DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF LIVERPOOL

# Optimization problems in network mechanism design

**Eleftherios Anastasiadis**

A thesis presented for the degree of
Doctor of Philosophy

September 2016

To my parents

# Abstract

We study approximation algorithms and design truthful mechanisms for optimization problems in networks that have direct applications in smart cities and urban planning. We present new models and new techniques which could be of independent interest.

More specifically, in Chapter 2 we introduce a new model for pollution control and propose two applications of this model. This is the first time this problem is studied from the computational perspective. The network is represented by a graph where nodes are the pollutants and edges between pollutants represent the effect of spread of pollution. The government sets bounds on the levels of emitted pollution in both local areas and the whole network. We mainly study the classes of planar graphs and trees which model air and water pollution and design truthful approximate mechanisms.

In Chapter 3 we introduce a new mechanism design model for a new model for the budgeted maximum lifetime coverage (BMLC) in wireless sensor networks (wsns). BMLC generalizes the known maximum lifetime coverage problem to the case where sensors are owned by selfish agents, where each agent has a private cost per unit time of how much to be paid for deploying his sensor. We introduce a random instances model for BMLC and design a novel approximate mechanism by reducing BMLC to the fractional knapsack which is truthful under some technical assumptions. For a closely related minimum coverage problem in wsns on unit disk graphs, we generalize a recent PTAS for this problem to obtain a truthful PTAS for the problem where sensors' costs are agents' private data.

In Chapter 4 we study approximation algorithms which are based on the primal dual method for network connectivity problems. We then prove that these algorithms are monotone and thus can lead to truthful mechanisms.

Finally in Chapter 5 we study the problem of facility location on the real line under non utilitarian objective functions. We extend previous models and derive inapproximability bounds for deterministic and randomized truthful mechanisms. As a byproduct we show that the same approximation guarantees hold for the social utility objective.

# Acknowledgments

I would first like to thank my supervisor Professor Piotr Krysta who introduced me to the area of Mechanism Design with the very interesting problems. I am grateful for his advice on research and his inspiring suggestions. I deeply appreciate his dedication and continuous efforts for the successful completion of the present work. I also thank him for his care for me to ensure financial aid so as I would carry on with research without problems. Without his guidance this thesis would not have been completed.

I am deeply indebted to Professor Paul Spirakis for his willingness to advise me and for his interest and care on the progress of the project. His experience resulted in very fruitful discussions.

I would also like to thank my second supervisor Dr. Martin Gairing and my advisors Dr. George Christodoulou and Dr. Rahul Savani for their effective advice and useful discussions we had during all four years of this research project.

My thanks with respect and appreciation to Professor Xiaotie Deng and Dr. Dionisis Kandris for their willingness to help. I would also like to thank my collaborators Dr. Minming Li, Dr. Han Qiao, Dr. Jinshan Zhang and my good friends Argyris Deligkas and Alkmini Sgouritsa for the lots of hours of fruitful discussions we had in the office.

Last but not least, I would like to thank my parents. My mother for her support and my father for his encouragement and his precious advice for completion of the thesis from the very beginning of the PhD.

# Contents

# List of Figures

# List of Algorithms

# Chapter 1

# Introduction

Internet is the most complex and valuable artifact among all the computational systems. This network of networks that grew with an immense speed since mid 1990's, has become undoubtedly a necessary means of communication in every day life. Its universality, its availability as well as its robustness have attracted rapidly the global economy. Through the years the economic activity has increased surprisingly on the Internet. Many markets have now moved to the Internet and plenty of new online stores have opened. Except from being an unlimited source of information and bringing the global communications at a higher level, its bloom coincided with the inception of online markets, e-services and e-commerce. It has become an environment where entities with diverse economic interests and goals interact. As Papadimitriou points out *"The most novel and defining characteristic of the Internet is its socio-economic complexity"* [114]. Game Theory and Mathematical Economics have been proved useful to model the online markets resulting in what is today called *Internet economics*. But how did this term arise?

Over the last decades one of the directions that Theoretical Computer Science has focused on, was the design of efficient algorithms for computationally intractable problems. Challenging and important problems have been studied extensively from an algorithmic perspective; the scheduling of tasks to machines, the allocation of memory to computer systems or the routing of messages in network environments are some paradigmatic examples. The input to these problems has been taken for granted without taking into account the human factor and the presence of incentives. In environments with multiple participating entities that require services from the owners of the resources, an algorithm must consider the different preferences of every participant. The environment can be seen as a somewhat multidimensional chessboard where the participating entities are the players, each having as goal to "win" in the game. Each of them will not play fair if cheating is more profitable. However the rules can change in a way that cheating is not beneficial. This is the desired outcome for the central authority who designs the game. In a larger scale the situation is pretty much similar.

On the other side, economic theory has given solutions to problems in which different entities interact, such as auctions, supply chains or types of markets to name a few. The field of game theory and economics which studies the design of economic mechanisms towards the optimization of a given objective in settings where the players act strategically is called *Mechanism Design*. Mechanism design studies solution concepts in a class of games where there is some information that is kept private by the participating players. A central authority, known in the literature as the *mechanism designer* who is interested in the outcome of a game, is responsible for the design of its structure. As Leonid Hurwicz points out in mechanism design the goal is given and the mechanism is unknown, in contrast to the traditional game theory where the attention is drawn in the analysis of a mechanism. This is the reason why mechanism design is also called *reverse game theory*.

In problems like the house allocation or facility location a mechanism is simply an algorithm. This area of problems is known under the term *mechanism design without money*. However, due to the impossibility results by Gibard [59] and Satterthwaite [124] the design of mechanisms that fulfill desired economic properties are restricted to very specific ones. In order to overcome this obstacle there are several ways one of which is the addition of money and thus the introduction of payments in the mechanisms. For problems such as combinatorial auctions and bilateral trading a mechanism is simply speaking an algorithm together with a payment scheme. This area of problems is known under the term *mechanism design with money*.

Previous studies had not taken into account the computational issues of computing a solution for a mechanism design problem. This factor was not considered until a decade ago. The works by Koutsoupias and Papadimitriou [87], Roughgarden and Tardos [123] and Nisan and Ronen [111] were the first to study algorithmic aspects of game theory and economic concepts that arise from problems on the Internet, establishing the field of *Algorithmic Game Theory*. Specifically [111] studied the design of computationally efficient mechanisms in the presence of incentives and applied the tools of mechanism design to algorithmic problems establishing the area of *Algorithmic Mechanism Design*.

Algorithmic Mechanism Design models mathematically problems where both computational and economical issues coincide. A direct application is on the Internet. As a first example, the sponsored search auctions taking place in search engines every second are one of the problems where Algorithmic Mechanism Design can be applied to. The applications of this new area are numerous and not limited to the Internet. Auctions constitute a problem of major importance that has been extensively studied, with applications in various sectors, apart from sponsored search. The sale of licences for the use of a band in the electromagnetic spectrum by telecommunication companies (spectrum auctions), the purchase of transportation services by a company from a number of bidding suppliers (transportation auctions) and the purchase of paths to

connect pairs of specified nodes in a communication network are some paradigmatic examples [113].

In the aforementioned applications the information which is given as input to the mechanisms (roughly speaking the algorithms that solve the problem) is provided by the participating entities, which from now on will be called *agents*[1]. The auctioneer, which in this case is the mechanism designer, aims at computing an outcome that is optimal according to an objective, e.g. his revenue, or the overall satisfaction of the agents, a notion termed in the literature as the *social welfare*. In general, every agent has a preference for each of the outcomes and thus orders them accordingly. Each one of them declares this information (for cardinal mechanisms studied in this thesis this preference is expressed by a number) to the mechanism designer and the latter computes an outcome based on it. There are two issues that arise:

1. First, the declared information might not correspond to the true preferences of the agents. Every agent acts selfishly and will try to manipulate the mechanism by lying, if this strategy of his will result in a more beneficial outcome for him.

2. Second, the underlying problem that needs to be solved is not always tractable.

What is needed is the design of a mechanism that will find efficiently a solution, possibly not the optimal, with respect to the objective, but which guarantees that no agent will have an incentive to lie about his preferences. In other words, an agent cannot gain by lying. In Mechanism Design the mechanisms that fulfill this property are called *truthful*. For the hard optimization problems, one way is to design first an approximation algorithm i.e. an algorithm that approximates the objective of the mechanism designer and then modify it accordingly to achieve the property of truthfulness. We note there is no general way to convert an algorithm to a truthful mechanism and so far there have been only seldom works in this direction. However, this task is very demanding and for many problems there is a trade-off in between efficiency, approximation and truthfulness. Roughly speaking, we cannot always achieve them all, thus we need to sacrifice one, or two of these factors in order to obtain the third e.g. get a worse approximation for the underlying optimization problem in order to obtain a truthful mechanism.

In this thesis we follow this line of research drawing our attention to Algorithmic Mechanism Design aiming at designing approximation algorithms and truthful mechanisms for optimization problems on networks. We extend the existing works, we propose new network models and design truthful mechanisms. The ultimate goal is to get a better understanding on what criteria a mechanism should fulfill in order to be truthful.

[1]For simplicity reasons we ascribe all agents the property of masculine gender.

## 1.1 Urban planning

Governments and local authorities in many cities worldwide have managed to ensure the success of their countries and cities in social, economic and environmental terms. Urban planning has been proved a very important tool that contributes to the overall urban planning modelling. With optimization being the base of sketch modelling, social planners and regulators are enabled to find solutions to several urban planning problems establishing minimum resource consumption, efficient routing of data on sensor devices, the control of pollution and the socially "suitable" positioning of facilities. Keirstead and Shah in a recent survey demonstrate the importance of optimization techniques in urban planning modelling [78]. In this thesis we study optimization problems on networks all related to urban planning. We extend previous models and introduce new ones investigating in parallel the underlying game and the incentives of the participants in it. Since most of these problems are intractable, we provide approximate solutions (in some cases the best we can hope for) and design mechanisms based on existing approximation algorithms or on their modifications.

One of the most important environmental problems in urban planning is the control of emitted pollution by industrial units and means of transportation. In Chapter 3 we introduce a new model for this problem. The government as a regulatory authority of a country, is responsible to make efficient environmental policies to ensure that a balance between the economic growth and the protection of the environment. To achieve this goal, restrictions are set on the allowable levels of pollution globally in whole country as well as locally in small regions along a country. Factories and cars consist the main sources of pollution. The government auctions a fixed number of licences for factories each one corresponding to a ton of $CO_2$. The factory owners have to comply with these pollution allowances, otherwise they are charged with a fine for any extra emission in the environment. If theses levels are exceeded significantly, then the government is responsible for shutting down the factories that are most harmful for the environment. Furthermore, pollution licences are distributed to the mayors who then sell them to car drivers.

The overall problem has been studied mainly in the environmental economics literature where the methodology of game theory is applied for the pollution control. In our network game model, we not only study the problem from an economic perspective but also make the first attempt (to the best of our knowledge) to analyze algorithmically pollution control from the perspective of the regulator. In the underlying game, each of the players (i.e. factory owners, mayors of cities) declares to the regulator his cost of cleaning the area around him (around the factory for the owner and around a city for the mayor). Each player has as strategy to be allocated as many licence permits as possible and thus might lie for a more profitable outcome for him. We derive algorithms that approximate the optimization problem and mechanisms for the game, studying it

from a computational perspective (to the best of our knowledge this is the first time in the literature).

The next problem after controlling pollution, is the detection of pollution levels in the atmosphere and waters as well as the detection of fire in forests. This is achieved through wireless sensor nodes i.e. multi-functional sensing devices that can measure the temperature, the humidity and the pollution levels in the atmosphere. In Chapter 4 we focus in two problems that have been extensively studied in the literature and investigate them from the mechanism design perspective. More precisely, we first study the problem of lifetime maximization i.e. the problem of finding a proper schedule of active/sleep wireless sensor nodes that monitor a set of target points in a predefined area. The goal of a base station, who acts as a regulatory authority, is to monitor these targets for the maximum time possible, known as the lifetime of the network under the battery constraints of the sensor nodes. We introduce the underlying game of the problem, in which each player (i.e. owner of a sensor node) declares an amount of cost per unit time of monitoring his sensing area, which is then paid by the base station. This cost consists a private information of the players, who may strategically lie about it to the base station in order to be paid more. We introduce a random instances model for this problem and design a novel approximate mechanism for the game.

We also extend this problem by introducing a budget constraint on the amount of money that the base station is allowed to spend. The second problem that we study is that of the weighted sensor cover in the plane. Given a set of target points and a set of weighted sensor nodes, we are asked to find a subset of the sensor nodes that can monitor all the target points having the minimum possible weight. We introduce the underlying game of this problem providing also a mechanism for it. The mechanism is based on the modification of a recently designed algorithm for the optimization problem. This modification does not affect the approximation of the produced solution.

After setting a wireless sensor network, the next step is to ensure that, once a signal is received or a measurement exceeds a specified level, this message will be broadcast in the network and routed to the base station as soon as possible. In this new problem we do not ask for an energy efficient solution but rather for the cheapest one. In large networks the components are heterogeneous and thus can be owned by different owners. The problem also differs from the previous one in that the owners do not possess nodes of the network (sensors in the previous problems) but rather its edges, which in this case are the interconnecting routes between sensor nodes in the network. In Chapter 5 we survey algorithms for these network design problems based on a specific algorithmic technique and introduce the underlying game. We prove that their approximation guarantee is preserved when we design the mechanisms for the game.

In cities with large populations it is the responsibility of mayoralty, which acts as regulatory authority, to make a plan for the location of facilities such as schools, facto-

ries, gym courts etc. in order that fairness is preserved among its inhabitants. Consider for example the case where the government is planning to build a school and a factory on a street. Citizens' preferences for these facilities might significantly differentiate. Those who work at the factory and also have children that go to school wish both facilities to be built close to their homes. Citizens without children might want the school to be build far because of the noise. Finally, those who do not work at the factory, prefer its location to be far from their home to avoid the emitted pollution. It is clear that preferences of the inhabitants are heterogeneous since each one of them might want to be close to a facility, be away from it, or be indifferent about its presence. In Chapter 6 we extend previous models for the problem of heterogeneous facility location, we derive inapproximability results and design mew mechanisms for this problem. We also note that in contrast to the other three Chapters, in the designed mechanisms of Chapter 6 there is no presence of money.

This thesis is based on the following papers:

- Eleftherios Anastasiadis, Xiaotie Deng, Piotr Krysta, Minming Li, Han Qiao, and Jinshan Zhang. "Network Pollution Games." In Proceedings of the International Conference on Autonomous Agents & Multiagent Systems, pp. 23-31. International Foundation for Autonomous Agents and Multiagent Systems, 2016.

- Eleftherios Anastasiadis, Xiaotie Deng, Piotr Krysta, Minming Li, Han Qiao, and Jinshan Zhang. "New Results for Network Pollution Games." In Computing and Combinatorics: 22nd International Conference, COCOON 2016, Ho Chi Minh City, Vietnam, August 2-4, 2016, Proceedings, vol. 9797, p. 39. Springer, 2016.

- Eleftherios Anastasiadis, Dionisis Kandris, Piotr Krysta, Paul Spirakis. "Mechanism design for coverage problems in wireless sensor networks", Unpublished manuscript.

- Eleftherios Anastasiadis, Argyrios Deligkas. "Maxmin Heterogeneous Facility Location Games on the Line". Submitted.

