*. North-Holland, 1991.

[27] N. Garg and J. Könemann. Faster and simpler algorithms for multicommodity flow and other fractional packing problems. In *39th Annual IEEE Symposium on Foundations of Computer Science*, pages 300–309, 1998.

[28] M. Goemans. Personal communication. 2005.

[29] M. Goemans, L. Li, V. S. Mirrokni, and M. Thottan. Market sharing games applied to content distribution in ad-hoc networks. In *Proceedings of the 5th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2004.

[30] M. Goemans and D. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, 42:1115–1145, 1995.

[31] Michel X. Goemans and David P. Williamson. New 3/4-approximation algorithms for the maximum satisfiability problem. *SIAM Journal of Discrete Mathematics*, 7:656–666, 1994.

[32] M.X. Goemans and M. Skutella. Cooperative facility location games. *Journal of Algorithms*, 50:194–214, 2004.

[33] C. Gomes, R. Regis, and D. Shmoys. An improved approximation algorithm for the partial Latin square extension problem. In *Proceedings of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2003.

[34] J. Green, E. Kohlberg, and J. J. Laffont. Partial equilibrium approach to the free rider problem. *Journal of Public Economics*, 6:375–394, 1976.

[35] M. Hajiaghayi, N. Immorlica, and V. S. Mirrokni. Power optimization in topology control algorithms for wireless multihop networks. In *Proceedings of the Ninth Annual International Conference on Mobile Computing and Networking (MOBICOM)*, pages 300–312, 2003.

[36] M. Hajiaghayi, M. Mahdian, and V. S. Mirrokni. Facility location with general cost functions. *Networks*, 42:42–47, 2003.

[37] M.T. Hajiaghayi, G. Kortsarz, V. S. Mirrokni, and Z. Nutov. Power optimization for connectivity problems. In *Integer Programming and Combinatorial Optimization, 12th International IPCO Conference*, page To appear, 2005.

[38] M. Halldorosn, J. Halpern, L. Li, and V. S. Mirrokni. On spectrum sharing games. In *Proceedings of Twenty-Third Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC)*, pages 107–114, 2004.

[39] N. Immorlica, D. Karger, M. Minkoff, and V. S. Mirrokni. On the costs and benefits of procrastination: Approximation algorithms for stochastic combinatorial optimization problems. In *Proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 691–700, 2004.

[40] N. Immorlica, M. Mahdian, and V. S. Mirrokni. Cycle cover with short cycles. In *Proceedings of the 22nd Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 641–653, 2005.

[41] N. Immorlica, M. Mahdian, and V. S. Mirrokni. Limitations of cross-monotone cost sharings. In *Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, page To appear, 2005.

[42] K. Jain, M. Mahdian, and V. S. Mirrokni. 2-stage combinatorial optimization against adversary. In preparation, 2005.

[43] K. Jain, M. Mahdian, and M. Salavatipour. Packing stiner trees. In *Proceedings of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 266–274, 2003.

[44] K. Jain and V.V. Vazirani. Applications of approximation algorithms to cooperative games. In *Proceedings on 33rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 364–372, 2001.

[45] K. Jain and V.V. Vazirani. Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and lagrangian relaxation. *Journal of the ACM*, 48:274–296, 2001.

[46] K. Jain and V.V. Vazirani. Equitable cost allocations via primal-dual-type algorithms. In *Proceedings on 34th Annual ACM Symposium on Theory of Computing (STOC)*, pages 313–321, 2002.

[47] D. Johnson, C.H. Papadimitriou, and M. Yannakakis. How easy is local search? *Journal of Computer and System Sciences*, 37:79–100, 1988.

[48] M. Kandori, G. J. Mailath, and R. Rob. Learning, mutation, and long-run equilibria in games. *Econometrica*, 61:29–56, 1993.

[49] K. Kent and D. Skorin-Kapov. Population monotonic cost allocation on MST's. *In Operational Reearch Proceedings KOI*, pages 43–48, 1996.

[50] J. Koenemann, S. Leonardi, and G. Schaefer. A group-strategyproof mechanism for Steiner forests. *Proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2005. To appear.

[51] J. Koenemann, S. Leonardi, G. Schaefer, and S. van Zwam. From primal-dual to cost shares and back: A stronger LP relaxation for the Steiner forest problem. In *Automata, Languages and Programming: 32nd International Colloquium(ICALP)*, page To appear, 2005.

[52] E. Kohlberg and J. Mertens. On the strategic stability of equilibria. *Econometrica*, 54(5):1003–1037, 1986.

[53] E. Koutsoupias and C. Papadimitriou. Worst-case equilibria. In *Proceedings of the 16th Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 404–413, 1999.

[54] L. Li, M. Mahdian, and V. S. Mirrokni. Secure overlay network design: A combinatorial approach. Submitted, 2004.

[55] I. Milchtaich. Congestion games with player-specific payoff functions. *Games and Economics Behavior*, 13:111–124, 1996.

[56] V. S. Mirrokni and A. Sidiropolous. Convergence in cut games. Submitted, 2004.

[57] V. S. Mirrokni, M. Thottan, H. Uzunalioglu, and S. Paul. A simple polynomial time framework to reduced path decomposition in multi-path routing. In

Proceedings of 23rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), 2004.

