GRAN SASSO SCIENCE INSTITUTE
SCHOOL OF ADVANCED STUDIES
Scuola Universitaria Superiore

PhD Thesis

Selfishness and optimization for multi-agent packing and coverage problems

PhD Program in Computer Science: XXXI cycle

Author:

Francesco Cellinese

Advisors:

Dr. Gianpiero Monaco
Dr. Gianlorenzo D'Angelo

July 2019

GSSI Gran Sasso Science Institute
Viale Francesco Crispi, 7 - 67100 L'Aquila - Italy

To my family and friends. You helped me
to conclude this incredible 10-years academic journey.

"A man provided with paper, pencil, and rubber, and subject to strict discipline, is in effect a universal machine."

Alan Turing

"Young man, in mathematics you don't understand things. You just get used to them."

John von Neumann

"Imagine if every Thursday your shoes exploded if you tied them the usual way. This happens to us all the time with computers, and nobody thinks of complaining."

Jeff Raskin

"Where is the 'any' key?"

Homer Simpson, in response to the message "Press any key"

Abstract

In the last decades of computer science development the focus gradually moved from centralized systems to a network perspective, mainly motivated by the quick and deep influence of the Internet. Numerous new problems arose, i.e. the need of efficient algorithms for IOT (*internet of things*) or models of behavior of large numbers of people trough social networks. In this scenario, many classical optimization problems are again under the spotlight since they are rethink under the perspective of multi-agent and distributed systems. Two classical examples are problems where the resources are decentralized, such as in web servers loading distribution, or simply game theoretical setting where some part of the problem is controlled by agents which aim at pursing their own goals.

In this thesis we focus on two classical optimization problems, the coverage problem and the bin packing problem, and we analyze them from both centralized and multi-agent perspectives. Both of them have plenty of real-world applications such as job scheduling, facility locations and resource allocations. Firstly, we consider multi-agent coverage where agents use their own budget to cover elements. We are interested in maximizing the total revenue defined as the sum of the revenue of each agent. For this problem we study two settings that differ in how the agents share the utilities of the elements. For each sub-setting we define a game and analyze the Nash equilibrium and its properties. We also consider the centralized problem where we maximize the total revenue, while satisfying the agents' budget constraints.
The second setting that we consider is a generalization of the budgeted coverage problem, where the profit is measured by a monotone submodular function over the elements. We give an approximation factor algorithm for the general case and discuss solutions for relevant instances of the problem.
Finally, we study a colorful bin packing game in which a set of items, each one controlled by a selfish agent, is to be packed into a minimum number of unit capacity bins. A bin has a unitary cost which is shared among the items it contains, so that agents are interested in selecting a bin of minimum shared cost. Moreover, each item has a color and two items with the same color can not be adjacent in a bin. We show the existence of Nash equilibria for two standard cost sharing functions and provide a tight characterization of their efficiency. We also design an algorithm which returns Nash equilibria with best achievable performance for an interesting special setting.

List of Publications

In the following, the list of the author's publications which some chapters of this doctoral dissertation are based on:

1. Vittorio Bilò, Francesco Cellinese, Giovanna Melideo, Gianpiero Monaco: **"On Colorful Bin Packing Games"**. COCOON 2018: 280-292
2. Francesco Cellinese, Gianlorenzo D'Angelo, Gianpiero Monaco, Yllka Velaj: **"Generalized Budgeted Submodular Set Function Maximization"**. MFCS 2018: 31:1-31:14
3. Francesco Cellinese, Gianlorenzo D'Angelo, Gianpiero Monaco, Yllka Velaj: **"Multi-Agent Coverage Problems"**. *(Submitted to conference, under peer review)*
4. Vittorio Bilò, Francesco Cellinese, Giovanna Melideo, Gianpiero Monaco: **"On Colorful Bin Packing Games"**. *(Journal version, under peer review)*
5. Francesco Cellinese, Gianlorenzo D'Angelo, Gianpiero Monaco, Yllka Velaj: **"Generalized Budgeted Submodular Set Function Maximization"**. *(Journal version, under peer review)*

Declaration of Authorship

I, Francesco Cellinese, declare that this thesis, titled "Selfishness and optimization for multi-agent packing and coverage problems", and the work presented in it is my own under the guidance of my supervisors Dr. Gianpiero Monaco and Dr. Gianlorenzo D'Angelo. I confirm that:

- Chapter 3 is based on the paper "Multi-Agent Coverage Problems", submitted to an international conference.
- Chapter 4 is based on the paper "Generalized Budgeted Submodular Set Function Maximization" [1], and its journal version submitted to Information and Computation.
- Chapter 5 is based on the paper "On Colorful Bin Packing Games" [2], and its journal version submitted to Journal of Combinatorial Optimization.

Signed:

_____

Date:

_____

Contents

Chapter 1

Introduction

Among the fundamental challenges in computer science, one of the main research topics is about combinatorial optimization problems. The development of the theoretical algorithms field moved in the past century through the analysis of a countless number of those problems. However, the vast majority of the classical optimization problems in computer science have been introduced before the rise of the Internet and in general the explotion of the network-based approaches that are now under the spotlight. In the last three decades, many of those problems have been re-designed in a variety of related problems to cope with the new challenges about game-theoretical settings, distributed algorithms and multi-agent settings. As a case study, in this thesis we consider two combinatorial optimization problems: Maximum Coverage (where we aim at maximizing the covered elements selecting a fixed number of subsets among a collection of subsets of those elements) and Bin Packing (where we want to pack items in the smallest amount of fixed size bins). Both problems are known to be NP-hard([3, 4]).

Although they are fundamental combinatorial optimization problems studied for decades, we found interesting gaps in the literature: for Maximum Coverage problem we first define a multi-agent setting where each agent aims at covering some elements while maximizing her revenue over them. Then, we define two game-theoretical settings based on the multi-agent problem and study the property of Nash equilibria. There are several works in the literature related with the maximum coverage in multiple agent scenarios, and we will show how we relate with those problems.

The second major result that we provide is about a generalization of Maximum Coverage where the goal is maximizing the value of a submodular function defined over the covered elements, with the constraint that the overall cost of the covered elements is at most a given budget. Our setting is strictly related to many problems in the literature but not captured by any of them. In fact, we generalize some of those problems. Moreover, our setting is relevant for the adaptive seeding problem, which is an algorithmic challenge

in the influence maximization field.

The last topic that we consider is a game-theoretical setting based on the Bin Packing problem. In particular, we consider a generalization of the Bin Packing game (called Colorful Bin Packing), where we want to pack items into the minimum number of bins while respecting some additional constraints related to the colors of the items. We define two natural game-theoretical settings based on the Colorful Bin Packing problem and study the existence and performance of Nash equilibria. This thesis represents the first approach with Nash equilibria in colorful bin packing games.

In the Maximum Coverage (MC) problem we are given a ground set X, a collection S of subsets of X with unit cost, and a budget k. The goal is selecting a subset $S' \subseteq S$, such that $|S'| \leq k$, and the number of elements of X covered by S' is maximized. MC has several applications in job scheduling, facility locations and resource allocations [5, Ch. 3], as well as in influence maximization [6]. A natural greedy algorithm starts with an empty solution and iteratively adds a set with maximum number of uncovered elements until k sets are selected. This algorithm has an approximation factor of $1 - \frac{1}{e}$ [7] wich is tight given the inapproximability result due to Feige [3]. In the Budgeted Maximum Coverage (BMC) problem, which is an extension of the maximum coverage, the cost of the sets in S are arbitrary, and thus a solution is feasible if the overall cost of the selected subset $S' \subseteq S$ is at most k. In [8], the authors present a polynomial time (greedy) algorithm with approximation factor of $1 - \frac{1}{e}$. In the Generalized Maximum Coverage (GMC) problem every set $s \in S$ has a cost $c(s)$, and every element $x \in X$ has a different weight and cost that depend on which set covers it. In [9], a polynomial time (greedy) algorithm with approximation factor of $1 - \frac{1}{e} - \epsilon$, for any $\epsilon > 0$, has been shown.

Motivated by the applications in complex systems such as the Internet have spawned, a recent interest in studying situations involving multiple agents, in this thesis we consider the *multi-agent coverage problem*. We are given a set A of k autonomous agents, a set X of n elements, and a set $\mathcal{S}$ of m containers where any $S \in \mathcal{S}$ is a subset of elements of X (i.e. $S \subseteq X$). Each container S has cost c_S, each element $x \in X$ has utility u_x, and each agent $i \in A$ has her own budget b_i. An outcome of the problem is a vector $O = (O_1, \ldots, O_k)$ where each agent i buys a subset of containers $O_i \subseteq \mathcal{S}$ of overall cost at most b_i. In an outcome O each agent gets a revenue for the covered elements. In particular, we consider two natural scenarios called *distributed* multi-agent coverage problem (DMCP) and *proportional* multi-agent coverage problem (PMCP). In DMCP the utility u_x of an element x is equally split among all the agents covering x with some containers in O, while in PMCP the utility u_x of x is equally split among all the containers in O containing x. We notice that DMCP and PMCP are exactly the maximum coverage problem when $k = 1$.

We first consider the classical optimization problem of computing an outcome which maximizes the total revenue, which is defined as the sum of the agents' revenue. Then,

we investigate distributed multi-agent coverage games (DMCG) and proportional multi-agent coverage games (PMCG) in which agents are autonomous and selfish, and their only concern is to maximize their own revenue. In such a scenario we adopt pure Nash equilibria as stable outcomes and analyze their convergence, existence and performance. A strong motivation for our research is represented by the increasing number of works in the literature that analyze optimization problems with multiple agents. There are many examples of multi-agent problems related to the Maximum Coverage problem ([10–14]) which differs from our setting DMCP and PMCP. A relevant setting in this area has been introduced by Chekuri and Kumar ([38]). They show a variant of the maximum coverage problem called maximum coverage with group budget constraints (MCG), where we are given a ground set X, and a collection of sets over the elements. The collection of sets is partitioned into groups and the goal in MCG is to select k sets and maximizing the cardinality of the union, selecting at least one set from each group. The authors provide a $\frac{1}{2}$-approximation algorithm. Moreover, the authors present a cost version of the problem where each set has an associated cost. Again, the goal is to select at least one set from each group, using at most a specific budget for each group. Moreover, the total cost cannot exceed an overall budget. For this setting the authors give a $\frac{1}{12}$-approximation algorithm. This problem is closely related with our multi-agent coverage problem but cannot be used to model our setting: in [38] the assumption is that the groups are disjoint. In [54] Farbstein and Levin show that if the sets in the groups are not disjoint the MCG problem cannot be approximated within any constant factor.

In all the above problems the profit of a solution is given by the sum of the weights of the covered elements. An important and studied extension for the Maximum Coverage is adopting a nonnegative, nondecreasing, submodular function f, which assigns a profit to each subset of elements. In this thesis we consider the Generalized Budgeted submodular set function Maximization problem (GBSM). We are given a ground set of elements X, a set of containers S, and a budget k. The goal is to find a subset of elements along with an associated set of containers such that the overall costs of both is at most a given budget and the profit is maximized. Each container has its own cost, while the cost of each element depends on its associated container. Finally, the profit is measured by a monotone submodular function over the elements. As mentioned above, our setting has a revelant application in the adaptive seeding problem [15, 16]. In its non-stochastic version, the problem is to select amongst certain accessible nodes in a network, and then select amongst neighbors of those nodes, in order to maximize a global objective function. In particular, given a set X and its neighbors $N(X)$ there is a monotone submodular function defined on $N(X)$, and the goal is to select $t \leq k$ elements in X connected to a set of size at most $k-t$ for which the submodular function has the largest value. GBSM is an extension of the aforementioned problem since we

consider more general costs.

Another reason that motivated our interest in such a setting is the literature referring to related problems: in the Submodular set Function subject to a Knapsack Constraint maximization (SFKC) problem we have a cost $c(x)$ for any element $x \in X$, and the goal is selecting a set $X' \subseteq X$ of elements that maximizes $f(X')$, where f is a monotone submodular function subject to the constraint that the sum of the costs of the selected elements is at most k. This problem admits a polynomial time algorithm that is a $\left(1 - \frac{1}{e}\right)$-approximation [17]. Since the MC problem is a special case of SFKC problem, such approximation is tight. A more general setting was considered in [18], where the authors consider the following problem called Submodular Cost Submodular Knapsack (SCSK): given a set of elements $V = \{1, 2, \ldots, n\}$, two monotone non-decreasing submodular functions g and f ($f, g : 2^V \to \mathbb{R}$), and a budget b, the goal is finding a set of elements $X \subseteq V$ that maximizes the value $g(X)$ under the constraint that $f(X) \leq b$. They show that the problem cannot be approximated within any constant bound. Moreover, they give a $1/n$ approximation algorithm and mainly focus on bi-criterion approximation.

We emphasize that GBSM is not a special case of the Generalized Maximum Coverage problem, since we consider any monotone submodular functions for the profits. Moreover, since our cost function is not submodular, the setting SCSK considered in [18] does not generalize our model. Finally, we notice that our setting extends the SFKC problem, given that, the cost of an element is not fixed like in SFKC, but instead depends on the container used for covering it.

The last section of the thesis investigates Bin Packing, more specifically our game-theoretical setting based on the Colorful Bin Packing. In the classical definition of Bin Packing we are given a set of items with different sizes in $[0, 1]$, which have to be packed into the smallest possible number of unit capacity bins. This problem is known to be NP-hard (see [19] for a survey). The problem can model many practical scenarios, like bandwidth allocations problems, packet scheduling problems (see [20, 21]). The study of bin packing in a game theoretical context has been introduced in [20]. In such a setting, items are handled by selfish players and the unitary cost of each bin is shared among the items it contains. In the literature, two natural cost sharing functions have been considered: the *egalitarian* cost function, which equally shares the cost of a bin among the items it contains (see [22, 23]), and the *proportional* cost function, where the cost of a bin is split among the items proportionally to their sizes. Namely, each player is charged with a cost according to the fraction of the used bin space her item requires (see [20, 24]). We stress that for games with uniform sizes, where all the items have the same size s, the two cost functions coincide. Each player would prefer to choose a strategy that minimizes her own cost, where the strategy is the bin chosen by the player. Nash equilibria, i.e., packings in which no player can lower her cost by changing the selected

bin in favor of a different one, are mainly considered as natural stable outcomes for these games. The social cost function that we aim to minimize is the number of open bins (a bin is open if it stores at least one item).

In this thesis we consider *colorful bin packing* games, a generalization of the bin packing games where we are given a set of n selfish players, a set of $m \geq 2$ colors, and a set of n unit capacity bins. The special case of games with two colors are called *black and white* bin packing games. Each player controls an indivisible colored item of size in $[0, 1]$. Each item needs to be packed into a bin. The items in a bin are ordered in a sequence. Each bin cannot exceed its capacity and no item can be misplaced, that is no item is adjacent to another one of the same color in the bin. We use both the egalitarian and the proportional cost functions, where we set the cost of any misplaced item as infinite, and adopt Nash equilibria as stable outcomes of the games. We notice that in any Nash equilibrium no player can be charged with an infinite cost since any player can move to an empty bin and getting cost 1. We notice that, if all the items have different colors, these games correspond to the bin packing ones. However, when there are items with the same color, the stable outcomes of the games are structurally different than bin packing ones. In fact, we show that in our games, Nash equilibria perform very differently than in bin packing ones. Colorful bin packing games can model many practical scenarios (see [25–27]) like television and radio stations which schedule a set of programs of variuos genre on different channels, or displays on websites that alternate between different types of information and advertisements, or software which renders user-generated content and assigns it to columns which are to be displayed.

1.1 Main contributions

This thesis deals with various algorithmic challenges regarding Maximum Coverage and Bin Packing. Since we cope with NP-hard optimization problems, our goal is to design algorithms to solve them approximately. When we consider game-theoretical settings, our goal is to characterize the existence of Nash equilibria and to show the efficiency of them.

In the case of our multi-agent coverage settings, we first consider the optimization problem of maximizing the total revenue, which is defined as the sum of the agents' revenue. We provide an $1 - \frac{1}{\sqrt{e}}$-approximation algorithm for the case when the maximum cost of a container is upper-bounded by the minimum budget of an agent, i.e. the case in which each agent is allowed to buy any container, and a randomized Monte-Carlo algorithm for the general case that runs in polynomial time, satisfies the budget constraint in expectation, and guarantees an expected $1 - \frac{1}{e}$ approximation factor. Finally, we investigate the distributed and proportional multi-agent coverage games. We show that DMCG can

be modeled via another work in the literature and show that it always guarantees the existence of a Nash equilibrium and a tight bound of $2 - \frac{1}{k}$ for price of anarchy and stability. Finally, we show that the existence of a Nash equilibrium for PMCG is not guaranteed.
For the GBSM problem we first present an algorithm that guarantees an approximation factor of $\frac{1}{2}\left(1 - \frac{1}{e^{\alpha}}\right)$, where $\alpha \leq 1$ is the approximation factor of an algorithm for a sub-problem, namely α is the approximation factor of an algorithm used to select a subset of elements whose ratio between marginal increment in the objective function and marginal cost is maximum. We give two polynomial-time algorithms to solve this sub-problem. The first one gives us $\alpha = 1 - \epsilon$ if the costs satisfies a specific condition, which is fulfilled in several relevant cases, including the unitary costs case and the problem of maximizing a monotone submodular function under a knapsack constraint. The second one guarantees $\alpha = 1 - \frac{1}{e} - \epsilon$ for the general case. The gap between our approximation guarantees and the known inapproximability bounds is $\frac{1}{2}$. We extend our algorithm to a bi-criterion approximation algorithm in which we are allowed to spend an extra budget up to a factor $\beta \geq 1$ to guarantee a $\frac{1}{2}\left(1 - \frac{1}{e^{\alpha\beta}}\right)$-approximation. If we set $\beta = \frac{1}{\alpha}\ln\left(\frac{1}{2\epsilon}\right)$, the algorithm achieves an approximation factor of $\frac{1}{2} - \epsilon$, for any arbitrarily small $\epsilon > 0$.
Considering Colorful Bin Packing games we show that under the considered cost functions they do not converge in general to a Nash equilibrium, however Nash equilibria are guaranteed to exist. We provide algorithms to build such equilibria in polynomial time under the egalitarian cost function and pseudo-polynomial time for a constant number of colors under the proportional one. We then measure the quality of Nash equilibria. We show that the prices of anarchy and stability are unbounded under both cost functions with at least 3 colors, while they are equal to 3 with 2 colors. We finally focus on the subcase of games with uniform sizes (i.e., all items have the same size). We show a tight characterization of the efficiency of Nash equilibria and design an algorithm which returns Nash equilibria with best achievable performance.

1.2 Outline of the thesis

The rest of this thesis is organized as follows. In Chapter 2 we introduce the basic definitions regarding the problems discussed in the thesis. We provide an overview of the basic concepts about algorithmic game theory, Maximum Coverage and Bin Packing. We introduce a variety of related problems and give the basic definitions about submodular functions.
In Chapter 3 we focus on our multi-agent settings for Maximum Coverage. We present two settings that differ only in the definition of the revenue of each agent and investigate the existence and the efficiency of Nash equilibria for the related games.

In Chapter 4 we give a generalization for the Maximum Coverage where we aim at maximizing a submodular function. We first present an algorithm where the approximation factor is related to a sub-problem, and show two polynomial-time algorithms as subroutines to solve this sub-problem.
In Chapter 5 we introduce Colorful Bin Packing games, and provide a complete characterization of the efficiency of Nash equilibria under both cost functions. Finally, in Chapter 6, we conclude and present several future research directions.

Chapter 2

Preliminaries

In this chapter, we give a series of definitions that will be used in the rest of the thesis and in the next sections we focus on the fundamental results known in the literature regarding the main topics investigated in this thesis. In the first section introduce the basics of algorithmic game theory, such as Nash equilibria, then we describe the Coverage problem and its submodular generalizations. Finally, we use the last section to introduce the Bin Packing problem and a collection of well-known results.

2.1 Algorithmic game theory

This section introduces some basic notions in the field of algorithmic game theory[1], building a framework for understanding the results in the next chapters. An extensive discussion is available in [28].

2.1.1 Games and strategies

The purpose of game theory is to model strategic settings where some actors act in a shared environment, and they affect each other. Each actor (usually called a *player* or *agent*) has an outcome based on the choices made by any other actor.
We start with the most common example of a game: the *prisoner's dilemma*. In this game, there are two prisoners (called $P1$ and $P2$) which are held in different cells and cannot talk to each other. They are accused of a major crime, and every prisoner can choose between being silent and confessing the crime. If both remain silent they will

[1] Many of the definitions reported in this section are from [28, 29]

have only a short prison time because the authorities will not be able to prove all the charges. If only one prisoner confesses, his confession will be used as proof against the other: the prisoner who confessed will spend 1 year in prison, the second will stay in jail for 5 years. The last option is that both of them confesses. In this case they will be accused for the crime but will have a 4 year deal (rather then 5) for the cooperation with the authorities.

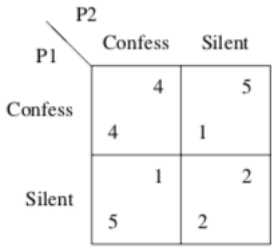

FIGURE 2.1: Possible years of prison for $P1$ and $P2$. (image from [28])

As depicted in Figure 2.1 the outcome of each prisoner does not depend only on his decision, but also on the behavior of the other. When a prisoner need to decide which is the best behavior, we assume that he is rational and selfish. In other words, he will try to choose in such a way that he will spend the minimum possible time in prison. For both of them, as a group, the best choice is to stay silent and spend only 2 years in prison. But even if both of them choose to stay silent, every prisoner is tempted to confess, because he can reduce his jail-time to one year. Moreover, if he stays silent maybe the other prisoner chooses to confess and in this he would increase his jail time from 2 to 5 years. For this reasons, the best selfish decision for them is to confess and we say that this is the only *stable* solution (a set of strategies where each agent cannot improve his condition by change his behavior). But this is a good deal for the authorities, in fact they are getting 2 confessions and giving a total of 8 years of prison.

The *prisoner dilemma* represents what in the literature is called a *strategic game*. Informally, a strategy for an agent is the choice that he made considering his payoff and the strategies of all the other agents.

Definition 2.1 (Strategic game)**.** A strategic game $\mathcal{G}$ consists of a set P of $n \geq 2$ where each agent p_i has a finite set of strategies S_i and a payoff function $\omega_i : S_1 \times \cdots \times S_n \to \mathbb{R}$ for $1 \leq i \leq n$.

One of the main relevant properties of a strategic game is the possibility to reach a stable profile of strategies, where nobody is interested in changing his strategy. A particular situation where an agent is not interested in a change of strategy is in the case of dominant strategies. Informally we say that a strategy is dominant for an agent if its

payoff is superior than any other possible strategy, for every possible choice of strategies of the other agents.

Definition 2.2 (Dominant strategy)**.** In a strategic game with n agents, we say that the strategy $s_i \in S_i$ of an agent p_i strictly dominates his strategy s_i' if $\omega_i(s_1, \ldots, s_i, \ldots, s_n) \geq \omega_i(s_1, \ldots, s_i', \ldots, s_n)$ for every strategy $s_j \in S_j$ adopted by any other agent p_j with $j \neq i$. A dominant strategy for agent p_i is a strategy $s_i \in S_i$ that strictly dominates any other strategy $s_i' \in S_i$.

Armed with the previous definition we can specify that the *confess* strategy of the *prisoner dilemma* is the best option of any agent because is a *dominant strategy*. Since we are under the assumption of rational agents, a *dominant strategy* is always chosen by an agent if it is available.

2.1.2 Nash equilibria

The concept of dominant strategy is a strong property for a game, but unfortunately is also strict and not very common in many games. For this reason the gold standard in the field of game theory when we talk about stable solutions is represented by the concept of Nash equilibria. Introduced by Nash in 1950 ([30]), it was a revolutionary addition to the young field of game theory, and that discovery led Nash to a Nobel prize in economics in 1994.

Definition 2.3 (Nash equilibria)**.** A state $s = (s_1, \ldots, s_n)$ is a Nash equilibrium (NE) if for every agent p_i with a strategy s_i it holds that

$$\omega_i(s_1, \ldots, s_i, \ldots, s_n) \geq \omega_i(s_1, \ldots, s_i', \ldots, s_n)$$

for any $s_i' \in S_i$.

As we anticipated, the concept of Nash equilibria is similar to that of a dominant strategies, but weaker and easier to find in a game. In a Nash equilibria any agent is not interested in a change of strategy given the actual strategies of the other agents.
However, that does not mean that that strategy is always a good strategy. Maybe for an agent is possible to have a much greater payoff, but this possibility is subjected to the change of strategy of another agent. In this scenario, and in the results that we consider in the thesis, we say that the game are *non cooperative*. In those kind of games each agent does not form any coalition with other agents, even if a combined change of strategy could lead to a greater payoff for each agent. In a non cooperative game each agent decide only about his strategy, without any consideration about the possible

decision of other agents.
It is interesting to notice that a game may admit more than one Nash equilibria, which leads to the problem of discriminating among the possible NE. Even worse, we know that non-cooperative games do not guarantee the existence of an NE. Nash, in a famous theorem ([30]), determined that every finite non-cooperative game guarantees the existence of a *mixed strategy* Nash equilibria. In this setting an agent can choose different strategies with some probabilities, and the payoff function is influenced by the probability distribution of the strategies.
When we need a distinction with the above mentioned *mixed strategy* NE, definition 2.3 is called *pure* NE. In our case we stick with the original definition and we study only the case of *pure* Nash equilibria.

After this introduction about some basics concept of game theory, it is important to explain the role of game theory in computer science. In the last four decades the world changed in many aspects, but there was one new factor that changed the history of the human kind at a faster pace than ever: the internet. In every field of science the concept of any kind of network (human, information, computation network, etc.) became more and more relevant. In computer science, many classical optimization problems (especially those about networks) have been reviewed under this new paradigm. Many new problems arose, since the perspective of decentralized settings permits to model many realistic and interesting scenarios, where actors related to the problem are not interested in a common well-being typical of the centralized algorithmic solutions, but want to maximize their own payoff. However, the lack of cooperation between agents may easily lead to a bad overall solution. Here we need to introduce the concept of *social welfare*, that is - usually - the desired solution by the entire community of agents. It can be defined as any function $f : S \to \mathbb{R}$ defined over the set of states of the game. Depending on the specific settings, there are games where the social welfare is not directly related the the agents' payoffs. However, a common definition for the social welfare is the sum of the agents payoffs. To give an example of the previous definition, we recall the solution of the *prisoner dilemma*: even if each prisoner chooses his dominant strategy, the *social welfare* is the sum of the outcome of the agents. Since we are considering years of prison, the *social welfare* aims to minimize the total number of prison time. As we said in that section, the two dominant strategies result in 8 years of prison. The optimal solution in terms of *social welfare* (usually called *social optimum*) was to choose the "silent" strategy for both agents and have only 4 years of prison.
When we analyze a problem from the point of view of the algorithmic game theory, we want to define and study games where this degradation of *social welfare* due to the agents' selfishness is limited, or at least bounded. There are several techniques to design

games where the agents can maximize their payoff while the social function remains acceptable for the problem. Since we need to model problems with games and the overall outcome is related with the choices of the agents and eventual stable states, we need to formally measure the results that a game provide.

2.1.3 Properties of Nash equilibria

In what follows we define some properties about the existence, the quality and the computational efficiency of Nash equilibria.

Existence and convergence to NE

As we said before, the existence of a (pure) Nash equilibria in a game is not guaranteed. Therefore, the first interesting property to investigate in a game is the existence of at least one Nash equilibria.
If the existence of a NE in a game is shown, the next question became natural: it is possible to converge to a NE? In each step, an agent can change his strategy. Obviously that will happen only if the agent can improve his payoff. More precisely, we say that an agent that change his strategy to improve his payoff is performing an *improving step* (or *improving deviation*). We define a game $\mathcal{G}$ as *convergent* if is always possible to reach a NE starting from any strategy profile and letting agents perform any improving deviation. Informally, a game is not convergent if is possible to create a cycle of improving deviations of some agents that leads again to the starting strategy profile.
A common technique to prove the convergence of a game is to find a suitable potential function $\Phi : S \to \mathbb{R}^+$. Proved in a seminal work by Rosenthal [31], the existence of such a potential function tells us that the sequence of improving deviations will end in a Nash equilibria.

Compute a Nash equilibria

Since we are interested in finding and studying Nash equilibria, another natural problem arises: we need to consider the computational complexity of this problem. The problem itself of finding a Nash equilibria does not fit in the typical computational classes of P and NP. Dealing with search problems concerning the computation of discrete fixed points, Papadimitrou in 1991 ([32]) introduced the complexity class known as PPAD (Polynomial Parity Arguments on Directed graphs). In 2005 Chen and Deng [33] proved that even for a non-cooperative 2-agents game, the problem of finding a Nash equilibria in this game in PPAD-complete. It is believed that the class PPAD contains hard

problems, but the PPAD-completeness is a weaker proof of intractability respect to the NP-completeness.
When a game is studied it is usually interesting to find extreme equilibria, such as those equilibria where the social function is minimized or maximized. These kinds of equilibria are frequently shown to be hard to compute.

Price of Anarchy and Price of Stability

In the previous section the debate was only considering the existence of a Nash equilibria in a game. But when we consider a game from the perspective of the algorithmic game theory, we are also interest in the *quality* of the Nash equilibria. Since we consider the equilibrium point of the game as a solution, we want to investigate if that equilibrium point is far from the optimal solution. The first definition that try to investigate such gap is the Price of Anarchy (PoA). Introduced by Koutsoupias and Papadimitriou [34], the PoA aims to measure the minimum possible loss that we have to face in a NE respect to the optimal solution. Comparing the study of the quality of a NE with a worst-case analysis of an algorithm, the Price of Anarchy is related to the approximation ratio: it give us the measurement of the loss in optimality in our game. Formally,

Definition 2.4 (Price of anarchy)**.** Given a game $\mathcal{G}$ and a social function f, let $\mathcal{N}$ be the set of all NE of $\mathcal{G}$ and OPT be a state of $\mathcal{G}$ optimizing f. The price of anarchy $PoA_{\mathcal{G}}(f)$ of a game $\mathcal{G}$ according to f is defined as

$$PoA_{\mathcal{G}}(f) = \sup_{s \in \mathcal{N}} \frac{f(s)}{f(OPT)}$$

The price of stability (PoS) is the ratio between the solution given by the best equilibria in terms of social value and the optimal solution. Its definition is similar to the previous one, but in this case we consider the minimum possible ratio between the social value of a NE and the optimum.