# Chapter 2

# Background

We continue by giving some basic definitions from optimization and mechanism design which are necessary for the chapters that follow. We note that some of the definitions are for maximization problems. For minimization problems the definition can be derived in a similar way.

## 2.1 Optimization problems

We study constrained optimization problems under the presence of incentives that can be formulated as Linear Programs (LPs) or an Integer Linear Programs (ILPs). More formally let $x \in \mathbb{R}^n$ be a vector of $n$ variables, $A \in \mathbb{N}^{m \times n}$ be a known matrix of coefficients and $c \in \mathbb{N}^n$ and $b \in \mathbb{N}^m$ be a vector of coefficients of the variables and a vector of the bounds of the constraints, respectively. Let also opt $\in \{\max, \min\}$ and $\preceq \in \{\leq, \geq\}$. The LP formulation of an optimization problem is the following:

$$\begin{aligned} \text{opt} \quad & c^T x \\ \text{s.t.} \quad & A \cdot x \preceq b \\ & x \geq 0 \end{aligned}$$

We refer to opt $c^T \cdot x$ as the objective function. Similarly, an Integer Linear Program has as additional requirements that the values of the variables be discrete i.e. $x \in \mathbb{N}^n$.

If in the above formulation opt = max, $\preceq = \leq$ and both $A$ and $b$ have non-negative entries, then the maximization problem is called a *Packing LP (PLP)*. The most notable problem of this class is the Set Packing problem; given a universe of elements and a family of subsets of the elements the goal is to find the maximum number of subsets which are pair-wise disjoint.

In a similar way if opt = min, $\preceq = \geq$ and both $A$ and $b$ have non-negative entries, then the minimization problem is called a *Covering LP (CLP)*. The most notable problem of this class is the Set Cover problem; given a universe of elements and a

family of subsets of the elements the goal is to find the minimum number of subsets the union of which is the universe.

Both PLP's and CLP's constitute very important classes of problems in computer science, optimization and operations research. These problems as well as special cases of them have numerous applications and have been extensively studied in the literature.

Although a problem in the continuous space can be tractable, it usually becomes hard to solve efficiently if we impose the discretization requirements on $x$. Most of the optimization problems of paramount importance, which can be be formulated as ILPs, are NP-hard and thus no polynomial time algorithm exists unless P=NP. In order to cope with the intractability of NP-hard problems, the research turns to approximation algorithms, the study of heuristics and special cases of the problems as well as parameterized complexity. In this thesis the study is focused on approximation algorithms for optimization problems. More formally let $OPT$ denote the optimal value of an optimization problem $\Pi$ and let $c(\mathcal{A}, I)$ denote the value of the objective function that polynomial-time algorithm $\mathcal{A}$ achieves for $\Pi$ on instance $I$. We denote by $|I|$ is the binary encoding of input $I$. A $\beta$-approximation algorithm is defined as follows:

**Definition 1.** *An algorithm $\mathcal{A}$ is a $\beta$-approximation for a maximization problem $\Pi$ if on any instance $I$ of $\Pi$, the value $c(\mathcal{A}, I) \geq \frac{1}{\beta} \cdot OPT$, where $\beta \geq 1$. If $\Pi$ is a minimization problem then $c(\mathcal{A}, I) \leq \beta \cdot OPT$. The factor $\beta$ is called the approximation ratio of $\mathcal{A}$.*

The approximation ratio can be a function that depends on the size of the input of the problem. Several problems are hard to approximate within some factor. For example, the Set Packing problem is known to be hard to approximate within $O(m^{1/2-\epsilon})$, where $m$ is the number of elements in the universe [66]. On the other hand, there are problems for which the approximation ratio can be arbitrarily close to 1 whose running time is inversely proportional to the approximation ratio. Roughly speaking we can trade the approximation of the solution we want to derive with the running time. We say that these problems admit a *PTAS*:

**Definition 2.** *A polynomial time algorithm $\mathcal{A}$ is a Polynomial Time Approximation Scheme (PTAS) for a maximization problem $\Pi$ if for any $\epsilon > 0$ and any instance $I$ of $\Pi$ it returns a solution with value $c(\mathcal{A}, I) \geq (1-\epsilon)OPT(I)$ and runs in time $O((\frac{1}{\epsilon}|I|)^{g(\frac{1}{\epsilon})})$, where $g$ is a function independent from $I$. Similarly for a minimization problem the value of the solution is $c(\mathcal{A}, I) \leq (1+\epsilon)OPT(I)$.*

**Definition 3.** *An algorithm $\mathcal{A}$ is an Efficient Polynomial Time Approximation Scheme (EPTAS) for a maximization problem $\Pi$ if for any $\epsilon > 0$ and any instance $I$ of $\Pi$ it returns a solution with value $c(\mathcal{A}, I) \geq (1-\epsilon)OPT(I)$ and runs in time $O(g(\frac{1}{\epsilon})poly(|I|))$, where $g$ is a function independent from $I$. Similarly for a minimization problem the value of the solution is $c(\mathcal{A}, I) \leq (1+\epsilon)OPT(I)$.*

The best family of algorithms we can hope for an NP-hard optimization problem, assuming that P≠NP is an FPTAS:

**Definition 4.** *An algorithm $\mathcal{A}$ is a Fully Polynomial Time Approximation Scheme (FPTAS) for a maximization problem $\Pi$ if for any $\epsilon > 0$ and any instance $I$ of $\Pi$ it returns a solution with value $c(\mathcal{A}, I) \geq (1-\epsilon)OPT(I)$ and runs in time $O(poly(\frac{1}{\epsilon}, |I|))$, where $g$ is a function independent from $I$. Similarly for a minimization problem the value of the solution is $c(\mathcal{A}, I) \leq (1+\epsilon)OPT(I)$.*

## 2.2 Mechanism design basics

A mechanism design problem is described by a set of *goods* or *alternatives* $\mathcal{X}$ and a set $\mathcal{N}$ of the participating *agents* or *players* (the two terms will be used interchangeably throughout the thesis). Every agent $i \in \mathcal{N}$ is associated with a value $\theta_i \in \Theta_i$ called his *type*, where $\Theta_i$ denotes the type space of agent $i$. The designer of the game also called *mechanism designer* or *social planner* has complete information about the type spaces $\Theta_1, \ldots, \Theta_n$. However, the type $\theta_i$ consists the private knowledge of agent $i$, for every agent $i \in \mathcal{N}$.

A mechanism maps a type profile $\theta = (\theta_1, \ldots, \theta_n)$ to an allowed outcome $o \in \mathcal{O} = 2^{\mathcal{X}}$. Every agent $i$ has a preference on an outcome which is expressed by a real valued function $v_i : \Theta_i \times \mathcal{O} \to \mathbb{R}^d$ called his *valuation* function. Let $\mathcal{V}_i \subseteq \Theta_i \times \mathcal{O}$ and $\mathcal{V} = \mathcal{V}_1 \times \ldots \times \mathcal{V}_n$. Let also $\mathcal{V}_{-i} = \mathcal{V}_1 \times \ldots \times \mathcal{V}_{i-1} \times \mathcal{V}_{i+1} \times \ldots \times \mathcal{V}_n$ and $\mathcal{V} = (\mathcal{V}_i, \mathcal{V}_{-i})$. In this thesis we follow this standard notation for players' vectors with subscript $-i$.

**Definition 5.** *A mechanism $M = (\mathcal{A}, p)$ for a given game $\Gamma = (\mathcal{O}, \mathcal{N})$ is composed of two elements: An allocation function $\mathcal{A} : \mathcal{V} \to \mathcal{O}$ specified by an algorithm and a payment scheme $p = (p_1, \ldots, p_n)$, where $p_i : \mathcal{V} \to \mathbb{R}$ for each agent $i$. More specifically a (direct revelation) mechanism consists of the following steps:*

- *Revelation: Every agent $i$ submits a bid $b_i \in \Theta_i$ to the mechanism designer. The bid of an agent represents his strategy in the game.*
- *Allocation: The mechanism designer decides on an allocation of the alternatives to the agents based on the declared strategy profile $b = (b_1, \ldots, b_n)$.*
- *Payment scheme: The mechanism designer charges every agent some monetary payment (if the payment is negative, then the mechanism designer is charged to pay this amount).*

The payment scheme in the above definition applies only to mechanisms with money. In mechanisms without money a mechanism is simply the allocation function. Every agent is associated with a *utility* function $u_i \in \mathcal{V} \times \mathcal{O} \to \mathbb{R}$, where $u_i(b_i|\theta_i, b_{-i}, o)$ denotes the satisfaction of agent $i$ over outcome $o = \mathcal{A}(b)$ when his bid is $b_i$, his type is

$\theta_i$ and the bids of the other agents are $b_{-i}$. In mechanisms without money the utility of agent $i$ is his valuation whereas in mechanisms with money the utility is the *quasi linear* function $u_i(b_i|\theta_i, b_{-i}, o) = v_i(\theta_i, o) - p_i(b)$ where $p_i$ denotes the amount of money that agent $i$ has to pay to the mechanism designer (or receive if $p_i$ is negative). Let $\mathcal{U}_i$ denote the set of utility functions of agent $i$ and let $\mathcal{U} = \mathcal{U}_1 \times \ldots \times \mathcal{U}_n$. We assume that the agents act *rationally* i.e. they aim at maximizing their utility regardless of the other agents' declarations.

**Definition 6.** *A bid $b_i$ is a dominant strategy of player $i$ if it maximizes his utility for any possible bids of the other agents. If all players have a dominant strategy and each plays these strategies in the game then we reach a dominant strategy equilibrium.*

The goal mechanism design is the design of mechanisms that guide the agents to behave in a desired way, assuming that each one has some private information. In game theoretic terms, we aim at designing a game that *implements* a *social choice function* $f : \mathcal{U} \to \mathcal{O}$ in *equilibrium*, given that the mechanism designer does not know the private information of the agents. In this thesis we study the implementation in dominant strategies:

**Definition 7.** *Given a game $\Gamma = (\mathcal{O}, \mathcal{N})$ and a set of utility functions $\mathcal{U}$, a mechanism with allocation function $\mathcal{A}$ is an implementation of a social choice function $f : \mathcal{U} \to \mathcal{O}$ if for any utility functions $\mathbf{u} = (u_1, \ldots, u_n) \in \mathcal{U}$ the game possesses a dominant strategy equilibrium $b^*$ such that $\mathcal{A}(b^*) = f(\mathbf{u})$.*

The importance of designing an implementation in dominant strategies arises from the fact that it leads to the design of *truthful* mechanisms:

**Definition 8** (Truthful mechanism)**.** *A mechanism $M = (\mathcal{A}, p)$ is called truthful or strategy proof if for any strategy profiles $b = (b_i, b_{-i})$ $b' = (b'_i, b_{-i})$ with respective outcomes $o \in \mathcal{O}$ and $o' \in \mathcal{O}$, it is a dominant strategy for all the agents to report their types i.e.*

$$u_i(b_i|\theta_i, b_{-i}, o) \geq u_i(b'_i|\theta_i, b_{-i}, o'), \;\; \forall b'_i \neq b_i = \theta_i, b_{-i} \in \mathcal{V}_{-i}$$

In other words, in a truthful mechanism the utility agent $i$ gets on outcome $o'$ when he reports $b'_i \neq \theta_i$ cannot be greater than the utility he gets on outcome $o$ when he declares his true type $b_i = \theta_i$. We note that if $o' = o$ for different declarations $b_i \neq b'_i$, the utility of every agent remains the same.
By the *revelation principle* (see below) a mechanism can be converted into a truthful one that implements the same social choice function:

**Theorem 1** (Revelation principle [113])**.** *If there exists a mechanism that implements a social choice function $f$ in dominant strategies, then there exists a truthful mechanism that implements $f$.*

For *randomized mechanisms* i.e. mechanisms whose allocation function is specified by a randomized algorithm, we have the following weaker notion of truthfulness:

**Definition 9.** *A randomized mechanism is truthful in expectation if for any* $b_{-i}$, $b_i$ *and* $b'_i$, $\mathbb{E}[u_i(b_i|\theta_i, b_{-i}, o)] \geq \mathbb{E}[u_i(b'_i|\theta_i, b_{-i}, o)]$, *where* $\mathbb{E}(\cdot)$ *is over the random bits.*

Randomized algorithms and thus randomized mechanisms are useful since in many cases they can achieve a better approximation ratio than the deterministic ones.

In a mechanism design optimization problem $\Pi$ the outcome is specified by an objective function $f(o, t)$ and a set of feasible outcomes $\mathcal{F} \subseteq \mathcal{O}$. In an optimal solution of $\Pi$ we require that $o \in \mathcal{F}$ such that $f(\cdot)$ is optimized, while in an approximate solution we require that $o \in \mathcal{F}$ and $f(o, t) \preceq \beta \cdot f(o', t)$, where $o' \in \mathcal{F}$, $\preceq \in \{\leq, \geq\}$ and $\beta$ is the approximation ratio. We say that a mechanism solves an optimization problem when it assures that a required outcome is given as a result. In this thesis we study the following objective functions:

- minimization of the total costs of the agents or equivalently the maximization of the *social welfare* i.e. the aggregation of the valuations of all the agents (Chapter 3, Section 4.4 and Chapter 5)
- the maximization of the *network lifetime* i.e. the total time that a set of constraints is fulfilled (Chapter 3)
- the maximization of the minimum utility of an agent (Chapter 6).

### Conditions for truthfulness

From the above definitions we have that $\mathcal{V}_i \subseteq \mathbb{R}^d$. When agent $i$ is *single-parameter* i.e. $d = 1$, *monotonicity* is a property of social choice functions that is sufficient and necessary condition to guarantee the truthfulness of a mechanism (see Chapter 12 in [113] and [14]). In simple words monotonicity means that if an agent changes his valuation and the social choice function changes too, then this is because the agent increased the value on the new choice in relation to the value of his old choice.

As an example consider the problem of combinatorial auction where the set of alternatives is a set of items to be allocated to the players and the objective is the maximization of the social welfare. Let $S = \sum_{i \in \mathcal{N}} v_i$ and suppose that player $i$ is allocated no item. If he changes his valuation to $v'_i \geq v_i$ and the allocation of the items also changes (and thus $i$ is allocated an item or a bundle of items), then the change in the social function $S' = \sum_{j \neq i} v_j + v'_i$ happened because of the change in $i$'s valuation.