[58] V. S. Mirrokni and A. Vetta. Convergence issues in competitive games. In *Proceedings of the 7th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (RANDOM-APPROX)*, pages 183–194, 2004.

[59] V. S. Mirrokni and A. Vetta. Sink equilibria and convergence. Submitted, 2005.

[60] D. Monderer and L. Shapley. Potential games. *Games and Economics Behavior*, 14:124–143, 1996.

[61] H. Moulin. *Axioms of cooperative decision making*, volume 1, chapter Cost sharing games and the core. Cambridge University Press, 1988.

[62] H. Moulin and S. Shenker. Strategyproof sharing of submodular costs: budget balance versus efficiency. *To appear in Economic Theory*, 1997.

[63] Hervé Moulin. Incremental cost sharing: Characterization by coalition strategyproofness. *Social Choice and Welfare*, 16:279–320, 1999.

[64] J. F. Nash. Non-cooperative games. *Annals of Mathematics*, 54:268–295, 1951.

[65] G. Nemhauser, L. Wolsey, , and M. Fisher. An analysis of the approximations for maximizing submodular set functions i. *Mathematical Programming*, 14:265–294, 1978.

[66] G. Nemhauser, L. Wolsey, , and M. Fisher. An analysis of the approximations for maximizing submodular set functions ii. *Mathematical Programming Studies*, 8:73–87, 1978.

[67] N. Nisan and A. Ronen. Algorithmic mechanism design. In *Proceedings on 31st Annual ACM Symposium on Theory of Computing (STOC)*, pages 129–140, 1999.

[68] Martin J. Osborne and Ariel Rubinstein. *A Course in Game Theory*. MIT Press, 1994.

[69] M. Pal and E. Tardos. Strategyproof mechanisms via primal-dual algorithms. In *Proceedings of 44th Symposium on Foundations of Computer Science (FOCS)*, pages 584–593, 2003.

[70] C. Papadimitriou. Algorithms, games, and the internet. In *Proceedings on 33rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 749–753, 2001.

[71] C.H. Papadimitriou, A. Schaffer, and M. Yannakakis. On the complexity of local search. In *Proceedings on 22nd Annual ACM Symposium on Theory of Computing (STOC)*, pages 438 – 445, 1990.

[72] E. Petrank. The hardness of approximation: Gap location. In *Computational Complexity*, pages 133–137, 1994.

[73] S. A. Plotkin, D. Shmoys, and É. Tardos. Fast approximation algorithms for fractional packing and covering problems. *Mathematics of Operations Research*, 20:257–301, 1995.

[74] S. Poljak. Integer linear programs and local search for max-cut. *Siam Journal of Computing*, 24(4):822–839, 1995.

[75] K. Roberts. The characterization of implementable choice rules. In J.J. Laffont, editor, *Aggregation and Revelation of Preferences*. Studies in Public Economics, Amsterdam, North Holland, 1979.

[76] R. W. Rosenthal. A class of games possessing pure-strategy Nash equilibria. *International Journal of Game Theory*, 2:65–67, 1973.

[77] A.E. Roth and J. H. Vande Vate. Random paths to stability in two-sided matchings. *Econometrica*, 58(1):1475–1480, 1990.

[78] T. Roughgarden and E. Tardos. How bad is selfish routing? *Journal of the ACM*, 49(2):236–259, 2002.

[79] A. Schaffer and M. Yannakakis. Simple local search problems that are hard to solve. *SIAM journal on Computing*, 20(1):56–87, 1991.

[80] P. Schuurman and T. Vredeveld. Performance guarantees of local search for multiprocessor scheduling. In *Integer Programming and Combinatorial Optimization, 8th International IPCO Conference*, pages 370–382, 2001.

[81] H. Shachnai and T. Tamir. Approximation schemes for generalized 2-dimensional vector packing with application to data placement. In *Proceedings of the 6th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX)*, pages 167–177, 2003.

[82] D. Shmoys and E. Tardos. An approximation algorithm for the generalized assignment problem. *Mathematical Programming*, 62(3):461–474, 1993.

[83] A. Srinivasan. Distributions on level-sets with applications to approximation algorithms. In *Proceedings of 44th Symposium on Foundations of Computer Science (FOCS)*, pages 588–597, 2001.

[84] Lloyd S.Shapley. On balanced sets and cores. *Naval Research Logistics Quarterly*, 14:453–460, 1967.

[85] M. Sviridenko. A note on maximizing a submodular set function subject to knapsack constraint. *Operations Research Letters*, 32:41–43, 2004.

[86] C. Swamy. Algorithms for the data placement problem. Unpublished, 2004.

[87] A. Vetta. Nash equilibria in competitive societies, with applications to facility location, traffic routing and auctions. In *Proceedings of 43rd Symposium on Foundations of Computer Science (FOCS)*, pages 416–425, 2002.

[88] N. Young. Randomized rounding without solving the linear program. In *Proceedings of the 7th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 170–178, 1996.

[89] N. Young. Sequential and parallel algorithms for mixed packing and covering. In *42nd Annual IEEE Symposium on Foundations of Computer Science*, pages 538–546, 2001.