Definition 2.5 (Price of stability)**.** Given a game $\mathcal{G}$ and a social function f, let $\mathcal{N}$ be the set of all NE of $\mathcal{G}$ and OPT be a state of $\mathcal{G}$ optimizing f. The price of stability $PoS_{\mathcal{G}}(f)$ of a game $\mathcal{G}$ according to f is defined as

$$PoA_{\mathcal{G}}(f) = \inf_{s \in \mathcal{N}} \frac{f(s)}{f(OPT)}$$

2.2 Maximum Coverage

Maximum Coverage is a fundamental combinatorial optimization problem, where we are given a ground set X, a collection S of subsets of X with unit cost, and a budget k. The goal is selecting a subset $S' \subseteq S$, such that $|S'| \leq k$, and the number of elements of X covered by S' is maximized. The Maximum Coverage (MC) which has several applications in job scheduling, facility location ([5, Ch. 3]) and resource allocation [6].
The Maximum Coverage problem can be approximated by a factor $1 - \frac{1}{e}$, through a well known algorithm given by Feige [3]. The algorithm starts with an empty solution and iteratively adds a set with maximum number of uncovered elements until k sets are selected.
From Maximum Coverage followed a series of special cases and sub-problems. For instance, in Maximum h-Coverage, h denotes the maximum size of each subset of elements. If the size of the subsest is small, the previous inapproximability results does not hold. Namely, this problem is known to be APX-hard for any $h \geq 3$ [35]. Moreover, notice that when $h = 2$ that problem is equal to the maximum matching problem. A simple polynomial local search heuristic has an approximation ratio very close to $\frac{2}{h}$ [36], and there exists a polynomial time algorithm that achieves an approximation factor of $\frac{5}{6}$ for the case when $h = 3$ [37].
In 1999, Khuller, Moss and Naor ([8]) introduced the Budgeted Maximum Coverage (BMC). This problem extends the maximum coverage, setting the cost of the subsets of elements to an arbitrary value. Clearly, in this problem the solution is feasible if the sum of the costs of the selected subsets is at most equal to the given budget. In their paper, Khuller et al. present a polynomial time greedy algorithm with approximation factor of $1 - \frac{1}{e}$, which is tight due to the inapproximability results of the Maximum Coverage.
In 2008, Cohen and Katzir introduced the Generalized Maximum Coverage (GMC), which a further generalization of BMC. In this problem every subset has a cost, and every element has a different weight and cost that depend on which set covers it. In [9], a polynomial time (greedy) algorithm with approximation factor of $1 - \frac{1}{e} - \epsilon$, for any $\epsilon > 0$, has been shown.
In the maximum coverage with group budgeted constraints, the subset of elements are partitioned into groups, and the goal is to pick k subsets to maximize the cardinality of their union with the restriction that at most one subset can be picked from each group. In [38], the authors propose a $\frac{1}{2}$-approximation algorithm for this problem, and smaller constant approximation algorithm for the cost version. In the ground-set-cost budgeted maximum coverage problem, given a budget and a hypergraph, where each vertex has a non-negative cost and a non-negative profit, we want to select a set of hyperedges such that the total cost of the covered vertices is at most the budget and the total profit of all

covered vertices is maximized. This problem is strictly harder than budgeted max coverage. The difference of this problem to the budgeted maximum coverage problem is that the costs are associated with the covered vertices instead of the selected hyperedges. In [39], the authors obtain a $\frac{1}{2}\left(1-\frac{1}{\sqrt{e}}\right)$-approximation algorithm for graphs (which means having sets of size 2) and an FPTAS if the incidence graph of the hypergraph is a forest (i.e. the hypergraph is Berge-acyclic).

In all the above problems the profit of a solution is given by the sum of the weights of the covered elements. An important and studied extension is adopting a nonnegative, nondecreasing, submodular function f, which assigns a profit to each subset of elements. In order to introduce those kind of problems, in what follows we briefly introduce some basic notions about submodularity, and then we show a series of results regarding the above mentioned problems.

2.2.1 Submodular set functions

Submodularity is a property of set functions with deep theoretical consequences and various applications: in computer science it has recently been identified and utilized in domains such as information gathering, image segmentation and speeding up satisfiability solvers.

Definition 2.6 (Submodular Set Function)**.** A set function f is called *submodular* if satisfies

$$f(S \cup \{v\}) - f(S) \geq f(T \cup \{v\}) - f(T)$$

for all elements $v \notin T$ and all pairs of sets $S \subseteq T$.

The marginal gain from adding an element to a set S is at least as high as the marginal gain from adding the same element to a superset of S.

In the further chapter we will use a particular type of submodular function, which is also *monotone*.

Definition 2.7 (Monotone Submodular Function)**.** A submodular function f is called *monotone* if satisfies

$$f(S \cup \{x\}) \geq f(S), \quad \forall x. \tag{2.1}$$

Definition 2.8 (Coverage Function)**.** Let $\Omega = \{E_1, E_2, \ldots, E_n\}$ be a collection of subsets of some ground set Ω'. The function $f(S) = |\cup_{E_i \in S} E_i|$ for $S \subseteq \Omega$ is called a coverage function.

There are multiple alternative (and equivalent) definitions for submodular functions, Nemhauser et al. [40] proved the equivalence between each of them. In the same paper they introduced a number of interesting properties about submodular functions that makes them interesting for many research field. Kempe et al. [41] deeply used one of these properties as a foundation for their results: let f be a submodular function. Suppose that f takes only non-negative values and is monotone, in the sense that adding an element to a set cannot cause f to decrease: $f(S \cup \{v\}) \geq f(S), \forall$ elements v and sets S. Searching a k-element set S for which $f(S)$ is maximized is an NP-hard optimization problem but a property of [40] shows that the simple greedy hill-climbing algorithm approximates the optimum to within a factor of $(1 - 1/e)$ (where e is the base of the natural logarithm): start with the empty set, and repeatedly add an element that gives the maximum marginal gain.

Theorem 2.9. *[40, 41] For a non-negative, monotone submodular function f, let S be a set of size k obtained by selecting elements one at a time, each time choosing an element that provides the largest marginal increase in the function value. Let S^* be a set that maximizes the value of f over all k-element sets. Then $f(S) \geq (1 - 1/e) f(S^*)$.*

In other words, S provides a $(1 - 1/e)$ approximation.

2.2.2 Submodular maximization problems

Since in the next chapters we consider a setting of Maximum Coverage where we aim at maximizing the value of a submodular function, in this section we show some of the related works in the literature. As we show in chapter 4, our setting is not generalized or captured by any of the related problem that we are about to show.
The first problem that we consider is the submodular set function subject to a Knapsack Constraint maximization (SFKC) problem. This problem is presented in [17] by Sviridenko. In this problem we have a cost for any element, and the goal is selecting a set X of elements that maximizes $f(X)$, where f is a monotone submodular function subject to the constraint that the sum of the costs of the selected elements is at most equal to the budget. This problem admits a polynomial time algorithm that is an $\left(1 - \frac{1}{e}\right)$-approximation. Since the Maximum Coverage problem is a special case of SFKC problem, this result is tight.
In 2013, Iyer and Bilmes ([18]) introduced a more general setting, called Submodular Cost Submodular Knapsack (SCSK). In this problem we are given a set of elements $V = \{1, 2, \ldots, n\}$, two monotone non-decreasing submodular functions g and f $(f, g : 2^V \rightarrow \mathbb{R})$, and a budget b, the goal is finding a set of elements $X \subseteq V$ that maximizes the value $g(X)$ under the constraint that $f(X) \leq b$. Their two main results

are to show that the problem cannot be approximated within any constant bound and providing a $1/n$ approximation algorithm and they mainly focus on bi-criterion approximation.
Another relevant problem which is related to submodular optimization and is related to the problems that we show in this thesis is the Influence Maximization. In this problem we are given a graph where each node represents a potential customer of a product and we are interested in finding a set of people that maximizes the expected number of active nodes through a diffusion process. Initially we have access only to a subset of nodes called *core set*. Given a budget k, we can activate a node in the core set with unitary cost. The active nodes initiate the diffusion process (i.e. via the independent cascade model). The *influence* of a set A of nodes, denoted $\sigma(A)$, is the expected number of active nodes at the end of the diffusion process, given that A is this initial set of active nodes chosen in the core set. Formally, $\sigma(A) = E[|A_t|]$, where A_t is the (random) time of quiescence.

Definition 2.10. (Influence Maximization) [41] Given a budget k, select a k-set of nodes A such that $A \subseteq J$ and $\sigma(A)$ is maximized.

As reported in section 2.2.1, Kempe et al. [41, 42] used the properties of submodular functions to achieve interesting results in different diffusion models. Their strategy consists in showing that for several diffusion models the resulting influence function $\sigma(\cdot)$ is submodular. An influence function quantifies the expected number of nodes that are influenced when a set of individuals initiates a cascade. Based on this research, numerous techniques and improvements were developed, ranging from sophisticated predictive models of influence [43–47] to fast approximation methods [48–51]. Kempe et al. shows that the greedy algorithm may not evaluate the influence function exactly. However, by simulating the diffusion process and sampling the resulting active sets, it is possible to obtain arbitrarily close approximations to $\sigma(A)$, with high probability. For the most common diffusion models the problem is intractable, but there are some basic algorithms that provide $(1 - 1/e)$-approximation results [41].

2.2.3 Multi-agent problems

In the next chapters we will address multi-agent coverage problems. There is a growing body of literature that recognizes the importance of analyzing optimization problems with multiple agents. The first setting that we consider is presented in [10], where the authors study a class of combinatorial problems with multi-agent submodular cost functions. In such problems we are given a set of elements X and a collection of sets over the elements. We are also given k agents, where each agent i specifies a normalized

monotone submodular cost function $f_i : 2^X \to \mathbb{R}^+$. The goal is to find a set S in the collections and a partition $S_1, \ldots, S_k$ of S such that $\sum_i f_i(S_i)$ is minimized. The authors manage to fix the collection to some combinatorial structures, the authors define a subclass of fundamental optimization problems: vertex cover, shortest path, perfect matching and spanning tree. The authors provide bounds for all these problems., and the results are extended in [11] and in [12].

Another relevant setting that we mention which is related to multiple agents coverage problems is the general covering problem. We are given a set of elements X where each $x \in X$ is associated to a positive integer weight. Moreover, we are given n collections $S_1, \ldots, S_n$ of subsets of X where $S_i \subset 2^X$ is a subset of the power-set of the elements. The goal is to choose one subset s_i from each collection S_i such that their union $\cup_{i\in[n]} S_i$ has maximum total weight. In [52] the authors consider the general covering problem where the choice in each collection is made by an independent agent. For covering an element, the agents receive a revenue defined by a non-increasing revenue sharing function. This function defines the fraction that each covering agent receives from the elements. They study how to define a revenue sharing function such that every Nash equilibrium approximates the optimal solution by a factor of $1 - \frac{1}{e}$. They also show a centralized $1 - \frac{1}{e}$ approximation algorithm for the general covering problem.

Chekuri and Kumar in [53] introduce a variant of the maximum coverage problem called maximum coverage with group budget constraints (MCG). In this problem we are given a collection of sets $\mathcal{S} = \{S_1, \ldots, S_n\}$ where each set S_i is a subset of a given ground set X. The collection $\mathcal{S}$ is partitioned into groups $G_1, \ldots, G_l$. In MCG the goal is to pick k sets from $\mathcal{S}$ in order to maximize the cardinality of their union, with the additional constraint of picking at least one set from each group. For this setting the authors provide a $\frac{1}{2}$-approximation algorithm. Even if this is not a paper about multi-agent settings, MCG can model a variety of multi-agent problems related to coverage. The authors also present a cost version of the problem where each set S_i has an associated cost $c(S_i)$, there is a budget B_j for each group G_j, and an overall budget B. In the cost version the goal is to maximize the cardinality of the union of the selected sets, respecting also the budget constraints: the total cost of the sets selected in each group cannot exceed the group's budget and the overall cost of the selected sets cannot exceed B. The authors give a $\frac{1}{12}$-approximation algorithm for the cost version of the problem. This factor was improved to $\frac{1}{5}$ by Farbstein and Levin [54].

Chakrabarty and Goel [13] consider the maximum budgeted allocation problem where we are given a set of indivisible items and agents. Each agent i is willing to pay b_{ij} on item j and has a budget b_i. The goal is to allocate items to agents to maximize the revenue, that is, the sum, over all the agents, of the price for the items she bought. The main results described in the paper are: a 3/4-approximation algorithm that exploits a natural LP relaxation of the problem, and a 15/16 hardness of approximation result.

We also consider the setting proposed by Vondrak ([14]). This problem considers the submodular welfare problem: m items are to be distributed among n agents with utility functions $w_i : 2^{[m]} \to \mathbb{R}^+$. Assuming that agent i receives a set of items S_i, the goal is to maximize the total utility $\sum_{i=1}^{n} w_i(S_i)$. In this paper, the authors work in the value oracle model where the only access to the utility functions is through a black box returning $w_i(S)$ for a given set S. They develop a randomized continuous greedy algorithm which achieves a $(1-1/e)$-approximation for their problem in the value oracle model. A last work that is worth to mention is [55] where the authors consider a class of games, called the distributed welfare games, that can be utilized to model the game theoretical version of the our distributed multi-agent coverage problem, that will be introduced and discussed in Chapter 3.

2.3 Bin Packing

Bin packing is a classical problem in combinatorial optimization, where we want to pack items of different sizes in $[0, 1]$ into the smallest possible number of unit capacity bins. During the 1970s was one of the first problems where computer scientists investigated the worst-case performance guarantees. This problem has been addressed in many works in the literature (see [19] for a survey) and countless sub-problems were derived from it. Bin packing has a variety of real-world applications, such as the optimization of trucks space or the schedule of television series and commercial. If we consider further constraints, e.g. the impossibility of packing similar categories of items into the same bin we can capture a broader set of real-life scenarios. Following in the section, we will introduce two relevant examples of such subproblems: the *colorful Bin Packing* and the *black&white Bin Packing*. Furthermore, we will introduce some game theoretical settings derived from the bin packing that are known in the literature as *selfish Bin Packing*.

Definition 2.11 (Bin Packing)**.** In the bin packing problem, we are given a set of items $X = \{x_1, \ldots, x_n\}$ where the element x_i has size $s_i \in (0, 1]$ and a set of unitary capacity bins $\mathcal{B} = \{B_1 \ldots, B_n\}$. The goal is to pack the items into a minimum number of bins without exceeding their capacity, i.e. $\forall B_j, \sum_{x_i \in B_j} s_i \leq 1$.

2.3.1 Algorithms for Bin Packing

The simplest algorithm considered for bin packing is the *Next Fit* [56]. In this straightforward online algorithm, there is a currently open bin, and any new element is placed into this bin. When an item does not fit into the bin, a new empty bin is opened and became the current bin. The algorithm stops when all the items are placed. Next Fit runs

in polynomial time and it is not difficult to prove that the algorithm is 2-approximating. A natural improvement to the Next Fit algorithm is to consider all the bins with some empty space as open bins. We consider a new rule where we place an item in the first bin into which it will fit, otherwise we open a new bin and we place the current item as first item of the new bin. This algorithm is known as *First Fit*. The trivial implementation of this algorithm achieves a quadratic-time, but as shown in [56] it is possible to develop a $O(n \log n)$ running time implementation of the First Fit using an appropriate data structure. The wider range of destinations for each item results in a better approximation respect the the *Next Fit*. As Ullman proved in the early 70s ([57]) the Next Fit strategy achieves a 1.7-approximation. This approximation is tight since it possible build a queue of an arbitrarily large number of items with a 1.7-approximation using First Fit.

The packing rules that we considered for *Next Fit* and *First Fit* are only two of the many possible rules that were considered in the literature. A very interesting packing rule is used in the *Best Fit* algorithm. As in the *First Fit*, any bin with some empty space is considered as open, and the current item is placed inside the bin with the minimum empty space left which is enough to store the current item. Johnson shows that *Best Fit* has a tight 1.7-approximation ratio ([56]). Despite this algorithm does not give any theoretical improvement respect to *First Fit*, the two algorithms can give significantly different results on individual lists. Johnson introduced several examples where differently structured lists of items results in a 50% difference between the two packing rules. Similar results are given for *Almost Any Fit*: a tight 1.7-approximation algorithm that can be used to achieve better results on individual lists respect to the previous algorithms. All the previous algorithm shares an interesting property, namely the approximation ratio of each algorithm improves significantly as the maximum item size declines ([56]).

When the online restriction is removed, it is possible to improve dramatically the efficiency of the algorithms. A simple and effective idea is to adapt the *First Fit* to the offline case, applying a sorting step before applying the packing rule. Sorting the items in size-decreasing order leads us to the *First Fit Decreasing*. This algorithm achieves a 1.22-approximation factor ([56]). In an example proposed by Johnson, this ratio is shown to be tight. Note that there are two more algorithms called *Best Fit Decreasing* and *Next Fit Decreasing*, which are defined analogously: after a decreasing sorting of the items, the *Best Fit* or *First Fit* packing is applied. For these algorithms the same results are given. As with the earlier algorithm, the approximation factor of the Decreasing algorithms improves when the size of the maximum element decreases.

2.3.2 Colorful Bin Packing

To provide a consistent background for the results proposed in the next chapters we need to introduce two recent variants of Bin Packing problem: the *Black&White Bin Packing* (B&W BP) and its generalization *Colorful Bin Packing* (CBP). Both these problems can be considered in the online ([58], [27]) and offline([59],[60]) version. In the CBP each item has a color and a size and the goal is to pack those items in the minimum number of unitary size bins. To have a valid packing, two consecutive items in a bin cannot have the same color. Note that the offline CBP is a generalization of the classical Bin Packing if each item has a different color. Clearly, this implies that CBP is NP-hard. The B&W BP is a special case of CBP where the items have only two colors. This problem has been introduced in 2012 by Balogh et al. ([61]). It is important to notice that one of the main differences between this problems and the original Bin Packing is that the order in which items are packed matters, and this restriction makes the previous algorithms less efficient, especially in the online case. For the online case of B&W BP Balogh et al. ([58]) shows that the previous algorithms "Fit-like" have a competitive ratio which is lower-bounded by 3. At the same time they give a 3-competitive algorithm. Moreover, they show that the competitive ratio for any deterministic online algorithm for this problem has a lower bound of 1.7213. This bound has been improved by Dósa and Epstein [27] showing a lower bound of 2.
Moving to the offline setting, Balogh et al. [59] show a variety of results. The first one is a 2-approximating heuristic that runs in $O(n \log n)$-time where n is the number of items to pack. Moreover, they propose an asymptotic PTAS (APTAS) algorithm, which is lately improved as asymptotic FPTAS (AFPTAS). Using the APTAS it is possible to reach an $\frac{3}{2}$-approximation algorithm with a bad running time respect to the heuristic. Contrariwise the AFPATS allows a faster running time algorithm with a worse constant additive error.

The Colorful Bin Packing problem is studied in recent works. Introduced by Dósa and Epstein in 2014 [27], they propose a series of results: they start showing that each combinatorial algorithm commonly used for the Bin Packing problem (such as *First Fit, Next Fit* and so on) have unbounded asymptotic competitive ratios for CBP. Then they propose a new algorithm called *Balanced-Pseudo* which is $4-$competitive. Furthermore, they show that the lower bound for any algorithm for CBP is 2. Finally, they introduce the sub-setting in which every item has size zero. For this particular setting they show that the lower bound for every possible competitive algorithm is $\frac{3}{2}$.
In follow-up research, Böhm et al. [25] extended and improved the results given by Dósa and Epstein. In this work they give a 3.5-competitive online algorithm and an higher

2.5 lower bound for any competitive algorithm on the online CBP. They studied also the zero-size setting, giving a 5/3-competitive algorithm. They improve the lower bound for zero-size, matching the value of 5/3 and concluding that their algorithm is optimal for this problem. Finally in the black&white zero-size setting they proposes an optimal 1-competitive online algorithm.
A significant improvement is given by Matsakis in [60], with a new offline algorithm for the CBP problem. He propose a 2-approximating algorithm that runs in $O(n \log n)$-time where n is the number of items to pack.

2.3.3 Selfish Bin Packing

The study of Bin Packing in a game theoretical context has been introduced in 2006 by Bilò ([20]). In a Bin Packing game each item is handled by a selfish agent, and the agents shares the costs of the chosen bin with the other agents in the same bin. Three main sharing functions are considered in the literature:

- *proportional* game: the cost of the bin is split among the items it contains proportionally to their size.
- *egalitarian* (or *unit*) game: the cost of the bin is equally shared among the items it contains.
- *general weight* game: in this game each item is associated with a positive weight, which is not related to its size. Similarly with the proportional case, the cost of the bin is split among the items it contains proportionally to their weight.

In 2013, Epstein published a survey ([62]) on the know results about bin packing with selfish items. As we introduced before, Bilò ([20]) published the first paper regarding a game theoretical approach on Bin Packing. In the same paper he introduces the *proportional* setting, showing a variety of results regarding the study of the Nash equilibrium. First of all, he shows that a proportional game always converges to a Nash equilibrium through the potential function method. Moreover, exploiting the same potential function, he derives a bound on the number steps needed to reach an approximate Nash equilibrium. Finally, he considered the quality of such Nash equilibrium, giving upper and lower bounds for the price of anarchy, respectively $\frac{5}{3}$ and $\frac{8}{5}$.
Those results have been improved multiple times during the last decade. In 2008, Yu and Zhang ([63]) give a polynomial time algorithm to find a Nash equilibrium in a proportional game, showing that computing a Nash equilibrium is an easy problem. Moreover, they improve the bounds for the price of anarchy of the game, showing a lower bound of 1.6416 and an upper bound of 1.6575.

In 2011, Epstein and Kleiman ([21]) improve those results to nearly tight lower and upper bounds of 1.6416 and 1.6428, respectively. Furthermore, they consider the Strong Nash equilibrium[2](SNE) showing that a packing is a Strong Nash equilibrium if and only if it is produced by the Subset Sum algorithm for Bin Packing. From this result, they are able to prove that in such bin packing games SPoA (*strong* PoA) = SPoS (*strong* PoS). Finally they show that there is no polynomial time algorithm to compute the Strong Nash equilibria, unless P = NP.

The most relevant paper regarding the egalitarian bin packing games has been published by Ma et al. in 2012 ([23]). The authors firstly introduce the setting, then they show that Next Fit Increasing (see Section 2.3.1 for related algorithms) creates an NE for every input. The approximation ratio of Next Fit Increasing is equal approximately to 1.69103. This value is the approximation ratio of a number of algorithms, such as Next Fit Increasing and Next Fit Decreasing (see [23]). Regarding the quality of the Nash equilibrium, the authors shows that the PoA is at least 1.7 (since the NE packing is the output of a First Fit algorithm) and they give an example where the PoA is exactly 1.7, concluding that their bound is tight for the egalitarian setting. In [24], Dòsa and Epstein gives further results about this setting. They shows that SPoA is equal to 1.69103, since Next Fit Increasing creates the worst possible strong Nash equilibrium. Moreover, in this case the result is not tight, since the SPoS is lower, and its value is approximately 1.611824.

The general weight games have been addressed by Dòsa and Epstein ([24]). Firstly, they show that any game with general weights admit a Nash equilibrium and they show that those games converge to Nash equilibria. Then, they consider several measures of types of Nash equilibria: strong Nash equilibria but also strictly Pareto optimal equilibria and weakly Pareto optimal equilibria. They prove that any game of this class admits all these types of equilibria. Finally, they show that the case of general weights is strongly related to the First Fit algorithm (see Section 2.3.1), and all the four PoA values are equal to 1.7. Regarding the PoS, in any game of this class the values are equal to 1, except for those of strong equilibria, which is equal to 1.7.

[2]A Strong Nash equilibrium is a packing where there exists no subset of agents, all agents in which can profit from jointly moving their items to different bins.

Chapter 3

Multi-Agent Coverage Problems

3.1 Model and Preliminaries

In the *multi-agent coverage problem* we are given a set A of k autonomous agents, a set X of n elements, and a set $\mathcal{S}$ of m containers, where, for any $S \in \mathcal{S}$, $S \subseteq X$. We denote as $c_S \in \mathbb{R}^+$ the cost of $S \in \mathcal{S}$, and as $u_x \in \mathbb{R}^+$ the utility of the element $x \in X$. Given any $Y \subseteq X$, we define the total utility of elements belonging to Y as $U_Y = \sum_{x \in Y} u_x$. We are also given a set of k budgets $B = \{b_1, \ldots, b_k\}$, where b_i is the budget of agent i. An outcome of the problem is a vector $O = (O_1, \ldots, O_k)$ where, for any $i = 1, \ldots, k$, the strategy $O_i \subseteq \mathcal{S}$ of agent i is such that $\sum_{S \in O_i} c_S \leq b_i$. That is, each agent buys a subset of elements of $\mathcal{S}$ which overall cost is at most her own budget (sometimes we say that a container is *assigned* to an agent). We say that an agent i *covers* an element $x \in X$ if there exists a container $S \in O_i$ such that $x \in S$. Moreover, we define as $C_i(O) = \bigcup_{S \in O_i} S$ the set of elements covered by agent i in the outcome O. We call $p_x(O)$ the number of agents that cover the element $x \in X$ in the outcome O, i.e., $p_x(O) = |\{i \in A : x \in C_i(O)\}|$. Furthermore, for any element $x \in X$ and agent $i \in A$, we denote by $t_x^i(O)$ the number of times that a container containing x is bought by i in the outcome O, formally, $t_x^i(O) = |\{S \in O_i : x \in S\}|$. We denote the overall number of times that a container containing x is bought in the outcome O as $t_x(O) = \sum_{i=1}^{k} t_x^i(O)$.

We consider two different settings which differ only in the definition of the revenue of each agent. In the first setting, that we call the *distributed* multi-agent coverage problem (DMCP), given an outcome O, the utility of each element is equally split among all the agents covering it in O. Formally, the revenue of agent i in the outcome O is $r_i^D(O) = \sum_{x \in C_i(O)} \frac{u_x}{p_x(O)}$.

In the second setting, that we call *proportional* multi-agent coverage problem (PMCP), given an outcome O, the utility of each element is equally split among all the containers

covering it in O, that is, for each element $x \in C_i(O)$, the agent i gets revenue equal to $\frac{t_x^i(O)}{t_x(O)} u_x$. The overall revenue of agent i in the outcome O is defined as

$$r_i^P(O) = \sum_{x \in C_i(O)} \frac{t_x^i(O)}{t_x(O)} u_x = \sum_{S \in O_i} \sum_{x \in S} \frac{u_x}{t_x(O)}.$$

For this problem we consider two different scenarios. The first one is the classical centralized approach where the objective is computing an outcome that maximizes the sum of the revenues of the agents which we call the *total revenue*. We remark that, since the agents are autonomous, they have to satisfy their own budget constraints. We also observe that the sum of the player's revenue is the same for DMCP and PMCP. In fact, for any outcome O, we have that $\sum_{i \in A} r_i^D(O) = \sum_{i \in A} r_i^P(O)$, which corresponds to the sum of the utilities of all the elements bought by at least one agent. We denote this sum as $R(O)$.

As we introduced in Section 2.2.3, there are several relevant works regarding the field. We emphasize that our centralized multi-agent setting is not captured by any research in the literature. The problem considered by Goel et al. ([10–12]) differs from our setting, since here agents have budget constraints to satisfy. Moreover, we have that a single element can belong to more than one container. Finally, we are interested in maximizing the sum of the utilities of the covered elements. We notice that our problem also differs from the one considered in [13]. In fact, in our setting, we have bins and a single element can belong to more than one container, i.e., an element can give revenue to more than one agent. Moreover, we have costs for the containers and utilities for the elements. Our setting differs from the one presented by Vondrak ([14]): our problem is different because we have bins with their own costs, a single element can belong to more than one bin, and agents have a budget constraint to satisfy. With respect to the setting presented by Chekuri and Kumar ([38]) we notice that the cost version of MCG cannot be used to model our setting since the assumption is that the groups are disjoint. As shown in [54], if the sets in the groups are not disjoint the MCG problem cannot be approximated within any constant factor. Finally, our multi-agent coverage problem is a special case of the general covering problem ([52]). Indeed, for each agent i, S_i can be defined has the different subsets of elements that can be covered by using budgets b_i. However, notice that S_i can be exponential in $|X|$, which implies that the centralized algorithm proposed in [52] is not polynomial in our setting.

The second scenario considers distributed multi-agent coverage games (DMCG) and the proportional multi-agent coverage games (PMCG) where autonomous and selfish agents are concerned about maximizing their own revenue. Formally, a distributed multi-agent coverage game or a proportional multi-agent coverage game $\mathcal{G} = (A, X, \mathcal{S})$ is defined by

the set A of k agents, the set X of n elements, and the set $\mathcal{S}$ of m containers. Let (O_{-i}, O'_i) denote the outcome O' obtained from O by changing the strategy of agent i from O_i to O'_i. In DMCG (PMCG, resp.), given an outcome $O = (O_1, \ldots, O_k)$, an *improving deviation* of agent i in the outcome O is a strategy O'_i such that $r_i^D((O_{-i}, O'_i)) > r_i^D(O)$ ($r_i^P((O_{-i}, O'_i)) > r_i^P(O)$, resp.). An outcome is a pure Nash equilibrium (NE) for $\mathcal{G}$ if and only if no agent can perform an improving deviation. Formally, an outcome O is a NE for DMCG (PMCG, resp.) if $r_i^D(O) \geq r_i^D((O_{-i}, O'_i))$ ($r_i^P(O) \geq r_i^P((O_{-i}, O'_i))$, resp.) for any possible strategy O'_i and for any agent $i \in A$.
We notice that DMCG and PMCG in general yield two different games. However, when all the agents have the same budget b and all the containers have the same cost b, the two games are equivalent, since each player can only choose exactly one container. The distributed setting of our multi-agent coverage game can be modeled via a distributed welfare game with anonymous resource specific welfare function([55]). We report the results in the next sections and show some relevant proofs. To our knowledge we are the first to consider proportional multi-agent coverage games.