Let now $\mathcal{X}$ consist of identical and divisible items. Suppose that each of these items can be divided into $k$ pieces. If $k \to \infty$ then $\mathcal{X}$ becomes a continuous set. For an outcome $o \in \mathcal{O}$ we define as *load* (sometimes denoted as *work*) $w_i : \mathcal{O} \to \mathbb{R}$ the amount of $\mathcal{X}$ allocated to agent $i$. Let us also define $c_i(\theta_i, o) = \theta_i \cdot w_i(o)$ to be the cost that

agent $i$ incurs when he is assigned load $w_i(o)$ over the outcome $o = \mathcal{A}(b)$. The valuation function of agent $i$ is then $v_i(\theta_i, o) = -c_i(\theta_i, o)$ and the type $\theta_i$ denotes the cost per unit of load assigned to agent $i$. We will abuse notation writing $w_i(b)$ instead of $w_i(o)$ when the allocation function $\mathcal{A}$ is known in the context. Let $b_{-i}$ be fixed. Then the produced outcome $o$ is a non-increasing function if the load curve $w_i(b_i, b_{-i})$ is non-increasing on $b_i$ for every agent $i$. More formally:

**Definition 10** (Monotone algorithm [14])**.** *Let the bids $b_{-i}$ be fixed and let $w_i(b_{-i}, b_i)$ be a single variable function of $b_i$ called the load or work of agent $i$. The allocation function $\mathcal{A}$ is non-increasing if each of the associated load curves in non-increasing i.e. $w_i(b_{-i}, b_i)$ is a non-increasing function of $b_i$. An algorithm that preserves the non-increasing load property for all agents is called monotone.*

We will say that a mechanism satisfies the *voluntary participation* condition if every agent who declares his true type to the mechanism does not incur a net loss. Archer and Tardos provided a characterization of truthful mechanisms for single-parameter agents by the following:

**Theorem 2** (Payment scheme [14])**.** *For single parameter agents, a mechanism is truthful and admits voluntary participation if and only if the load of each agent is non-increasing $\int_0^\infty w_i(b_{-i}, u)du < \infty$ for all $i$, $b_{-i}$ and the payments are*

$$p_i(b_{-i}, b_i) = b_i \cdot w_i(b_{-i}, b_i) + \int_{b_i}^{\infty} w_i(b_{-i}, u)du. \tag{2.1}$$

With the above characterization, in order to design a truthful mechanism it is sufficient and necessary to design a monotone algorithm to specify the allocation function together with a payment scheme as stated in equation (2.1) (for maximization problems), assuming that no externalities exist, i.e. the agents care only about their assigned loads.

In discrete settings, monotonicity for a social welfare maximization problem means that if agent $i$ with original valuation $v_i$ is allocated a bundle of alternatives (a winning declaration), then he will either be allocated the same bundle or a different one with higher valuation for $i$, if he declares to the mechanism designer a valuation $v_i' \geq v_i$. Similarly, for a cost minimization problem agent $i$ with original cost $c_i$ will be allocated the same or another bundle of alternatives with higher valuation, when he declares a cost $c_i' \leq c_i$. More formally for discrete settings we have:

**Definition 11.** *An algorithm $A$ is monotone if for every agent $i$ and any declarations of the other agents $b_{-i}$, if $v_i$ is a winning declaration then every higher declaration $v_i' \geq v_i$ also wins.*

By this definition, for a monotone allocation algorithm $A$ and for any $v_{-i}$ there exists a *critical value* $\zeta_i$ such that for all $v_i > \zeta_i$, $v_i$ is a winning declaration and for all $v_i < \zeta_i$, $v_i$ is a losing declaration.

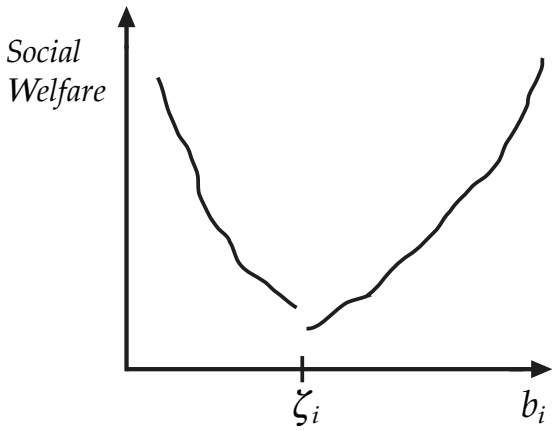



Figure 2.1: A bitonic curve of a bitonic allocation algorithm

Consider now a setting where the objective is the maximization of the social welfare. Let the allocation function be specified by an algorithm with several subroutines. Mu'Alem and Nisan presented an array of algorithmic techniques in [107] to obtain truthful mechanisms for special cases of algorithms as the one above. One of the studied cases is the "max" construct i.e. taking the best outcome as a solution from a combination of different monotone algorithms. In the case where an allocation function is specified by an algorithm $A$ that returns the outcome of one of its monotone subroutines, then $A$ is not necessarily monotone. Thus, they provided the stronger notion of *bitonicity* which is sufficient for special cases of monotone allocation algorithms which are called *bitonic*. For a monotone algorithm $A$, bitonicity provides a connection between $b_i$ and the social welfare of the allocation $A(b_{-i}, b_i)$. Roughly speaking, in the allocation by a bitonic algorithm, the social welfare does not increase with respect to $b_i$, when agent $i$ loses i.e. $b_i < \zeta_i$, and does not increase with respect to $b_i$ when $i$ wins i.e. $b_i > \zeta_i$ (see Figure 2.1 reproduced from [107]). More formally:

**Definition 12** (Bitonic algorithm [107])**.** *A monotone algorithm $A$ is bitonic if for every agent $i$ and every $b_{-i}$, the social welfare denoted as $SW_A(b_{-i}, b_i)$ of the allocation is a non-increasing function of $b_i$ when $b_i < \zeta_i$ and a non-decreasing function of $b_i$ when $b_i > \zeta_i$. The value of the social welfare is then $SW_A(b_{-i}, b_i) \leq \max\{\lim_{b_i \to \zeta_i^-} SW_A(b_{-i}, b_i), \lim_{b_i \to \zeta_i^+} SW_A(b_{-i}, b_i)\}$.*

Generally speaking, an algorithm is bitonic if it is monotone and the computed cost is proportional to the parameters i.e. improves when they improve. We now have the following composition theorem which states that the combination of bitonic algorithms for single-parameter agents is a monotone algorithm:

**Theorem 3** ([21])**.** *Consider a game with single-parameter agents and let $M$ be a procedure with $m$ subproblems $P_1, \ldots, P_m$. Each subproblem is solved by a different*

*procedure $M_i$ and returns the optimal solution with respect to a cost function $c_i(\cdot)$. If every procedure $M_i$ is bitonic with $c_i(\cdot)$, then $M$ is monotone.*

We note that for multi-parameter agents monotonicity is not a sufficient condition for truthfulness. In such cases, we need the stronger condition of cycle monotonicity [94]. If we restrict to social welfare maximization for multi-parameter agents, then the celebrated VCG mechanism due to the works by Vickrey [140] Clarke [33] and Groves [65] is known to be the optimal mechanism. In order to ensure truthfulness, VCG charges payments to the agents, based on Clarke's pivot rule [33]. Although this mechanism solves the economic problem of maximizing the social welfare, it does not take into account that the allocation algorithm can be NP-hard. Thus, despite the fact that VCG can ensure truthfulness it cannot be computed in polynomial time for a plethora of problems. A natural way to avoid this obstacle and have a computationally efficient mechanism would be to compute an approximately optimal allocation based on the solution produced by an approximation algorithm combined with the VCG payments. The results of Lehmann et al. [96] and Nisan and Ronen [111] indicate that such mechanisms are not truthful. Further study on this direction was made by Nisan and Ronen [112], Holzman et al. [68] and Lavi et al. [93].

However, there are approximation algorithms that can be used to obtain truthful mechanisms for social welfare maximization, when we use the VCG payments:

**Definition 13.** *Let $\mathcal{R}$ be a subset of the allocations' space. An algorithm is called Maximum In Range (MIR) if on any possible input of valuations of the agents, it returns the allocation which maximizes the social welfare in $\mathcal{R}$.*

The randomized variant of MIR mechanisms is to consider a set $\mathcal{D}$ of distributions over the allocation set. After choosing set $\mathcal{D}$ the resulting allocation rule, called *Maximal in Distributional Range (MIDR)* is to choose the distribution that maximizes the expected social welfare taking into account the reported valuations of the agents. Based on MIDR mechanisms Lavi and Swamy [95] proposed a general technique to obtain truthful in expectation mechanisms given an approximation algorithm for the problem. The technique is based on the LP relaxation of the problem using the VCG mechanism.

# Chapter 3

# Network pollution games

## 3.1 The problem

The advance of technology and commercial freedom have fused and accelerated the development process to an unprecedented scale. Environmental degradation however has accompanied this progress, resulting in global water and air pollution. In many developing countries, this has caused wide public concerns. As an example, in 2012, China discharged 68.5 billion tons of industrial wastewater and the $CO_2$ emissions reached 21.2 million tons (National Bureau of Statistics of China, 2013). China has become one of the most polluted countries in the world with industrial emissions as the main source of its pollution. The recent annual State of the Air report of the American Lung Association finds that 47% of Americans live in counties with frequently unhealthy levels of either ozone or particulate pollution, see [5]. The latest assessment of air quality, by the European Environment Agency, finds that around 90% of city inhabitants in the European Union are exposed to very damaging air pollutants at harmful levels, see [1]. Environmental research suggests that water pollution is one of the very significant factors affecting water security worldwide [142]. It is the role of regulatory authorities to make efficient environmental policies in balancing economic growth and environment protection. Pollution control regulations are inspired by the managerial approaches in environment policies, where models based on game theory are proposed and analysed.

From a different point of view, Dasgupta, Hammond and Maskin [37] focus on minimizing the sum of pollution damages, abatement costs and individual rationality for consumers. Spulber [129] develops a market model of environmental regulation with interdependent production, pollution abatement costs and heterogeneous firms who have private information about costs and pursue Bayes-Nash strategies in communication with the regulator. Their paper illustrates that the full information optimum cannot be attained unless gains from trade in the product market net of external damages exceed the information rents earned by firms and aggregate output and externality levels are lower at the regulated equilibrium than at the full information social optimum. A more

extended literature overview can be found at the end of the chapter.

Pollution has a diffusion nature: emitted from one source, it will have an effect on its neighbours at some decreased level. We consider two applications using a network model. In the first application, the vertices represent pollution sources and edges are routes of pollution transition from one source to another similar to Belitskaya [18]. Our model measures the pollution diminishing transition by arbitrary weights on the edges, which is also present in the model presented by Montgomery [106]. The polluters' privately known clean-up cost and damage of the emitted pollution in our model are inspired by Kwerel [92]. In the second application, the vertices represent mayors of cities and the edges represent the roads between cities. The percentage of cars moving from one city to another is represented by the weight of the corresponding edge.

Our model covers both aforementioned applications with details given in Section 3.2. The government, as the regulator, can decide to either shut down or keep open a pollution source taking into account the diffusion nature of pollution. It sets bounds on the global and local levels of pollution, while trying to optimize the social welfare. The emissions that exceed the licences, if any, must be cleaned-up (hence, agent's clean-up cost). Furthermore our model allows the regulator to auction pollution licences for cars to mayors. In this case, the pollution level of an agent (mayor), i.e., the number of allocated licences, is set by the regulator together with the prices that the agent pays to get them.

Furthermore we study water pollution in rivers modelled by tree networks. In water pollution the government decides which pollution sources should be shut down so that the effluent level in water is as low as possible. Water pollution cost sharing was introduced in [109] where the network is a path (single river). This model was extended to tree networks (a system of rivers) in [44]. We model a system of rivers as a tree, but study a different pollution control model.

As a variant of the first application described above, we also consider the case in which the government is allowed to sell licences to the pollution sources instead of deciding to shut it down or keep it open. This is a widely used approach to control pollution levels by auctioning a fixed number of licences or pollution allowances. For instance, the European Emissions Trading System sells EU Emission Allowances (EUAs), each one representing the right to emit one ton of $CO_2$. In such an auction, firm's bid is a number of EUAs and per EUA price. The auction ranks all the bids in descending order of per EUA price and determines the per EUA clearing price. The clearing price is the first bid price such that the total volume of EUAs in the bids (demand) in this descending order meets the total volume of EUAs offered by the regulator (supply). All the bids above this clearing price are awarded and they all pay the clearing price, see, e.g., [2]. This very simple auction does not take into account the diffusion relations between polluters, etc.

| | | General objective function | Linear objective function | | |
|---|---|---|---|---|---|
| | | Bounded Degree $\Delta$ | Trees | | Planar |
| Lower bound | | $\Omega\left(\frac{\Delta}{\log \Delta^2}\right)$ | NP-hard | | strongly NP-hard ($\delta$ violation) |
| PG(poly) | | $O(\Delta)^a$ | FPTAS TiE | $O(1)$ DT | PTAS ($\delta$ violation) |
| PG(general) | | $O(\Delta)$ TiE $^b$ | FPTAS TiE $^c$ | | $O(1)$ TiE [10] |

$^a$ Monotone increasing obj. function. $^b$ Piece-wise linear obj. function with one shift and an additional mild assumption. $^c$ Running time is polynomial in $q$.

Table 3.1: Our results. TiE/DT: truthful in expectation/deterministic truthful mechanism. PG(poly) is PG with poly-size integer variables, PG(general) without this assumption.

Finding an optimal social welfare solution to our problem, which we call Pollution Game (PG), is NP-hard, since even in the special case of tree structures the well known Knapsack problem can be reduced to PG. That is why we study polynomial time approximation algorithms which can lead to incentive compatible (truthful) mechanisms. We study linear cost and damage functions and derive approximation algorithms and truthful mechanisms focusing on planar network topologies. In contrast, Belitskaya [18] assumes quadratic cost functions and linear damage functions deriving optimal social welfare and Nash equilibria solutions by explicit analytic formulas. We focus our study on planar network topologies which model realistic scenarios.

Most of the cited economics papers derive equilibria by closed analytic formulas. Some of these papers provide computational mechanisms without investigating polynomial running time. Our approach is algorithmic and focuses on efficiently computing these solutions. We also analyze the computational complexity/hardness, of computing the social optimum in our model. To the best of our knowledge, this work is the first attempt to algorithmically analyze pollution control from the perspective of regulators by a network game model with information asymmetry between regulators and polluters. Our results are summarized in Table 3.1.

## 3.2 Model and applications

We first give a brief description of our model for PG. In this model we are given a set of agents each of whom gets a benefit by doing some work (production of goods, traffic on roads) that results in emission of pollution. The benefit of each agent is expressed by a function. The government sets some bounds on the total emitted pollution in local area around every source and in the global area. Its objective is the maximization of the social welfare (the sum of the benefits of the agents). More precisely PG is defined as follows:

**Definition 14** (Pollution Game)**.** *We are given a network represented by a graph $G = (V, E, w)$ where $V$ is the set of polluters, $E$ is the set of connecting edges (roads)*

*between them and $w_{uv}$ is a percentage of the effect of pollution from $u$ to $v$. Let $p$ and $p_u$ be bounds on the total allowable emitted pollution in whole network and in the local area around polluter $u_i$, $\forall u_i \in V$ respectively. Let also $r_u(\cdot)$ denote the valuation function of polluter $u$. In the Pollution Game we are asked to find an allocation of pollution permits $x = (x_u, x_{-u})$ for the polluters of the network such that the social welfare $\sum_{u\in V} r_u(x_u)$ is maximized under the constraints that the emitted pollution does not exceed the bounds both locally $x_u + \sum_{v\in\delta_G(u)} w_{vu}x_v \leq p_v,\ \ \forall u \in V$ and globally $\sum_{u\in V} x_u \leq p$.*

The game will become more clear in the following two applications. In the first the government as the regulator of the game, allocates pollution licences to mayors of cities which are then sold to car drivers. In the second application, the government has to decide which pollution sources among many need to be shut down and which can remain open.

We note that in this model no geometric assumptions are made.

### 3.2.1 Application 1: Allocation of pollution licences

Consider an area of $n$ cities, each administered by its mayor (agent). In every city, statistical observations are used to measure the traffic to the neighboring cities. More precisely let $G = (V, E, w)$ be a weighted digraph representing a network, where $V$ is the set of $n$ agents (mayors of the cities), $E$ is the set of roads connecting cities such that $(u, v) \in E$ if and only if $u$ and $v$ are neighbouring cities, and $w : E \to \mathbb{R}$ represents the percentage of cars entering a city from a neighboring one i.e. $w_{uv}$ denotes the percentage of cars driving from $u$ to $v$ in some time interval measured by observations.