3.2 Centralized case

In this section we address the centralized case where we look for an outcome O that maximizes the total revenue $R(O) = \sum_{i \in A} r_i^D(O) = \sum_{i \in A} r_i^P(O)$, while satisfying the agents' budget constraints.

We first introduce the case of a single agent, which corresponds to the budgeted maximum coverage problem, and that can be optimally approximated to within a constant factor. We then show how to exploit the algorithm for single agent to obtain a constant factor approximation for the case in which the maximum cost of a container is upper-bounded by the minimum budget of an agent (i.e. the case in which each agent in A can buy any container in $\mathcal{S}$). Finally, we give a randomized algorithm for the general case that guarantees a constant factor approximation in expectation.

3.2.1 Single-agent case

The single-agent case corresponds to the budgeted maximum coverage problem for which an algorithm that guarantees a $1 - \frac{1}{e}$ approximation factor has been proposed in [8]. The problem generalizes the maximum coverage problem which is known to be *NP*-hard to approximate to within a factor greater than $1 - \frac{1}{e}$, therefore the algorithm in [8] is optimal from the approximation point of view.

Algorithm 1: Greedy algorithm for single-agent case.

Input : $\mathcal{S}, X, b_1$

1 $O_1 := \emptyset$; $X' := X$; $\mathcal{S}' := \mathcal{S}$;
2 **repeat**
3 Select $S' \in \mathcal{S}$ that maximizes $\frac{\sum_{x \in S' \cap X'} u_x}{c_{S'}}$;
4 **if** $\sum_{S \in O_1} c_S + c_{S'} \leq b_1$ **then**
5 $O_1 := O_1 \cup \{S'\}$;
6 $X' := X' \setminus S'$;
7 $\mathcal{S}' := \mathcal{S}' \setminus \{S'\}$
8 **until** $\mathcal{S}' = \emptyset$;
9 **return** $O = (O_1)$;

Two algorithms are given in [8]: the former one is a greedy algorithm that achieves an approximation factor of $1 - \frac{1}{\sqrt{e}}$; the latter one improves the approximation factor to $1 - \frac{1}{e}$ by first guessing three containers that are contained in an optimal solution, and then completing the solution with the greedy algorithm.

In the following we describe the greedy algorithm and show a property that will be used in the next section to prove an approximation bound on the multi-agent case. The pseudo-code is reported in Algorithm 1. The algorithm starts with an empty solution O_1 and, at each iteration selects the container S' that maximizes the ratio between the utility of the elements in S' that are not yet covered and the cost of S' (line 3). If the sum of the cost of S' and that of the solution computed so far does not exceed the budget b_1, then S' is added to O_1 (lines 4–5).

Let O^* be an optimal solution. Let h be the number of iterations of Algorithm 1 until the first container is not added to O_1 because it violates the budget constraint (i.e. either $h+1$ is the first iteration in which the condition at line 4 is not satisfied, or all the containers in $\mathcal{S}$ are added to O_1). For each $j = 1, \ldots, h$, let S_j be the container selected at iteration j and O_1^j be the set of containers O_1 computed at the end of iteration j. The next lemma is used in [8] to show the approximation ratio of Algorithm 1.

Lemma 3.1. *[8] For each $j = 1, \ldots, h$, the following holds:*

$$R(O_1^j) \geq \left[1 - \prod_{\ell=1}^{j} \left(1 - \frac{c_{S_\ell}}{b_1}\right)\right] R(O^*).$$

The following proposition will be used in the next section to prove an approximation guarantee in the multi-agent case.

Proposition 3.2. *Let O^* be an optimal solution for the single-agent case and let $\alpha \in [0, 1]$. For each $j = 1, \ldots, h$, if the agent spends at least αb_1 budget by iteration j, then $R(O_1^j) \geq \left(1 - \frac{1}{e^\alpha}\right) R(O^*)$.*

Algorithm 2: Algorithm for the case $c_{\max} \leq b_{\min}$.

Input : $\mathcal{S}, X, B$

Run Algorithm 1 with input $(\mathcal{S}, X, \sum_{i=1}^{k} b_i)$;
Let $O'_1 = \{S_1, \ldots, S_g\}$ be the output of Algorithm 1, where S_j is the container chosen at iteration j, for $j = 1, \ldots, g$;
$O_i := \emptyset$, for each $i = 1, \ldots, k$;
for $j = 1, \ldots, g$ **do**
    if There exists an index $i \leq k$ such that $\sum_{S \in O_i} c_S + c_{S_j} \leq b_i$ **then**
        Let i be the smallest index such that $\sum_{S \in O_i} c_S + c_{S_j} \leq b_i$;
        $O_i := O_i \cup \{S_j\}$;
return $O = (O_1, \ldots, O_k)$;

Proof. By hypothesis, $\sum_{S \in O_1^j} c_S \geq \alpha b_1$. Then, by Lemma 3.1, we have:

$$\begin{aligned} R(O) &\geq \left[1 - \prod_{\ell=1}^{j} \left(1 - \frac{c_{S_\ell}}{b_1}\right)\right] R(O^*) \\ &\geq \left[1 - \prod_{\ell=1}^{j} \left(1 - \frac{\alpha c_{S_\ell}}{\sum_{S \in O_1^j} c_S}\right)\right] R(O^*) \\ &\geq \left[1 - \left(1 - \frac{\alpha}{j}\right)^j\right] R(O^*) \\ &\geq \left(1 - \frac{1}{e^\alpha}\right) R(O^*), \end{aligned}$$

where the last two inequalities are due to the following observation (see [1]): For a sequence of numbers $a_1, \ldots, a_n$ such that $\sum_{\ell=1}^{n} a_\ell = A$, the function $\left[1 - \prod_{\ell=1}^{n} \left(1 - \frac{a_\ell \cdot \alpha}{A}\right)\right]$ achieves its minimum when $a_\ell = \frac{A}{n}$ and

$$\left[1 - \prod_{\ell=1}^{n} \left(1 - \frac{a_\ell \cdot \alpha}{A}\right)\right] \geq 1 - \left(1 - \frac{\alpha}{n}\right)^n \geq 1 - e^{-\alpha}.$$

□

3.2.2 Smallest budget greater than or equal to the highest cost

In this section, we consider the multi-agent case in which all the agents have enough budget to be able to buy any container in $\mathcal{S}$. Formally, if $c_{\max} = \max_{S \in \mathcal{S}} c_S$ and $b_{\min} = \min_{i \in A} b_i$, then $c_{\max} \leq b_{\min}$. Without loss of generality, we assume that the agents are sorted in non-increasing order of budget, that is $b_1 \geq b_2 \geq \cdots \geq b_k = b_{\min}$, ties are broken arbitrarily.

We give an algorithm that achieves a $1 - \frac{1}{\sqrt{e}}$ approximation ratio. The algorithm, whose pseudo-code is given in Algorithm 2, first defines an instance of the single-agent case which has the same elements X and containers $\mathcal{S}$, while the budget of the unique agent is equal to $\sum_{i=1}^{k} b_i$, which is the sum of the budgets of all the agents in the multi-agent instance. Algorithm 2 approximately solves this instance by means of Algorithm 1, and then assigns some of the containers returned to the k agents in such a way that the budget constraints are not violated. In detail, let $\{S_1, \ldots, S_g\}$ be the containers returned by Algorithm 1, where S_j is the container chosen at iteration j of Algorithm 1, for $j = 1, \ldots, g$. Iteratively, for each $j = 1, \ldots, g$, the algorithm assigns S_j to the agent i such that i is minimum (i.e. her budget is maximum) and the cost of S_j plus that of the partial solution O_i does not exceed budget b_i, if such an agent exists, otherwise it discards S_j (lines 4–7).

The next theorem shows a constant approximation bound for Algorithm 2. The assumption that $c_{\max} \leq b_k$ allows us to show that the cost of the containers assigned to the agents is at least half of the entire budget $\sum_{i=1}^{k} b_i$. Moreover, these assigned containers satisfy the hypotheses of Proposition 3.2, therefore, we can exploit it with $\alpha = \frac{1}{2}$ to show the statement.

Theorem 3.3. *Let O^* be an optimal solution and O be the solution returned by Algorithm 2, then $R(O) \geq \left(1 - \frac{1}{\sqrt{e}}\right) R(O^*)$.*

Proof. Let us consider the last iteration h of Algorithm 1 for which the budget constraint is not violated (i.e. either $h + 1$ is the first iteration in which the condition at line 4 is not satisfied or Algorithm 1 includes all the containers in O'_1).

We first show that the cost of containers $S_1, \ldots, S_h$ is at least a fraction $1 - \frac{1}{k}$ of the entire budget $\sum_{i=1}^{k} b_i$ or Algorithm 1 includes all the containers in $\mathcal{S}$ to O'_1 by iteration h. Then, we show that there exists a $j \leq h$ such that Algorithm 2 is able to assign to the k agents all the containers $S_1, \ldots, S_j$ and which the overall cost of these containers is at least half of the entire budget used by Algorithm 1. Finally, we exploit Proposition 3.2 to show the statement.

If Algorithm 1 does not include all the containers in $\mathcal{S}$ to O'_1 by iteration h, then at iteration $h + 1$ the budget $\sum_{i=1}^{k} b_i$ is violated, that is $\sum_{j=1}^{h+1} c_{S_j} > \sum_{i=1}^{k} b_i$. Therefore,

$$\sum_{j=1}^{h} c_{S_j} > \sum_{i=1}^{k} b_i - c_{S_{h+1}} \geq \sum_{i=1}^{k} b_i - c_{\max} \geq \sum_{i=1}^{k} b_i - b_k \geq \left(1 - \frac{1}{k}\right) \sum_{i=1}^{k} b_i,$$

where the third inequality is due to the hypothesis that $c_{\max} \leq b_k$, and the last one is implied by the fact that b_k is the minimum budget, which implies that $b_k \leq \frac{1}{k} \sum_{i=1}^{k} b_i$.

Therefore, the cost of $S_1, \ldots, S_h$ is at least a fraction $1 - \frac{1}{k}$ of the entire budget $\sum_{i=1}^{k} b_i$ or Algorithm 1 includes all the containers in $\mathcal{S}$ to O'_1 by iteration h.

We now show that there exists a $j \leq h$ such that Algorithm 2 is able to assign to the k agents all the containers $S_1, \ldots, S_j$ and that the overall cost of these containers is at least half of the entire budget.

If Algorithm 2 is able to assign all the containers $S_1, \ldots, S_h$ to the agents, then either the overall cost of these containers is at least $\left(1 - \frac{1}{k}\right) \sum_{i=1}^{k} b_i \geq \frac{1}{2} \sum_{i=1}^{k} b_i$, for $k \geq 2$, or we found an optimal solution which assigns all the containers in $\mathcal{S}$ to the agents. In this case $j = h$.

Otherwise, let $j < h$ be an index such that containers $S_1, \ldots, S_j$ are all assigned to the agents and $j + 1 \leq h$ is the first iteration of the cycle at lines 4–7 of Algorithm 2 such that the condition at line 5 does not hold, that is S_{j+1} is the first container in the greedy ordering that is not assigned to an agent because $\sum_{S \in O_i} c_S + c_{S_{j+1}} > b_i$, for each $i = 1, \ldots, k$, at iteration $j + 1$. Note that $j \geq 1$ because at least one container is always assigned, assuming that $c_{\max} \leq b_k$.

We show that $\sum_{f=1}^{j} S_f \geq \frac{1}{2} \sum_{i=1}^{k} b_i$. Equivalently, we show that at the beginning of iteration $j + 1$ (at the end of iteration j), $\sum_{i=1}^{k} \sum_{S \in O_i} c_S \geq \frac{1}{2} \sum_{i=1}^{k} b_i$.

Since S_{j+1} is not assigned to any agent and $c_{S_{j+1}} \leq b_i$, for each agent i, then all the agents are assigned with at least one container before iteration $j + 1$, that is $O_i \neq \emptyset$, for each i. Formally, $\sum_{S \in O_i} c_S + c_{S_{j+1}} > b_i$ and $c_{S_{j+1}} \leq b_i$, imply $\sum_{S \in O_i} c_S > 0$, that is $O_i \neq \emptyset$, for each $i = 1, \ldots, k$.

If $\sum_{S \in O_i} c_S \geq \frac{1}{2} b_i$, for each agent i, then the statement holds. Otherwise, let ℓ be the smallest index such that $\sum_{S \in O_\ell} c_S < \frac{1}{2} b_\ell$. We analyze two cases:

- If $\ell = k$, then the containers assigned to O_k in previous iterations have not been assigned to any agent $i < k$, which implies that

$$\sum_{S \in O_i} c_S + \sum_{S \in O_k} c_S > b_i \geq \frac{1}{2}(b_i + b_k).$$

  This holds in particular for agent $k-1$. For any other agent $i < k-1$, by hypothesis we have $\sum_{S \in O_i} c_S \geq \frac{1}{2} b_i$. Therefore, the overall cost of assigned containers is then

$$\sum_{i=1}^{k-2} \sum_{S \in O_i} c_S + \sum_{S \in O_{k-1}} c_S + \sum_{S \in O_k} c_S \geq \sum_{i=1}^{k-2} \frac{1}{2} b_i + \frac{1}{2}(b_{k-1} + b_k) = \frac{1}{2} \sum_{i=1}^{k} b_i.$$

- If $\ell < k$, let us split the agents different from ℓ into two groups according to their ordering. For each agent $i > \ell$, the containers assigned to O_i in previous iterations have not been assigned to agent ℓ, which implies that

$$\sum_{S \in O_\ell} c_S + \sum_{S \in O_i} c_S > b_\ell.$$

From $\sum_{S \in O_\ell} c_S < \frac{1}{2} b_\ell$, follows that

$$\sum_{S \in O_i} c_S > b_\ell - \sum_{S \in O_\ell} c_S > \frac{1}{2} b_\ell \geq \frac{1}{2} b_i.$$

In particular, let us consider agent $i = \ell + 1$ (note that this agent always exists, as $\ell < k$). From $\sum_{S \in O_\ell} c_S + \sum_{S \in O_i} c_S > b_\ell$ and $b_\ell \geq b_{\ell+1}$ follows that

$$\sum_{S \in O_\ell} c_S + \sum_{S \in O_{\ell+1}} c_S > \frac{1}{2}(b_{\ell+1} + b_\ell).$$

Finally, for any agent $i < \ell$, by hypothesis we have $\sum_{S \in O_i} c_S \geq \frac{1}{2} b_i$. Therefore, the overall cost of the assigned containers is

$$\begin{aligned} &\sum_{i=1}^{\ell-1} \sum_{S \in O_i} c_S + \sum_{S \in O_\ell} c_S + \sum_{S \in O_{\ell+1}} c_S + \sum_{i=\ell+2}^{k} \sum_{S \in O_i} c_S \\ &\geq \sum_{i=1}^{\ell-1} \frac{1}{2} b_i + \frac{1}{2}(b_\ell + b_{\ell+1}) + \sum_{i=\ell+2}^{k} \frac{1}{2} b_i = \frac{1}{2} \sum_{i=1}^{k} b_i. \end{aligned}$$

Let $O_1'' = \{S_1, \ldots, S_j\}$. We have just proved that the cost of O_1'' is a fraction $\frac{1}{2}$ of the budget given to Algorithm 1. Therefore, we can apply Proposition 3.2 with $\alpha = \frac{1}{2}$ to obtain $R(O_1'') \geq \left(1 - \frac{1}{\sqrt{e}}\right) R(O^{**})$. Where O^{**} is an optimal solution to the single-agent instance that has elements X, containers $\mathcal{S}$, and budget $\sum_{i=1}^{k} b_i$. Since this is a relaxation of the original multi-agent instance in which the assignment constraints are relaxed, then $R(O^{**}) \geq R(O^*)$. As Algorithm 2 assigns all the containers in O_1'' to the k agents, the statement holds.

□

3.2.3 Randomized algorithm for the general case

In this section, we give a randomized algorithm for the general case that returns a solution that satisfies the budget contraints in expectation and achieves an expected

Algorithm 3: Randomized algorithm for the general case.

Input : $\mathcal{S}, X, B$

1 Solve the relaxation of (IP) where Constr. (3.4) are replaced with $y_x, z_S^i \in [0,1]$, for $x \in X, (S,i) \in D$, and let (y^*, z^*) be an opt. fractional solution;

2 For each $S \in \mathcal{S}$, independently, select at most one variable z_S^i, $(S,i) \in D$, to be set to 1. Each variable is selected with probability z_S^{i*} and no variable is selected with probability $1 - \sum_{(S,i)\in D} z_S^{i*}$;

3 Let $O = (O_1, \ldots, O_k)$, where $O_i = \{S \mid z_S^i = 1\}$;

4 **return** $O = (O_1, \ldots, O_k)$;

approximation ratio of $1 - \frac{1}{e}$. We leave open the deterministic approximation for the general case.

We start by defining an integer program for the general multi-agent case. Let D be all the pairs of a single container and a single agent that satisfies the budget constraint, that is, $D := \{(S,i) \in \mathcal{S} \times A \mid c_S \leq b_i\}$. We define two types of binary variables: y_x indicates whether element x is covered by any agent, for each $x \in X$, and z_S^i, indicates whether container S is assigned to agent i, for each $(S,i) \in D$.

The integer program is then as follows.

$$\max \sum_{x\in X} u_x y_x \tag{IP}$$

$$\sum_{(S,i)\in D} c_S z_S^i \leq b_i \qquad \text{for } i \in A \tag{3.1}$$

$$\sum_{(S,i)\in D} z_S^i \leq 1 \qquad \text{for } S \in \mathcal{S} \tag{3.2}$$

$$\sum_{S:x\in S} \sum_{(S,i)\in D} z_S^i \geq y_x \qquad \text{for } x \in X \tag{3.3}$$

$$y_x, z_S^i \in \{0,1\} \qquad \text{for } x \in X, (S,i) \in D \tag{3.4}$$

Constraints (3.1) guarantee that the cost of the containers assigned to an agent does not exceed her budget. Constraints (3.2) guarantee that each container is assigned to at most one agent. Note that this is not required by the problem definition, however, in the centralized setting, any optimal solution satisfy this condition. Indeed, for any solution that does not satisfy this condition it is possible to define another solution that satisfy it and that does not decrease the value of the objective function. Therefore we can add this constraint without loss of generality. Constraints (3.3), guarantee that, if an element x is covered (i.e. $y_x = 1$), then there exists at least a container S that contains x and that is assigned to some agent i (i.e. $z_S^i = 1$, for some $(S,i) \in D$ such that $x \in S$).

The randomized algorithm is reported in Algorithm 3. It first solves the linear relaxation of (IP) where the integrality constraints are replaced with bounds between 0 and 1 on the variables. Let (y^*, z^*) be an optimal fractional solution of this relaxation. To round this fractional solution to binary values, variables z_S^i are interpreted as probabilities. In detail, let us consider each container $S \in \mathcal{S}$, independently. Among all variables z_S^i, $(S,i) \in D$, we set at most a variable to 1 and all the other ones to 0. The probability that z_S^i is set to 1 is proportional to the value of the optimal fractional variable z_S^{i*}, while the probability that all variables z_S^i, $(S,i) \in D$, are set to 0 is $1 - \sum_{(S,i)\in D} z_S^{i*}$. Eventually, Algorithm 3 defines O_i as $O_i = \{S \mid z_S^i = 1\}$.

By the above process, we have that the probability that a container S belongs to O_i, for each $(S,i) \in D$, is $\mathbf{P}[S \in O_i] = \mathbf{P}[z_S^i = 1] = z_S^{i*}$; the probability that a container $S \in \mathcal{S}$ is assigned to some agent is equal to $\mathbf{P}[S \in \bigcup_{(S,i)\in D} O_i] = \sum_{(S,i)\in D} z_S^{i*}$, and the probability that S is not assigned to any agent is $\mathbf{P}[S \notin \bigcup_{(S,i)\in D} O_i] = 1 - \sum_{(S,i)\in D} z_S^{i*}$.

The next lemma shows that the solution $O = (O_1, \ldots, O_k)$ returned by Algorithm 3 satisfies the budget constraints in expectation.

Lemma 3.4. *For each agent i, $\mathbf{E}[\sum_{S\in O_i} c_S] \leq b_i$.*

Proof. $\mathbf{E}[\sum_{S\in O_i} c_S] = \sum_{(S,i)\in D} c_S \mathbf{P}[S \in O_i] = \sum_{(S,i)\in D} z_S^{i*} \leq b_i$, where the last inequality is due to Constraint (3.1). □

The solution $O = (O_1, \ldots, O_k)$ returned by Algorithm 3 satisfies the budget constraints in expectation.

The following theorem shows the expected approximation ratio of Algorithm 3.

Theorem 3.5. *Let O^* be an optimal solution and O be the solution returned by Algorithm 3, then $\mathbf{E}[R(O)] \geq \left(1 - \frac{1}{e}\right) R(O^*)$.*

Proof. For each $x \in X$, let us denote by w_x the random binary variable that is equal to 1 if element x is covered. The probability that $w_x = 1$ is equal to the probability that x belongs to $\cup_{i\in A} C_i(O)$, that is,

$$\mathbf{P}[w_x = 1] = \mathbf{P}[x \in \cup_{i\in A} C_i(O)] = 1 - \mathbf{P}[x \notin \cup_{i\in A} C_i(O)].$$

The probability that x does not belong to $\cup_{i\in A} C_i(O)$ is equal to the probability that each container S that contains x is not selected by Algorithm 3. Since containers are

selected independently, this is equal to

$$\mathbf{P}[x \notin \cup_{i\in A} C_i(O)] = \prod_{S:x\in S} \mathbf{P}[S \notin \bigcup_{(S,i)\in D} O_i] = \prod_{S:x\in S} \left(1 - \sum_{(S,i)\in D} z_S^{i*}\right).$$

Since $1 - p \leq e^{-p}$, for any $p \in [0,1]$ and $\sum_{(S,i)\in D} z_S^{i*} \in [0,1]$ by Constraint (3.2), then this value is equal to

$$\prod_{S:x\in S} \exp\left(-\sum_{(S,i)\in D} z_S^{i*}\right) = \exp\left(-\sum_{S:x\in S}\sum_{(S,i)\in D} z_S^{i*}\right).$$

By Constraint (3.3), $\sum_{S:x\in S}\sum_{(S,i)\in D} z_S^{i*} \geq y_x^*$ hence $\mathbf{P}[x \notin \cup_{i\in A} C_i(O)] \leq \exp(-y_x^*)$. Therefore,

$$\mathbf{P}[w_x = 1] \geq 1 - e^{-y_x^*} \geq \left(1 - \frac{1}{e}\right) y_x^*,$$

where the last inequality is due to the fact that $1 - e^{-p} \geq (1 - e^{-1})p$, for any $p \in [0,1]$, [1] and $y_x^* \in [0,1]$.

The expected value of $R(O)$ is equal to

$$\mathbf{E}[R(O)] = \mathbf{E}\left[\sum_{x\in X} u_x w_x\right] = \sum_{x\in X} u_x \mathbf{P}[w_x = 1] \geq \sum_{x\in X} u_x \left(1 - \frac{1}{e}\right) y_x^* \geq \left(1 - \frac{1}{e}\right) R(O^*),$$

since $\sum_{x\in X} u_x y_x^*$ is the optimum value for the linear relaxation of IP.

□

3.3 Game theoretical setting

In this section we consider the strategic versions of the multi-agent coverage problems. We first focus on the distributed multi-agent coverage games (DMCG) and then discuss the proportional multi-agent coverage games (PMCG).

3.3.1 Distributed multi-agent coverage games

We recall that in DMCG the utility of an element is equally shared among all the agents that are covering it. In 2013 Marden and Wierman introduced a class of games called Distributed Welfare Games ([55]). One of the settings they proposed, is called the

[1] To see this, observe that function $1 - e^{-p}$ is concave for $p > 0$ and it is equal to 0 when $p = 0$ and to $1 - e^{-1}$ when $p = 1$.

distributed welfare game with anonymous resource specific welfare function, can be used to model DMCG games, then our setting shares the same results as theirs. As shown in their paper (Proposition 1), the DMCG are convergent and the existence of a Nash equilibria is always guaranteed. DMCG are potential games [64], that is, they admit a potential function ϕ which assigns to each possible outcome a non-negative real value, such that, from any outcome O, if a single agent i performs an improving deviation leading the dynamics to the new outcome O', then the change in the potential value $\phi(O') - \phi(O)$ is exactly the change in the revenue of agent i, which is $r_i^D(O') - r_i^D(O)$. This implies that any improving deviation increases the value of ϕ in a proportional way. Thus, since our games are finite, after a finite number of improving deviations the dynamics reaches a Nash equilibrium.

Theorem 3.6. *[55] DMCG are potential games.*

Proof. Given any outcome O, we define the potential function ϕ as follows:

$$\phi(O) = \sum_{x \in X} \sum_{h=1}^{p_x(O)} f_x(h)$$

where $f_x(h) = \frac{u_x}{h}$.

Consider any agent i that can perform an improving deviation in the outcome O. The change in the potential value only concerns the new elements covered by agent i in the outcome O', and the ones that agent i were covering in O and that she is not covering anymore in O'. Formally:

$$\begin{aligned}
\phi(O') - \phi(O) &= \sum_{x \in X} \sum_{h=1}^{p_x(O')} f_x(h) - \sum_{x \in X} \sum_{h=1}^{p_x(O)} f_x(h) \\
&= \sum_{x \in C_i(O')} \frac{u_x}{p_x(O')} - \sum_{x \in C_i(O)} \frac{u_x}{p_x(O)} \\
&= r_i^D(O') - r_i^D(O).
\end{aligned}$$

□

We now analyze the performance of Nash equilibria in DMCG. We start by showing an upper bound to the price of anarchy. We remark that the same result can be found in [55] and holds for a more general case. Anyway, we show our proof which is simpler in our setting with respect to the one used by Marden and Wierman.

Theorem 3.7. *[55] The price of anarchy of DMCG is at most* $2 - \frac{1}{k}$.

Proof. Fix a distributed multi-agent coverage game $\mathcal{G}$ and a pair of outcomes $O^* = \{O_1^*, \ldots, O_k^*\}$ and $O = \{O_1, \ldots, O_k\}$ such that O^* is an optimum and O is a Nash equilibrium for $\mathcal{G}$, respectively. Let us define $C(O) = \bigcup_{i=1,\ldots,k} C_i(O)$ as the set of elements covered in the outcome (Nash equilibrium) O, and with $\overline{C}(O) = X \setminus C(O)$ the set of elements not covered by O.
For any agent $i = 1, \ldots, k$, let us partition the elements that i covers in the optimum. In particular, let $\hat{C}_i(O^*) = C_i(O^*) \cap C(O)$ be the set of elements covered by agent i in the optimum and also covered in the Nash equilibrium O, and let $\overline{C}_i(O^*) = C_i(O^*) \cap \overline{C}(O)$ be the set of elements covered by agent i in the optimum and not covered in the Nash equilibrium O. It is easy to see that, for any $i = 1, \ldots, k$, $C_i(O^*) = \hat{C}_i(O^*) \cup \overline{C}_i(O^*)$ and $\hat{C}_i(O^*) \cap \overline{C}_i(O^*) = \emptyset$.
Let $C(O^*) = \bigcup_{i=1,\ldots,k} C_i(O^*)$, $\hat{C}(O^*) = \bigcup_{i=1,\ldots,k} \hat{C}_i(O^*)$, and $\overline{C}(O^*) = \bigcup_{i=1,\ldots,k} \overline{C}_i(O^*)$. We notice that the optimum total revenue is such that

$$R(O^*) = \sum_{i=1}^{k} r_i^D(O^*) = U_{C(O^*)} = U_{\overline{C}(O^*)} + U_{\hat{C}(O^*)}.$$

Since O is a Nash equilibrium, it holds that, for any $i = 1, \ldots, k$, the revenue of agent i in O is such that:

$$r_i^D(O) \geq r_i^D((O_{-i}, O_i^*)) \geq U_{\overline{C}_i(O^*)} + \frac{U_{\hat{C}_i(O^*)}}{k},$$

because in the outcome (O_{-i}, O_i^*), the elements in $\overline{C}_i(O^*)$ are covered only by agent i and the elements in $\hat{C}_i(O^*)$ are covered by at most k agents. We get that

$$\begin{aligned} R(O) &= \sum_{i=1}^{k} r_i^D(O) \\ &\geq \left(U_{\overline{C}(O^*)} + \frac{U_{\hat{C}(O^*)}}{k} \right) \\ &= U_{\overline{C}(O^*)} + U_{\hat{C}(O^*)} - \frac{k-1}{k} U_{\hat{C}(O^*)} \\ &\geq R(O^*) - \frac{k-1}{k} R(O), \end{aligned}$$

where the last inequality holds because $\hat{C}(O^*) \subseteq C(O)$. It follows that $\left(2 - \frac{1}{k}\right) R(O) \geq R(O^*)$, and the theorem holds. □

As shown in [55](Example 2), the price of stability of DMCG is at least $2 - \frac{1}{k} - \epsilon$, for any (small) $\epsilon > 0$, even for very simple instances where the budgets and the cost of the containers are all equal to 1 and each element is contained in exactly one container. We

notice that this result is tight due to the upper bound to the price of anarchy proved in Theorem 3.7.

Theorem 3.8. *[55] The price of stability of DMCG is at least* $2 - \frac{1}{k} - \epsilon$*, for any (small)* $\epsilon > 0$.

3.3.2 Proportional multi-agent coverage game

We now consider the proportional multi-agent coverage game (PMCG), where the utility of an element is equally shared among all the subsets covering it. For this setting we show that the existence of a Nash equilibrium is not guaranteed even for a very simple instance with only 2 agents with identical budget and where all the containers have cost 1.