The duty of the regulator is to allocate a number of pollution licences (i.e. a licence additional to the car licence) to the agents (mayors) such that the total welfare is maximized while fulfilling a number of constraints. We denote by $x_u$ the number of licences allocated to agent $u$. The agent with $x_u$ licences gains a benefit of $b_u(x_u)$ which is a monetary income coming from selling these $x_u$ licences to car drivers (our model does not model this explicitly but just assumes for simplicity that all $x_u$ licences are sold). We assume that $b_u(x_u)$ is a concave increasing function (economic diminishing marginal utility phenomenon)[1] with $b_u(0) = 0$.

The pollution damage caused in the area of agent $u$ is given by the non-decreasing damage function $d_u\big(x_u + \sum_{v\in\delta_G^-(u)} w_{vu}x_v\big)$ (the damaging effect of more emitted pollution is accelerating), where $\delta_G^-(u) = \{v \in V : (v, u) \in E\}$. The total valuation of agent $u$ is his benefit minus his damage cost:

[1] [92] uses cost function rather than benefit function, which can be viewed as $M_u - b_v(x_u)$, with $M_v$ a large constant for any $u \in V$. The author assumes that cost function is convex decreasing and it is equivalent to $b_u(x_u)$ being a concave increasing function. We use benefit function rather than cost function for ease of analysis.

$$r_u = b_u(x_u) - d_u\left(x_u + \sum_{v \in \delta_G^-(u)} w_{vu} x_v\right) \tag{3.1}$$

The damage function shows that player $u$ is affected by the damage of his own discharged pollution if $x_u = 1$ and by the total discounted pollution of his neighbours. This models the fact that pollution spreads along the edges of $G$. The total utility that the regulator aims to maximize is: $\sum_{u \in V} b_u(x_u) - d_u\big(x_u + \sum_{v \in \delta_G^-(u)} w_{vu} x_v\big)$.

The total number of licences for the whole network is bounded by a number $p$ given in the input of the problem. This constraint is called *the global constraint*:

$$\sum_{v \in V} x_v \leq p \tag{3.2}$$

Naturally a percentage of cars with licences from city $u$ remains in $u$ and the rest is split and drives into the neighboring cities. We denote by $w_u$ the percentage of cars remaining in $u$ and $w'_{vu}$ the percentage of cars entering $u$ from neighbouring city $v$. It's realistic to assume that $w_u \neq 0$ since not all the cars with licence from city u move to neighbouring cities at the same time. The maximum number of cars (maximum number of licences) allowed at any moment in city $u$ is bounded by $p'_u$ also given in the input. This is represented by the local constraint: $w_u x_u + \sum_{v \in \delta_G^-(u)} w'_{vu} x_v \leq p'_u$. The last inequality can equivalently be written to the following constraint, called *the local constraint* of $u$:

$$x_u + \sum_{v \in \delta_G^-(u)} w_{vu} x_v \leq p_u \tag{3.3}$$

where $w_{vu} = w'_{vu}/w_u$ and $p_u = p'_u/w_u$. Furthermore, the number of licences that can be issued in city $u, \forall u \in V$, is bounded by $q_u$ which is decided by the government and is given on input.

Combining the objective with all the above constraints, the problem of social welfare maximization can be formulated in the general form by the following integer program:

$$\max \quad R(x) = \sum_{v \in V} \left(b_v(x_v) - d_v\left(x_v + \sum_{u \in \delta_G^-(v)} w_{uv} x_u\right)\right) \tag{3.4}$$

$$\text{s.t.} \quad \sum_{v \in V} x_v \leq p \tag{3.5}$$

$$x_v + \sum_{u \in \delta_G^-(v)} w_{uv} x_u \leq p_v, \ \ \forall v \in V \tag{3.6}$$

$$x_v \in \{0, 1, \ldots, q_v\}, \ \ \forall v \in V \tag{3.7}$$

We call the above convex integer program Pollution Game (PG) with integer variables (if $x_v \in \mathbb{Z}$) or with binary variables (if $x_v \in \{0, 1\}$). For an instance $I$ of PG, $|I|$ denotes

the number of bits to encode $I$, and if $q \in poly(|I|)$, where $q = \max_{v \in V}\{q_v\}+1$, we call this problem PG with polynomial size integer variables.

We assume that the networks of this application are represented by planar graphs. These graphs are close to real applications and it is natural to study planar networks [139]. Imagine a collection of cities (each being a contiguous geographic area) and roads connecting them. This defines a planar map where we only consider edges (roads) between neighbouring cities, which implies a planar graph. We disregard other roads and we consider only frequent driving patterns in a time interval measured by observations. They correspond to frequent commuters, e.g., between house and work, which typically are neighbouring cities.

### 3.2.2 Application 2: Regulation of pollution sources

We are given an area of pollution sources (e.g. factories) each one owned by an agent. The goal of the government as a regulator is to optimize the social welfare while restricting the levels of emitted pollution. More formally, given a weighted digraph $G = (V, E)$, where $V$ is the set of $n$ pollution sources (players, agents) and edge $(u, v) \in E$ represents the fact that $u$ and $v$ are geographic neighbours i.e. $(u, v) \in E$ if and only if the pollution emitted by $u$ affects $v$. For each $(u, v) \in E$ weight $w_{(u,v)} = w_{uv}$ denotes a discount factor of the pollution discharged by player $u$ affecting its neighbour $v$. Without loss of generality we may suppose that $w_{uv} \in (0, 1]$, $\forall (u, v) \in E$.

The government has to decide which pollution sources must remain open and which must be shut down. For each source $u$ the variable $x_u \in \{0, 1\}$ denotes this where 0 means it must be shut down and 1 it will remain open. Furthermore the government sets the total pollution quota discharged to the environment (by the number of pollution sources that remain open) to be $p$. The global constraint, as it will be called in the following, is:

$$\sum_{v \in V} x_v \leq p \tag{3.8}$$

Each agent $v$ has a non-decreasing benefit function $b_v : \mathbb{R}_{\geq 0} \longrightarrow \mathbb{R}_{\geq 0}$, where $b_v(x_v)$ is a concave increasing function (economic diminishing marginal utility phenomenon)[2] with $b_v(0) = 0$ which represents the benefit incurred by $v$. Each $v$ also has a non-decreasing damage function $d_v : \mathbb{R}_{\geq 0} \longrightarrow \mathbb{R}_{\geq 0}$ (the damaging effect of more emitted pollution is accelerating). The total valuation $r_v$ of player $v$ is his benefit minus his damage cost:

$$r_v = b_v(x_v) - d_v\left(x_v + \sum_{u \in \delta_G^-(v)} w_{uv} x_u\right) \tag{3.9}$$

[2] [92] uses cost function rather than benefit function, which can be viewed as $M_v - b_v(x_v)$, with $M_v$ a large constant for any $v \in V$. The author assumes that cost function is convex decreasing and it is equivalent to $b_v(x_v)$ being a concave increasing function. We use benefit function rather than cost function for ease of analysis.

where, $\delta_G^-(v) = \{u \in V : (u,v) \in E\}$, $\delta_G^+(v) = \{u \in V : (v,u) \in E\}$. Thus, the damage function shows that player $v$ is affected by the damage of his own discharged pollution if $x_v = 1$ and by the total discounted pollution of his neighbours. This models the fact that pollution spreads along the edges of $G$. We assume that the government decides on the allowable local level of pollution $p_v$, for every $v \in V$. This imposes the following local constraints, as will be called in the following for every player $v \in V$:

$$x_v + \sum_{u \in \delta_G^-(v)} w_{uv} x_u \leq p_v \tag{3.10}$$

$$x_v \leq q_v \tag{3.11}$$

Thus we can model this application by the convex program (3.4)-(3.6) assuming now that $x_v \in \{0,1\}$ and $q_v = 1$, $\forall v \in V$.

### 3.2.3 Basic definitions

We will introduce some basic definitions and propositions which will be necessary later in proving the approximations of our algorithms. Let $G = (V,E)$ be a graph and $I = (G, \mathbf{b}, \mathbf{d}, \mathbf{p}, \mathbf{q}, \mathbf{w}, p)$ be an instance of PG, where $\mathbf{b} = (b_v)_{v \in V}$ is the benefit function, $\mathbf{d} = (d_v)_{v \in V}$ is the damage function $\mathbf{p} = (p_v)_{v \in V}$ is the bound of the local constraint, $\mathbf{w} = (w_{uv})_{uv \in E}$, $p$ is the bound of the global constraint and $\mathbf{q} = (q_v)_{v \in V}$ denotes the total quota of every player ($b_v$ is assumed private information of $v$ and other parameters are public). Let $\mathcal{I}$ be the set of all instances, and $\mathcal{X}$ the set of feasible allocations. For a given digraph $G = (V,E)$ we consider the undirected graph $G^{un} = (V, E^{un})$ where $E^{un} = \{(u,v) \in E \text{ or } (v,u) \in E\}$. An undirected graph $G$ is $k$-outerplanar if for $k = 1$, $G$ is outerplanar and for $k > 1$, $G$ has a planar embedding such that if all vertices on the exterior face are deleted, the connected components of the remaining graph are all $(k-1)$-outerplanar. A planar graph is $k$ outerplanar where $k$ can be equal to $+\infty$. We will use the notation $i \in [N]$ to denote $i \in \{1, \ldots, N\}$.

In the following we will denote by $OPT_G^{fr}(PG)$ the value of the optimal fractional solution of PG on $G$. Similarly $OPT_G^{in}(PG)$ denotes the optimal integral solution. The integrality gap of PG on $G$ is defined as $\frac{OPT_G^{fr}(PG)}{OPT_G^{in}(PG)}$. The approximation ratio of an algorithm $\mathcal{A}$ with respect to $OPT_G^{in}(PG)$ ($OPT_G^{fr}(PG)$ respectively) is $\rho^{in}(\mathcal{A}) = \frac{OPT_G^{in}(PG)}{R(\mathcal{A})}$ ($\rho^{fr}(\mathcal{A}) = \frac{OPT_G^{fr}(PG)}{R(\mathcal{A})}$ respectively), where $R(\mathcal{A})$ is the objective function of the solution produced by $\mathcal{A}$. Unless stated otherwise, the approximation $\rho$ will be with respect to $OPT_G^{fr}(PG)$.

When $b_v$ and $d_v$ are both linear functions we assume they have slopes $s_v^0$ and $s_v^1$ respectively, i.e. $b_v(x) = s_v^0 x$ and $d_v(y) = s_v^1 y$, for any $v \in V$. Let $V = \{v_1, \ldots, v_n\}$. The social welfare function is:

$$
\begin{aligned}
R(x) &= \sum_{v\in V}\left[b_v(x_v) - d_v\left(x_v + \sum_{u\in\delta_G^-(v)} w_{uv}x_u\right)\right] \\
&= \sum_{v\in V}\left[s_v^0 x_v - s_v^1\left(x_v + \sum_{u\in\delta_G^-(v)} w_{uv}x_u\right)\right] = \sum_{v\in V}\left[(s_v^0 - s_v^1)x_v - s_v^1 \sum_{u\in\delta_G^-(v)} w_{uv}x_u\right] \\
&= \left[(s_{v_1}^0 - s_{v_1}^1)x_{v_1} - s_{v_1}^1 \sum_{u\in\delta_G^-(v_1)} w_{uv_1}x_u\right] + \ldots + \left[(s_{v_n}^0 - s_{v_n}^1)x_{v_1} - s_{v_n}^1 \sum_{u\in\delta_G^-(v_n)} w_{uv_n}x_u\right] \\
&= \left[s_{v_1}^0 - s_{v_1}^1 - \sum_{u\in\delta_G^-(v_1)} s_u^1 w_{v_1u}\right]x_{v_1} + \ldots + \left[s_{v_n}^0 - s_{v_n}^1 - \sum_{u\in\delta_G^-(v_n)} s_u^1 w_{v_nu}\right]x_{v_n} \\
&\sum_{v\in V}\left[s_v^0 - s_v^1 - \sum_{u\in\delta_G^-(v)} s_u^1 w_{vu}\right]x_v = \sum_{v\in V}\omega_v x_v
\end{aligned}
\tag{3.12}
$$

where $\omega_v = s_v^0 - s_v^1 - \sum_{u\in\delta_G^-(v)} s_u^1 w_{vu}$. For every variable $x_{v_i}$ in the fourth line of the above equations we have considered all the occurrences of $x_{v_i}$ in all the terms in line 3.

In the following sections we will consider $k$-column sparse ILP packing problems i.e. those in which each variable $j$ participates in at most $k$ constraints. The following propositions will be useful in obtaining approximation algorithms and mechanisms for packing problems:

**Proposition 1** ([17], [117])**.** *There is a polynomial-time deterministic algorithm for $k$-column sparse linear packing programming problem with binary variables, achieving the approximation ratio $\rho^{fr} = \gamma_k$, where $\gamma_k = (e + o(1))k = O(k)$ for a fixed $k$.*

**Proposition 2** ([17])**.** *There is a polynomial-time deterministic algorithm for $k$-column sparse convex packing programming problem with binary variables, achieving the approximation ratio $\rho^{fr} = \frac{e\gamma_k}{e-1}$, where $\gamma_k = (e + o(1))k = O(k)$ for a fixed $k$.*

**Proposition 3** ([95])**.** *For any linear packing programming problem, if there is a polynomial deterministic algorithm with the approximation ratio $\rho^{fr}$ for this problem, then there is a polynomial, randomized, individually rational, $\rho^{fr}$-approximation mechanism for the same problem that is truthful in expectation.*

## 3.3 Hardness

Observe that PG is weakly NP-hard even on stars and even without the global constraint with linear valuation functions. Consider the star where the central node $u$ is connected to nodes $v_1, v_2, \ldots, v_n$ with valuations $\omega_{v_1}, \omega_{v_2}, \ldots, \omega_{v_n}$ and the edges that connect them have weights $w_{uv_1}, w_{uv_2}, \ldots, w_{uv_n}$ respectively. Then PG on this instance (omitting the local constraints of all the nodes except $u$) can be formulated by the following ILP:

$$
\begin{aligned}
\max \quad & \omega_u x_u + \sum_{i=1}^{n} \omega_{v_i} x_{v_i} \\
\text{s.t.} \quad & x_u + \sum_{v_i \in \delta_G^-(u)} w_{v_i u} x_{v_i} \leq p_u \\
& x_{v_i} \in \{0,1\}, \ \ \forall v_i \in \delta_G^-(u)
\end{aligned}
$$

The above ILP is the Knapsack problem [137]. In the Knapsack problem we are given $n$ items each having a weight $w_i$ and a value $v_i, \forall i = 1, \ldots, n$. We are also given capacity $W$ and we are asked to choose those items whose total value is maximized under the constraint that the capacity is not exceeded. In the instance described above the weights on the edges are the weights of the items and the values of the nodes are the values of the items. Knapsack in known to be weakly NP-hard, thus PG on stars is also weakly NP-hard.