Theorem 3.9. *The existence of a NE for PMCG is not guaranteed.*

Proof. Consider the following instance $\mathcal{G}$ of PMCG with only two agents (i.e., $|A| = 2$) who we call i and j, and five elements $X = \{x_1, x_2, x_3, x_4, x_5\}$ with utilities $u_1 = 9, u_2 = 6, u_3 = 7, u_4 = 6, u_5 = 8$, respectively. Moreover, we have four containers $\mathcal{S} = \{A, B, C, D\}$, all of cost 1, where $A = \{x_1, x_2, x_4\}$, $B = \{x_2, x_3\}$, $C = \{x_1, x_5\}$ and $D = \{x_2, x_4, x_5\}$. Finally, $b_1 = b_2 = 2$. A graphical description of the containers and elements of the game $\mathcal{G}$ is reported in the Figure 3.1.

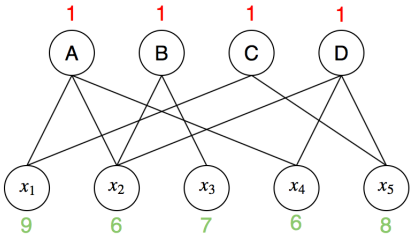

FIGURE 3.1: An instance of PMCG admitting no Nash equilibrium.

We will show that, for any possible outcome of $\mathcal{G}$, there exists at least one agent that can perform an improving deviation. It implies that $\mathcal{G}$ does not admit any Nash equilibrium. We begin by noticing that both agents are willing to use their entire budget by buying exactly two containers. It implies that any outcome where an agent buys only one container can not be a Nash equilibrium since she can increase her revenue by buying a second one. Clearly this is true independently from the other agent's strategy. As a consequence, we only need to consider all the possible outcomes where each agent buys exactly two containers. We first show that any outcome where an agent i chooses a strategy of the form $O_i = \{S, S\}$, for any $S \in \mathcal{S}$, is not a NE (the same is true for agent j). In particular, we prove that, independently from the other agent's strategy, moving

from the strategy O_i to $O'_i = \{A, D\}$ is always an improving deviation. In the following we assume that O (resp. O') is the outcome formed by O_i (resp. O'_i) for agent i and an arbitrary fixed strategy for agent j. Formally, we want to show that $r_i^P(O') > r_i^P(O)$. We distinguish among four different cases.

- $O_i = \{A, A\}$. In this case we have that $t^i_{x_2}(O) = t^i_{x_2}(O')$ and $t^i_{x_4}(O) = t^i_{x_4}(O')$. Therefore, x_2 and x_4 give the same revenue to agent i in both outcomes, and we ignore them in our analysis. In O' the agent i gets revenue at least $\frac{u_5}{3}$ from x_5. In the outcome O we have that $t^i_{x_1}(O) = 2$ and $0 \leq t^j_{x_1}(O) \leq 2$. If in O the agent i gets revenue $\frac{u_1}{2}$ from x_1 (that is $t^j_{x_1}(O) = 2$), since $\frac{u_1}{2} < \frac{u_1}{3} + \frac{u_5}{3}$, we have that $r_i^P(O') > r_i^P(O)$. If in O the agent i gets revenue $\frac{2}{3}u_1$ from x_1, since $\frac{2}{3}u_1 < \frac{u_1}{2} + \frac{u_5}{3}$, it holds that $r_i^P(O') > r_i^P(O)$. Finally, if in O the agent i is the only one who covers x_1, obviously $u_1 < u_1 + \frac{u_5}{3}$ and thus $r_i^P(O') > r_i^P(O)$.

- $O_i = \{B, B\}$. Also in this case $t^i_{x_2}(O_i) = t^i_{x_2}(O'_i)$ (and we ignore x_2). In O' the agent i gets revenue at least $\frac{u_1}{3}$, $\frac{u_4}{3}$ and $\frac{u_5}{3}$ from x_1, x_4 and x_5, respectively. We conclude that $r_i^P(O') > r_i^P(O)$ because $\frac{u_1}{3} + \frac{u_4}{3} + \frac{u_5}{3} > u_3$.

- $O_i = \{C, C\}$. In this case, in O', the agent i gets revenue at least $\frac{u_2}{2}$ and $\frac{u_4}{2}$ from x_2 and x_4, respectively. If in O the agent i gets revenue $\frac{u_1}{2}$ and $\frac{u_5}{2}$ from x_1 and x_5 respectively, we have $\frac{u_1}{2} + \frac{u_5}{2} < \frac{u_1}{3} + \frac{u_5}{3} + \frac{u_2}{2} + \frac{u_4}{2}$. If she gets revenue $\frac{2}{3}u_1 + \frac{2}{3}u_5$ from x_1 and x_5 respectively, it holds that $\frac{2}{3}u_1 + \frac{2}{3}u_5 < \frac{u_1}{2} + \frac{u_2}{2} + \frac{u_4}{2} + \frac{u_5}{2}$. Finally, if i is the only agent who covers x_1 and x_5, we have that $u_1 + u_5 < u_1 + u_5 + \frac{u_2}{2} + \frac{u_4}{2}$. In all the cases it holds that $r_i^P(O') > r_i^P(O)$.

- $O_i = \{D, D\}$. As in the first previous case, we have that $t^i_{x_2}(O) = t^i_{x_2}(O')$ and $t^i_{x_4}(O) = t^i_{x_4}(O')$ (and we ignore them). In O' the agent i gets revenue at least $\frac{u_1}{3}$ from x_1. In the outcome O we have that $t^i_{x_5}(O) = 2$ and $0 \leq t^j_{x_5}(O) \leq 2$. If in O the agent i gets revenue $\frac{u_5}{2}$ from x_5 (that is $t^j_{x_5}(O) = 2$), since $\frac{u_5}{2} < \frac{u_1}{3} + \frac{u_5}{3}$, we have that $r_i^P(O') > r_i^P(O)$. If in O the agent i gets revenue $\frac{2}{3}u_5$ from x_5, since $\frac{2}{3}u_5 < \frac{u_1}{3} + \frac{u_5}{2}$, it holds that $r_i^P(O') > r_i^P(O)$. Finally, if in O the agent i is the only one who covers x_5, obviously $u_5 < u_5 + \frac{u_1}{3}$ and thus $r_i^P(O') > r_i^P(O)$.

We proved that any outcome where an agent buys only one container or two identical containers is not a Nash equilibria. It remains to consider any other possible outcome where each of the two agents chooses a pair of different containers. In Table 3.1 we provide an enumeration of each possible outcome of this kind. In the table we show that for any outcome there exists at least an improving deviation for an agent, and this concludes the proof.

□

ID	$O = (O_i, O_j)$	$r_i^P(O)$	$r_j^P(O)$	**ID of improving deviation**
1	({A,B},{A,C})	17	19	2
2	({A,B},{A,D})	16,5	19,5	31
3	({A,B},{B,C})	18	18	2
4	({A,B},{B,D})	18,5	18,5	2
5	({A,B},{C,D})	18,5	17,5	2
6	({A,C},{A,D})	14	15	8
7	({A,C},{B,C})	19	17	8
8	({A,C},{B,D})	18	18	4
9	({A,C},{C,D})	$14,\overline{6}$	$14,\overline{3}$	5
10	({A,D},{B,C})	18,5	17,5	35
11	({A,D},{B,D})	20	16	10
12	({A,D},{C,D})	$15,\overline{1}$	$13,\overline{8}$	10
13	({B,C},{B,D})	18,5	17,5	14
14	({B,C},{C,D})	$17,\overline{1}$	$18,\overline{8}$	5
15	({B,D},{C,D})	$16,\overline{6}$	$19,\overline{3}$	14
16	({A,B},{A,B})	14	14	1
17	({A,C},{A,C})	14,5	14,5	1
18	({A,D},{A,D})	14,5	14,5	2
19	({B,C},{B,C})	15	15	3
20	({B,D},{B,D})	13,5	13,5	4
21	({C,D},{C,D})	14,5	14,5	5
22	({A,C},{A,B})	19	17	23
23	({A,D},{A,B})	19,5	16,5	10
24	({B,C},{A,B})	18	18	23
25	({B,D},{A,B})	18,5	18,5	23
26	({C,D},{A,B})	17,5	18,5	23
27	({A,D},{A,C})	15	14	29
28	({B,C},{A,C})	17	19	29
29	({B,D},{A,C})	18	18	25
30	({C,D},{A,C})	$14,\overline{3}$	$14,\overline{6}$	29
31	({B,C},{A,D})	17,5	18,5	28
32	({B,D},{A,D})	16	20	31
33	({C,D},{A,D})	$13,\overline{8}$	$15,\overline{1}$	31
34	({B,D},{B,C})	17,5	18,5	32
35	({C,D},{B,C})	$18,\overline{8}$	$17,\overline{1}$	26
36	({C,D},{B,D})	$19,\overline{3}$	$16,\overline{6}$	26

TABLE 3.1: All the remaining possible outcomes and the corresponding improving deviations.

Chapter 4

Generalized budgeted Submodular set function maximization

4.1 Model and Preliminaries

We are given a set X of n elements and a set S of m containers. Let us denote the cost of a container $s \in S$ by $c(s) \in \mathbb{R}_{\geq 0}$. For each container $s \in S$ and element $x \in X$, we denote by $c(s, x)$ the cost of associating x to s. Given a budget $k \in \mathbb{R}_{\geq 0}$, and a monotone submodular function $f : 2^X \to \mathbb{R}_{\geq 0}$[1], our goal is to find a subset X' of X and a subset $S' \neq \emptyset$ of S such that $c(S', X') = \sum_{s \in S'} c(s) + \sum_{x \in X'} \min_{s \in S'} c(s, x) \leq k$, and $f(X')$ is maximum. We call this problem the Generalized Budgeted Submodular set function Maximization problem (GBSM).

Our problem generalizes several well-known problems, most of which have been introduced in Chapter 2. Indeed, by setting $c(s, x) = \infty$, we do not allow the association of element x to container s, while by setting $c(s, x) = 0$ we allow to assign element x to container s with no additional cost. Moreover, we relax the constraints related to the association of elements to containers by setting $c(s) = 0$ for each $s \in S$, and $c(s_1, x) = c(s_2, x)$, for each $s_1, s_2 \in S$ and $x \in X$. By suitably combining these conditions we can capture the following problems: the budgeted maximum coverage problem [8]; the non-stochastic adaptive seeding problem [15] (also with knapsack constraints [65]); monotone submodular set function subject to a knapsack constraint maximization [17].

[1]For a ground set X, a function $f : 2^X \to \mathbb{R}_{\geq 0}$ is submodular if for any pair of sets $S \subseteq T \subseteq X$ and for any element $x \in X \setminus T$, $f(S \cup \{x\}) - f(S) \geq f(T \cup \{x\}) - f(T)$.

Moreover, our cost function is not submodular and thus the setting considered in [18] does not generalize our model.

Let us consider a partial solution (S', X'). Given a set $T \subseteq X \setminus X'$, we denote by $c_{\min}(T)$ the minimum cost of associating the elements in T with a single container in S, considering that the cost of containers in S' has been already paid, formally:

$$c_{\min}(T) = \min_{s \in S} \left\{ c_{S'}(s) + \sum_{x \in T} c(s, x) \right\},$$

where $c_{S'}(s) = c(s)$ if $s \notin S'$, and $c_{S'}(s) = 0$ if $s \in S'$. We call $c_{\min}(T)$ the *marginal cost* of T with respect to the partial solution (S', X'). We define $s_{\min}(T)$ as the container $s \in S$ needed to cover T with cost $c_{\min}(T)$. Moreover, we denote by $\bar{c}(T)$ the cost of associating the elements in T to $s_{\min}(T)$, $\bar{c}(T) = c_{\min}(T) - c_{S'}(s_{\min}(T))$.

The *marginal increment* of $T \subseteq X$ with respect to the partial solution (S', X') is defined as $f(X' \cup T) - f(X')$. To simplify the notation, we use $g(T) = f(X' \cup T) - f(X')$ to denote the marginal increment.

In the algorithm in the next section, we will look for subsets of X that maximize the ratio between the marginal increment and the marginal cost with respect to some partial solution. In the following we define a family of subsets of X containing a set that approximates such maximal ratio. Given a partial solution (S', X'), we denote by $\mathcal{F}$ the family of subsets T of X that can be associated to containers in $S' \cup \{s\}$, for some single container $s \in S$, with a cost such that $c(S' \cup \{s_{\min}(T)\}, T) \leq k$, formally:

$$\mathcal{F} = \left\{ T \in 2^{X \setminus X'} \mid c(S' \cup \{s_{\min}(T)\}, T) \leq k \right\}.$$

A sub-family of $\mathcal{F}$ is an α-*list* with respect to (S', X') if it contains a subset T whose ratio between marginal increment and marginal cost is at least α times the optimal such ratio amongst all the subsets $\mathcal{F}$. Formally, $L \subseteq \mathcal{F}$ is an α-list with respect to (S', X') if

$$\max \left\{ \frac{g(T)}{c_{\min}(T)} \mid T \in L, c_{\min}(T) > 0 \right\} \geq \alpha \cdot \max \left\{ \frac{g(T)}{c_{\min}(T)} \mid T \in \mathcal{F}, c_{\min}(T) > 0 \right\}.$$

Note that the sets that maximize the above formula are not necessarily singletons due to the container opening cost. Moreover, the algorithm given in the next section build partial solutions (S', X') in such a way that $c_{\min}(T) > 0$, for each $T \in \mathcal{F}$.

4.2 Greedy Algorithm

In this section we give an algorithm that guarantees a $\frac{1}{2}\left(1-\frac{1}{e^{\alpha}}\right)$-approximation to the GBSM problem, if we assume that we can compute, in polynomial time, an α-list of polynomial size. In the next sections we will give two algorithms to compute such lists for bounded values of α.

The pseudo-code is reported in Algorithm 4. In the first step (line 3) we add all zero-cost containers to the solution. Then, the algorithm finds two candidate solutions. The first one is found at lines 4–11 with a greedy strategy as follows. The algorithm iteratively constructs a partial solution (S', X') by adding a subset $\hat{T}$ to X' and a container $s_{\min}(\hat{T})$ to S'. In particular, at each iteration, it first adds all the elements that can be associated to S' with cost 0 (line 5). Then, it selects a subset $\hat{T}$ that maximizes the ratio between the marginal increment and the marginal cost amongst the elements of an α-list L. Here, we assume that we have an algorithm to compute an α-list L w.r.t. (S', X') (see line 6). In the next sections, we will show how to compute L in polynomial time for some bounded α. The algorithm stops when adding the element with the maximum ratio would exceed the budget k or when $X' = X$. Without loss of generality, we can assume that at each iteration, the sets in the α-list L do not contain any element in X', since such elements do not increase the value of the marginal increment and possibly increase the marginal cost. This implies that at each iteration of the greedy procedure at least a new element in X is added to X' and then the number of iterations is $O(n)$.

Let (S_G, X_G) be the first candidate solution computed at the end of the greedy procedure. The second candidate solution (lines 12–14) is computed by using the set $\hat{T}$ that is discarded in the last iteration of the greedy procedure because adding $\{s_{\min}(\hat{T})\}$ and $\hat{T}$ to (S_G, X_G) would exceed the budget. Indeed, the second candidate solution is $(S_G \cup \{s_{\min}(\hat{T})\}, \hat{T})$. Note that this solution is feasible because $\hat{T}$ is contained in the α-list L computed in the last iteration of the greedy algorithm. Therefore, by definition of α-list, $c(S_G \cup \{s_{\min}(\hat{T})\}, \hat{T}) \leq k$.

The algorithm returns one of the two candidate solutions that maximizes the objective function.

The computational complexity of Algorithm 4 is $O(n \cdot (|L_{\max}| + cl))$, where $L_{\max}$ is the largest α-list computed and cl is the computational complexity of the algorithm at line 6. In the next sections we will show that our algorithms construct the α-lists in such a way that both $|L_{\max}|$ and cl are polynomially bounded in the input size.

In what follows we analyze the approximation ratio of Algorithm 4. The proof generalizes known arguments for monotone submodular maximization, see e.g. [8, 9, 17].

Algorithm 4: General Algorithm

Input : S, X

1 $S' := \emptyset$;
2 $X' := \emptyset$;
3 **foreach** $s \in S$ s.t. $c(s) = 0$ **do** $S' := S' \cup \{s\}$;
4 **repeat**
5 **foreach** $x \in X \setminus X'$ s.t. $c(s', x) = 0$ and $s' \in S'$ **do** $X' := X' \cup \{x\}$;
6 Build an α-list L w.r.t. (S', X');
7 $\hat{T} := \arg\max_{T \in L} \frac{f(X' \cup T) - f(X')}{c_{\min}(T)}$;
8 **if** $c(S' \cup \{s_{\min}(\hat{T})\}, X' \cup \hat{T}) \leq k$ **then**
9 $S' := S' \cup \{s_{\min}(\hat{T})\}$;
10 $X' := X' \cup \hat{T}$;
11 **until** $c(S' \cup \{s_{\min}(\hat{T})\}, X' \cup \hat{T}) > k$ or $X' = X$;
12 **if** $f(\hat{T}) \geq f(X')$ **then**
13 $S' := S' \cup \{s_{\min}(\hat{T})\}$;
14 $X' := \hat{T}$;
15 **return** (S', X');

We give some additional definitions that will be used in the proof. We denote an optimal solution by (S^*, X^*). Let us consider the iterations executed by the greedy algorithm. Let $l + 1$ be the index of the iteration in which an element in the α-list is not added to X' because it violates the budget constraint[2]. For $i = 1, 2, \ldots, l$, we define X'_i and S'_i as the sets X' and S' at the end of the i-th iteration of the algorithm, respectively. Moreover, let $X'_{l+1} = X'_l \cup \{\hat{T}\}$ and $S'_{l+1} = S'_l \cup \{s_{\min}(\hat{T})\}$, where $\hat{T}$ is the element selected at line 7 of iteration $l + 1$ (see Algorithm 4). Let c_i be the value of $c_{\min}(\hat{T})$ as computed at iteration i of the greedy algorithm. The next lemma will be used in the proof of Theorem 4.3.

Lemma 4.1. *After each iteration $i = 1, 2, \ldots, l + 1$,*

$$f(X'_i) \geq \left(1 - \prod_{j=1}^{i} \left(1 - \alpha \frac{c_j}{k}\right)\right) f(X^*).$$

The next lemma will be used in the proof of the Lemma 4.1.

Lemma 4.2. *After each iteration $i = 1, 2, \ldots, l + 1$, the following holds*

$$f(X'_i) - f(X'_{i-1}) \geq \frac{c_i}{k} \alpha (f(X^*) - f(X'_{i-1})).$$

[2]We can assume that this iteration exists, as otherwise the algorithm is able to select $X' = X$, which is the optimum.

Proof. Let us consider a partition of the elements in $X^* \setminus X'_{i-1}$ according to the containers they are assigned to in the optimal solution (S^*, X^*), that is the elements of each set T_j in the partition are associated to the same container in (S^*, X^*) Formally, $X^* \setminus X'_{i-1} = T_1 \cup T_2 \cup \ldots \cup T_\ell$ such that for each $j = 1, 2, \ldots, \ell$ and for each $x_1, x_2 \in T_j$, $\arg\min_{s \in S^*}\{c(s, x_1)\} = \arg\min_{s \in S^*}\{c(s, x_2)\}$ and T_j is maximal.

We first show the following:

$$f(X^*) - f(X'_{i-1}) \leq \sum_{j=1}^{\ell} (f(X'_{i-1} \cup T_j) - f(X'_{i-1})).$$

Indeed,

$$\begin{aligned} f(X^*) - f(X'_{i-1}) \leq & f(X^* \cup X'_{i-1}) - f(X'_{i-1}) \\ = & \sum_{j=1}^{\ell} \big(f(X'_{i-1} \cup T_1 \cup \ldots \cup T_j) - f(X'_{i-1} \cup T_1 \cup \ldots \cup T_{j-1})\big) \\ \leq & \sum_{j=1}^{\ell} (f(X'_{i-1} \cup T_j) - f(X'_{i-1})), \end{aligned}$$

where the last inequality follows from submodularity of f.

Let us denote by $c_i^*(T_j)$ the marginal cost of adding the elements T_j to solution (S'_{i-1}, X'_{i-1}), that is $c_i^*(T_j) = c(S'_{i-1} \cup \{s^*(T_j)\}, X'_{i-1} \cup T_j) - c(S'_{i-1}, X'_{i-1})$, where $s^*(T_j)$ is the container in S^* which all the elements of T_j are associated with. By definition of α-list and the maximum at line 7 it follows that, for each $j = 1, 2, \ldots, \ell$,

$$\frac{f(X'_i) - f(X'_{i-1})}{c_i} \geq \alpha \frac{f(X'_{i-1} \cup T_j) - f(X'_{i-1})}{c_i^*(T_j)}.$$

Therefore,

$$\begin{aligned} \sum_{j=1}^{\ell} (f(X'_{i-1} \cup T_j) - f(X'_{i-1})) & \leq \sum_{j=1}^{\ell} \frac{c_i^*(T_j)}{\alpha} \frac{f(X'_i) - f(X'_{i-1})}{c_i} \\ & = \frac{f(X'_i) - f(X'_{i-1})}{\alpha c_i} \sum_{j=1}^{\ell} c_i^*(T_j) \\ & \leq \frac{f(X'_i) - f(X'_{i-1})}{\alpha c_i} k. \end{aligned}$$

It follows that:

$$f(X^*) - f(X'_{i-1}) \leq \frac{k}{c_i \alpha} \Big(f(X'_i) - f(X'_{i-1})\Big),$$

which implies the statement. □

Proof of Lemma 4.1. For $i = 1$ from Lemma 4.2 follows that $f(X'_1) \geq \alpha\frac{c_1}{k}f(X^*)$. Applying Lemma 4.2 and the inductive hypothesis we obtain:

$$\begin{aligned}
f(X'_i) &= f(X'_{i-1}) + (f(X'_i) - f(X'_{i-1})) \\
&\geq f(X'_{i-1}) + \alpha\frac{c_i}{k}(f(X^*) - f(X'_{i-1})) \\
&= f(X'_{i-1})\left(1 - \alpha\frac{c_i}{k}\right) + \alpha\frac{c_i}{k}f(X^*) \\
&\geq \left(1 - \prod_{j=1}^{i-1}\left(1 - \alpha\frac{c_j}{k}\right)\right) f(X^*)\left(1 - \alpha\frac{c_i}{k}\right) + \alpha\frac{c_i}{k}f(X^*) \\
&= \left(1 - \prod_{j=1}^{i}\left(1 - \alpha\frac{c_j}{k}\right)\right) f(X^*).
\end{aligned}$$

□

Based on Lemma 4.1, we can prove Theorem 4.3.

Theorem 4.3. *Algorithm 4 guarantees an approximation factor of* $\frac{1}{2}\left(1 - \frac{1}{e^\alpha}\right)$ *for GBSM.*

Proof. We observe that since (S'_{l+1}, X'_{l+1}) violates the budget, then $c(S'_{l+1}, X'_{l+1}) > k$. Moreover, for a sequence of numbers $a_1, a_2, \ldots, a_n$ such that $\sum_{\ell=1}^{n} a_\ell = A$, the function $\left[1 - \prod_{i=1}^{n}\left(1 - \frac{a_i \cdot \alpha}{A}\right)\right]$ achieves its minimum when $a_i = \frac{A}{n}$ and that

$$\left[1 - \prod_{i=1}^{n}\left(1 - \frac{a_i \cdot \alpha}{A}\right)\right] \geq 1 - \left(1 - \frac{\alpha}{n}\right)^n \geq 1 - e^{-\alpha}.$$

Therefore, by applying Lemma 4.1 for $i = l+1$ and observing that $\sum_{\ell=1}^{l+1} c_\ell = c(S'_{l+1}, X'_{l+1})$, we obtain:

$$f(X'_{l+1}) \geq \left[1 - \prod_{\ell=1}^{l+1}\left(1 - \frac{c_\ell \cdot \alpha}{k}\right)\right] f(X^*) \tag{4.1}$$

$$> \left[1 - \prod_{\ell=1}^{l+1}\left(1 - \frac{c_\ell \cdot \alpha}{c(S'_{l+1}, X'_{l+1})}\right)\right] f(X^*) \tag{4.2}$$

$$\geq \left[1 - \left(1 - \frac{\alpha}{(l+1)}\right)^{l+1}\right] f(X^*) \geq \left(1 - \frac{1}{e^\alpha}\right) f(X^*). \tag{4.3}$$

Since, by submodularity, $f(X'_{l+1}) \leq f(X'_l) + f(\hat{T})$, where $\hat{T}$ is the set selected at iteration $l+1$, we get

$$f(X'_l) + f(\hat{T}) \geq \left(1 - \frac{1}{e^\alpha}\right) f(X^*).$$

Hence, $\max\{f(X'_l), f(\hat{T})\} \geq \frac{1}{2}\left(1 - \frac{1}{e^\alpha}\right) f(X^*)$. The theorem follows by observing that $\hat{T}$ is the set selected as the second candidate solution at lines 12–14 of Algorithm 4. □

4.3 Computing an α-list for a particular case

In this section, we give a polynomial time algorithm to find a $(1-\epsilon)$-list with respect to a partial solution (S', X') for the particular case in which, for a given parameter $\epsilon \in (0,1)$, the following condition holds:

$$\sum_{x \in T} c(s,x) \geq \frac{1}{\epsilon} c(s), \tag{4.4}$$

for each $s \in S$ and for each $T \subseteq X$ such that $|T| = \frac{1}{\epsilon}$. We observe that this condition is fulfilled for any $\epsilon \in (0,1)$ in the case in which $c(s) = 1$ and $c(s,x) \geq 1$, for each $s \in S$ and for each $x \in X$, which generalizes the non-stochastic adaptive seeding problem [15]. Indeed, in this case $\sum_{x \in T} c(s,x) \geq |T| = \frac{1}{\epsilon} c(s)$, for each $s \in S$ and for each $T \subseteq X$, such that $|T| = \frac{1}{\epsilon}$.

We give a simple algorithm that returns a $(1-\epsilon)$-list with respect to a partial solution (S', X'). The algorithm works as follows: build a list which contains all the subsets T of $X \setminus X'$ such that $|T| \leq \frac{1}{\epsilon}$ and $c(S' \cup \{s_{\min}(\hat{T})\}, \hat{T}) \leq k$.

Plugging this algorithm into line 6 of Algorithm 4, we can guarantee an approximation factor of $\frac{1}{2}\left(1 - \frac{1}{e}\right) - \epsilon'$, where $\epsilon' = \frac{1}{2e}\left(e^{\epsilon} - 1\right)$ for GBSM.

We observe that the case in this section contains the problem of maximizing a submodular set function under a knapsack constraint as a special case. Indeed, it is enough to set $c(s) = 0$, for each $s \in S$, and $c(s_1, x) = c(s_2, x)$, for each $s_1, s_2 \in S$ and $x \in X$. Note that in this case Condition 4.4 is satisfied for any $\epsilon \in (0,1)$. A special case of submodular set function maximization is the maximum coverage problem, and since this latter is NP-hard to be approximated within a factor greater than $\left(1 - \frac{1}{e}\right)$ [3], then the gap between the approximation factor of our algorithm and the best achievable one in polynomial time is $\frac{1}{2}$, unless $P = NP$.

It is easy to see that the computational complexity required by the algorithm in this section is $O(n^{\frac{1}{\epsilon}})$ and that $|L_{\max}| = O(n^{\frac{1}{\epsilon}})$.

In what follows, we assume that any set T^* that maximizes the ratio between marginal increment and marginal cost has size greater than $\frac{1}{\epsilon}$, as otherwise the α-list returned by our algorithm would contain such set. The following two technical lemmata will be used in the analysis of the algorithm.

Lemma 4.4. *Given a monotone submodular set function $f : 2^X \to \mathbb{R}_{\geq 0}$, then, for any $X' \subseteq X$, the function $g(T) = f(X' \cup T) - f(X')$ is monotone and submodular.*

Proof. It is easy to prove that g is monotone. We show that for each pair of sets T, S such that $T \subseteq S \subseteq (X \setminus X')$ and for each $x \in X \setminus (X' \cup S)$, the increment in the value of g that element x causes in $S \cup \{x\}$ is smaller than the increment it produces in $T \cup \{x\}$, that is

$$g(T \cup \{x\}) - g(T) \geq g(S \cup \{x\}) - g(S).$$

By applying the definition we have:

$$f(X' \cup T \cup \{x\}) - f(X') - f(X' \cup T) + f(X') \geq f(X' \cup S \cup \{x\}) - f(X') - f(X' \cup S) + f(X'),$$

which is equivalent to:

$$f(X' \cup T \cup \{x\}) - f(X' \cup T) \geq f(X' \cup S \cup \{x\}) - f(X' \cup S).$$

The statement follows because f is submodular. □

Lemma 4.5. *Let us consider a monotone submodular set function $f : 2^X \to \mathbb{R}_{\geq 0}$ and a cost function $c : 2^X \to \mathbb{R}_{\geq 0}$ such that $c(T) = \sum_{x \in T} c(\{x\})$, for each $T \subseteq X$. For each set $T \subseteq X$, if T_y denotes the subset of T such that $\frac{f(T_y)}{c(T_y)}$ is maximum and $|T_y| = y$, then $\frac{f(T)}{c(T)} \leq \frac{f(T_y)}{c(T_y)}$, for any $y \leq |T|$.*

Proof. We show the following equivalent statement: given a set $T \subseteq X$, there exists a set $T' \subseteq T$, such that $|T'| = |T| - 1$ and $\frac{f(T)}{c(T)} \leq \frac{f(T')}{c(T')}$. Let $T = \{t_1, t_2, \ldots, t_\ell\}$, where $c(\{t_i\}) \leq c(\{t_j\})$, for each $i < j$, and let δ_i denote the marginal increment given by adding element t_i to the set $\{t_1, t_2, \ldots, t_{i-1}\}$, that is, $\delta_i = f(\{t_1, t_2, \ldots, t_i\}) - f(\{t_1, t_2, \ldots, t_{i-1}\})$. We have that $f(T) = \sum_{i=1}^{\ell} \delta_i$ and $c(T) = \sum_{i=1}^{\ell} c(\{t_i\})$. We define $T' = T \setminus \{t_\ell\}$, i.e. we remove from T an element with the maximum cost. We obtain $f(T') = \sum_{i=1}^{\ell-1} \delta_i$ and $c(T') = \sum_{i=1}^{\ell-1} c(\{t_i\})$. Since $c(\{t_\ell\})$ is maximum, then

$$c(\{t_\ell\}) \geq \frac{\sum_{i=1}^{\ell-1} c(\{t_i\})}{\ell - 1} = \frac{c(T')}{\ell - 1}.$$

Moreover, by submodularity of f, $\delta_\ell \leq \frac{\sum_{i=1}^{\ell-1} \delta_i}{\ell-1} = \frac{f(T')}{\ell-1}$. Therefore:

$$\frac{f(T)}{c(T)} = \frac{f(T') + \delta_\ell}{c(T') + c(\{t_\ell\})} \leq \frac{f(T') + \frac{f(T')}{\ell-1}}{c(T') + \frac{c(T')}{\ell-1}} = \frac{f(T')}{c(T')}.$$

□

The next theorem shows the approximation ratio of the algorithm. The main idea is to consider the subset $\hat{T}$ that maximizes the ratio between the marginal increment and marginal cost in L and to derive a series of inequalities to lead us state that this value is

greater than the ratio given by the optimal subset T^* times the factor $(1-\epsilon)$. We first compare the ratio computed for $\hat{T}$ with that for $T^*_{\frac{1}{\epsilon}}$ that is a subset of cardinality $\frac{1}{\epsilon}$ of maximal ratio, then, by rewriting the marginal cost formula according to its definition and by exploiting Lemmata 4.4 and 4.5, and Condition (4.4) we compare this ratio to that given by the subset T^* and this last inequality concludes the theorem.