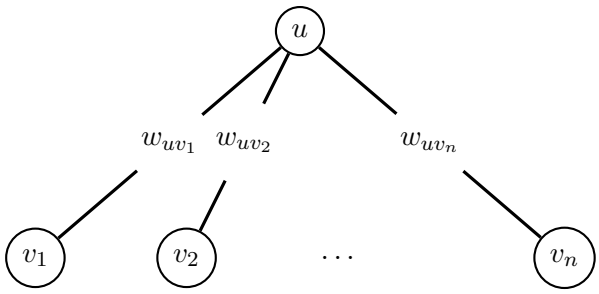



Figure 3.1: An instance of PG on a star

We note that inequality (3.5) can also be written as equality

$$\sum_{u \in V} x_u = p \tag{3.13}$$

since the upper limit of the total amount of the pollution is controlled by the government. If $\sum_{v \in V} x_v < p$ in the final allocation, then the government can simply set $p$ equal to $\sum_{v \in V} x_v$ without any changes. However, for computational issues, these two representations lead to different computational complexity of the problem. If inequality (3.5) is replaced by (3.13), then even finding a feasible solution to PG is NP-complete as shown in the following theorem:

**Theorem 4.** *Finding a feasible solution to PG when $p_v = 1 \, \forall v \in V$ and $w_{uv} > 0$ for any $(u, v) \in E$ and after replacing constraint (3.6) with (3.13), is NP-complete.*

*Proof.* It is straightforward that the problem is in NP. Consider now a formula of monotone 1-in-3 SAT where an instance of this problem consists of $n$ Boolean variables and $m$ clauses. A YES instance is one in which there exists an assignment to its Boolean

variables such that exactly one literal from each clause is true. The problem is known to be NP-Complete even when there are no negations [125]. The following proof is inspired by the reduction of 3-SAT to Independent Set (p. 248 [38]).

Let us represent a clause, say $(x \vee y \vee z)$, by a triangle with vertices labeled $x$, $y$, $z$. Repeat this construction for all clauses. Next consider one of the literals, say $x$, which appears in $k$ clauses $C_{i_1}, \ldots, C_{i_k}$. Let $Tr_{i_1}, \ldots, Tr_{i_k}$ be the triangles of the clauses $C_{i_1}, \ldots, C_{i_k}$ respectively. Then connect $x$ of $Tr_{i_1}$ with all the vertices of $Tr_{i_2}, \ldots, Tr_{i_k}$ except those labeled with $x$. Repeat this construction for $x$ in all these triangles and for all literals. For example consider the formula $\Phi = (x_1 \vee x_2 \vee x_3)(x_1 \vee x_4 \vee x_5)$ (see Figure 3.2). First construct the triangles labeled $x_1, x_2, x_3$ and $x_1, x_4, x_5$ for the two clauses respectively. Then connect vertex $x_1$ of the first clause with the vertices $x_4$ and $x_5$ of the second clause. Similarly, connect vertex $x_1$ of the second clause with the vertices $x_2$ and $x_3$ of the first clause.

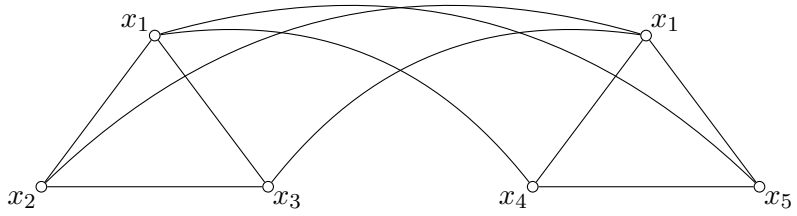



Figure 3.2: A gadget of reduction

Consider now an instance of 1-in-3 SAT which is true and let $G = (V, E)$ be the corresponding graph constructed as explained above. Furthermore for every vertex $u \in V$, let $p_u = 1$, $p = m$ and $0 < w_{uv} \leq 1$, $\forall v \in \delta_G^-(u)$. Suppose that we have a truth assignment which satisfies all the clauses in 1-in-3 sense. This means we choose $p$ vertices in $G$ without violating any of the constraints. Indeed if any two vertices have the same label, they are not connected. If they have different labels, say $x$ from clause $C_1$ and $y$ from clause $C_2$ and they are connected, then their corresponding clauses have a common literal, either $x$ or $y$. Thus if one of them has value true, the other will have value false for the formula to be satisfiable. Finally, if two vertices belong to the same clause, only one of them will have the value true.

Suppose now that we can decide in polynomial time whether there is a solution in an instance of a graph constructed by a formula as described above with $p = m$ vertices when $p_v = 1$, $\forall v \in V$. Then setting the literal in the set $\{v : x_v = 1\}$ is a solution of 1-in-3 SAT. The argument is as follows: in each triangle, there is exactly one vertex such that its value is one since at most one vertex in each triangle can be selected and there are $m$ triangles and $p = m$. By the construction of $G$, these vertices consist of a solution of 1-in-3 SAT. Thus exactly one literal in every clause has the value true in the formula. □

In the following, unless stated otherwise, we will assume inequality (3.5) as a constraint of PG.

**Theorem 5.** *It is strongly NP-hard to find an optimal solution to PG when $p_v, \forall v \in V$ is any constant number $\geq 1$, $b_v(x_v)$ is linear, $d_v(y)$ is piecewise linear (with at most two pieces) and for any $(v,u) \in E$, $w_{vu}$ is a positive constant.*

*Proof.* The reduction is based on the following construction. Given a graph $G$ and an instance of the Maximum Independent Set (MIS) problem on $G$ we construct a bipartite graph $G'$ and an instance of PG on $G'$. In MIS we are given a graph and we are asked to choose the maximum number of pairwise disjoint nodes.

We will only consider undirected graphs, however, our reduction also applies to directed graphs. Let $G = (V, E)$ be a graph with degree $d(G) \leq d$. Next construct a bipartite graph $G' = (V', U', E')$ with $|V'| = |V|$ and $|U'| = |E|$, where each vertex of $V'$ corresponds to a vertex of $V$ and each vertex of $U'$ corresponds to an edge of $E$. Connect a vertex $v \in V'$ with a vertex $u \in U'$ if the corresponding vertex of $v$ is incident to the corresponding edge of $u$ in G. It can easily be seen that every $v \in V'$ has degree at most $d$ and every $u \in U'$ has degree 2. Let $b_v(x_v) = x_v$ and $d_v(x_v) = 0, \forall v \in V'$. Furthermore, for any $u \in U'$, let $b_u(x_u) = 0$ and $d_u(y) = \frac{|V|(y - \max\{w_{vu}, w_{v'u}\})}{\min\{1, w_{vu}+w_{v'u}\} - \max\{w_{vu}, w_{v'u}\}}$ if $y > \max\{w_{vu}, w_{v'u}\}$ and $d_u(y) = 0$ otherwise, where $(v,u) \in E'$ and $(v',u) \in E'$ . Let $W$ be an independent set of $G$ with $|W| = k \leq p$. Then the welfare of $W$ for PG on $G'$ is $k$. Suppose now there is a better solution $W'$ with $|W'| > |W|$. We then have the following two claims:

**Claim 1.** *$W' \cap U' = \emptyset$.*

Suppose $W' \cap U' \neq \emptyset$. If a vertex $u \in U'$ is included in $W'$, then the valuation ((3.1)) is $r_u = b_u(x_u) + d_u(y) \leq 0$, where $b_u(\cdot)$ and $d_u(\cdot)$ are as defined above. Hence, removing $u$ from $W' \cap U'$ will not decrease the total welfare.

**Claim 2.** *Any two vertices $u, v \in W'$ are not connected to the same vertex in $U'$.*

Let $u, v \in W'$ be two vertices connected to the same vertex in $u' \in U'$. Then $r_u = r_v = 1$ since $u, v \in V'$ (Claim 1) and $b_v(x_v) = x_v$, $d_v(x_v) = 0$, as defined above. However, since for the local level of pollution in $u'$ is $y \geq w_{vu} + w_{v'u} > \max\{w_{vu}, w_{v'u}\}$, we have $r_{u'} = -d_{u'}(y) \leq -|V|$. Hence, the total welfare achieved by $W'$ is at most $|W' \cap V'| - |V| \leq 0 < k$ . Removing either $u$ or $v$ from $W'$ will increase welfare by $|V| - 1$.

Therefore, $W'$ corresponds to an independent set in $G$ with size larger than $|W|$. Thus, any independent set $W$ gives a welfare of $|W|$ in $G'$. As a consequence, if we can find the optimal solution of PG on $G'$, we can find a maximum independent set on $G$. □

From [56] we know that it is strongly NP-hard to find the maximum independent set on a planar graph with degree at most 3.

**Corollary 1.** *For a planar graph $G = (V, E)$ with degree at most 3, the problem of finding an optimal solution in PG setting as in Theorem 5.10 is strongly NP-hard.*

*Proof.* For any planar graph $G = (V, E)$, the constructed graph $G' = (V', U', E')$ in the proof of Theorem 5.10 is planar. To see this, just add one vertex to the center of each edge in $G$ representing the edge vertex in $U'$. The resulting graph is planar and the same as $G'$. The corollary follows from the reduction in the proof of Theorem 5.10. □

**Theorem 6.** *PG is Unique Games-hard [3] to approximate within $n^{1-\epsilon}$ and within $\frac{\Delta}{\log^2 \Delta}$ for graph $G$ with degree $\Delta$ when $p_v$ is any constant number $\geq 1$, $b_v(x_v)$ is linear and $d_v(y)$ is piecewise linear (with two pieces) $\forall v \in V$ and $w_{vu}$ is positive constant for any $(v, u) \in E$.*

*Proof.* According to [80], maximum independent set is Unique Games-hard to approximate within $n^{1-\epsilon}$ in general graphs and within $\frac{\Delta}{\log^2 \Delta}$ for graph $G$ with degree $\Delta$. The theorem follows from the reduction in the proof of Theorem 5.10. □

**Theorem 7.** *There is no EPTAS for PG with binary variables on the directed planar graph $G = (V, E)$ when $b_v$ and $d_v$ are both linear functions, for any $v \in V$.*

*Proof.* We reduce PG with binary variables on planar graphs to the two-dimensional Knapsack problem. Recall (see beginning of section 3.3) that in Knapsack we are given $n$ weighted items and a maximum weight capacity. In the two-dimensional Knapsack the weight of item $i$ is given by a two dimensional vector $w_i = (w_{i1}, w_{i2}), \forall i = 1, \ldots, n$. Furthermore the Knapsack has a two-dimensional capacity $W = (W_1, W_2)$.

Consider now PG on the following simple planar graph. There are $n + 2$ vertices labeled as $\{v_1, v_2, \ldots, n, u, x\}$ and the edge set $E = E_1 \cup E_2$ where $E_1 = \{(u, v_i), i \in [n]\}$ with weights $w_{u,v_i}$, $i \in [n]$ and $E_2 = \{(x, v_i), i \in [n]\}$ with weights $w_{x,v_i}$, $i \in [n]$. Let $p_u$ and $p_x$ be the bounds of the local constraints and $\omega_v$ and $\omega_x$ be the objectives of nodes $u$ and $x$ respectively. Clearly for any two-dimensional Knapsack problem, there exists an instance of PG with binary variables without the global constraint on such a simple graph exactly corresponding to this two-dimensional Knapsack problem, where $W_1 = p_u$, $W_2 = p_x$, the weights of set $E_1$ correspond to the weights $w_{i1}, \forall i = 1, \ldots, n$ and the weights of set $E_2$ correspond to the weights $w_{i2}, \forall i = 1, \ldots, n$. According to [91], there is no EPTAS for two dimensional knapsack. Hence, there is no EPTAS for PG on this simple planar graph. □

[3] The Unique Games Conjecture(UCG) [79] is used to prove inapproximability results for NP-Complete problems which researchers are not able to prove otherwise. The conjecture is based on the inapproximability of the *Unique Game*. A detailed survey can be found in [81].

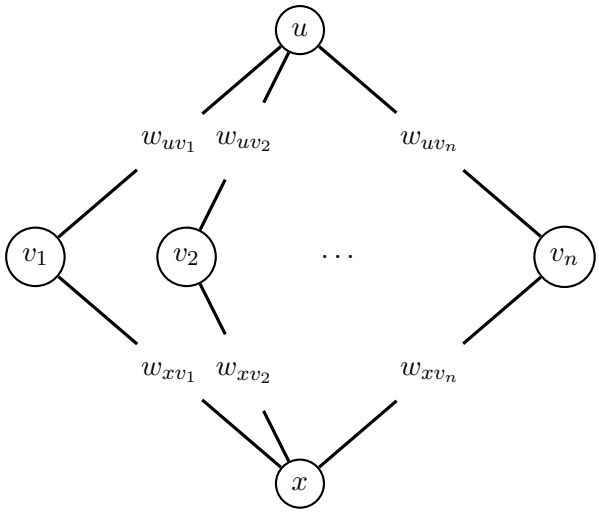



Figure 3.3: The two dimensional knapsack on an instance of a planar graph

## 3.4 Directed trees

We present approximation algorithms and mechanisms for PG on directed trees. A digraph $G$ is called a *directed tree* if the undirected graph $G^{un}$ is a tree. We consider trees whose arcs are directed towards the leaves. We first obtain a truthful in expectation FPTAS for PG on directed trees by a two level dynamic programming approach and a 3-approximate deterministic truthful mechanism which is Maximal in Range (MIR).

We will also need the following tool from mechanism design for packing problems. Recall that an integer linear packing problem with binary variables is a problem of maximising a linear objective function over a set of linear packing constraints, i.e., constraints of form $a \cdot x \leq b$ where $x \in \{0,1\}^n$ is a vector of binary variables, and $a, b \in \mathbb{R}^n_{\geq 0}$.

**Proposition 4** ([46])**.** *Given an FPTAS for an integer linear packing problem with binary variables, there is a truthful in expectation mechanism that is an FPTAS.*

### 3.4.1 Truthful in expectation mechanisms

We obtain our truthful in expectation FPTAS for PG with binary variables on any directed tree by a two-level dynamic programming (DP) approach. The first bottom-up level is based on a careful application of the standard single-dimensional knapsack FPTAS. The second level is by an interesting generalization of an FPTAS of [26] for a special multiple choice multi-dimensional knapsack problem with a constant number of constraints most of whose coefficients are of size $poly(|I|)$ where $I$ denotes the instance. This FPTAS generalizes the results in [26], where the authors consider the one dimensional knapsack problem with cardinality constraint. We first present an FPTAS on directed trees without global constraint which captures our main technical ingredients.

**FPTAS without global constraint**

The algorithm uses a dynamic programming approach and the FPTAS for knapsack problem as a subroutine. Note that on a star, any instance of knapsack can be reduced to a PG instance without global constraints. Thus, an FPTAS is the best we can hope for such PG unless $P = NP$ (see the reduction to Knapsack in the beginning of the section).

We keep four values for each $v \in V$. Suppose that the father of $v$ is $v'$ and let $n_v$ denote the number of children of $v$. Let also $M_{v\text{in}}^{v'\text{in}}$ denote the optimal value of PG on the subtree rooted at $v$ when both $v'$ and $v$ are selected in the solution. Similarly, we also have $M_{v\text{out}}^{v'\text{in}}$, $M_{v\text{in}}^{v'\text{out}}$ and $M_{v\text{out}}^{v'\text{out}}$, where *in* denotes the fact that a node is chosen and *out* that a node is not chosen in the solution. Let $u_i$, $i = 1, \ldots, n_v$ denote the children of $v$. Suppose $M_{u_i\text{in}}^{v\text{in}}$, $M_{u_i\text{out}}^{v\text{in}}$, $M_{u_i\text{in}}^{v\text{out}}$ and $M_{u_i\text{out}}^{v\text{out}}$ have been calculated, for any $i = 1, \ldots, n_v$. Some of them may be undefined due to infeasibility. We will calculate now $M_{v\text{in}}^{v'\text{in}}$. Observe that $M_{v\text{in}}^{v'\text{in}}$ is equal to the optimal value of the following knapsack $(IP_1)$:

$$\begin{aligned} \max \quad & \sum_{i\in[n_v]} (M_{u_i\text{in}}^{v\text{in}} x_{u_i} + M_{u_i\text{out}}^{v\text{in}}(1 - x_{u_i})) + \omega_v \\ \text{s.t.} \quad & 1 + w_{v'v} + \sum_{i\in[n_v]} w_{u_i v} x_{u_i} \leq p_v \\ & x_{u_i} \in \{0, 1\} \ \ \forall i \in [n_v] \end{aligned} \tag{$IP_1$}$$

where $M_{u_i\text{in}}^{v\text{in}}$ and $M_{u_i\text{out}}^{v\text{in}}$ have finite values (otherwise we remove them).

If this knapsack problem has a feasible solution, we get the value $M_{v\text{in}}^{v'\text{in}}$, otherwise we set $M_{v\text{in}}^{v'\text{in}}$ to be undefined. Similarly we calculate $M_{v\text{out}}^{v'\text{in}}$, $M_{v\text{in}}^{v'\text{out}}$ and $M_{v\text{out}}^{v'\text{out}}$. Thus, if we can calculate an optimal solution at each step, this solution will be obtained by the above DP approach. For knapsack with $n_v$ variables, there is an FPTAS. Hence, at each step we get an approximate value $\bar{M}_{v\text{in}}^{v'\text{in}} \geq (1 - \epsilon) M_{v\text{in}}^{v'\text{in}}$ in time polynomial in $n_v$ and $\frac{1}{\epsilon}$ by knapsack's FPTAS. In a similar way we compute approximately the other three values. Thus, in the final solution, $\bar{M}_{root} \geq (1 - \epsilon)^k M_{root}$, where $k$ is the number of levels of the tree and $M_{root}$ is the optimal value of PG without global constraints, terminating in $poly(|I|, \frac{1}{\epsilon})$ time where $|I|$ is the input size. If we let $1 - \epsilon' = (1 - \epsilon)^k$, we have that $\epsilon = \Theta(\frac{\epsilon'}{k})$. The running time is $poly(|I|, \frac{k}{\epsilon'}) = poly(|I|, \frac{1}{\epsilon'})$ due to $k \leq |I|$, giving an FPTAS for PG without global constraint.

**FPTAS with global constraint**

Suppose without loss of generality that $p \leq n$, otherwise let $p = n$. For each vertex $v$, we will keep $4p$ values. Suppose that the father of $v$ is $v'$. Let $M_{v\text{in}}^{v'\text{in}}(s)$ denote the optimal value of PG on the subtree rooted at $v$ when both $v'$ and $v$ are selected in the

solution, and the total pollution level allocated to the subtree rooted at $v$ is no more than $s$, $s = 0, 1, \ldots, p$. Similarly, we also have $M^{v'\text{in}}_{v\text{out}}(s)$, $M^{v'\text{out}}_{v\text{in}}(s)$ and $M^{v'\text{out}}_{v\text{out}}(s)$. Let $u_i$, $i \in [n_v]$ denote the children of $v$. Suppose that $M^{v\text{in}}_{u_i\text{in}}(s)$, $M^{v\text{in}}_{u_i\text{out}}(s)$, $M^{v\text{out}}_{u_i\text{in}}(s)$ and $M^{v\text{out}}_{u_i\text{out}}(s)$ have been calculated, for any $i \in [n_v]$ and $s = 0, 1, \ldots, p$. Some of them may be undefined due to infeasibility. Note that $M^{v\text{in}}_{u_i\text{in}}(0)$, $M^{v\text{out}}_{u_i\text{in}}(0)$ are undefined and $M^{v\text{out}}_{u_i\text{out}}(0) = M^{v\text{in}}_{u_i\text{out}}(0) = 0$. Now we calculate $M^{v'\text{in}}_{v\text{in}}(\ell)$. Observe that $M^{v'\text{in}}_{v\text{in}}(\ell)$ is equal to the optimal value of the following knapsack problem (denoted $\text{KNAPSACK}_v(\ell)$) plus $\omega_v$:

$$
\begin{aligned}
\max \quad & \sum_{i\in[n_v]} \sum_{s\in[p]} (M^{v\text{in}}_{u_i\text{in}}(s)x_{is} + M^{v\text{in}}_{u_i\text{out}}(s)y_{is}) && (IP_2)\\
\text{s.t.} \quad & \sum_{i\in[n_v]} \sum_{s\in[p]} s(x_{is} + y_{is}) \le \ell - 1 \\
& \sum_{s=0}^{p} (x_{is} + y_{is}) = 1, \ \ \forall i \in [n_v] \\
& 1 + w_{v'v} + \sum_{i\in[n_v]} [w_{u_iv}(\sum_{s=0}^{p} x_{is})] \le p_v \\
& x_{is}, y_{is} \in \{0, 1\}, \ \ \forall i \in [n_v], s = 0, 1, \ldots, p.
\end{aligned}
$$

If $M^{v\text{out}}_{u_i\text{in}}(\text{s})$ and $M^{v\text{out}}_{u_i\text{out}}(s)$ do not have finite values they are removed from $\text{KNAPSACK}_v(\ell)$. Note that $x_{i0} \equiv 0$, for any $i \in [n_v]$. If $\text{KNAPSACK}_v(\ell)$ has a feasible solution, then we get the value $M^{v'\text{in}}_{v\text{in}}(\ell)$, otherwise we set $M^{v'\text{in}}_{v\text{in}}(\ell)$ to be undefined. Similarly we calculate $M^{v'\text{in}}_{v\text{out}}(\ell)$, $M^{v'\text{out}}_{v\text{in}}(\ell)$ and $M^{v'\text{out}}_{v\text{out}}(\ell)$, $\ell = 0, 1, \ldots, p$. From the analysis of the dynamic programming approach without global constraints, we know that if there is an FPTAS for $\text{KNAPSACK}_v(\ell)$, then there is FPTAS for $\text{KNAPSACK}_{root}(p)$ giving an FPTAS for PG with binary variables on directed trees. Note that the constraint $\sum_{s=0}^{p}(x_{is} + y_{is}) = 1$ can be replaced by $\sum_{s=1}^{p}(x_{is}+y_{is}) \le 1, \forall i \in [n_v]$. This constraint indicates that exactly one variable from all the possible levels will be chosen in the solution.

**FPTAS for the Special Knapsack Problem**

We will now show how to obtain an FPTAS to compute $(IP_2)$. Consider the following instance $I$ of a Special multiple choice and multi-dimensional Knapsack Problem (denoted as SKP):

$$
\begin{aligned}
\max \quad & H(x) = \sum_{j\in[J]} \sum_{k\in[K]} C_{jk} x_{jk} && (SKP)\\
\text{s.t.} \quad & \sum_{j\in[J]} \sum_{k\in[K]} A'_{jk} x_{jk} \le B' \\
& \sum_{k\in[K]} x_{jk} \le 1 \ \ \forall j \in [J] \\
& \sum_{j\in[J]} \sum_{k\in[K]} A_{ijk} x_{jk} \le B_i, \ \ \forall i \in [N] \\
& x_{jk} \in \{0,1\}, \ \ \forall j \in [J],\, k \in [K]
\end{aligned}
$$

The coefficients of the objective $C_{jk}$ correspond to the coefficients $M^{vin}_{u_i\text{in}}(s)$ and $M^{vin}_{u_i\text{out}}(s)$ $J$ is the number of items available for selection, $K$ denotes the number of different classes of items where at most one item can be chosen from each class and $N$ is the number of dimensions of the constraints or items, where $B_i = poly(|I|)$, $\forall\ i \in [N]$ and $N = O(1)$. Without loss of generality suppose all the parameters in the above knapsack problem are integers and $A_{ijk} \le B_i = poly(|I|)$, $\forall\ i \in [N]$, $j \in [J]$, $k \in [K]$. Let $C = OPT(I)$ and $B = \max_{i\in[N]} B_i$.

**Lemma 1.** *There is a pseudo polynomial optimal algorithm for SKP, terminating in $O(CJKB^N)$ time.*

*Proof.* Let $\ell = (\ell_1, \ell_2, \cdots, \ell_N)$. Consider now the following linear program:

$$
\begin{aligned}
\min \quad & h_s(M,\ell) = \sum_{j\in[s]} \sum_{k\in[K]} A'_{jk} x_{jk} \\
\text{s.t.} \quad & \sum_{j\in[s]} \sum_{k\in[K]} C_{jk} x_{jk} = M \\
& \sum_{j\in[s]} \sum_{k\in[K]} A_{ijk} x_{jk} = \ell_i, \ \ \forall i \in [N] \\
& \sum_{k\in[K]} x_{jk} \le 1, \forall j \in [s]
\end{aligned}
$$

Initially, $h_0(M,\ell) = +\infty$, for all $M$, $\ell$. Then set $h_0(0,0) = 0$. As a result, the recursion can be calculated as follows:

$$
h_s(M,\ell) = \min\{h_{s-1}(M,\ell), \min_{k\in[K]}\{h_{s-1}(M - C_{sk}, (\ell_i - A_{isk})_{i\in[N]})\}\}.
$$

Then the optimal solution of SKP is

$$
\max_{M\le C, \ell_i \le B_i, i\in[N]} \{M : h_J(M,\ell) \le B'\}
$$

Note that the running time of this dynamic programming approach is $O(CJKB^N)$. $\square$

Let $C_{max} = \max_{j\in[J],k\in[K]} C_{jk}$. Note that

$$\frac{OPT^{in}(I)}{J} \leq C_{max} \leq OPT^{in}(I)$$

We now scale all the coefficients in the objective function $H(x)$. Let

$$\tilde{C}_{jk} = \lceil \frac{C_{jk}J}{\epsilon C_{max}} \rceil \leq \frac{C_{jk}J}{\epsilon C_{max}} + 1, \forall j \in [J], k \in [K]$$

The optimal value $\tilde{C}$ of scaled SKP is then upper bounded by

$$\frac{CJ}{\epsilon C_{max}} + J \leq \frac{J^2}{\epsilon} + J$$

Consider the dynamic programming approach running on the scaled SKP as $\mathcal{A}^{scaled}$. Then we have

**Theorem 8.** *$\mathcal{A}^{scaled}$ is an FPTAS for SKP, terminating in $O(\frac{J^3KB^N}{\epsilon})$ time.*

*Proof.* We only need to show the approximation part i.e. that the optimal solution returned is within $1-\epsilon$ of the optimal value of SKP, since the running time straightforwardly follows from the running time of the dynamic programming approach

$$O(\tilde{C}JKB^N) = O((\frac{J}{\epsilon}+1)J^2KB^N) = O(\frac{J^3KB^N}{\epsilon})$$

Let $\tilde{S}$ and $S$ denote the optimal solution of the scaled and the original SKP respectively. Note that $\tilde{S}$ is a feasible solution to the original SKP. By scaling,

$$\frac{\epsilon(\tilde{C}_{jk}-1)C_{max}}{J} \leq C_{jk} \leq \frac{\epsilon\tilde{C}_{jk}C_{max}}{J} \tag{3.14}$$

Then

$$\begin{aligned} H(S) - H(\tilde{S}) &\leq \frac{\epsilon C_{max}}{J}(\tilde{C}(S) - \tilde{C}(\tilde{S}) + |S|) \\ &\leq \frac{\epsilon C_{max}|S|}{J} \leq \epsilon C_{max} \leq \epsilon H(S) \end{aligned}$$

where the last inequality comes from $|S| \leq J$. □

By Proposition 4 we have the following:

**Theorem 9.** *There is a truthful in expectation mechanism for PG with binary variables on directed trees, which is an FPTAS.*

For general $x_v \in \mathbb{Z}$, we can replace each $x_v$ by $q_v$ duplicated variables $x_{vj}$, $j = 1, \cdots, q_v$, i.e., $\{x_v \in \{0,1,\ldots,q_v\}\} = \{\sum_{j\in[q_v]} jx_{vj} \mid \sum_{j\in[q_v]} x_{vj} \leq 1, x_{vj} \in \{0,1\}\}$. Note that this transforms a polynomial size integer constraint into a multiple choice, one dimensional knapsack constraint. Hence, for directed trees, by a DP approach, we can construct a pseudo polynomial time algorithm to compute the exact optimal

value of PG with integer variables, in time $poly(|V|, q, OPT^{in}(PG))$. In addition, we can remove $OPT^{in}(PG)$ from the running time by a loss of $\epsilon$ of the optimal value using scaling techniques. Thus, there is a $(1-\epsilon)$-approximation algorithm for PG with integer variables with running time in $poly(|V|, q, 1/\epsilon)$. Finally, by Proposition 4, we obtain the following:

**Theorem 10.** *There is a truthful in expectation mechanism for PG with polynomial size integer variables on directed trees, which is an FPTAS.*

### 3.4.2 Deterministic truthful mechanisms on directed trees

We will use a maximal in range (MIR) mechanism to obtain a $(3+\epsilon)$ approximate deterministic truthful mechanism for PG with polynomial size integer variables on directed trees. By transformation from integer constraint into multiple choice and one dimensional knapsack constraint (see paragraph before Theorem 10), we only need to show such an approximation algorithm for binary variables. Our mechanism is based on a recent deterministic truthful PTAS for two-dimensional knapsack problem[4] [28,42,88]. We will first need the following:

**Definition 15.** *A vertex in a rooted directed tree is called at level $i$ if the distance between the vertex and the root is $i$ in the undirected version of the tree.*

Let $L_i$ denote the set of vertices of level $3k+i$, $k = 0, 1, 2, \ldots, \mu$, for any $i \in [3]$ where $\mu = \lceil \frac{d}{3} \rceil$ and $d$ is the depth of the tree. For each vertex $v$, suppose that the number of children of $v$ is $n_v$ and that children are $u_1, u_2, \cdots, u_{n_v}$. Let $\Delta = \max_{v \in V}\{n_v\}+1$. Let $G_v$ denote the subtree constructed by $v$ and its children. Then restricting PG on $G_v$ with capacity (global constraint) $c_v$ (enumerating on the values of $c_v < p$) and $x_v = 0$ is equivalent to solving the following linear programming problem (denoted as $PG_v$):

$$
\begin{aligned}
\max \quad & \sum_{i \in [n_v]} \omega_{u_i} x_{u_i} && (PG_v)\\
\text{s.t.} \quad & \sum_{i \in [n_v]} w_{u_i v} x_{u_i} \le p_v \\
& \sum_{i \in [n_v]} c_{u_i v} x_{u_i} \le c_v \\
& x_{u_i} \in \{0,1\}, \ \ \forall i \in [n_v]
\end{aligned}
$$

where $c_{u_i v} = 1$, for $i \in [n_v]$. For any solution $s_v$ of $PG_v$, we use $\omega(s_v)$ to denote the objective value of this solution (recall from (3.12) that linear functions $b(\cdot)$ and $d(\cdot)$ are replaced by $\omega(\cdot)$) given the input $I = (G, \mathbf{b}, \mathbf{d}, \mathbf{p}, \mathbf{q}, \mathbf{w}, p)$. Let us denote by $I_v = (G_v, \omega(s_v), p, q, \mathbf{w}, c_v)$ an instance of $PG_v$ restricted on $G_v$ and $OPT(PG_v(c_v))$ to denote the optimal value of $PG_v(c_v)$ given input $I_v$.