Theorem 4.6. *If for each $T \subseteq X$ such that $|T| = \frac{1}{\epsilon}$ and for each $s \in S$ we have $\sum_{x \in T} c(s,x) \geq \frac{1}{\epsilon} c(s)$, then the list L made of all the subsets of $X \setminus X'$ of size at most $\frac{1}{\epsilon}$ and cost at most k is a $(1-\epsilon)$-list.*

Proof. We recall that $g(T) = f(X' \cup T) - f(X')$. Given a subset T of $X \setminus X'$, we denote by T_y a subset of T such that $|T_y| = y$ and $\frac{f(T_y)}{\bar{c}(T_y)}$ is maximum. Let T^* be the subset of $X \setminus X'$ that maximizes the ratio between the marginal increment and the marginal cost. Let $\hat{T}$ be the set containing the element of L that maximizes $\frac{g(\hat{T})}{c_{\min}(\hat{T})}$. Since $|\hat{T}| \leq \frac{1}{\epsilon}$, then

$$\frac{g(\hat{T})}{c_{\min}(\hat{T})} \geq \frac{g\left(T^*_{\frac{1}{\epsilon}}\right)}{c_{\min}\left(T^*_{\frac{1}{\epsilon}}\right)} = \frac{g\left(T^*_{\frac{1}{\epsilon}}\right)}{c_{S'}\left(s_{\min}\left(T^*_{\frac{1}{\epsilon}}\right)\right) + \bar{c}\left(T^*_{\frac{1}{\epsilon}}\right)}.$$

By the hypothesis of the theorem, $\bar{c}\left(T^*_{\frac{1}{\epsilon}}\right) \geq \frac{1}{\epsilon} c\left(s_{\min}\left(T^*_{\frac{1}{\epsilon}}\right)\right)$, moreover, $c\left(s_{\min}\left(T^*_{\frac{1}{\epsilon}}\right)\right) \geq c_{S'}\left(s_{\min}\left(T^*_{\frac{1}{\epsilon}}\right)\right)$ and then $c_{S'}\left(s_{\min}\left(T^*_{\frac{1}{\epsilon}}\right)\right) \leq \epsilon \bar{c}\left(T^*_{\frac{1}{\epsilon}}\right)$. Therefore,

$$\frac{g\left(T^*_{\frac{1}{\epsilon}}\right)}{c_{S'}\left(s_{\min}\left(T^*_{\frac{1}{\epsilon}}\right)\right) + \bar{c}\left(T^*_{\frac{1}{\epsilon}}\right)} \geq \frac{g\left(T^*_{\frac{1}{\epsilon}}\right)}{\epsilon\bar{c}\left(T^*_{\frac{1}{\epsilon}}\right) + \bar{c}\left(T^*_{\frac{1}{\epsilon}}\right)} = \frac{g\left(T^*_{\frac{1}{\epsilon}}\right)}{(\epsilon+1)\bar{c}\left(T^*_{\frac{1}{\epsilon}}\right)}.$$

Since f is monotone and submodular, then, by Lemma 4.4, also $g\left(T^*_{\frac{1}{\epsilon}}\right)$ is submodular. By Lemma 4.5 it follows that

$$\frac{g\left(T^*_{\frac{1}{\epsilon}}\right)}{(\epsilon+1)\bar{c}\left(T^*_{\frac{1}{\epsilon}}\right)} \geq \frac{g(T^*)}{(\epsilon+1)\bar{c}(T^*)}.$$

We now focus on the denominator, and we obtain that:

$$\frac{1}{(\epsilon+1)\bar{c}(T^*)} = \frac{1+\epsilon-\epsilon}{(\epsilon+1)\bar{c}(T^*)} = \frac{1}{\bar{c}(T^*)} - \frac{\epsilon}{(\epsilon+1)\bar{c}(T^*)} \geq \frac{1}{\bar{c}(T^*) + c_{S'}(s_{\min}(T^*))} - \frac{\epsilon}{\epsilon\bar{c}(T^*) + \bar{c}(T^*)}.$$

By applying the hypothesis $\bar{c}(T^*) \geq \frac{1}{\epsilon} c(s_{\min}(T^*))$, it follows that:

$$\frac{1}{\bar{c}(T^*) + c_{S'}(s_{\min}(T^*))} - \frac{\epsilon}{\epsilon\bar{c}(T^*) + \bar{c}(T^*)} \geq$$

$$\frac{1}{\bar{c}(T^*) + c_{S'}(s_{\min}(T^*))} - \frac{\epsilon}{c(s_{\min}(T^*)) + \bar{c}(T^*)} \geq$$
$$\frac{1}{\bar{c}(T^*) + c_{S'}(s_{\min}(T^*))} - \frac{\epsilon}{\bar{c}(T^*) + c_{S'}(s_{\min}(T^*))} = \frac{1-\epsilon}{c_{\min}(T^*)}.$$

To conclude:

$$\frac{g(\hat{T})}{c_{\min}(\hat{T})} \geq (1-\epsilon)\frac{g(T^*)}{c_{\min}(T^*)}.$$

□

4.4 Computing an α-list for the general case

In this section we give a polynomial time algorithm that builds a $\left(1-\frac{1}{e}\right)(1-\epsilon)$-list with respect to a partial solution (S', X'), for any $\epsilon \in (0,1)$. Using this algorithm as a subroutine at line 6 of Algorithm 4, we can guarantee an approximation factor of

$$\frac{1}{2}\left(1 - \frac{1}{e^{\left(1-\frac{1}{e}\right)(1-\epsilon)}}\right)$$

for GBSM. We observe that this case generalizes the non-stochastic adaptive seeding with knapsack constraints problem, which cannot be approximated within a factor greater than $\left(1-\frac{1}{e^{1-\frac{1}{e}}}\right)$, unless $P = NP$ [65]. Then, the gap between the approximation factor of our algorithm and the best achievable one in polynomial time is $\frac{1}{2}$, unless $P = NP$. In the algorithm of this section we make use of a procedure called `GreedyMaxCover` to maximize the value of a monotone submodular function $g : 2^X \to \mathbb{R}_{\geq 0}$, given a certain budget and costs associated to the elements of X. It is well-known that there exists a polynomial-time procedure that guarantees a $\left(1-\frac{1}{e}\right)$-approximation for this problem [17]. Let us denote by $\hat{c}$ the minimum possible positive value of functions $c(s)$ and $c(s,x)$, amongst all elements x and containers s, i.e. $\hat{c} = \min\{\min\{c(s) : s \in S, c(s) > 0\}, \min\{c(s,x) : s \in S, x \in X, c(s,x) > 0\}\}$. The main idea is to build an α-list L which contains approximate solutions to the problem of maximizing a monotone submodular set function subject to a knapsack constraint in which the budget increases by a factor $1+\epsilon$ starting from $\hat{c}$, and the cost of the elements are given by the cost of associating them to a single container. In particular, we consider $q = \left\lfloor \log_{1+\epsilon}\left(\frac{k}{\hat{c}}\right)\right\rfloor + 1$ different budgets B_i that iteratively increase by a factor $1+\epsilon$, i.e. $B_0 = \hat{c}$ and $B_i = (1+\epsilon)B_{i-1}$, for $i = 1, \ldots, q$. Moreover we define $B_{q+1} = k$. For each $i = 0, \ldots, q+1$ and for each container $s \in S$, we apply procedure `GreedyMaxCover` with ground set X, budget B_i, and the cost of associating the elements to container s as cost function. Then, we add the set returned by `GreedyMaxCover` to L. In this way we consider a budget that is at most a factor $1+\epsilon$ greater than the cost of an optimal

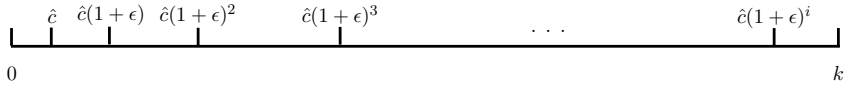

FIGURE 4.1: Growth of the budget B_i in the inner cycle of the algorithm.

Algorithm 5: Exponential Budget Greedy

Input : $S, X, S', X', k, \epsilon$

1 $L := \emptyset$;
2 **foreach** $s \in S$ **do**
3 $i := 0$;
4 **while** $\hat{c}(1+\epsilon)^i < k$ **do**
5 $B_i := \hat{c}(1+\epsilon)^i$;
6 $T_i(s) :=$ `GreedyMaxCover`$(X, s, B_i - c_{S'}(s))$;
7 $L := L \cup \{T_i(s)\}$;
8 $i := i + 1$
9 $B_i := k$;
10 $T_i(s) :=$ `GreedyMaxCover`$(X, s, B_i - c_{S'}(s))$;
11 $L := L \cup \{T_i(s)\}$;
12 **return** L;

solution and the solution returned by `GreedyMaxCover` for this budget has a value that is at most $1 - \frac{1}{e}$ times smaller than that of the optimal solution.

The pseudo-code of the algorithm is reported in Algorithm 5. The outer cycle at lines 2–11 iteratively selects a container s in S and finds a list of sets of elements assigned to container s. The inner cycle at lines 4–8, at each iteration i, calls procedure `GreedyMaxCover` which uses g as function to maximize, $\hat{c}(1+\epsilon)^i - c_{S'}(s)$ as budget and the cost of associating the elements to container s as cost function (to compute these costs, we need only to pass s as a parameter to `GreedyMaxCover`). The budget is increased by a factor $(1+\epsilon)$ until $\hat{c}(1+\epsilon)^i \geq k$. Finally the algorithm runs `GreedyMaxCover` with the full budget k. See Figure 4.1 for an illustration.

We call q the value of i at the end of the last iteration in the inner cycle of the algorithm. Let T_j be the set in L that maximizes the ratio between $g(T_j)$ and its assigned budget, that is:

$$T_j = \arg\max \left\{ \frac{g(T_i(s))}{B_i} : s \in S, i = 0, 1, \ldots, q+1 \right\}. \tag{4.5}$$

In order to bound the approximation ratio, we consider $\bar{X}^*$ as the set with the optimal ratio $\frac{g(\bar{X}^*)}{c_{\min}(\bar{X}^*)}$ amongst any possible subset of items. Let B_l be the smallest value of B_i, for $i \in \{0, 1, \ldots, q+1\}$, that is greater than or equal to the cost of an optimal solution, that is the smallest B_l such that $B_l \geq c_{\min}(\bar{X}^*)$. See figure 4.2 for an illustration. We call T_l^* the set in L that has the highest ratio $\frac{g(T_l)}{B_l}$ amongst those computed by

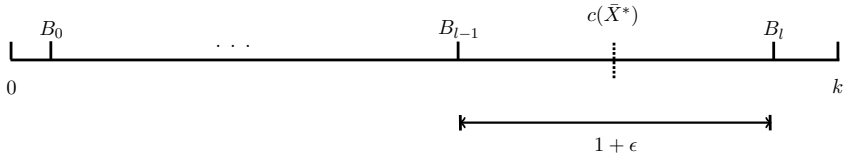

FIGURE 4.2: Notation used in Theorem 4.7.

`GreedyMaxCover` with budget B_l, i.e. $T_l^* = \max\left\{\frac{g(T_l(s))}{B_l} : s \in S\right\}$. We also denote the set that maximizes $g(X_l^*)$ with budget B_l by X_l^*.

The idea of the approximation analysis is that an optimal solution $\bar{X}^*$ has a value of g that is at most $g(X_l^*)$ and a cost that is at most $1+\epsilon$ times smaller than B_l, while the number of iterations remains polynomial since the size of the intervals grows exponentially. The next two theorems show the bounds on approximation ratio, computational complexity and size of L.

Theorem 4.7. *The list L built by Algorithm 5, is a $\left(1-\frac{1}{e}\right)(1-\epsilon)$-list.*

Proof. Since, by construction, $c_{\min}(T_j) \leq B_j$, and, by Equation 4.5, $\frac{g(T_j)}{B_j}$ is maximum, then

$$\frac{g(T_j)}{c_{\min}(T_j)} \geq \frac{g(T_j)}{B_j} \geq \frac{g(T_l^*)}{B_l}.$$

Procedure `GreedyMaxCover` guarantees a $\left(1-\frac{1}{e}\right)$-approximation, then

$$\frac{g(T_l^*)}{B_l} \geq \left(1-\frac{1}{e}\right)\frac{g(X_l^*)}{B_l}.$$

Moreover, since function g is monotone and $c_{\min}(\bar{X}^*) \leq B_l$, then $g(X_l^*) \geq g(\bar{X}^*)$, and therefore:

$$\left(1-\frac{1}{e}\right)\frac{g(X_l^*)}{B_l} \geq \left(1-\frac{1}{e}\right)\frac{g(\bar{X}^*)}{B_l}.$$

We defined B_l as the smallest value of B_i that is at least $c_{\min}(\bar{X}^*)$, this implies that $B_{l-1} \leq c_{\min}(\bar{X}^*)$. Moreover the ratio between B_l and B_{l-1} is $1+\epsilon$. It follows that $B_l \leq (1+\epsilon)c_{\min}(\bar{X}^*)$, which implies:

$$\left(1-\frac{1}{e}\right)\frac{g(\bar{X}^*)}{B_l} \geq \left(1-\frac{1}{e}\right)\left(\frac{1}{1+\epsilon}\right)\frac{g(\bar{X}^*)}{c_{\min}(\bar{X}^*)} \geq \left(1-\frac{1}{e}\right)(1-\epsilon)\frac{g(\bar{X}^*)}{c_{\min}(\bar{X}^*)}$$

The last inequality holds since $\frac{1}{1+\epsilon} = 1 - \frac{\epsilon}{1+\epsilon} \geq 1-\epsilon$, for any $\epsilon > 0$, and this concludes the proof. □

Theorem 4.8. *Algorithm 5 requires $O\left(\frac{1}{\epsilon}m \cdot gr(n) \cdot \log\frac{k}{\bar{c}}\right)$ computational time, where $gr(n)$ is the best known computational time of* `GreedyMaxCover`*, and $|L_{\max}| = O\left(\frac{1}{\epsilon}m\log\frac{k}{\bar{c}}\right)$.*

Algorithm 6: Bi-criterion Algorithm

Input : S, X

1 $S' := \emptyset$;
2 $X' := \emptyset$;
3 **foreach** $s \in S$ s.t. $c(s) = 0$ **do** $S' := S' \cup \{s\}$;
4 **repeat**
5 **foreach** $x \in X \setminus X'$ s.t. $c(s', x) = 0$ and $s' \in S'$ **do** $X' := X' \cup \{x\}$;
6 Build an α-list L w.r.t. (S', X');
7 $\hat{T} := \arg\max_{T \in L} \frac{f(X' \cup T) - f(X')}{c_{\min}(T)}$;
8 **if** $c(S' \cup \{s_{\min}(\hat{T})\}, X' \cup \hat{T}) \leq \beta k$ **then**
9 $S' := S' \cup \{s_{\min}(\hat{T})\}$;
10 $X' := X' \cup \hat{T}$;
11 **until** $c(S' \cup \{s_{\min}(\hat{T})\}, X' \cup \hat{T}) > \beta k$ or $X' = X$;
12 **if** $f(\hat{T}) \geq f(X')$ **then**
13 $S' := S' \cup \{s_{\min}(\hat{T})\}$;
14 $X' := \hat{T}$;
15 **return** (S', X');

Proof. The outer for cycle requires m iterations. We now bound the number q of iteration of the inner cycle of the algorithm. By the exit condition of the cycle, we have: $\hat{c} \cdot (1 + \epsilon)^q < k$, which is equivalent to: $q < \log_{1+\epsilon}\left(\frac{k}{\hat{c}}\right)$. Since for $\epsilon < 1$, $\log_{1+\epsilon}\left(\frac{k}{\hat{c}}\right) = O\left(\frac{1}{\epsilon}\log\frac{k}{\hat{c}}\right)$, the statement follows. □

We observe that $O(\log\frac{k}{\hat{c}})$ is polynomially bounded in the size of the input.

4.5 Bi-criterion approximation algorithm

In this section we extend the results given in Section 4.2 providing a bi-criterion approximation algorithm that guarantees a $\frac{1}{2}\left(1 - \frac{1}{e^{\alpha\beta}}\right)$-approximation to the GBSM problem, if we allow an extra budget up to a factor $\beta \geq 1$. We notice that, if $\beta = 1$, i.e. we do not increase the budget, the approximation factor is $\frac{1}{2}\left(1 - \frac{1}{e^{\alpha}}\right)$, while if $\beta = \frac{1}{\alpha}\ln\left(\frac{1}{2\epsilon}\right)$ the algorithm achieves an approximation factor of $\frac{1}{2} - \epsilon$, for any arbitrarily small $\epsilon > 0$.

The algorithm is slightly different from Algorithm 4 and it is reported in Algorithm 6. In this algorithm, we allow to exceed the given budget k by a factor β. In particular we modify lines 8 and 11, admitting a greater budget respect to Algorithm 4.

In the next theorem we show the approximation ratio of this algorithm.

Theorem 4.9. *There exists an algorithm that guarantees a* $\left[\frac{1}{2}\left(1 - \frac{1}{e^{\alpha\beta}}\right), \beta\right]$ *bi-criterion approximation for GBSM, for any* $\beta \geq 1$.

Proof. We observe that since (S'_{l+1}, X'_{l+1}) violates the budget, then $c(S'_{l+1}, X'_{l+1}) > \beta k$. Moreover, for a sequence of numbers $a_1, a_2, \ldots, a_n$ such that $\sum_{\ell=1}^{n} a_\ell = A$, the function $\left[1 - \prod_{i=1}^{n} \left(1 - \frac{a_i \cdot \alpha}{A}\right)\right]$ achieves its minimum when $a_i = \frac{A}{n}$ and that

$$\left[1 - \prod_{i=1}^{n} \left(1 - \frac{a_i \cdot \alpha}{A}\right)\right] \geq 1 - \left(1 - \frac{\alpha}{n}\right)^n \geq 1 - e^{-\alpha}.$$

Therefore, by applying Lemma 4.1 for $i = l{+}1$ and observing that $\sum_{\ell=1}^{l+1} c_\ell = c(S'_{l+1}, X'_{l+1})$, we obtain:

$$f(X'_{l+1}) \geq \left[1 - \prod_{\ell=1}^{l+1} \left(1 - \frac{c_\ell \cdot \alpha}{k}\right)\right] f(X^*) \tag{4.6}$$

$$> \left[1 - \prod_{\ell=1}^{l+1} \left(1 - \frac{c_\ell \cdot \alpha \cdot \beta}{c(S'_{l+1}, X'_{l+1})}\right)\right] f(X^*) \tag{4.7}$$

$$\geq \left[1 - \left(1 - \frac{\alpha \cdot \beta}{(l+1)}\right)^{l+1}\right] f(X^*) \geq \left(1 - \frac{1}{e^{\alpha\beta}}\right) f(X^*). \tag{4.8}$$

Since, by submodularity, $f(X'_{l+1}) \leq f(X'_l) + f(T)$, where T is the set selected at iteration $l+1$, we get

$$f(X'_l) + f(T) \geq \left(1 - \frac{1}{e^{\alpha\beta}}\right) f(X^*).$$

Hence, $\max\{f(X_l), f(T)\} \geq \frac{1}{2}\left(1 - \frac{1}{e^{\alpha\beta}}\right) f(X^*)$. The theorem follows by observing that T is the set selected as the second candidate solution at lines 12–14 of Algorithm 6. □

Chapter 5

On colorful Bin Packing games

5.1 Model and Preliminaries

In a *colorful bin packing game* $G = (N, C, \mathcal{B}, (s_i)_{i\in N}, (c_i)_{i\in N})$ we have a set of n agents $N = \{1, \ldots, n\}$, a set of $m \geq 2$ colors $C = \{1, \ldots, m\}$ and a set of n unit capacity bins $\mathcal{B} = \{B_1 \ldots, B_n\}$. Each agent $i \in N$ controls an indivisible item, denoted for convenience as x_i (i.e., we denote by $X = \{x_1, \ldots, x_n\}$ the set of items), having size $s_i \in [0, 1]$ and color $c_i \in C$ which needs to be packed into one bin in $\mathcal{B}$ without exceeding its capacity. Game G has *uniform sizes* if $s_i = s_j$ for every $i, j \in N$. The special case in which G has $m = 2$ colors is called the *black and white bin packing game*; we shall define color 1 as black, color 2 as white and denote by $\#B$ and $\#W$ the number of black and white items in G, respectively.

A strategy profile is modeled by an n-tuple $\boldsymbol{\sigma} = (\sigma_1, \ldots, \sigma_n)$ such that, for each $i \in N$, $\sigma_i \in \mathcal{B}$ is the bin chosen by agent i. We denote by $B_j(\boldsymbol{\sigma}) = \{x_i \in X : \sigma_i = B_j\}$ the set of items packed into B_j according to the strategy profile $\boldsymbol{\sigma}$. Similarly, we also write $\sigma_i(\boldsymbol{\sigma}) = \{x_l \in X : \sigma_l = \sigma_i\}$ to indicate the set of items packed in the same bin as x_i (i.e., the bin chosen by agent i), according to $\boldsymbol{\sigma}$. Given any bin B_j, we assume to pack the items in a fixed internal order, going from bottom to top, that is the sequential order in which agents have chosen the bin B_j as strategy. This is what basically happens in queue systems. Namely, for any pair of items x_i and x_l in $B_j(\boldsymbol{\sigma})$, we say that x_i precedes x_l inside the bin B_j, and we write $x_i \prec_{\boldsymbol{\sigma}} x_l$, if agent i chose bin B_j before l. Formally, given any strategy profile $\boldsymbol{\sigma}$, each item x_i occupies a precise position p_i in the sequential order of items in bin σ_i, counting from bottom to top, computed as $p_i(\boldsymbol{\sigma}) = 1 + |\{x_l \in \sigma_i(\boldsymbol{\sigma}) : x_l \prec_{\boldsymbol{\sigma}} x_i\}|$. We notice that with such packing, the last agent (say i), choosing the bin σ_i, occupies the top position in σ_i.

Denoted by $\ell_{B_j}(\boldsymbol{\sigma}) = \sum_{x_i \in B_j(\boldsymbol{\sigma})} s_i$ the total size of items packed into $B_j(\boldsymbol{\sigma})$, we always assume that $\ell_{B_j}(\boldsymbol{\sigma}) \leq 1$, so that every strategy profile induces a packing of items in $\mathcal{B}$ and vice versa. We say that an item x_i is *misplaced* if there exists an item x_l with $c_i = c_l$ such that $\sigma_i = \sigma_l$ and $|p_i - p_l| = 1$, that is, x_i is adjacent to an item of the same color in the bin. A bin is *feasible* if it stores no misplaced items. In particular, an empty bin is feasible. A strategy profile is feasible if so are all of its bins. For games with uniform sizes $s_i = s$ for every $i \in N$, we denote by $\kappa = \left\lfloor \frac{1}{s} \right\rfloor$ the maximum number of items that can be packed into any (even non-feasible) bin. We only consider the cases in which $\kappa > 1$ as, otherwise, the game is trivial.

We shall denote by $cost_i(\boldsymbol{\sigma})$ the cost that agent $i \in N$ pays in the strategy profile $\boldsymbol{\sigma}$ and each agent aims at minimizing it. We consider two different cost functions: the *egalitarian cost function* and the *proportional cost function*. We have $cost_i(\boldsymbol{\sigma}) = \infty$ under both cost functions when x_i is a misplaced item, while, for non-misplaced ones, we have $cost_i(\boldsymbol{\sigma}) = \frac{1}{|\sigma_i(\boldsymbol{\sigma})|}$ under the egalitarian cost function and $cost_i(\boldsymbol{\sigma}) = \frac{s_i}{\ell_{\sigma_i}(\boldsymbol{\sigma})}$ under the proportional one. Note that, for games with uniform sizes, the two cost functions coincide. For a fixed strategy profile $\boldsymbol{\sigma}$, we say that a bin is a *singleton* bin if it stores only one item. Moreover, when considering the egalitarian (resp. proportional) cost function, we denote by $\overline{B}_{\boldsymbol{\sigma}}$ the bin storing the maximum number of items (resp. the fullest bin) in the packing corresponding to $\boldsymbol{\sigma}$, breaking ties arbitrarily.

A *deviation* for an agent i in a strategy profile $\boldsymbol{\sigma}$ is the action of changing the selected bin σ_i in favor of another bin, say B_j, such that $\ell_{B_j}(\boldsymbol{\sigma}) + s_i \leq 1$. We shall denote as $(\boldsymbol{\sigma}_{-i}, B_j)$ the strategy profile realized after the deviation. Formally, $\boldsymbol{\sigma}' = (\boldsymbol{\sigma}_{-i}, B_j) = (\sigma'_1, , \ldots, \sigma'_n)$ is defined as follows: $\sigma'_i = B_j$ and $\sigma'_l = \sigma_l$ for each agent $l \neq i$. We consider deviations of the following form: x_i is removed from σ_i and packed on top of B_j, consistently with the sequential order of items in a bin. An *improving deviation* for an agent i in a strategy profile $\boldsymbol{\sigma}$ is a deviation towards a bin B_j such that $cost_i(\boldsymbol{\sigma}_{-i}, B_j) < cost_i(\boldsymbol{\sigma})$. Fix a feasible strategy profile $\boldsymbol{\sigma}$. Under the egalitarian cost function, agent i admits an improving deviation in $\boldsymbol{\sigma}$ if there exists a bin $B_j \in \mathcal{B} \setminus \{\sigma_i\}$ such that (i) the item on top of B_j has a color different than c_i and (ii) $|\sigma_i(\boldsymbol{\sigma})| \leq |B_j(\boldsymbol{\sigma})|$. Under the proportional cost function, agent i admits an improving deviation in $\boldsymbol{\sigma}$ if there exists a bin $B_j \in \mathcal{B} \setminus \{\sigma_i\}$ such that (i) the item on top of B_j has a color different than c_i and (ii) $\ell_{\sigma_i}(\boldsymbol{\sigma}) < \ell_{B_j}(\boldsymbol{\sigma}) + s_i$. Conversely, when a strategy profile $\boldsymbol{\sigma}$ is unfeasible, under both cost functions, an agent controlling a misplaced item x always possesses an improving deviation, for instance, by moving x to an empty bin which is always guaranteed to exist as there are n items, n bins and the bin storing x is non-singleton. We note that, as a side-effect of an improving deviation, $(\boldsymbol{\sigma}_{-i}, B_j)$ may be unfeasible even if $\boldsymbol{\sigma}$ is feasible: this happens when x_i separates two items of the same color. We say that an improving deviation is *valid* whenever the destination bin is feasible before the deviation.

A strategy profile $\boldsymbol{\sigma}$ is a (pure) Nash equilibrium if $cost_i(\boldsymbol{\sigma}) \leq cost_i(\boldsymbol{\sigma}_{-i}, B_j)$ for each $i \in N$ and $B_j \in \mathcal{B}$, that is, no agent has an improving deviation in $\boldsymbol{\sigma}$. Let $\mathsf{NE}(G)$ denote the set of Nash equilibria of game G. It is easy to see that any Nash equilibrium is a feasible strategy profile. This implies that $m = 1$ would force each item to be packed into a different bin: this justifies our choice of $m \geq 2$. A game G has the *finite improvement path property* if it does not admit an infinite sequence of improving deviations. Clearly, if G enjoys the finite improvement path property, it follows that $\mathsf{NE}(G) \neq \emptyset$. Given a strategy profile $\boldsymbol{\sigma}$, let $\emptyset(\boldsymbol{\sigma}) \subseteq \mathcal{B}$ be the set of open bins in $\boldsymbol{\sigma}$, where a bin is *open* if it stores at least one item. Let $\mathsf{F}(\boldsymbol{\sigma})$ be the number of open bins in $\boldsymbol{\sigma}$, i.e., $\mathsf{F}(\boldsymbol{\sigma}) = |\emptyset(\boldsymbol{\sigma})|$. We shall denote with $\boldsymbol{\sigma}^*(G)$ the *social optimum*, that is, any strategy profile minimizing function F. It is easy to see that any social optimum is a feasible strategy profile.
Let $\mathcal{G}_m$ denote the set of all colorful bin packing games with m colors and $\mathcal{U}_m^{odd}$ (resp. $\mathcal{U}_m^{even}$) denote the set of all colorful bin packing games with m colors and uniform sizes for which κ is odd (resp. even). Finally, denote $\mathcal{U}_m = \mathcal{U}_m^{even} \cup \mathcal{U}_m^{odd}$.