[4]This PTAS also works for multiple choice and constant dimensional knapsack problem, which will be used for PG with polynomial size integer variables.

**Lemma 2** ([28,88]). *There exists a range $\mathcal{S}_v(c_v)$ of solutions of $PG_v(c_v)$, which does not depend on the declarations in $I_v$ and only depends on $c_v$ such that $\max_{s_v\in\mathcal{S}_v(c_v)}\{\omega(s_v)\}\geq(1-\epsilon)OPT$. Besides, there exists an $O(\Delta^{4+\frac{1}{\epsilon}})$ algorithm $\mathcal{A}_v(c_v)$ that finds the optimal solution of the range $\mathcal{S}_v(c_v)$, for any $\epsilon>0$.*

Denote now by $PG_i$ the restriction of PG on $L_i$ and let $\mathcal{S}_i=\bigcup_{v\in\Phi}\mathcal{S}_v(c_v)$, where $\Phi=\{v\in L_i|c_v\in[n_v]\wedge\sum_{v\in L_i}c_v\leq p\}$. In other words $\mathcal{S}_i$ is a range restricted to $L_i$ such that the nodes which are in its solution are no more than $p$. Then $\mathcal{S}_i$ is a range of $PG_i$, $i\in[3]$. The high level idea of the range is to omit pairs of consecutive levels of nodes such that the externalities between neighbouring nodes are avoided.

**Lemma 3.**

*(1).* $\max_{s_i\in\mathcal{S}_i}\{\omega(s_i)\}\geq(1-\epsilon)OPT(PG_i)$.

*(2). There exists an $O(|L_i|\Delta^{6+\frac{1}{\epsilon}})$ algorithm $\mathcal{A}_i$ that finds the optimal solution of the range $\mathcal{S}_i$, for any $\epsilon>0$.*

*Proof.* Suppose that in the optimal solution of $PG_i$, each vertex $PG_v$ is allocated $c_v^*$ amount of global pollution level. As we know $\sum_{v\in L_i}c_v^*\leq p$. Then

$$\max_{s_i\in\mathcal{S}_i}\{\omega(s_i)\}\geq\sum_{v\in L_i}\max_{s_v\in\mathcal{S}_v(c_v^*)}(\omega(s_v))\geq(1-\epsilon)\sum_{v\in L_i}OPT(PG_v(c_v^*))=(1-\epsilon)OPT(PG_i)$$

where the first inequality comes from $\sum_{v\in L_i}c_v^*\leq p_v$, the second one is due to Lemma 2, and the third one is from the definition. Suppose the fathers of vertices in $L_i$ are labeled as $v_1,v_2,\cdots,v_{\ell_i}$. Let $g_i(C)$ denote the optimal value of PG restricted to vertices with fathers $v_1,v_2,\cdots,v_i$ on the range $\mathcal{S}_i$ when the capacity allocated to this subproblem is no more than $C$. We have the following recursive function:

$$g_{i+1}(C)=\max_{c_{v_{i+1}}\leq C}\{g_i(C-c_{v_{i+1}})+OPT(PG_{v_{i+1}}(c_{v_{i+1}}))\}$$

where $OPT(PG_i)=\max_{i\in[\ell_i],C\leq p}g_i(C)$. The total running time of this dynamic programming approach is $O(|L_i|\Delta^2\Delta^{4+\frac{1}{\epsilon}})=O(|L_i|\Delta^{6+\frac{1}{\epsilon}})$. □

**Theorem 11.** *There is a deterministic $\rho^{in}$-approximate truthful mechanism for PG with polynomial size integer variables on directed trees, where $\rho^{in}=3+\epsilon$. For binary variables the mechanism terminates in $O(|V|^2\Delta^{6+\frac{1}{\epsilon}})$ time.*

*Proof.* We only need to prove this theorem for PG with binary variables. Note that $\max_{i\in[3]}\{OPT(PG_i)\}\geq\frac{1}{3}OPT(PG)$ since $OPT(PG)=\sum_{i=1}^3OPT(PG_i)$. Then by Lemma 3 we have

$$\max_{i\in[3]}\{\max_{s_i\in\mathcal{S}_i}\{\omega(s_i)\}\}\geq\frac{1-\epsilon}{3}OPT(PG)$$

Using VCG payment rule on the range $\mathcal{S}=\bigcup_{i\in[3]}\mathcal{S}_i$, we can get a deterministic truthful mechanism for PG on directed trees, achieving $\frac{1-\epsilon}{3}OPT(PG)$ social welfare. The running time $O(|V|^2\Delta^{6+\frac{1}{\epsilon}})$ follows directly by Lemma 3 and the payment rule of VCG. □

## 3.5 Planar graphs

We present two algorithms for PG on planar graphs. The first has a constant approximation ratio, obtained by decomposing the graph without violating any constraint. The second algorithm is a PTAS, obtained by rounding the variables and a dynamic programming approach on a tree decomposition of the graph. This PTAS violates the local constraints by a small value $\delta > 0$.

### 3.5.1 Constant approximation without violations

Given a digraph $G = (V, E)$ and a subset $U \subset V$, we call significant neighbours of $U$, $SN_G(U)$, all the vertices in $V \setminus U$ with at least two neighbours in $U$ (see Figure 3.4). Consider a partition $\{V^i\}_{i=1}^{\alpha}$ of $V$. Now let $SN_{G^{un}}(V^i) = \{u \notin V^i \,|\, \exists v \in V^i, \text{s.t. } u \text{ is a significant neighbour of } v \text{ w.r.t. } V^i\}$ denote the significant neighbours of $V^i$ in $G^{un}$. Let $G^i$ be the induced subgraph of $V^i \cup SN_{G^{un}}(V^i)$ in $G^{un}$. A partition $\{V^i\}_{i=1}^{\alpha}$ of $V$ is called an $(\alpha, \beta)$-partition (or $(\alpha, \beta)$-decomposition) of $G$ if for any $i \in [\alpha]$ and $v \in V^i$, $|\delta^-_{G^i(v)}| \leq \beta$, where $\alpha, \beta$ are two given positive integers and $|\delta^-_{G^i(v)}|$ is the number of neighbours of $v$ in $G^i$.

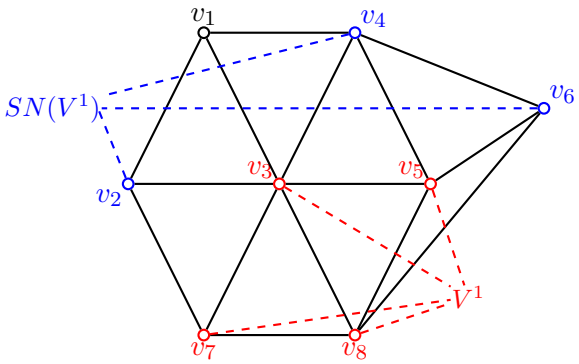



Figure 3.4: Significant neighbours $SN(V^1) = \{v_2, v_4, v_6\}$ of $V^1 = \{v_3, v_5, v_7, v_8\}$ in the graph of solid black lines.

According to the following Lemma 4, we can obtain a constant approximation for PG with integer variables for any graph with $(\alpha, \beta)$-decomposition, where $\alpha$ and $\beta$ are constants. Such a decomposition of planar graphs will be presented later. Recall from Propositions 1 and 2 that $\gamma_k = (e + o(1))k = O(k)$.

**Lemma 4.** *If a directed graph $G$ has an $(\alpha, \beta)$-decomposition, then there is a deterministic $(\rho^{fr} = \alpha\gamma_{\beta+2} + 1)$-approximation algorithm for PG with integer variables, and, a truthful in expectation mechanism for the same problem with the same approximation.*

*Proof.* If there is an $\rho^{fr}$-approximation algorithm for a linear packing problem with binary variables, then there is an $(\rho^{fr}+1)$-approximation algorithm for the same problem with integer variables [17]. Hence, it is sufficient to show that there is an $(\rho^{fr} = \alpha\gamma_{\beta+2})$-

approximation algorithm for PG with binary variables. Now we consider PG with binary variables. Let $\{V^i\}_{i=1}^{\alpha}$ be an $(\alpha, \beta)$-decomposition of graph $G$. Let $x^*$ be the optimal fractional solution of PG with binary variables. Then $R(x^*) \leq \alpha \max_{i\in[\alpha]}\{R(x^*_{V^i})\}$, where $x^*_{V^i}$ is a fractional solution such that its value is equal to $x^*_v$, for any $v \in V^i$ and 0 otherwise. Let $PG_i$ denote the PG on $G$ by setting $x_v = 0$, for any $v \notin V^i$. Note that $x^*_{V^i}$ is a feasible solution for $PG_i$, which gives $R(x^*_{V^i}) \leq OPT^{fr}(PG_i)$. Without loss of generality, we suppose $w_{uv} \leq p_v$, for any $(u, v) \in E$ and $v \in V$ (otherwise $x_v = 0$ for PG). Observe that in $PG_i$, only $x_v$, $v \in V^i$ are variables. Now for any $v \in V^i$, let us see how many constraints in $PG_i$ contain $x_v$. Suppose $u \in V \setminus V^i$ is a neighbour of $v$ in $G^{un}$. If $u$ is not a significant neighbour of $v$, since $w_{vu} \leq p_u$, we can remove the constraint $w_{vu}x_v \leq p_v$ in $PG_i$. Hence, only the local constraints of the significant neighbours of $v$ remain containing variable $x_v$. As $\{V^i\}_{i=1}^{\alpha}$ is an $(\alpha, \beta)$-decomposition of graph $G$, there are at most $\beta + 1$ local constraints containing variable $x_v$ (which includes the local constraint of vertex $v$ itself). Together with the global constraint, we know $x_v$ appears in at most $\beta+2$ constraints in $PG_i$, for any $v \in V^i$, which means $PG_i$ is $\beta+2$ column sparse. Therefore, by Proposition 1, there is a polynomial deterministic algorithm for $PG_i$ with binary variables, finding an integer solution $y_i$ for $PG_i$ such that $\gamma_{\beta+2}R(y_i) \geq OPT^{fr}(PG_i)$, for any $i \in [\alpha]$. Then

$$\alpha\gamma_{\beta+2} \max_{i\in[\alpha]}\{R(y_i)\} \geq \alpha \max_{i\in[\alpha]}\{OPT^{fr}(PG_i)\} \geq \alpha \max_{i\in[\alpha]}\{R(x^*_{V^i})\} \geq R(x^*) = OPT^{fr}(PG)$$

A truthful in expectation mechanism with the same approximation ratio is guaranteed by Proposition 3. □

**Planar graphs.** The integrality gap of PG on planar graphs is at least 4 as shown by a complete graph with four vertices. For a small $\epsilon > 0$, let $w_{uv} = \epsilon$, for any $(u, v) \in E$, and $p_v = \omega_v = 1$, for any $v \in V$. There is no global constraint. The optimal integer solution of PG on this graph is $x_v = 1$ for some $v \in V$ and $x_u = 0$ for all $u \neq v$, implying the optimal objective value 1. However, setting $x_v = 1 - 4\epsilon$, for any $v \in V$ provides a feasible fractional solution, which gives the objective value $4 - 16\epsilon$. Therefore, the integrality gap is at least 4, meaning that our LP relaxation cannot lead to better than 4 (e.g., PTAS) approximations.

We provide an $(\alpha, \beta)$-decomposition of any planar graph, with $\alpha = 18$, $\beta = 6$. We did not attempt to optimize these two parameters. However, we note that this is the first algorithm with constant approximation for PG on planar graphs.

**Theorem 12.** *There is an $(\alpha, \beta)$-decomposition of a directed planar graph $G = (V, E)$, where $(\alpha, \beta) = (18, 6)$.*

*Proof.* Let $G' = G^{un}$. Suppose $G'$ is connected, otherwise we can run the algorithm on each connected component respectively. Define the sequence of vertex sets $\{N_i\}_i$ of $G'$

as follows. Fix an arbitrary vertex $v_0 \in V$, and let $N_1 = \{v_0\}$; $N_i$ is defined recursively as

$$N_i = \{v \in V \setminus \bigcup_{j=1}^{i-1} N_j \,|\, (v,u) \in G', \text{for some}\, u \in N_{i-1}\},$$

for $i = 1, 2, \ldots, |V|$. By this definition, for any $v \in N_i$ and $u \in N_j$, if $|i-j| \geq 2$, then $(u,v) \notin E'$. We also observe that $N_i$ is the set of vertices with distance $i-1$ to $v_0$ in $G'$ (i.e., the shortest path distance with respect to the number of edges). Suppose the length of the sequence $\{N_i\}_i$ is $K$. Let $S_i = \{j \equiv i \,(\mathrm{mod}\, 3) \,|\, j \in [K]\}$, $i \in [3]$. Let $S_0 = S_3$, and $V^i = \bigcup_{j \in S_i} N_j$, $i \in [3]$. We will need the following Lemmas 5, 6 and 7.

**Lemma 5.** *For each $v \in N_j$, the number of significant neighbours of $v$ in $N_{j-1}$ with respect to $N_j$ is at most two.*

*Proof.* Suppose there exists $v_1 \neq v_2 \neq v_3 \in N_{j-1} \cap \delta_{G'}(v)$ and $v \neq u_1, u_2, u_3 \in N_j$ such that $(u_i, v_i) \in G'$, $i \in [3]$ (see Figure 3.5). By the definition of $N_j$, there is a path from $v_0$ to $v_i$, $i \in [3]$, and $(v, v_i) \in G'$, $i \in [3]$. Suppose without loss of generality that $v_2$ is inside the circle constructed from the path of $v_0$ to $v_1$, $v_3$ and edges $(v, v_1)$ and $(v, v_3)$ in the planar embedding. Then $(u_2, v_2)$ will intersect this circle, which contradicts that $G'$ is planar. □

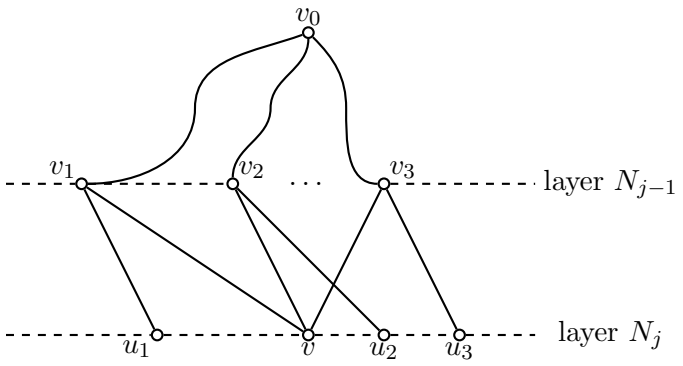



Figure 3.5: An illustration of relations between $N_j$ and $N_{j-1}$

Next, we partition $N_j$ into two sets $N_j^1$ and $N_j^2$ such that each vertex in $N_j^i$, $i \in [2]$, has at most two significant neighbours in $N_{j+1}$. We say two vertices $v, u \in N_j$ are connected by a zigzag path if there exists a path $(v, v_1, v_2, v_3, \ldots, v_s, u)$ in $G'$ such that $v_i$ and $v_{i+1}$ alternatively belong to $N_{j+1}$ and $N_j$, i.e., $v_1 \in N_{j+1}$ and $v_2 \in N_j$. Note that $s$ must be odd. We define the zigzag length of this zigzag path as $\frac{s+1}{2}$. The zigzag distance between $v$ and $u$, denoted $d_{uv}^z$, is defined as the zigzag length of the shortest zigzag path between $v$ and $u$ if there exists one. Otherwise we set it to be undefined. Note that the zigzag distance of $v$ to itself is zero. The partition algorithm $\mathrm{PA}_1$ works as follows (see Algorithm 1). Let $N_j^1 = A_1$ and $N_j^2 = A_2$, where $A_1, A_2$ is output of $\mathrm{PA}_1$. (Note that $\mathrm{PA}_1$ is run for each $j \in [K]$.)