As we show in Section 2.3, the literature contains plenty of works regarding the colorful bin packing and the classical bin packing from a game theoretical perspective. However, to the best of our knowledge, this thesis is the very first study of Nash equilibria in colorful bin packing games. Notice that, if all the items have different colors, these games correspond to the classical bin packing ones. However, when there are items with the same color, the stable outcomes of the games are structurally different than bin packing ones. In fact, we show that in our games, Nash equilibria perform very differently than in bin packing ones.

5.2 Existence and Efficiency of Nash equilibria in General Games

In this section, we first show that, without any particular restriction on the type of improving deviations performed by the agents, even games with uniform sizes and only two colors may not admit the finite improvement path property (Proposition 5.1). However, if one allows the agents to perform only valid improving deviations, then any game possesses the finite improving path property under both cost functions (Theorems 5.2 and 5.3). These two theorems, together with the fact that in any strategy profile which is not a Nash equilibrium there always exists a valid improving deviation, imply the existence of Nash equilibria for colorful bin packing games under both cost functions.

Proposition 5.1. *There exists a black and white bin packing game with uniform sizes not possessing the finite improvement path property.*

Proof. Let G be a black and white bin packing game with uniform sizes defined by three black items, denoted as b_1, b_2 and b_3, and three white items, denoted as w_1, w_2 and w_3. All items have size $1/4$, so that $\kappa = 4$. Figure 5.1 depicts a cyclic sequence of (non-valid) improving deviations which shows the claim. □

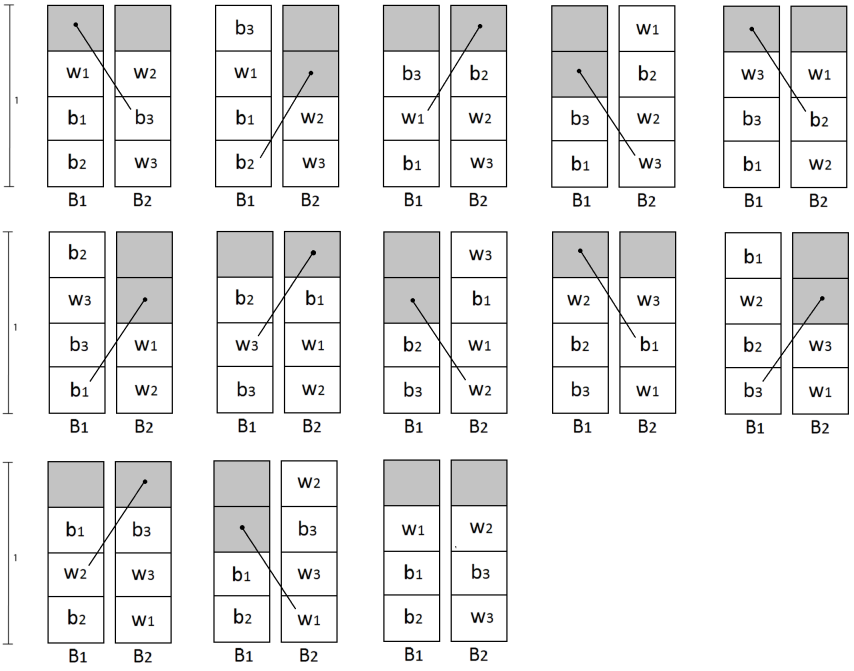

FIGURE 5.1: A cyclic sequence of (non-valid) improving deviations.

As we show in the next theorem, the previous result does not rule out the existence of a Nash equilibria in colorful bin packing games.

Theorem 5.2. *If agents are restricted to perform only valid improving deviations, then each colorful bin packing game under the egalitarian cost function admits the finite improvement path property.*

Proof. To prove the claim, we define a suitable potential function which strictly increases each time an agent performs a valid improving deviation. Given a strategy profile $\boldsymbol{\sigma}$, consider the potential function

$$\Phi(\boldsymbol{\sigma}) = \sum_{B_j \in \emptyset(\boldsymbol{\sigma}) : B_j \text{ is feasible}} |B_j(\boldsymbol{\sigma})|^{|B_j(\boldsymbol{\sigma})|}$$

and assume that an agent i performs a valid improving deviation by moving x_i onto bin $B_j \neq \sigma_i$. We distinguish between two cases.

If bin σ_i is not feasible since both $B_j(\boldsymbol{\sigma})$ and $B_j(\boldsymbol{\sigma}_{-i}, B_j)$ can be placed in feasible bins, we obtain

$$\Phi(\boldsymbol{\sigma}_{-i}, B_j) - \Phi(\boldsymbol{\sigma}) \geq (|B_j(\boldsymbol{\sigma})| + 1)^{|B_j(\boldsymbol{\sigma})|+1} - |B_j(\boldsymbol{\sigma})|^{|B_j(\boldsymbol{\sigma})|} > 0$$

when $B_j(\boldsymbol{\sigma})$ is open and $\Phi(\boldsymbol{\sigma}_{-i}, B_j) - \Phi(\boldsymbol{\sigma}) \geq 1^1 - 0 > 0$ otherwise.

If σ_i is feasible, under the egalitarian cost function, this implies that $|B_j(\boldsymbol{\sigma})| \geq |\sigma_i(\boldsymbol{\sigma})|$. Observe that, since σ_i is feasible, $cost_i(\boldsymbol{\sigma}) \leq 1$, which implies that it must be $|B_j(\boldsymbol{\sigma})| \geq 1$. For the ease of notation, set $|\sigma_i(\boldsymbol{\sigma})| = \alpha$ and $|B_j(\boldsymbol{\sigma})| = \beta$, so that $\beta \geq \alpha$ and $\beta \geq 1$; we get

$$\begin{aligned}\Phi(\boldsymbol{\sigma}_{-i}, B_j) - \Phi(\boldsymbol{\sigma}) &\geq (\beta+1)^{\beta+1} - \alpha^\alpha - \beta^\beta = (\beta+1)(\beta+1)^\beta - \alpha^\alpha - \beta^\beta \\ &\geq 2(\beta+1)^\beta - \alpha^\alpha - \beta^\beta \geq 2(\beta+1)^\beta - 2\beta^\beta > 0,\end{aligned}$$

where the second inequality comes from $\beta \geq 1$ and the third one comes from $\beta \geq \alpha$. Thus, in any case, $\Phi(\boldsymbol{\sigma}_{-i}, B_j) > \Phi(\boldsymbol{\sigma})$ which, since the number of possible strategy profiles is finite, implies the claim. □

The next proof uses a different function than the one used above.

Theorem 5.3. *If agents are restricted to perform only valid improving deviations, then each colorful bin packing game under the proportional cost function admits the finite improvement path property.*

Proof. To prove the claim, we define a suitable potential function which strictly increases each time an agent performs a valid improving deviation. Denote $\mathcal{P}(N)$ as the power set of N and, given a set $X \in \mathcal{P}(N)$, denote $S(X) = \sum_{i \in X} s_i$. Define

$$\delta = \min_{X,Y \in \mathcal{P}(N): S(X) \neq S(Y)} |S(X) - S(Y)| > 0$$

and let $\rho >> 1$ be a number such that $\rho^\delta > 2$. Given a strategy profile $\boldsymbol{\sigma}$, consider the potential function

$$\Phi(\boldsymbol{\sigma}) = \sum_{B_j \in \emptyset(\boldsymbol{\sigma}): B_j \text{ is feasible}} \rho^{\ell_{B_j(\boldsymbol{\sigma})}}$$

and assume that an agent i performs a valid improving deviation by moving x_i onto bin $B_j \neq \sigma_i$, which implies $s_i > 0$. We distinguish between two cases.

If bin σ_i is not feasible, since both $B_j(\boldsymbol{\sigma})$ and $B_j(\boldsymbol{\sigma}_{-i}, B_j)$ are feasible, we obtain

$$\Phi(\boldsymbol{\sigma}_{-i}, B_j) - \Phi(\boldsymbol{\sigma}) \geq \rho^{\ell_{B_j(\boldsymbol{\sigma})} + s_i} - \rho^{\ell_{B_j(\boldsymbol{\sigma})}} > 0$$

.

If σ_i is feasible, under the proportional cost function, this implies that $\ell_{B_j}(\boldsymbol{\sigma}) + s_i > \ell_{\sigma_i}(\boldsymbol{\sigma})$. For the ease of notation, set $\ell_{\sigma_i}(\boldsymbol{\sigma}) = \alpha$ and $\ell_{B_j}(\boldsymbol{\sigma}) = \beta$, so that $\beta + s_i > \alpha$, which, by the definition of δ, implies that $\beta + s_i \geq \max\{\alpha, \beta\} + \delta$; we get

$$\begin{aligned}\Phi(\boldsymbol{\sigma}_{-i}, B_j) - \Phi(\boldsymbol{\sigma}) &\geq \rho^{\beta+s_i} - \rho^{\alpha} - \rho^{\beta} \\ &\geq \rho^{\max\{\alpha,\beta\}+\delta} - 2\rho^{\max\{\alpha,\beta\}} = \rho^{\max\{\alpha,\beta\}}\left(\rho^{\delta} - 2\right) > 0,\end{aligned}$$

where the last inequality comes from the definition of ρ. Thus, in any case, $\Phi(\boldsymbol{\sigma}_{-i}, B_j) > \Phi(\boldsymbol{\sigma})$ which, since the number of possible strategy profiles is finite, implies the claim. □

In the following we give a tight characterization of the efficiency of Nash equilibria in colorful bin packing games under both cost functions. For games with at least three colors, Theorems 5.4 and 5.5 show that, under both cost functions, the PoS can be unbounded, thus, in the worst-case, no efficient Nash equilibria are guaranteed to exist.

Theorem 5.4. *Under the egalitarian cost function,* $\mathsf{PoS}(\mathcal{G}_m)$ *is unbounded for each* $m \geq 3$.

Proof. Fix a value $m \geq 3$. We show the theorem by proving that there exists a game $G_m \in \mathcal{G}_m$ with n agents such that

$$\mathsf{PoS}(G_m) \geq \begin{cases} \frac{n}{32} & \text{if } m = 3, \\ \frac{n(2m-5)}{18(m-1)} & \text{if } m \in \{4, 5, 6\}, \\ \frac{n(m-3)}{8(m-1)} & \text{if } m \geq 7. \end{cases}$$

By taking the limit for n going to ∞, the theorem will follow.

We construct game G_m as follows. There are:

- hk white items of size $1/k - h\delta$,
- for every color other than white, $\frac{h(k-1)}{m-1}$ items of size δ,

where h and k are suitable integers such that $h \geq 2$ and k is a multiple of m, while $\delta > 0$ is an arbitrarily small real value such that $\delta < (hk(k+1))^{-1}$. Observe that, since a combination of k white items and $k-1$ non-white items can be feasibly packed into the same bin, and there are exactly hk white items and $h(k-1)$ non-white items, there exists a feasible strategy profile using exactly h bins. Moreover, note that $k+1$ white

items do not fit into a bin, as their total size is equal to $1 + 1/k - h(k+1)\delta > 1$ by the definition of δ.

We proceed by showing that any Nash equilibrium for G_m needs to use at least

$$k\left(h - 1 - \frac{h}{m-1}\right)$$

bins. To this aim, fix a Nash equilibrium $\boldsymbol{\sigma}$ for G_m and consider bin $\overline{B}_{\boldsymbol{\sigma}}$[1]. If the item on top of $\overline{B}_{\boldsymbol{\sigma}}$ is white, then all non-white items have to be stored in $\overline{B}_{\boldsymbol{\sigma}}$. In fact, since $\overline{B}_{\boldsymbol{\sigma}}$ can contain at most k white items, for an overall occupation of $1 - hk\delta$, it follows that $\overline{B}_{\boldsymbol{\sigma}}$ has enough space to accommodate all the $h(k-1)$ non-white items. Thus, since $\overline{B}_{\boldsymbol{\sigma}}$ stores all the non-white items and can store at most k white items, it follows that $\boldsymbol{\sigma}$ needs to use at least $k(h-1)$ singleton bins to feasibly store all white items not stored in $\overline{B}_{\boldsymbol{\sigma}}$, so that

$$\mathsf{F}(\boldsymbol{\sigma}) \geq 1 + k(h-1) \geq k\left(h - 1 - \frac{h}{m-1}\right).$$

Hence, assume that the item on top of $\overline{B}_{\boldsymbol{\sigma}}$ has a color c which is not white. By the same arguments exploited above, it follows that $\overline{B}_{\boldsymbol{\sigma}}$ have to contain all non-white items having color different than c. Again, since $\overline{B}_{\boldsymbol{\sigma}}$ can contain at most k white items, $\boldsymbol{\sigma}$ needs to pack at least $k(h-1)$ white items outside bin $\overline{B}_{\boldsymbol{\sigma}}$. To do this, at most $\frac{h(k-1)}{m-1} - 1$ non-white items (the ones having color c) can be used. The only way to pack these items so that $\boldsymbol{\sigma}$ results in a Nash equilibrium is to use a first-fit-like algorithm. Note that, in order to pack k white items in the same bin, at least $k-1$ item of color c are needed. Hence, by using $\frac{h(k-1)}{m-1} - 1$ items of color c, at most $\frac{hk}{m-1}$ white items can be packed in a non-singleton bin, so that at least $k\left(h - 1 - \frac{h}{m-1}\right)$ white items need to be stored in a singleton bin.

Thus, we can conclude that

$$\mathsf{PoS}(G_m) \geq \frac{k}{h}\left(h - 1 - \frac{h}{m-1}\right).$$

Set $h = 4$ for $m = 3$, $h = 3$ for $m \in \{4, 5, 6\}$, and $h = 2$ for $m \geq 7$. For sufficiently high values of k, by using $k > \frac{n}{2h}$, the claim follows. □

Theorem 5.5. *Under the proportional cost function,* $\mathsf{PoS}(\mathcal{G}_m)$ *is unbounded for each* $m \geq 3$.

Proof. We prove the claim under the hypothesis of $m = 3$. It is easy to adapt the proof so as to deal with any number of colors $m \geq 3$.

[1]Recall that we denote by $\overline{B}_{\boldsymbol{\sigma}}$ the bin storing the maximum number of items (resp. the fullest bin) in the packing corresponding to $\boldsymbol{\sigma}$, breaking ties arbitrarily.

Consider the following instance with n agents, where n is a multiple of 4. There are $\frac{n}{2}$ items of color c_1, $\frac{n}{4}$ of color c_2, and $\frac{n}{4}$ of color c_3. The sizes are such that, one item of color c_1 has size a, while all the others $\frac{n}{2}-1$ ones have size b. Finally, each item of color c_2 or c_3 has size c. The values of the sizes a, b, c are as follows:

- $a = 1 - \frac{2}{n} + \epsilon$, for any $0 < \epsilon < \frac{2}{n}$ (i.e., $a < 1$).
- $b = 1 - a$
- $0 < c \leq \frac{2}{n}(\frac{2}{n} - \epsilon)$

Notice that $\sum_{i=1}^{n} s_i > 1$, therefore any solution cannot use less than two bins. An optimal solution can be achieved by assigning the item of size a to one bin, and all the other items to a second bin, as depicted in Figure 5.2. Notice that, *(i)* the total size of items packed in the second bin is at most a. Moreover, *(ii)* $a + c\frac{n}{2} \leq 1$, i.e., the item of size a and all the items of size c can be packed together in a bin. We further notice that, the item of size a cannot be packed with any other item of the same color c_1, because at least one element of different color is needed between them, and $a + b + c > 1$.

Consider any Nash equilibrium where the item of size a is packed in a bin B. We now show that at most two items of size c, are not packed in B. Let us suppose, by contradiction, that there exist at least three items of size c that are not packed in the bin B. If among such three items, there exist one of color c_2 and another one of color c_3, then, by properties *(i)* and *(ii)*, we get a contradiction with the fact that it is a Nash equilibrium. Thus, the three items must have the same color, and without loss of generality, suppose that it is c_2. In this case we also get a contradiction. Indeed, it is easy to see that, it is not possible to pack $\frac{n}{2}$ items of color c_3 in the bin B, by using at most $\frac{n}{2} - 3$ items of color c_2. We conclude by noticing that, at least $\frac{n}{2} - 5$ items of sizes b must be packed into singleton bins, and this concludes the proof. □

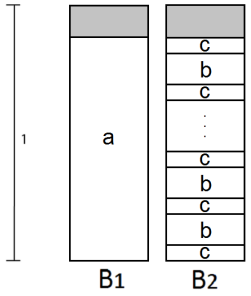

FIGURE 5.2: An optimal configuration for the instance considered in Theorem 5.5.

We conclude the section by presenting a simple algorithm, namely Algorithm 7, for computing a Nash equilibrium in colorful bin packing games under both cost functions.

Algorithm 7:

Input : A colorful bin packing game G

$X \leftarrow \{x_1, \ldots, x_n\}$;
while $(X \neq \emptyset)$ **do**
  if *(G is defined under the egalitarian cost function)* **then**
    Let B be a solution to Max Cardinality Colorful Packing(X);
  else
    Let B be a solution to Colorful Subset Sum(X);
  Open a new bin and assign it the set of items B;
  $X \leftarrow X \setminus B$;
return the strategy profile induced by the set of open bins

In particular, we shall prove that its running time is polynomial for the egalitarian cost function and pseudo-polynomial for the proportional one for the special case of constant number of colors. Algorithm 7 is based on the computation of a solution for the following two optimization problems.

Max Cardinality Colorful Packing: Given a set of items $X = \{x_1, \ldots, x_n\}$, where each item x_i has size $s_i \in [0, 1]$ and color c_i, compute a set of items of maximum cardinality which can be packed into a feasible bin without exceeding its capacity.

Colorful Subset Sum: Given a set of items $X = \{x_1, \ldots, x_n\}$, where each item x_i has size $s_i \in [0, 1]$ and color c_i, compute a set of items, of maximum total size, which can be packed into a feasible bin without exceeding its capacity.

Algorithm 7 is defined as follows.

Next lemma shows the correctness of the algorithm.

Lemma 5.6. *Algorithm 7 computes a Nash equilibrium for any colorful bin packing game G.*

Proof. Consider the strategy profile $\boldsymbol{\sigma}$ returned by Algorithm 7 and assume, for the sake of contradiction, that it is not a Nash equilibrium. Then, there exist an item x_i packed into a bin B_j and a bin $B_k \neq B_j$ such that x_i can be feasibly packed into B_k and $|B_j(\boldsymbol{\sigma})| \leq |B_k(\boldsymbol{\sigma})|$ under the egalitarian cost function, while $\ell_{B_j}(\boldsymbol{\sigma}) < \ell_{B_k}(\boldsymbol{\sigma}) + s_i$ under the proportional one. Now, if B_j is opened before B_k by Algorithm 7, we have that the set of items packed into B_k together with x_i is a better packing than B_j, thus contradicting its optimality. Conversely, if B_k is opened before B_j by Algorithm 7, we have that the set of items packed into B_k together with x_i is a better packing than B_k, thus contradicting its optimality. Hence, in both cases, we get a contradiction. □

Lemma 5.7. Max Cardinality Colorful Packing *can be solved in polynomial time.*

Proof. Our proof is based on the following two fundamental observations: *(i)* a set B of colored items can be feasibly packed into a bin without exceeding its capacity if and only if the frequency of the dominant color in B is at most $\lceil |B|/2 \rceil$, *(ii)* if there exists a set B of colored items that can be feasibly packed into a bin without exceeding its capacity whose dominant color c^* has frequency k^*, then the set of items computed as follows:

- choose the k^* items of color c^* having minimum size;
- let Y be the set of items obtained by choosing from X, for each color other than c^*, the k^* items of minimum size (or all items if there are less than k^* items of that color);
- number all items in Y in non-decreasing order of sizes;
- choose the first $|B| - k^*$ items in Y;

can also be feasibly packed into a bin without exceeding its capacity.

Let B^* be an optimal solution to the problem, c^* be the dominant color in B^* and k^* be the frequency of the items of color c^* in B^*. If we knew both c^* and k^*, because of observation *(ii)*, a solution B' such that $|B'| = |B^*|$ can be constructed in time $O(n \log n)$. Thus, the claim follows by guessing all possible n^3 pairs $(c^*, k^*, |B^*|)$, with $k^* \leq \lceil |B^*|/2 \rceil$, and then returning the best feasible solution.[2] □

Lemma 5.8. Colorful Subset Sum *can be solved in pseudo-polynomial time as long as the number of colors is constant.*

Proof. Assume to know the exact number of items of each color $c_1^*, \ldots, c_m^*$ that belong to an optimal solution. Note that Colorful Subset Sum becomes a variant of a Knapsack problem in which item are partitioned into m sets according to their colors and one wants to maximize the sum of the volumes of the items packed in the knapsack without exceeding its capacity by choosing exactly c_i^* items from each set. By suitably extending the dynamic programming algorithm for the knapsack problem and guessing the $O(n^m)$ possible tuples $c_1^*, \ldots, c_m^*$ such that $c^* = \max\{c_1^*, \ldots, c_m^*\} \leq \left\lceil \frac{\sum_i c_i^* - c^*}{2} \right\rceil$, the claim follows. □

As a consequence of Lemmas 5.6, 5.7 and 5.8, we obtain the following result.

[2] Indeed, it is possible to lower the computational complexity of the algorithm by considering only the n^2 pairs (c^*, k^*) by noting that the algorithm used within observation *(ii)* can be easily adapted to deal with the case in which $|B|$ is not known.

Theorem 5.9. *A Nash equilibrium for colorful bin packing games can be computed in polynomial time under the egalitarian cost function and in pseudo-polynomial time for a constant number of colors under the proportional one.*

5.3 Efficiency of Nash equilibria in Black and White Games

For black and white bin packing games, things get much more interesting, as we show an upper bound of 3 on the PoA and a corresponding lower bound on the PoS. To address this particular case, given a black and white bin packing game G, we make use of the following additional notation. Given a strategy profile $\boldsymbol{\sigma}$, we denote by $S_b(\boldsymbol{\sigma})$ the set of singleton bins storing a black item, by $S_w(\boldsymbol{\sigma})$ the set of singleton bins storing a white item, by $M_b(\boldsymbol{\sigma})$ the set of non-singleton bins having a black item on top, and by $M_w(\boldsymbol{\sigma})$ the set of non-singleton bins having a white item on top.

The following lemma relates the set of open bins of a feasible strategy profile with that of a social optimum. Recall that $\mathsf{F}(\boldsymbol{\sigma})$ is the number of open bins in $\boldsymbol{\sigma}$, i.e., $\mathsf{F}(\boldsymbol{\sigma}) = |\emptyset(\boldsymbol{\sigma})|$.

Lemma 5.10. *Fix a feasible strategy profile $\boldsymbol{\sigma}$ and a social optimum $\boldsymbol{\sigma}^*$ for a black and white bin packing game G. Then, $|S_b(\boldsymbol{\sigma})| - |S_w(\boldsymbol{\sigma})| - |M_w(\boldsymbol{\sigma})| \leq \mathsf{F}(\boldsymbol{\sigma}^*)$.*

Proof. First, we observe that, since in any open bin of $\boldsymbol{\sigma}^*$ the absolute value of the difference between the number of black and white items is at most 1, we have

$$\#B \leq \#W + \mathsf{F}(\boldsymbol{\sigma}^*). \tag{5.1}$$

Now, let us denote with z_b the number of black items packed into bins belonging to $M_b(\boldsymbol{\sigma}) \cup M_w(\boldsymbol{\sigma})$ and with z_w the number of white items packed into bins belonging to $M_b(\boldsymbol{\sigma}) \cup M_w(\boldsymbol{\sigma})$. We have $z_b = \#B - |S_b(\boldsymbol{\sigma})|$ and $z_w = \#W - |S_w(\boldsymbol{\sigma})|$. Moreover, the number of black items packed into bins belonging to $M_b(\boldsymbol{\sigma}) \cup M_w(\boldsymbol{\sigma})$ is at least the number of white items packed into bins belonging to $M_b(\boldsymbol{\sigma}) \cup M_w(\boldsymbol{\sigma})$ minus $|M_w(\boldsymbol{\sigma})|$, so that, by putting all together, we obtain

$$\#B \geq \#W + |S_b(\boldsymbol{\sigma})| - |S_w(\boldsymbol{\sigma})| - |M_w(\boldsymbol{\sigma})|. \tag{5.2}$$

By combining inequalities (5.1) and (5.2), the claim follows. □

The following theorem gives an upper bound on the PoA of black and white bin packing games under both cost functions.

Theorem 5.11. *Under both cost functions,* $\mathsf{PoA}(\mathcal{G}_2) \leq 3$.

Proof. Given a black and white bin packing game G under a certain cost function, i.e. with both of the two cost functions, fix a Nash equilibrium $\boldsymbol{\sigma}$ and a social optimum $\boldsymbol{\sigma}^*$. Let $S = \sum_{i \in N} s_i$ be the sum of the sizes of all the items. Notice that $\mathsf{F}(\boldsymbol{\sigma}^*) \geq \lceil S \rceil$. Assume without loss of generality that $|S_b(\boldsymbol{\sigma})| \geq |S_w(\boldsymbol{\sigma})|$ (if this is not the case, we simply swap the two colors).

Let P be the set of pairs of bins constructed as follows: each bin in $S_w(\boldsymbol{\sigma})$ is paired with a bin in $S_b(\boldsymbol{\sigma})$, each remaining bin in $S_b(\boldsymbol{\sigma})$ is paired with a bin in $M_w(\boldsymbol{\sigma})$, finally, all the remaining bins in $M_w(\boldsymbol{\sigma})$ and all the bins in $M_b(\boldsymbol{\sigma})$ are joined into pairs until possible. It is easy to check that, for each created pair of bins (B_j, B_k), it must be that

$$\ell_{B_j}(\boldsymbol{\sigma}) + \ell_{B_k}(\boldsymbol{\sigma}) > 1 \tag{5.3}$$

under both cost functions, otherwise the hypothesis that $\boldsymbol{\sigma}$ is a Nash equilibrium would be contradicted. Moreover, by (5.3), it follows that $S > |P|$ which implies that $\mathsf{F}(\boldsymbol{\sigma}^*) \geq \lceil S \rceil \geq |P| + 1$.

Now two cases may occur:

- no bin in $S_b(\boldsymbol{\sigma})$ is left unmatched by P, which implies that $|S_b(\boldsymbol{\sigma})| + |S_w(\boldsymbol{\sigma})| + |M_b(\boldsymbol{\sigma})| + |M_w(\boldsymbol{\sigma})| \leq 2|P| + 1$, as at most one bin from the set $M_w(\boldsymbol{\sigma}) \cup M_b(\boldsymbol{\sigma})$ may remain unmatched. Thus, we obtain

$$\mathsf{F}(\boldsymbol{\sigma}) = |S_b(\boldsymbol{\sigma})| + |S_w(\boldsymbol{\sigma})| + |M_b(\boldsymbol{\sigma})| + |M_w(\boldsymbol{\sigma})| \leq 2|P| + 1 < 2\mathsf{F}(\boldsymbol{\sigma}^*);$$

- at least one bin in $S_b(\boldsymbol{\sigma})$ is unmatched by P, which implies that $|S_b(\boldsymbol{\sigma})| + |S_w(\boldsymbol{\sigma})| + |M_b(\boldsymbol{\sigma})| + |M_w(\boldsymbol{\sigma})| \leq 2|P| + 1 + |S_b(\boldsymbol{\sigma})| - |S_w(\boldsymbol{\sigma})| - |M_w(\boldsymbol{\sigma})|$. Thus, we obtain

$$\begin{aligned} \mathsf{F}(\boldsymbol{\sigma}) &= |S_b(\boldsymbol{\sigma})| + |S_w(\boldsymbol{\sigma})| + |M_b(\boldsymbol{\sigma})| + |M_w(\boldsymbol{\sigma})| \\ &\leq 2|P| + 1 + |S_b(\boldsymbol{\sigma})| - |S_w(\boldsymbol{\sigma})| - |M_w(\boldsymbol{\sigma})| \\ &\leq 2|P| + 1 + \mathsf{F}(\boldsymbol{\sigma}^*) < 2\mathsf{F}(\boldsymbol{\sigma}^*) + \mathsf{F}(\boldsymbol{\sigma}^*) = 3\mathsf{F}(\boldsymbol{\sigma}^*), \end{aligned}$$

  where the second inequality comes from Lemma 5.10.

□

In the next two theorems, we show a matching lower bound on the PoS of black and white bin packing games under both cost functions.

Theorem 5.12. *Under the egalitarian cost function,* $\mathsf{PoS}(\mathcal{G}_2) \geq 3$.

Proof. We prove the theorem by showing that, for any $\epsilon > 0$, there exists a black and white bin packing game $G_\epsilon \in \mathcal{G}_2$ such that $\mathsf{PoS}(G_\epsilon) \geq 3 - \epsilon$. G_ϵ is defined by the following set of items:

- $2k$ white items having size $\frac{1}{k} - 2\delta$, denoted as items of type (1);
- $k/2$ black items having size 1, denoted as items of type (2);
- $2k$ black items having size δ, denoted as items of type (3);
- k white items having size 0, denoted as items of type (4);

where k is an even integer such that $k \geq \max\left\{\frac{10-4\epsilon}{\epsilon}, 19\right\}$ and $\delta > 0$ is arbitrarily small.

Denote with $\boldsymbol{\sigma}$ the strategy profile such that $\emptyset(\boldsymbol{\sigma}) = (B_1, \ldots, B_h)$ with $h = \frac{3k}{2} + 1$ and such that

- bin B_1 contains k items of type (1), k items of type (4) and $2k$ items of type (3),
- each bin from B_2 to B_{k+1}, for a total of k bins, contains one item of type (1),
- each bin from B_{k+2} to B_h, for a total of $k/2$ bins, contains one item of type (2).

In the definition of $\boldsymbol{\sigma}$, we avoid considering the order in which the items are packed within each bin as it is irrelevant to our purposes. We only stress the fact that there exists a proper ordering of the items which makes $\boldsymbol{\sigma}$ a feasible strategy profile.

Now, denote with $\boldsymbol{\sigma}^*$ the strategy profile such that $\emptyset(\boldsymbol{\sigma}^*) = (B_1^*, \ldots, B_{h^*}^*)$ with $h^* = \frac{k}{2} + 2$ and such that

- bins B_1^* and B_2^* both contain k items of type (1) and k items of type (3),
- each bin from B_3^* to $B_{h^*}^*$, for a total of $k/2$ bins, contains one item of type (2) and 2 items of type (4).

Also in this case, we avoid considering the order in which the items are packed within each bin and stress the fact that there exists a proper ordering of the items which makes $\boldsymbol{\sigma}^*$ a feasible strategy profile.

In order to show the claimed lower bound on $\mathsf{PoS}(G_\epsilon)$, we proceed by proving that the packing of items corresponding to any Nash equilibrium for G_ϵ coincides with the one corresponding to $\emptyset(\boldsymbol{\sigma})$. This is achieved by exploiting a sequence of results.

First of all, we observe the following basic fact.

Fact 5.13. In any Nash equilibrium for G_ϵ, any bin containing an item of type (2) can store at most 3 items.

We continue by proving some basic properties possessed by bin $\overline{B}_{\boldsymbol{\tau}}$, for any Nash equilibrium $\boldsymbol{\tau}$ for G_ϵ.

Lemma 5.14. *Fix a Nash equilibrium $\boldsymbol{\tau}$ for G_ϵ. Then, $\overline{B}_{\boldsymbol{\tau}}$ stores at least 4 items.*

Proof. Assume, for the sake of contradiction, that $\overline{B}_{\boldsymbol{\tau}}$ stores at most 3 items. Since the total number of items is $\frac{11k}{2}$, it follows that $\boldsymbol{\tau}$ is made up of at least $\lceil \frac{11k}{6} \rceil$ bins. This implies that

$$\begin{aligned}\mathsf{PoA}(G_\epsilon) &\geq \frac{\mathsf{F}(\boldsymbol{\tau})}{\mathsf{F}(\boldsymbol{\sigma}^*)} \geq \frac{\lceil \frac{11k}{6} \rceil}{\frac{k}{2}+2} \geq \frac{\frac{11k}{6}}{\frac{k}{2}+2} \\ &= \frac{11}{3} - \frac{44}{3(k+4)} = 3 + \frac{2}{3} - \frac{44}{3(k+4)} > 3.\end{aligned}$$

Since $k > 18$: a contradiction to Theorem 5.11. □

As a consequence of Fact 5.13 and Lemma 5.14, we get the following corollary.

Corollary 5.15. *Fix a Nash equilibrium $\boldsymbol{\tau}$ for G_ϵ. No item of type (2) is packed into $\overline{B}_{\boldsymbol{\tau}}$.*

Lemma 5.16. *Fix a Nash equilibrium $\boldsymbol{\tau}$ for G_ϵ. All items of type (4) are packed into $\overline{B}_{\boldsymbol{\tau}}$.*

Proof. Assume, for the sake of contradiction, that there exists an item of type (4), say x_i, which is not packed into $\overline{B}_{\boldsymbol{\tau}}$. Since $\boldsymbol{\tau}$ is a Nash equilibrium, the item on top of $\overline{B}_{\boldsymbol{\tau}}$ must be white, otherwise agent i would lower her cost by migrating to $\overline{B}_{\boldsymbol{\tau}}$. By Corollary 5.15, $\overline{B}_{\boldsymbol{\tau}}$ can only store items of type (1), (3) and (4). The load coming from items of type (1) packed into $\overline{B}_{\boldsymbol{\tau}}$ can be at most $1 - 2k\delta$ (since at most k items of type (1) can be packed into the same bin), so that bin $\overline{B}_{\boldsymbol{\tau}}$ can potentially store all items of type (3). This implies that all of these items must indeed be packed into $\overline{B}_{\boldsymbol{\tau}}$, otherwise the agent owning any of the leftover ones would lower her cost by migrating to $\overline{B}_{\boldsymbol{\tau}}$. Hence, we can conclude that $\overline{B}_{\boldsymbol{\tau}}$ stores $2k$ black items and the item on top of $\overline{B}_{\boldsymbol{\tau}}$ is white. This implies that $\overline{B}_{\boldsymbol{\tau}}$ has to store at least $2k$ white items. Now, since at most k items of type (1) can be packed into the same bin, $\overline{B}_{\boldsymbol{\tau}}$ needs to store all items of type (4). □

Lemma 5.17. *Fix a Nash equilibrium $\boldsymbol{\tau}$ for G_ϵ. All items of type (3) are packed into $\overline{B}_{\boldsymbol{\tau}}$.*

Proof. Assume, for the sake of contradiction, that there exists an item of type (3), say x_i, which is not packed into $\overline{B}_{\boldsymbol{\tau}}$. Again, by Corollary 5.15, $\overline{B}_{\boldsymbol{\tau}}$ can only store items of type (1), (3) and (4) and the load coming from items of type (1) packed into $\overline{B}_{\boldsymbol{\tau}}$ can be at most $1-2k\delta$, so that bin $\overline{B}_{\boldsymbol{\tau}}$ can potentially store all items of type (3). Hence, since $\boldsymbol{\tau}$ is a Nash equilibrium, the item on top of $\overline{B}_{\boldsymbol{\tau}}$ must be black, otherwise agent i would lower her cost by migrating to $\overline{B}_{\boldsymbol{\tau}}$. This implies that the total load of $\overline{B}_{\boldsymbol{\tau}}$ has to exceed $1-1/k+2\delta$, otherwise any agent owning an item of type (1) would lower her cost by migrating to $\overline{B}_{\boldsymbol{\tau}}$. Given that no item of type (2) can be stored in $\overline{B}_{\boldsymbol{\tau}}$, we have that, in order to achieve a load of more than $1-1/k+2\delta$, $\overline{B}_{\boldsymbol{\tau}}$ has to store exactly k items of type (1). Moreover, by Lemma 5.16, all items of type (4) are packed in $\overline{B}_{\boldsymbol{\tau}}$. Hence, we can conclude that $\overline{B}_{\boldsymbol{\tau}}$ stores exactly $2k$ white items and the item on top of $\overline{B}_{\boldsymbol{\tau}}$ is black. This implies that $\overline{B}_{\boldsymbol{\tau}}$ has to store at least $2k$ black items none of which belonging to type (2), that is, $\overline{B}_{\boldsymbol{\tau}}$ has to store all items of type (3). □

We can finally prove that all Nash equilibria for G_ϵ correspond to the same packing of items.

Lemma 5.18. *Fix a Nash equilibrium $\boldsymbol{\tau}$ for G_ϵ. Then, $\emptyset(\boldsymbol{\tau})$ and $\emptyset(\boldsymbol{\sigma})$ are equal up to a renumbering of the bins.*

Proof. Fix a Nash equilibrium $\boldsymbol{\tau}$ for G_ϵ. By Lemmas 5.16 and 5.17, it follows that $\overline{B}_{\boldsymbol{\tau}}$ stores all items of type (3) and (4), that is, $2k$ black items and k white items. Hence, in order to obtain a feasible strategy profile, $\overline{B}_{\boldsymbol{\tau}}$ needs to store at least $k-1$ items of type (1). Moreover, $\overline{B}_{\boldsymbol{\tau}}$ cannot store more than k items of type (1). If $\overline{B}_{\boldsymbol{\tau}}$ stores $k-1$ items of type (1), the agent owning any of the leftover items of type (1) would lower her cost by migrating to $\overline{B}_{\boldsymbol{\tau}}$. This implies that $\overline{B}_{\boldsymbol{\tau}}$ needs to store exactly k items of type (1). Since the remaining k items of type (1) and all items of type (2) can only be packed into different bins, it follows that there exists a suitable renumbering of the bins in $\boldsymbol{\tau}$ which gives $\emptyset(\boldsymbol{\tau}) = \emptyset(\boldsymbol{\sigma})$. □

We can conclude our proof by lower bounding the price of stability of G_ϵ. Because of Lemma 5.18, we get

$$\mathsf{PoS}(G_\epsilon) \geq \frac{\mathsf{F}(\boldsymbol{\sigma})}{\mathsf{F}(\boldsymbol{\sigma}^*)} = \frac{\frac{3k}{2}+1}{\frac{k}{2}+2} = 3 - \frac{10}{k+4} \geq 3-\epsilon,$$

since $k \geq \frac{10-4\epsilon}{\epsilon}$ implies that $\epsilon \geq \frac{10}{k+4}$. □

Theorem 5.19. *Under the proportional cost function,* $\mathsf{PoS}(\mathcal{G}_2) \geq 3$.

Proof. We prove the theorem by showing that, for any $\epsilon > 0$, there exists a black and white bin packing game $G_\epsilon \in \mathcal{G}_2$ such that $\mathsf{PoS}(G_\epsilon) \geq 3 - \epsilon$. G_ϵ is defined by the following set of items:

- $2k$ white items having size $\frac{1}{k} - 3\delta$, denoted as items of type (1);
- $k/2$ black items having size $1 - 5k\delta$, denoted as items of type (2);
- $2k$ black items having size δ, denoted as items of type (3);
- k white items having size δ, denoted as items of type (4);

where k is an even integer such that $k \geq \max\left\{\frac{10-4\epsilon}{\epsilon}, 2\right\}$ and $\delta > 0$ is an arbitrarily small number satisfying $\delta < (k(5k+3))^{-1}$.

Denote with $\boldsymbol{\sigma}$ the strategy profile such that $\emptyset(\boldsymbol{\sigma}) = (B_1, \ldots, B_h)$ with $h = \frac{3k}{2} + 1$ and such that

- bin B_1 contains k items of type (1), k items of type (4) and $2k$ items of type (3),
- each bin from B_2 to B_{k+1}, for a total of k bins, contains one item of type (1),
- each bin from B_{k+2} to B_h, for a total of $k/2$ bins, contains one item of type (2).

In the definition of $\boldsymbol{\sigma}$, we avoid considering the order in which the items are packed within each bin as it is irrelevant to our purposes. We only stress the fact that there exists a proper ordering of the items which makes $\boldsymbol{\sigma}$ a feasible strategy profile.

Now, denote with $\boldsymbol{\sigma}^*$ the strategy profile such that $\emptyset(\boldsymbol{\sigma}^*) = (B_1^*, \ldots, B_{h^*}^*)$ with $h^* = \frac{k}{2} + 2$ and such that

- bins B_1^* and B_2^* both contain k items of type (1) and k items of type (3),
- each bin from B_3^* to $B_{h^*}^*$, for a total of $k/2$ bins, contains one item of type (2) and 2 items of type (4).

Also in this case, we avoid considering the order in which the items are packed within each bin and stress the fact that there exists a proper ordering of the items which makes $\boldsymbol{\sigma}^*$ a feasible strategy profile.

In order to show the claimed lower bound on $\mathsf{PoS}(G_\epsilon)$, we proceed by proving that the packing of items corresponding to any Nash equilibrium for G_ϵ coincides with the one corresponding to $\emptyset(\boldsymbol{\sigma})$. This is achieved by exploiting a sequence of results.

Fact 5.20. No bin can store more than k items of type (1).

Proof. $k+1$ items of type (1) require a total space of $(k+1)(1/k-3\delta) = 1+1/k-3(k+1)\delta > 1$ since $\delta < (k(5k+3))^{-1} < (3k(k+1))^{-1}$. □

Fact 5.21. No bin can store more than 1 item of type (2).

Proof. Two items of type (2) require a total space of $2-10k\delta > 1$ since $\delta < (k(5k+3))^{-1} < (10k)^{-1}$. □

Fact 5.22. No bin can simultaneously store items of types (1) and (2).

Proof. One item of type (1) and one item of type (2) require a total space of $1/k-3\delta+1-5k\delta = 1+1/k-(5k+3)\delta > 1$ since $\delta < (k(5k+3))^{-1}$. □

Fact 5.23. For any strategy profile $\boldsymbol{\tau}$, $\overline{B}_{\boldsymbol{\tau}}$ either contains exactly one item of type (2) and no items of type (1) or exactly k items of type (1) and no items of type (2).

Proof. Clearly, we have

$$\ell_{\overline{B}_{\boldsymbol{\tau}}}(\boldsymbol{\tau}) \geq 1-5k\delta \tag{5.4}$$

(a bin containing an items of type (2) has at least this occupation) which, given that $\delta < (k(5k+3))^{-1} < (8k)^{-1}$, $\overline{B}_{\boldsymbol{\tau}}$ cannot contain only items of types (3) and (4). If $\overline{B}_{\boldsymbol{\tau}}$ contains items of type (2), then, by Facts 5.21 and 5.22, it has to contain exactly one item of this type. On the contrary, if $\overline{B}_{\boldsymbol{\tau}}$ contains items of type (1), then by Facts 5.20 and 5.22, it can only contain k items of this time, but no more than k of them. If $\overline{B}_{\boldsymbol{\tau}}$ contains at most $k-1$ items of type (1), then, as $\overline{B}_{\boldsymbol{\tau}}$ can additionally contain all items of types (3) and (4), we have $\ell_{\overline{B}_{\boldsymbol{\tau}}}(\boldsymbol{\tau}) \leq (k-1)(1/k-3\delta)+3k\delta = 1-1/k+3\delta < 1-5k\delta$ as $\delta < (k(5k+3))^{-1}$, thus contradicting inequality (5.4). □

Fact 5.24. For any strategy profile $\boldsymbol{\tau}$, $\overline{B}_{\boldsymbol{\tau}}$ has enough unused space to store all items of types (3) and (4) which are not packed into $\overline{B}_{\boldsymbol{\tau}}$.

Proof. Fix a strategy profile $\boldsymbol{\tau}$. By Fact 5.23, we need to distinguish between two cases only. If $\overline{B}_{\boldsymbol{\tau}}$ contains an item of type (2) and no items of type (1), since all items of types (3) and (4) can be packed into the leftover space of $5k\delta$, the claim follows. If $\overline{B}_{\boldsymbol{\tau}}$ contains k item of type (1) and no items of type (2), since all items of types (3) and (4) can be packed into the leftover space of $3k\delta$, the claim follows. □

We continue by showing a fundamental structural property.

Lemma 5.25. *Fix a Nash equilibrium $\boldsymbol{\tau}$ for G_ϵ. Then, $\overline{B}_{\boldsymbol{\tau}}$ stores all items of type (4).*

Proof. Fix a Nash equilibrium $\boldsymbol{\tau}$ for G_ϵ and assume, by way of contradiction, that at least one item x_j of type (4) is not packed into $\overline{B}_{\boldsymbol{\tau}}$. By Fact 5.24, it follows that the item on top of $\overline{B}_{\boldsymbol{\tau}}$ is white, otherwise agent j would lower her cost by migrating to $\overline{B}_{\boldsymbol{\tau}}$. Again, by Fact 5.24, this implies that all items of type (3) are packed into $\overline{B}_{\boldsymbol{\tau}}$, otherwise the agent owing a leftover item would lower her cost by migrating to $\overline{B}_{\boldsymbol{\tau}}$. Thus, we have that $\overline{B}_{\boldsymbol{\tau}}$ stores at least $2k$ black items. As, by hypothesis $\overline{B}_{\boldsymbol{\tau}}$ does not store all items of type (4), by Fact 5.23, the number of white items packed into $\overline{B}_{\boldsymbol{\tau}}$ is at most $2k-1$. Thus, we get a contradiction as it is not possible to feasible pack $2k$ black items and $2k-1$ white items in such a way that the item of top of the bin is white. □

We can now prove that all Nash equilibria for G_ϵ correspond to the same packing of items.

Lemma 5.26. *Fix a Nash equilibrium $\boldsymbol{\tau}$ for G_ϵ. Then, $\emptyset(\boldsymbol{\tau})$ and $\emptyset(\boldsymbol{\sigma})$ are equal up to a renumbering of the bins.*

Proof. Fix a Nash equilibrium $\boldsymbol{\tau}$ for G_ϵ. We show the claim by proving that $\overline{B}_{\boldsymbol{\tau}}$ contains k items of type (1) and all items of types (3) and (4). By Fact 5.23, we have to distinguish between two cases only.

Assume first that $\overline{B}_{\boldsymbol{\tau}}$ contains an item of type (2) and no items of type (1). As $\overline{B}_{\boldsymbol{\tau}}$ can feasibly store all items of type (4) and no more than k items of type (3), we get $\ell_{\overline{B}_{\boldsymbol{\tau}}}(\boldsymbol{\tau}) \leq 1 - 5k\delta + 2k\delta = 1 - 3k\delta$. Now, let $\widetilde{B}_{\boldsymbol{\tau}}$ be the bin containing the maximum number of items of type (1) in $\boldsymbol{\tau}$. We claim that $\widetilde{B}_{\boldsymbol{\tau}}$ contains k items of type (1). Assume, by way of contradiction, that $\widetilde{B}_{\boldsymbol{\tau}}$ contains at most $k-1$ items of type (1). If the item on top of $\widetilde{B}_{\boldsymbol{\tau}}$ is black, we get a contradiction as the agent owning an item of type (1) not packed into $\widetilde{B}_{\boldsymbol{\tau}}$ would lower her cost by migrating to $\widetilde{B}_{\boldsymbol{\tau}}$. If the item on top of $\widetilde{B}_{\boldsymbol{\tau}}$ is white, it follows that $\widetilde{B}_{\boldsymbol{\tau}}$ can store at most $k-1$ items of type (3). Given that $\overline{B}_{\boldsymbol{\tau}}$ contains at most k items of type (3), it follows that there is an item of this type, say x_j which is packed neither into $\overline{B}_{\boldsymbol{\tau}}$ nor into $\widetilde{B}_{\boldsymbol{\tau}}$. Hence, we get a contradiction as agent j would lower her cost by migrating to $\widetilde{B}_{\boldsymbol{\tau}}$. Thus, $\widetilde{B}_{\boldsymbol{\tau}}$ contains k items of type (1). By the same argument exploited above, it also follows that $\widetilde{B}_{\boldsymbol{\tau}}$ also contains k items of type (3). Hence we get $\ell_{\widetilde{B}_{\boldsymbol{\tau}}}(\boldsymbol{\tau}) = k(1/k - 3\delta) + k\delta = 1 - 2\delta > \ell_{\overline{B}_{\boldsymbol{\tau}}}(\boldsymbol{\tau})$ thus contradicting the fact that $\overline{B}_{\boldsymbol{\tau}}$ contains an item of type (2) and no items of type (1). So, we conclude that this case cannot occur.

Now assume that $\overline{B}_{\boldsymbol{\tau}}$ contains k items of type k and no items of type (2). By Lemma 5.25, $\overline{B}_{\boldsymbol{\tau}}$ also contains all k items of type (4), so that it contains $2k$ white items. Moreover, if $\overline{B}_{\boldsymbol{\tau}}$ does not contain all items of type (3), by Fact 5.24, the item on top of $\overline{B}_{\boldsymbol{\tau}}$ must be black, otherwise the agent owing any leftover item would lower her cost my migrating

to $\overline{B}_{\boldsymbol{\tau}}$. But this raises a contradiction, since it is not possible to feasible pack $2k$ white items and $2k-1$ black one in such a way that the item on top of the bin is black. Hence, we have that $\overline{B}_{\boldsymbol{\tau}}$ contains all items of type (3) which shows the claim. □

We can conclude our proof by lower bounding the price of stability of G_ϵ. Because of Lemma 5.26, we get

$$\mathsf{PoS}(G_\epsilon) \geq \frac{\mathsf{F}(\boldsymbol{\sigma})}{\mathsf{F}(\boldsymbol{\sigma}^*)} = \frac{\frac{3k}{2}+1}{\frac{k}{2}+2} = 3 - \frac{10}{k+4} \geq 3-\epsilon,$$

since $k \geq \frac{10-4\epsilon}{\epsilon}$ implies that $\epsilon \geq \frac{10}{k+4}$. □

5.4 Efficiency of Nash equilibria in Games with Uniform Sizes

In this section, we provide a complete picture of the efficiency of Nash equilibria for games with uniform sizes. We remind the reader that, in this setting, the egalitarian and proportional cost functions are equivalent. For the sake of simplicity, we say that a bin is *full* if it contains κ items.

First, we give a lower bound of 2 on the PoS for games with any number of colors under the hypothesis that κ is an even number.

Theorem 5.27. [3] *For each $m \geq 2$, $\mathsf{PoS}(\mathcal{U}_m^{even}) \geq 2$.*

Proof. We prove the claim under the hypothesis of $m = 2$. It is easy to adapt the proof so as to deal with any number of colors $m \geq 2$. In particular, we show that, for any fixed $\epsilon > 0$, there exists a game $G_\epsilon \in \mathcal{U}_2^{even}$ such that $\mathsf{PoS}(G_\epsilon) \geq 2-\epsilon$.

Game G_ϵ is defined as follows: there are $n = k(k+1)/2$ items, of which $k^2/4 + k/2$ are white and the remaining $k^2/4$ are black, where k is an even number such that $k \geq 2(2-\epsilon)/\epsilon$. The size of each item is set in such a way that $\kappa = k$.

Let $\boldsymbol{\sigma}^*$ be the strategy profile such that $\emptyset(\boldsymbol{\sigma}^*) = (B_1, \ldots, B_{k/2+1})$ where

- each of the first $k/2$ bins contains $k/2$ white items and $k/2-1$ black items;
- bin $B_{k/2+1}$ contains $k/2$ white items and $k/2$ black items.

[3] Recall that $\mathcal{U}_m^{odd}$ (resp. $\mathcal{U}_m^{even}$) denote the set of all colorful bin packing games with m colors and uniform sizes for which κ is odd (resp. even). Finally, $\mathcal{U}_m = \mathcal{U}_m^{even} \cup \mathcal{U}_m^{odd}$.

We continue by proving that each Nash equilibrium for G_ϵ uses k open bins. Towards this end, assume, by way of contradiction, that there exists a Nash equilibrium with less than $k/2$ full bins. As k is even, each full bin stores exactly $k/2$ white items and $k/2$ black items, so that the set of full bins can store at most $\frac{k^2}{4} - \frac{k}{2}$ items of the same color. It follows that the set of non-full bins have to store at least k white items and at least $k/2$ black ones. Call these items *leftover* items. Let $\widetilde{B}$ be the non-full bin with the highest number of items. Clearly, it contains at most $k-1$ items and we consider two cases: if $\widetilde{B}$ has a black item on top, as $k-1$ is odd, $\widetilde{B}$ contains at most $\frac{k}{2}-1$ white items. An agent owning a leftover white item not packed into $\widetilde{B}$ (there are at least $k/2+1$ such items) has an improving deviation to $\widetilde{B}$ which raises a contradiction. If $\widetilde{B}$ has a white item on top, as $k-1$ is odd, $\widetilde{B}$ contains at most $\frac{k}{2}-1$ black items. An agent owning the only leftover black item not packed into $\widetilde{B}$ has an improving deviation to $\widetilde{B}$ which, again, raises a contradiction. Thus, each Nash equilibrium for G_ϵ uses at least $k/2$ full bins.

Now note that there are not enough black items in G_ϵ to create $k/2+1$ feasible full bins. Hence, since every Nash equilibrium is feasible, it follows that each Nash equilibrium $\boldsymbol{\sigma}$ has exactly $k/2$ full bins, where a total number of $k^2/4$ white items and $k^2/4$ black ones are packed. The remaining $k/2$ white items can be feasibly packed only into singleton bins, thus yielding $\mathsf{F}(\boldsymbol{\sigma}) = k$. By the definition of k, we get $\mathsf{PoS}(G_\epsilon) \geq \frac{k}{k/2+1} \geq 2-\epsilon$. □

We show that, unlike the case of general games considered in the previous section, under the hypothesis of uniform sizes, efficient Nash equilibria are always guaranteed to exist for any number of colors. In particular, we design an algorithm which, given a colorful bin packing game G with uniform sizes, returns a Nash equilibrium $\boldsymbol{\sigma}$ such that $\mathsf{F}(\boldsymbol{\sigma}) \leq 2\mathsf{F}(\boldsymbol{\sigma}^*(G))$ when κ is even and $\mathsf{F}(\boldsymbol{\sigma}) = \mathsf{F}(\boldsymbol{\sigma}^*(G))$ when κ is odd. Given the result on the price of stability of Theorem 5.27, these is the best achievable performance.

Theorem 5.28. *For each $m \geq 2$, $\mathsf{PoS}(\mathcal{U}_m^{even}) \leq 2$ and $\mathsf{PoS}(\mathcal{U}_m^{odd}) = 1$. Moreover, for any game $G \in \mathcal{U}_m$, a Nash equilibrium $\boldsymbol{\sigma}$ such that $\mathsf{F}(\boldsymbol{\sigma}) \leq 2\mathsf{F}(\boldsymbol{\sigma}^*(G))$ if $G \in \mathcal{U}_m^{even}$ and such that $\mathsf{F}(\boldsymbol{\sigma}) = \mathsf{F}(\boldsymbol{\sigma}^*(G))$ if $G \in \mathcal{U}_m^{odd}$ can be computed in pseudo-polynomial time.*

Proof. Fix an integer $m \geq 2$ and a game $G \in \mathcal{U}_m$. We prove the claim by showing that Algorithm 8 computes a Nash equilibrium $\boldsymbol{\sigma}$ for G such that $\mathsf{F}(\boldsymbol{\sigma}) \leq 2\mathsf{F}(\boldsymbol{\sigma}^*(G))$ if $G \in \mathcal{U}_m^{even}$ and such that $\mathsf{F}(\boldsymbol{\sigma}) = \mathsf{F}(\boldsymbol{\sigma}^*(G))$ if $G \in \mathcal{U}_m^{odd}$.

We start by showing that the strategy profile $\boldsymbol{\sigma}$ returned by Algorithm 8 is a Nash equilibrium for G. Let us partition $\boldsymbol{\sigma}$ into three sets, namely Γ, Δ, Θ, where Γ contains all the full bins, Δ contains all the non-full and non-singleton bins and Θ contains all the singleton bins. It is not difficult to see that, by the definition of Algorithm 8, Δ and

Algorithm 8:

Input : A colorful bin packing game G

```
1  X ← {x_1, ..., x_n};
2  i ← 1;
3  c_old ← 0;
4  while (X ≠ ∅) do
5      if (|B_i| < κ) && (∃x_j ∈ X s.t. c_j ≠ c_old)) then
6          c ← most frequent color among the items in X having color other than c_old;
7          Select an item x_j of color c;
8          X ← X \ {x_j};
9          c_old ← c;
10         σ_j ← B_i;
11     else
12         i ← i + 1;
13         c_old ← 0;
14 return σ
```

Θ are such that *(i)* Δ is either empty or contains only one bin, *(ii)* all items stored into bins belonging to Θ have the same color, denoted as c_Θ, *(iii)* the item on top of the bin in Δ (if any) has color c_Θ.

Now assume, by way of contradiction, that there exists an agent j possessing an improving deviation in $\boldsymbol{\sigma}$ towards a bin B_i. Clearly, this can only be possible if x_j is packed into a singleton bin and $B_i \in \Delta \cup \Theta$, but properties (*ii*) and *(iii)* above imply a contradiction. So, $\boldsymbol{\sigma}$ is a Nash equilibrium.

Let $n_z(c)$ be the number of items of color c belonging to X at the zth iteration of Algorithm 8.

Lemma 5.29. *If either* $|\Theta| \geq 2$ *or* $|\Delta| = |\Theta| = 1$*, then the color of each item occupying an odd position in a bin belonging to* $\Gamma \cup \Delta$ *is* c_Θ*.*

Proof. Consider an iteration z of Algorithm 8 such that $c_{old} \neq c_\Theta$ and an item of color $c \neq c_\Theta$ is selected. As color c_Θ is a candidate color among the ones considered at line 6 of the algorithm, it must be $n_z(c) \geq n_z(c_\Theta)$. For the case of $|\Theta| \geq 2$, let $\overline{z}$ be the first iteration at which the algorithm starts constructing singleton bins, while, for the case of $|\Delta| = |\Theta| = 1$, let $\overline{z}$ be the iteration at which the algorithm selects the last item packed into the unique bin in $|\Delta|$. In both cases, it follows that $n_{\overline{z}}(c_\Theta) - n_{\overline{z}}(c) \geq 2$.

Let z' be the first iteration, among the ones realized after iteration z, in which an item of color c_Θ is selected and such that $n_{z'}(c_\Theta) - n_{z'}(c) \geq 2$. Clearly, z' is well-defined because iteration $\overline{z}$ meets the required conditions. This implies that the difference between the number of items of color c and the number of items of color c_Θ selected by Algorithm

8 during all iterations going from z to $z' - 1$ is at least 2. Hence, there is an iteration z'' at which an item of color c is selected despite the fact that $n_{z''}(c_\Theta) = n_{z''}(c) + 1$. By line 5 of the algorithm, this can happen only if $c_{old} = c_\Theta$ which implies that, at iteration $z'' - 1$, an item of color c_Θ is selected which gives $n_{z''-1}(c_\Theta) = n_{z''}(c_\Theta) + 1 = n_{z''}(c) + 2$ which contradicts the minimality of z'.

Hence, we have proved that, at each iteration such that $c_{old} \neq c_\Theta$, Algorithm 8 selects an item of color c_Θ which implies the claim. □

By the previous lemma, we get the following corollary which gives us the number of items of color c_Θ and the number of items of color different that c_Θ.

Corollary 5.30. *If either* $|\Theta| \geq 2$ *or* $|\Delta| = |\Theta| = 1$*, then each bin in* $B_j \in \emptyset(\boldsymbol{\sigma})$ *contains* $\lceil |B_j(\boldsymbol{\sigma})|/2 \rceil$ *items of color* c_Θ*.*

Let $\#c_\Theta$ be the number of items having color c_Θ. We conclude by showing that $\mathsf{F}(\boldsymbol{\sigma}) \leq 2\mathsf{F}(\boldsymbol{\sigma}^*(G))$ when κ is even and that $\mathsf{F}(\boldsymbol{\sigma}) = \mathsf{F}(\boldsymbol{\sigma}^*(G))$ when κ is odd. Towards this end, we use Corollary 5.30 together with the simple basic fact.

Fact 5.31. $2\#c_\Theta \leq n + \mathsf{F}(\boldsymbol{\sigma}^*(G))$.

Let us start with the cases not covered by Corollary 5.30, that is, $|\Theta| = 0$ and $|\Theta| = 1 \wedge |\Delta| = 0$. In both cases, we have $\mathsf{F}(\boldsymbol{\sigma}) = \mathsf{F}(\boldsymbol{\sigma}^*(G))$ independently of the parity of κ, as $\emptyset(\boldsymbol{\sigma})$ contains at most one non-full bin. Thus, in the remaining of the proof, we can assume that Corollary 5.30 holds. Let $\delta \in \{0, \ldots, \kappa - 1\}$ be the number of items stored into the bin belonging to Δ ($\delta = 0$ models the case in which this bin does not exist).

For the case in which κ is odd, by Corollary 5.30, we have $\#c_\Theta = |\Gamma|\frac{\kappa+1}{2} + \lceil \frac{\delta}{2} \rceil + |\Theta|$ and $n = |\Gamma|\kappa + \delta + |\Theta|$. Assume, by way of contradiction, that $\mathsf{F}(\boldsymbol{\sigma}^*(G)) < \mathsf{F}(\boldsymbol{\sigma})$, that is, $\mathsf{F}(\boldsymbol{\sigma}^*(G)) \leq |\Gamma| + |\Delta| + |\Theta| - 1$. By Fact 5.31, we obtain

$$2\left\lceil \frac{\delta}{2} \right\rceil \leq \delta + |\Delta| - 1. \tag{5.5}$$

Now observe that, for $|\Delta| = 0$, which implies $\delta = 0$, (5.5) is not satisfied. Hence, it must be that $|\Delta| = 1$ which, as $|\Theta| \geq 1$ (recall that we are under the hypothesis in which Corollary 5.30 holds), implies that $\delta > 1$. Now, if δ is even, by Corollary 5.30, the item on top of the unique bin in Δ has color different than c_Θ. This means that an agent controlling an item packed into any bin in Θ has an improving deviation by migrating to the unique bin in Δ, thus contradicting the fact that $\boldsymbol{\sigma}$ is a Nash equilibrium. Thus, under the hypothesis of $|\Delta| = 1$ and δ odd, (5.5) is again not satisfied, thus rising a contradiction. Hence, it follows that $\mathsf{F}(\boldsymbol{\sigma}^*(G)) = \mathsf{F}(\boldsymbol{\sigma})$.

For the case in which κ is even, by Corollary 5.30, we have $\#c_\Theta = |\Gamma|\frac{\kappa}{2} + \lceil \frac{\delta}{2} \rceil + |\Theta|$ and $n = |\Gamma|\kappa + \delta + |\Theta|$. As $\mathsf{F}(\boldsymbol{\sigma}^*(G)) \geq |\Gamma| + |\Delta|$, if $|\Gamma| \geq |\Theta|$, it follows that

$$\mathsf{F}(\boldsymbol{\sigma}) = |\Gamma| + |\Delta| + |\Theta| \leq 2|\Gamma| + |\Delta| \leq 2\mathsf{F}(\boldsymbol{\sigma}^*(G)).$$

Thus, in the remaining of the proof, we assume that $|\Theta| > |\Gamma|$. Assume now, by way of contradiction, that $\mathsf{F}(\boldsymbol{\sigma}^*(G)) < \mathsf{F}(\boldsymbol{\sigma})/2$, which implies $\mathsf{F}(\boldsymbol{\sigma}^*(G)) \leq \frac{|\Gamma|+|\Delta|+|\Theta|}{2} - \frac{1}{2}$. By Fact 5.31, we obtain

$$2\left\lceil \frac{\delta}{2} \right\rceil + \frac{|\Theta|}{2} \leq \delta + \frac{|\Gamma| + |\Delta| - 1}{2}. \tag{5.6}$$

Using the hypothesis that $|\Theta| > |\Gamma|$ within (5.6), we obtain

$$2\left\lceil \frac{\delta}{2} \right\rceil \leq \delta + \frac{|\Delta|}{2} - 1 \tag{5.7}$$

which is never satisfied, thus rising a contradiction. Hence, it follows that $\mathsf{F}(\boldsymbol{\sigma}) \leq 2\mathsf{F}(\boldsymbol{\sigma}^*(G))$.

We now argue the complexity of Algorithm 8. We first notice that, for uniform sizes, the compact representation of the input has size $\Omega(m+\log n)$. Moreover, it is easy to see that Algorithm 8 has complexity $O(n)$. It turns out that when, for instance, $m = \Omega(n^{\frac{1}{h}})$, for some constant h, the algorithm has polynomial time complexity. However, when $m = O(\log n)$, the complexity is pseudo-polynomial. □

Theorems 5.27 and 5.28 completely characterize the PoS of colorful bin packing games with uniform sizes. For what concerns the PoA, we also obtain a complete picture by means of the following results.

For the case of at least three colors, the PoA can be arbitrarily high.

Theorem 5.32. *For each $m \geq 3$, both $\mathsf{PoA}(\mathcal{U}_m^{odd})$ and $\mathsf{PoA}(\mathcal{U}_m^{even})$ are unbounded.*

Proof. We show that there exist two colorful bin packing games $G \in \mathcal{U}_3^{even}$ and $G' \in \mathcal{U}_3^{odd}$ whose PoA grows asymptotically with the number of their agents. It is easy to adapt the proof so as to deal with any number of colors $m \geq 3$. Game G is as follows: there are $n = 4k$ agents, all of them of size ϵ, where $k \geq 1$ is a suitable integer, and $\epsilon > 0$ is an arbitrary small real value such that $\sum_{i=1}^{n} s_i \leq 1$, so that $\kappa = 4k$ is even. There are $2k$ items of color c_1 denoted as $b_1, \ldots, b_{2k}$, k items of color c_2 denoted as $w_1, \ldots, w_k$, and k items of color c_3 denoted as $r_1, \ldots, r_k$. Notice that $\mathsf{F}(\boldsymbol{\sigma}^*) = 1$, since all items can be feasibly packed into the same bin, as depicted in Figure 5.3 on the left side. In the same figure, on the right side, is also depicted a packing corresponding to a Nash equilibrium $\boldsymbol{\sigma} \in \mathsf{NE}(G)$ with social cost $\mathsf{F}(\boldsymbol{\sigma}) = 2k = \frac{n}{2}$, which yields $\mathsf{PoA}(G) = \Omega(n)$.

Game G' can be obtained by adding item b_0 of color c_1 to game G. This item can be feasibly packed at the bottom of both bins B_1 depicted in Figure 5.3, so that we get $\kappa = 4k + 1$, which is odd, and $\mathsf{PoA}(G') = \Omega(n)$. □

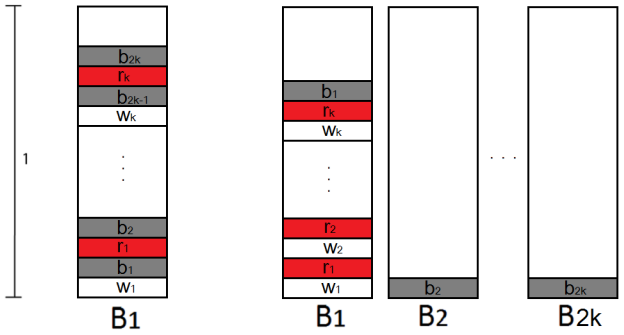

FIGURE 5.3: A colorful bin packing game $G \in \mathcal{U}_3$ with $\mathsf{PoA}(G) \geq \frac{n}{2}$: on the left side the packing corresponding to a social optimum and on the right side the packing corresponding to a Nash equilibrium with social cost $2k$.

For the case of black and white bin packing games, we show a lower bound of 3 on the PoA of games for which κ is an odd number, thus matching the upper bound showed in Theorem 5.11 which holds for general sizes.

Theorem 5.33. $\mathsf{PoA}(\mathcal{U}_2^{odd}) \geq 3$.

Proof. We prove the claim by showing that, for any fixed $\epsilon > 0$, there exists a game $G_\epsilon \in \mathcal{U}_2^{odd}$ such that $\mathsf{PoA}(G_\epsilon) \geq 3 - \epsilon$.

Game G_ϵ is defined as follows: there are $n = k(k+3)/2$ items, of which $(k^2 + 4k - 1)/4$ are white and the remaining $(k+1)^2/4$ are black, where k is an odd number such that $k \geq (8 - 3\epsilon)/\epsilon$. The size of each item is set in such a way that $\kappa = k$.

Let $\boldsymbol{\sigma}^*$ be the strategy profile such that $\emptyset(\boldsymbol{\sigma}^*) = (B_1, \ldots, B_{(k+1)/2+1})$ where

- each of the first $(k+1)/2$ bins contains $(k+1)/2$ white items and $(k-1)/2$ black items;
- bin $B_{(k+1)/2+1}$ contains $(k-1)/2$ white items and $(k+1)/2$ black items.

Since, by definition, at most k items can be packed into a bin and k is odd, we have $\mathsf{F}(\boldsymbol{\sigma}^*(G_\epsilon)) \geq \lceil \frac{n}{k} \rceil = \lceil \frac{k+3}{2} \rceil = (k+1)/2 + 1$, so that $\boldsymbol{\sigma}^*$ is a social optimum.

Let $\boldsymbol{\sigma}$ be the strategy profile such that $\emptyset(\boldsymbol{\sigma}^*) = (B_1, \ldots, B_{(3k+1)/2})$ where

- each of the first $(k+1)/2$ bins contains $(k-1)/2$ white items and $(k+1)/2$ black items;
- each of the remaining k bins contains a single white item.

Since all of the first $(k+1)/2$ bins are full and the remaining k bins are singletons and contain only white items, it follows that $\boldsymbol{\sigma}$ is a Nash equilibrium. By the definition of k, we get $\mathsf{PoA}(G_\epsilon) \geq \frac{3k+1}{k+3} \geq 3-\epsilon$. □

For the leftover case of black and white bin packing games for which κ is even, we show that the upper bound on the PoA drops to 2 which matches the lower bound given in Theorem 5.27 for the PoS.

Theorem 5.34. $\mathsf{PoA}(\mathcal{U}_2^{even}) \leq 2$.

Proof. Fix a game $G \in \mathcal{U}_2^{even}$ and a Nash equilibrium $\boldsymbol{\sigma}$ for G. Let us partition $\boldsymbol{\sigma}$ into three sets, namely Γ, Δ, Θ, where Γ contains all the full bins, Δ contains all the non-full and non-singleton bins and Θ contains all the singleton bins. It is not difficult to see that, by the fact that $\boldsymbol{\sigma}$ is a Nash equilibrium, Δ and Θ are such that *(i)* Δ is either empty or contains only one bin, *(ii)* all items stored into bins belonging to Θ have the same color, which we assume without loss of generality to be black, *(iii)* the item on top of the bin in Δ (if any) is black. Let $\delta \in \{0, \ldots, \kappa-1\}$ be the number of items packed into the unique bin in Δ ($\delta = 0$ model the case in which $\Delta = \emptyset$). Since κ is even and the item on top of the bin in Δ (if any) is black, we have $\#B = |\Gamma|\frac{\kappa}{2} + \lceil \frac{\delta}{2} \rceil + |\Theta|$ and $\#W = |\Gamma|\frac{\kappa}{2} + \lfloor \frac{\delta}{2} \rfloor$. By substituting the values within inequality (5.1), we obtain

$$|\Gamma|\frac{\kappa}{2} + \left\lceil \frac{\delta}{2} \right\rceil + |\Theta| \leq |\Gamma|\frac{\kappa}{2} + \left\lfloor \frac{\delta}{2} \right\rfloor + \mathsf{F}(\boldsymbol{\sigma}^*(G))$$

which implies $\mathsf{F}(\boldsymbol{\sigma}^*(G)) \geq |\Theta|$. Moreover, as all bins in Γ is full, we have $\mathsf{F}(\boldsymbol{\sigma}^*(G)) \geq |\Gamma| + |\Delta|$, so that $\mathsf{F}(\boldsymbol{\sigma}^*(G)) \geq \max\{|\Theta|, |\Gamma| + |\Delta|\}$. By the arbitrariness of $\boldsymbol{\sigma}$, we obtain

$$\mathsf{PoA}(G) = \frac{\mathsf{F}(\boldsymbol{\sigma})}{\mathsf{F}(\boldsymbol{\sigma}^*(G))} \leq \frac{|\Gamma| + |\Delta| + |\Theta|}{\max\{|\Theta|, |\Gamma| + |\Delta|\}} \leq 2.$$

□

Chapter 6

Conclusion

In the era of networks and distributed systems, there is a growing interest among computer scientists on the study of problems motivated by the social and technological development of the recent years, such as social networks and internet of things. There is a massive amount of real-life scenarios where we need to optimize resources which can be physically distributed in different places, or needs to be used by a large number of users or eventually can be managed by independent agents. Moreover, those agents could be either selfish or can act in a cooperative manner. To tackle all these problems, the research community developed a considerable body of literature. Part of the literature involves new problems, but many results are based on variation of classical optimization problems, which are now considered in their applications in network scenarios. In this dissertation the main focus is on multi-agent problems and algorithmic game theory. In particular, we tackle settings related to two well-known NP-hard problems: the Maximum Coverage problem and the Bin Packing problem.

The first contribution of the thesis regards a problem that we call multi-agent coverage. In this setting we begin by considering the optimization problem of maximizing the sum of the agents' revenue. For this problem we provide a polynomial algorithm that achieves an approximation factor of $1 - \frac{1}{\sqrt{e}}$ if the maximum cost of a container is upper-bounded by the minimum budget of an agent, and provide a polynomial Monte-Carlo algorithm with an expected approximation factor of $1 - \frac{1}{e}$. We also define distributed and proportional multi-agent coverage games, and investigate the existence and the quality of Nash equilibria. We show that the distributed game can be modeled as a distributed welfare game ([55]) and the existence of a Nash equilibrium is always guaranteed. We show a tight bound of $2 - \frac{1}{k}$ for price of anarchy and stability. We prove that in the proportional game the existence of a Nash equilibrium is not guaranteed.

The main open problems are that of finding a deterministic constant approximation algorithm for the general case and, in the case when $c_{\max} \leq b_{\min}$, closing the gap between

our deterministic $\left(1-\frac{1}{\sqrt{e}}\right)$-approximation algorithm and the $1-\frac{1}{e}$ hardness of approximation for the maximum coverage problem ([3]). One possible direction could be that of exploiting the framework of multilinear relaxation and contention resolution schemes given in [66]. We observe that the centralized multi-agent coverage problem cannot be directly reduced to any of the problems solved in [66], as the constraints of our problem cannot be directly modeled as knapsack, matroid, or sparse packing constraints (or their intersection). However, this does not exclude the existence of a suitable contention resolution scheme for our problem. Moreover, extending the definition of multi-agent coverage to more general utility functions (e.g. general submodular functions) is worth further investigation. Another interesting research direction regards the possibility of studying the existence of approximate Nash equilibria in proportional games.

In chapter 4 we consider the generalized budgeted submodular set function maximization problem (GBSM), where the goal is maximizing the value of a submodular function defined over the covered elements, with the constraint that the overall cost of the covered elements is at most a given budget. We introduce an algorithm that achieves an approximation factor of $\frac{1}{2}\left(1-\frac{1}{e^{\alpha}}\right)$, where $\alpha \leq 1$ is the approximation factor of an algorithm for a sub-problem. To cope with the sub-problem we show two polynomial-time algorithms. The first one guarantees $\alpha = 1-\epsilon$ if the costs satisfy a specific constraint that is often fulfilled in relevant scenarios. For the general case we give an algorithm that guarantees $\alpha = 1-\frac{1}{e}-\epsilon$. Finally we give a $\frac{1}{2}\left(1-\frac{1}{e^{\alpha\beta}}\right)$-approximating bi-criterion algorithm, in which we are allowed to spend an extra budget up to a factor $\beta \geq 1$. We are able to achieves an approximation factor of $\frac{1}{2}-\epsilon$, for any arbitrarily small $\epsilon > 0$, setting $\beta = \frac{1}{\alpha}\ln\left(\frac{1}{2\epsilon}\right)$.

The main open problem is to close the gap between the known hardness result of $1-\frac{1}{e^{\alpha}}$, where $\alpha = 1$ for the MC problem [3] and $\alpha = 1-\frac{1}{e}$ for the non-stochastic adaptive seeding problem with knapsack constraint problem [65], and our approximation bound of $\frac{1}{2}\left(1-\frac{1}{e^{\alpha}}\right)$. One possibility to get rid of the $\frac{1}{2}$ factor could be to use the partial enumeration technique exploited in specific subproblems (e.g. budgeted maximum coverage problem [8] and monotone submodular set function subject to a knapsack constraint maximization problem [17]). However, this requires that each greedy step selects a single element of X, to be added to a partial solution X', while our greedy algorithm selects a subset of $X \setminus X'$ that maximizes the ratio between its marginal increment in the objective function and its marginal cost. Note that this set can contain more than one element in order to ensure that the ratio is non-increasing at each iteration of the greedy algorithm, which is needed to apply the analysis in [8] and [17]. Other research directions, that deserve further investigation, include the study of the GBSM considering different cost functions and also different objective functions where the profit given by an element x depends on the container s which it is associated with.

The last problem considered in the thesis are Colorful Bin Packing games, where each

agent controls an indivisible item and the items are to be packed in unitary bins. We consider two different cost functions, called proportional and egalitarian, and we show that in either case the game do not converge in general to a Nash equilibrium. However, the Nash equilibria are always guaranteed to exist. We give algorithms to build Nash equilibria: for the egalitarian function we show a polynomial time algorithm, while we show a pseudo-polynomial algorithm for the proportional function. We investigate the quality of the equilibria showing that the prices of anarchy and stability are unbounded under both cost functions with at least 3 colors. However, the price of anarchy and stability are equal to 3 in the case of black and white games. Finally, we cope with the subcase of games with uniform sizes and give a complete characterization of the efficiency of Nash equilibria. We also give an algorithm that returns Nash equilibria with the best achievable performance.

Bibliography

[1] Francesco Cellinese, Gianlorenzo D'Angelo, Gianpiero Monaco, and Yllka Velaj. Generalized budgeted submodular set function maximization. In *Proceedings of the 43rd International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 117 of *LIPIcs*, pages 31:1–31:14, 2018.

[2] Vittorio Bilò, Francesco Cellinese, Giovanna Melideo, and Gianpiero Monaco. On colorful bin packing games. In *International Computing and Combinatorics Conference*, pages 280–292. Springer, 2018.

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

[4] Bernhard Korte, Jens Vygen, B Korte, and J Vygen. *Combinatorial optimization*, volume 2. Springer, 2012.

[5] D.S. Hochbaum. *Approximation Algorithms for NPHard Problems.* PWS Publishing Company,, Boston, MA, USA, 1997.

[6] David Kempe, Jon M. Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. *Theory of Computing*, 11:105–147, 2015.

[7] G.L. Nemhauser, L.A. Wolsey, and M.L. Fisher. An analysis of approximations for maximizing submodular set functions–I. *Mathematical Programming*, 14(1): 265–294, 1978.

[8] Samir Khuller, Anna Moss, and Joseph Seffi Naor. The budgeted maximum coverage problem. *Information Processing Letters*, 70(1):39–45, 1999.

[9] Reuven Cohen and Liran Katzir. The generalized maximum coverage problem. *Information Processing Letters*, 108(1):15–22, 2008.

[10] G. Goel, C. Karande, P. Tripathi, and L. Wang. Approximability of combinatorial problems with multi-agent submodular cost functions. In *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2009.

[11] Gagan Goel, Pushkar Tripathi, and Lei Wang. Combinatorial problems with discounted price functions in multi-agent systems. In *Proceedings of the Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, pages 436–446.

[12] Richard Santiago and F. Bruce Shepherd. Multi-agent submodular optimization. In *Proceedings of the 21st International Conference on Approximation Algorithms for Combinatorial Optimization Problems (APPROX/RANDOM)*, volume 116 of *LIPIcs*, pages 23:1–23:20.

[13] D. Chakrabarty and G. Goel. On the approximability of budgeted allocations and improved lower bounds for submodular welfare maximization and gap. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2008.

[14] Jan Vondrak. Optimal approximation for the submodular welfare problem in the value oracle model. In *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing (STOC)*, pages 67–74, 2008.

[15] Ashwinkumar Badanidiyuru, Christos H. Papadimitriou, Aviad Rubinstein, Lior Seeman, and Yaron Singer. Locally adaptive optimization: Adaptive seeding for monotone submodular functions. In *27th ACM-SIAM Symp. on Disc. Alg., (SODA)*, pages 414–429, 2016.

[16] Lior Seeman and Yaron Singer. Adaptive seeding in social networks. In *IEEE 54th Symp. on Foundations of Computer Science (FOCS)*, pages 459–468. IEEE, 2013.

[17] Maxim Sviridenko. A note on maximizing a submodular set function subject to a knapsack constraint. *Operation Research Letters*, 32(1):41–43, 2004.

[18] Rishabh K. Iyer and Jeff A. Bilmes. Submodular optimization with submodular cover and submodular knapsack constraints. In *27th Annual Conference on Neural Information Processing Systems (NIPS)*, pages 2436–2444, 2013.

[19] EG Coffman Jr, MR Garey, and DS Johnson. Approximation algorithms for bin packing: A survey. *Approximation Algorithms for NP-Hard Problems*, pages 46–93, 1996.

[20] Vittorio Bilò. On the packing of selfish items. In *Parallel and Distributed Processing Symposium, 2006. IPDPS 2006. 20th International*, pages 9–pp. IEEE, 2006.

[21] Leah Epstein and Elena Kleiman. Selfish bin packing. *Algorithmica*, 60(2):368–394, 2011.

[22] György Dósa and Leah Epstein. The convergence time for selfish bin packing. In *International Symposium on Algorithmic Game Theory*, pages 37–48. Springer, 2014.

[23] Ruixin Ma, György Dósa, Xin Han, Hing-Fung Ting, Deshi Ye, and Yong Zhang. A note on a selfish bin packing problem. *Journal of Global Optimization*, 56(4): 1457–1462, 2013.

[24] Gyorgy Dosa and Leah Epstein. Generalized selfish bin packing. *arXiv preprint arXiv:1202.4080*, 2012.

[25] Martin Böhm, Jiří Sgall, and Pavel Veselỳ. Online colored bin packing. In *International Workshop on Approximation and Online Algorithms*, pages 35–46. Springer, 2014.

[26] Martin Böhm, György Dósa, Leah Epstein, Jiří Sgall, and Pavel Veselỳ. Colored bin packing: Online algorithms and lower bounds. *Algorithmica*, 80(1):155–184, 2018.

[27] György Dósa and Leah Epstein. Colorful bin packing. In *Scandinavian Workshop on Algorithm Theory*, pages 170–181. Springer, 2014.

[28] Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay V Vazirani. *Algorithmic game theory*. Cambridge University Press, 2007.

[29] Luca Moscardelli. *The Impact of Non-Cooperativeness and of Limited Resources and Social Knowledge on Distributed Systems: Performances and Complexity*. PhD thesis, University of L'Aquila, IT, 2008. URL http://www.moscardelli.it/it/pubb.htm.

[30] John F Nash et al. Equilibrium points in n-person games. *Proceedings of the national academy of sciences*, 36(1):48–49, 1950.

[31] Robert W Rosenthal. A class of games possessing pure-strategy nash equilibria. *International Journal of Game Theory*, 2(1):65–67, 1973.

[32] Christos H Papadimitriou. On inefficient proofs of existence and complexity classes. In *Annals of Discrete Mathematics*, volume 51, pages 245–250. Elsevier, 1992.

[33] Xi Chen, Xiaotie Deng, and Shang-Hua Teng. Settling the complexity of computing two-player nash equilibria. *Journal of the ACM (JACM)*, 56(3):14, 2009.

[34] Elias Koutsoupias and Christos Papadimitriou. Worst-case equilibria. In *Annual Symposium on Theoretical Aspects of Computer Science*, pages 404–413. Springer, 1999.

[35] V Kann. Maximum bounded 3-dimensional matching is max snp-comple. *Information Processing Letters*, 37:27–35, 1991.

[36] Ioannis Caragiannis. Wavelength management in WDM rings to maximize the number of connections. *SIAM J. Discrete Math.*, 23(2):959–978, 2009.

[37] Ioannis Caragiannis and Gianpiero Monaco. A 6/5-approximation algorithm for the maximum 3-cover problem. *J. Comb. Optim.*, 25(1):60–77, 2013.

[38] Chandra Chekuri and Amit Kumar. Maximum coverage problem with group budget constraints and applications. In *7th Intl. Work. on Approximation Algorithms for Combinatorial Optimization Problems, APPROX*, pages 72–83, 2004.

[39] Irving van Heuven van Staereling, Bart de Keijzer, and Guido Schäfer. The Ground-Set-Cost Budgeted Maximum Coverage Problem. In *41st International Symposium on Mathematical Foundations of Computer Science (MFCS 2016)*, volume 58 of *LIPIcs*, pages 50:1–50:13, 2016.

[40] George L. Nemhauser, Laurence A. Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical Programming*, 14(1):265–294, 1978.

[41] D Kempe, J Kleinberg, and E Tardos. Maximizing the spread of influence in a social network. In *Proceeding 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 137–146, 2003.

[42] David Kempe, Jon Kleinberg, and Éva Tardos. Influential nodes in a diffusion model for social networks. In *International Colloquium on Automata, Languages, and Programming*, pages 1127–1138. Springer, 2005.

[43] Eytan Bakshy, Jake M Hofman, Winter A Mason, and Duncan J Watts. Everyone's an influencer: quantifying influence on twitter. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 65–74. ACM, 2011.

[44] Ittai Abraham, Shiri Chechik, David Kempe, and Aleksandrs Slivkins. Low-distortion inference of latent similarities from a multiplex social network. *SIAM Journal on Computing*, 44(3):617–668, 2015.

[45] Manuel Gomez-Rodriguez, Jure Leskovec, and Bernhard Schölkopf. Modeling information propagation with survival theory. In *ICML (3)*, pages 666–674, 2013.

[46] Manuel Gomez Rodriguez, Jure Leskovec, and Andreas Krause. Inferring networks of diffusion and influence. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1019–1028. ACM, 2010.

[47] Jure Leskovec, Lada A Adamic, and Bernardo A Huberman. The dynamics of viral marketing. *ACM Transactions on the Web (TWEB)*, 1(1):5, 2007.

[48] Elchanan Mossel and Sebastien Roch. On the submodularity of influence in social networks. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 128–134. ACM, 2007.

[49] Michael Mathioudakis, Francesco Bonchi, Carlos Castillo, Aristides Gionis, and Antti Ukkonen. Sparsification of influence networks. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 529–537. ACM, 2011.

[50] Ning Chen. On the approximability of influence in social networks. *SIAM Journal on Discrete Mathematics*, 23(3):1400–1415, 2009.

[51] Jure Leskovec, Andreas Krause, Carlos Guestrin, Christos Faloutsos, Jeanne VanBriesen, and Natalie Glance. Cost-effective outbreak detection in networks. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 420–429. ACM, 2007.

[52] Martin Gairing. Covering games: Approximation through non-cooperation. In *Proceedings of the 5th International Workshop on Internet and Network Economics (WINE)*, pages 184–195, 2009. doi: 10.1007/978-3-642-10841-9_18.

[53] Chandra Chekuri and Amit Kumar. Maximum coverage problem with group budget constraints and applications. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 72–83. Springer, 2004.

[54] Boaz Farbstein and Asaf Levin. Maximum coverage problem with group budget constraints. *Journal of Combinatorial Optimization*, 34(3):725–735, 2017.

[55] Jason R. Marden and Adam Wierman. Distributed welfare games. *Operations Research*, 61(1):155–168, 2013. doi: 10.1287/opre.1120.1137.

[56] David S Johnson. *Near-optimal bin packing algorithms*. PhD thesis, Massachusetts Institute of Technology, 1973.

[57] Michael R Garey, Ronald L Graham, and Jeffrey D Ullman. Worst-case analysis of memory allocation algorithms. In *Proceedings of the fourth annual ACM symposium on Theory of computing*, pages 143–150. ACM, 1972.

[58] János Balogh, József Békési, György Dósa, Leah Epstein, Hans Kellerer, and Zsolt Tuza. Online results for black and white bin packing. *Theory of Computing Systems*, 56(1):137–155, 2015.

[59] János Balogh, József Békési, György Dósa, Leah Epstein, Hans Kellerer, Asaf Levin, and Zsolt Tuza. Offline black and white bin packing. *Theoretical Computer Science*, 596:92–101, 2015.

[60] Nicolaos Matsakis. *Approximation algorithms for packing and buffering problems.* PhD thesis, University of Warwick, UK, 2015. URL http://wrap.warwick.ac.uk/82141/.

[61] János Balogh, József Békési, Gyorgy Dosa, Hans Kellerer, and Zsolt Tuza. Black and white bin packing. In *International Workshop on Approximation and Online Algorithms*, pages 131–144. Springer, 2012.

[62] Leah Epstein. Bin packing games with selfish items. In *International Symposium on Mathematical Foundations of Computer Science*, pages 8–21. Springer, 2013.

[63] Guosong Yu and Guochuan Zhang. Bin packing of selfish items. In *International Workshop on Internet and Network Economics*, pages 446–453. Springer, 2008.

[64] Dov Monderer and Lloyd Shapley. Potential games. *Games and Economic Behavior*, 14(1), 1996.

[65] Aviad Rubinstein, Lior Seeman, and Yaron Singer. Approximability of adaptive seeding under knapsack constraints. In *16th ACM Conf. on Economics and Computation*, pages 797–814. ACM, 2015.

[66] Chandra Chekuri, Jan Vondrák, and Rico Zenklusen. Submodular function maximization via the multilinear relaxation and contention resolution schemes. *SIAM J. Comput.*, 43(6):1831–1879, 2014.