**Algorithm 1:** ($\mathrm{PA}_1$)

**Input**: $A_1, A_2 \leftarrow \emptyset$, $B \leftarrow N_j$
**Output**: $A_1$, $A_2$

```
while B ≠ ∅ do
    Select a vertex v ∈ B;
    A_1 ← A_1 ∪ B_1; A_2 ← A_2 ∪ B_2;
    B ← B \ (B_1 ∪ B_2);
    forall the nodes u ∈ V discovered by BFS(v) do
        if d^z_uv is odd then
            B_1 = B_1 ∪ {u}
        else if d^z_uv is finite then
            B_2 = B_2 ∪ {u}
```

**Lemma 6.** *For each $v \in N_j^i$, $v$ has at most two significant neighbours in $N_{j+1}$ with respect to $N_j^i$, $i \in [2]$.*

*Proof.* First, note that if $v$ and $u$ are selected in different iterations of the while loop in Algorithm 1, there is no zigzag path between them. Therefore, for a single iteration of the while loop, suppose $v \in B$ is selected. We only need to show that for any $u \in B_i$, $u$ has at most two significant neighbours in $N_{j+1}$ with respect to $B_i$, $i \in [2]$. First, note that $v \in B_2$ (its zigzag distance to itself is 0). Since all the other vertices in $B_2$ have zigzag distance to $v$ at least two, $v$ has no significant neighbours with respect to $B_2$ in $N_{j+1}$. Now fix $i \in [2]$. Consider any two vertices $u_1, u_2 \in B_i$, $u_1$ and $u_2$ connect to the same vertex in $N_{j+1}$ only if they have the same zigzag distance to $v$. Suppose there exists three different vertices $v_1$, $v_2$, $v_3 \in N_{j+1}$, such that they are significant neighbours of $u_1$ with respect to $B_i$ (see Fig. 3.6). By similar arguments as above, there exists zigzag paths from $v$ to $v_i$, $i \in [3]$. Also note that edges $(u_1, v_i) \in G'$, $i \in [3]$. Without loss of generality, suppose $v_2$ is in the circle constructed from the zigzag paths $v$ to $v_1$, $v_3$ and edges $(u_1, v_1)$ and $(u_1, v_3)$. Since $G'$ is planar, there exists no edge between $v_2$ and another vertex in $B_i$ with the same zigzag distance to $u_1$. Therefore, $u_1$ has at most two significant neighbours with respect to $B_i$ in $N_{j+1}$. □

Next we will partition each set $N_j^i$, $i \in [2]$, $j \in [K]$ into a constant number of sets $\{N_j^{ik}\}_k$ such that each vertex in $N_j^{ik}$ has at most a constant number of significant neighbours with respect to $N_j^{ik}$ in $N_j$. We provide a partition algorithm which in spirit is similar to Algorithm 1. For any two vertices $v$, $u \in N_j^i$, we say they are connected by a $N_j$-path if there exists a path $(v, v_1, v_2, \cdots, v_s, u)$ in $G'$ such that $v_\ell \in N_j$, $\forall \ell \in [s]$. $N_j$-distance of two vertices $v$, $u \in N_j^i$, denoted $d_{uv}^{N_j}$, is defined as the number of edges of the shortest $N_j$-path between $v$ and $u$ if there exists one and $\infty$ otherwise. The process works as $\mathrm{PA}_2$ (Algorithm 2). Note that $v \in B_3$, because the $N_j$-distance from $v$ to

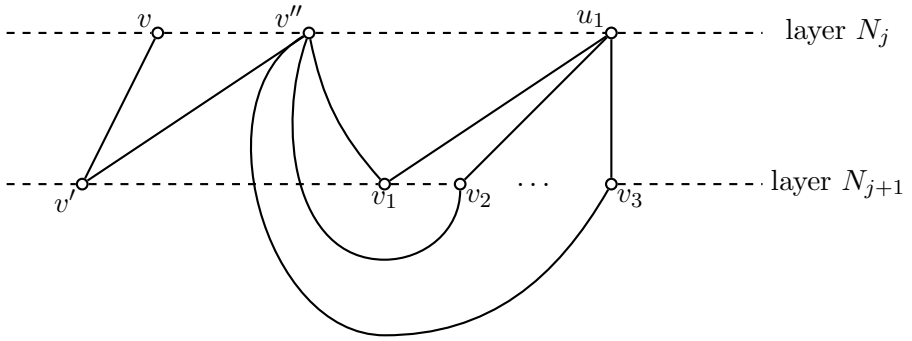



Figure 3.6: Relations between $N_j$ and $N_{j+1}$

itself is zero. Let $N_j^{ik} = A_k$, $k \in [3]$ (where $A_1$, $A_2$, $A_3$ are a partition of $N_j^i$ output by $\mathrm{PA}_2$).

**Algorithm 2:** ($\mathrm{PA}_2$)

**Input**: $A_1, A_2, A_3 \leftarrow \emptyset$, $B \leftarrow N_j^i$
**Output**: $A_1$, $A_2$, $A_3$
**while** $B \neq \emptyset$ **do**
  Select a vertex $v \in B$;
  Find $B_k$'s below by BFS.
  **for** $k \leftarrow 1$ **to** 3 **do**
    $B_k \leftarrow \{u \in N_j^i \mid d_{uv}^{N_j} \equiv k (\mathrm{mod}\ 3)\}$
    $A_k \leftarrow A_k \cup B_k$;
  $B \leftarrow B \setminus (B_1 \cup B_2 \cup B_3)$;

**Lemma 7.** *For any $k \in [3]$, and each $v \in N_j^{ik}$, $v$ has at most 2 neighbours in $N_j$, or has no neighbours in $N_j^{ik}$ nor significant neighbours with respect to $N_j^{ik}$ in $N_j \setminus N_j^{ik}$.*

*Proof.* First, note that if $v$ and $u$ are selected in different iterations of while loop in Algorithm 2, there is no $N_j$-path between them. Therefore, for a single iteration of the while loop, suppose $v \in B$ is selected. Since $v \in B_3$ ($N_j$ distance to itself is 0), $v$ has no neighbours in $B_3$ nor significant neighbours with respect to $B_3$ in $N_j \setminus B_3$ by $\mathrm{PA}_2$. Now fix $k \in [3]$. Consider any two vertices $u_1, u_2 \in B_k$, $u_1$ and $u_2$ connect to the same vertex in $N_j$ only if they have the same $N_j$-distance to $v$. Next we will show for any $u_1 \neq v$ and $u_1 \in B_k$, for any $k$, $u_1$ has at most two neighbours in $N_j$. Suppose there exist three different vertices $u_1$, $u_2$, $u_3 \in N_j$, such that $(u_1, u_2) \in G'$ and $(u_1, u_3) \in G'$. By similar arguments as above, there exists $N_j$-paths from $v$ to $u_i$, $i \in [3]$. Since $u_i \in N_j$, $i \in [3]$, there exist paths in $G'$ from $v_0$ to $u_i$, $i \in [3]$. We observe that it is only possible that $u_1$ is in the circle constructed from the $N_j$ paths $v$ to $u_2$, $u_3$ and paths from $v_0$ to $u_2$ and $u_3$ (the case where $u_2$ or $u_3$ is in the circle constructed by the other two vertices with $v$ and $v_0$ will violate the planarity of $G'$) (see Fig. 3.7). Since graph $G'$ is planar, there exists no edge between $u_1$ and another vertex in $N_j$ (due to that such a vertex

will have a path to $v$ and $v_0$ respectively). Therefore, $u_1$ has at most two neighbours in $N_j$. □

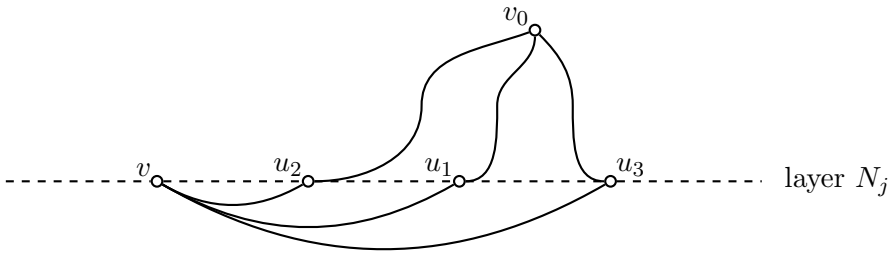



Figure 3.7: Relations between $N_j$ and $N_j$

Combining Lemmas 5, 6 and 7, $\{N_j^{ik}\}_{ijk}$ is an $(\alpha, \beta)$- decomposition of $G$ with $(\alpha, \beta) = (18, 6)$, thus finishing the proof. □

By Theorem 12 and Lemma 4, and observing that $18 \min\{\gamma_8, 3\gamma_6\} = 18\gamma_8 = O(1)$, we have

**Theorem 13.** *There is a randomized, individually rational and truthful in expectation* $(18\gamma_8 + 1)$*-approximation mechanism for PG on planar graphs with integer variables.*

### 3.5.2 Better approximation under some mild condition

We will use the 4-color theorem for planar graphs to present an improved $(6 + \epsilon)$-approximate truthful in expectation mechanism for PG under the following natural (and mild) assumption:

$$\sum_{u \in \delta_G^-(v)} w_{uv} \leq p_v \tag{3.15}$$

This constraint means that if each of $v$'s neighbours emits only one unit amount of pollution, the level of pollution in $v$ will not exceed $v$'s local level of pollution. Let $x^1$ be the optimal fractional solution of PG with binary variables without global constraint on planar graph $G$.

**Theorem 14.** *Suppose condition (3.15) holds and* $R(x^1) \geq 1$*. There is a randomized, individually rational,* $(\rho^{fr} = 6 + \epsilon)$*-approximation mechanism that is truthful in expectation for PG on planar graphs with integer variables, terminating in time* $poly(|I|, \log(\frac{1}{\epsilon}))$*.*

*Proof.* Note that if condition (3.15) holds, then every independent set is a feasible solution for PG with binary variables without global constraint. By 4-color theorem [12,121] for planar graphs, there is an independent set $S \subset V$ such that $4R(z_S) \geq R(x^1)$ where $z_S$ is defined by $z_v = 1$ if $v \in S$ and $z_v = 0$ otherwise. Further there is an $O(|V|^2)$

algorithm finding $z_S$ [121]. By Theorem 3 of [90] and $R(x^1) \geq 1$, there is a deterministic ($\rho^{fr} = 5 + \epsilon$)-approximation algorithm for PG with binary variables, running in $poly(|I|, \log(\frac{1}{\epsilon}))$ time. Then there is a deterministic ($\rho^{fr} = 6 + \epsilon$)-approximation algorithm for PG with integer variables, running in time $poly(|I|, \log(\frac{1}{\epsilon}))$ [17]. By Proposition 3, this ($\rho^{fr} = 6 + \epsilon$)-approximation mechanism is truthful in expectation for PG with integer variables. □

### 3.5.3 A PTAS with $\delta$ violation of constraints

**A PTAS with $\delta$-violation**: Our approach to obtain a PTAS has three main steps:

1. Round $PG$ to an equivalent problem $\bar{PG}_2$ with polynomial size integer variables.
2. Using the nice tree decomposition (see Definition 17 later on this chapter), we present a dynamic programming approach to solve $\bar{PG}_2$ optimally on a $k$-outerplanar graph.
3. By a shifting technique similar to [15], we obtain a PTAS with $1 + \delta$ violation of local constraints for $PG$.

**Step 1: Rounding Procedure.** Recall from (3.12) that PG is equivalent to the following integer linear program:

$$
\begin{aligned}
\max \quad & \sum_{v \in V} \omega_v x_v && (PG)\\
\text{s.t.} \quad & \sum_{v \in V} x_v \leq p \\
& w_{vv} x_v + \sum_{u \in \delta_G^-(v)} w_{uv} x_u \leq p_v, \ \ \forall v \in V \\
& x_v \in \{0, \ldots, q_v\}, \ \ \forall v \in V
\end{aligned}
$$

where $\omega_v = \max\{0, s_v^0 - s_v^1 - \sum_{u \in \delta_G^+(v)} s_u^1 w_{vu}\}$ and $w_{v,v} = 1 \ \forall v \in V$, and $b_v$ and $d_v$ are both linear with slopes $s_v^0$ and $s_v^1$. For each $v \in V$, suppose $q_v \in [2^{o_v - 1} - 1, 2^{o_v} - 1)$. We next consider an encoding of $x_v$ using a bit representation. Let $o_v = \lfloor \log_2(q_v) \rfloor + 1$ if $q_v \neq 2^{o_v - 1} - 1$ and $o_v = \lfloor \log_2(q_v) \rfloor + 2$ otherwise; $c_v^i = 2^{i-1}$, $i \in [o_v - 1]$ and $c_v^{o_v} = q_v - 2^{o_v - 1} + 1$. By simple calculations, we know

$$\{x_v \,|\, x_v \in \mathbb{Z}, 0 \leq x_v \leq q_v\} = \{\sum_{i=1}^{o_v} c_v^i y_v^i \,|\, y_v^i \in \{0, 1\}, i \in [o_v]\}$$

for any $v \in V$. Therefore, PG is equivalent to the following integer linear programming problem (denoted as $PG'$):

$$
\begin{aligned}
\max \quad & \sum_{v\in V}\sum_{i=1}^{o_v}\omega_v c_v^i y_v^i && (PG')\\
\text{s.t.} \quad & \sum_{v\in V}\sum_{i=1}^{o_v} c_v^i y_v^i \le p\\
& \sum_{i=1}^{o_v} w_{vv} c_v^i y_v^i + \sum_{u\in\delta_G^-(v)}\sum_{i=1}^{o_v} w_{uv} c_u^i y_u^i \le p_v,\ \ \forall v\in V\\
& y_v^i \in \{0,1\},\ \ \forall v\in V, i\in[o_v]
\end{aligned}
$$

Let $o^* = \max_{v\in V} o_v$ and $\rho = o^*|V|$. Recall that $q = \max_{v\in V}\{q_v\}+1$. For any $\delta > 0$, let

$$\bar{w}_{uv}^i = \lfloor \frac{2w_{uv}c_v^i\rho}{p_v\delta} \rfloor \text{ and } \bar{p}_v = \lceil \frac{2p_v\rho}{p_v\delta} \rceil = \lceil \frac{2\rho}{\delta} \rceil$$

for any $u, v \in V$. Then we have the following modified $PG'$ denoted as $\bar{PG}_1$:

$$
\begin{aligned}
\max \quad & \sum_{v\in V}\sum_{i=1}^{o_v}\omega_v c_v^i y_v^i && (PG_1)\\
\text{s.t.} \quad & \sum_{v\in V}\sum_{i=1}^{o_v} c_v^i y_v^i \le p\\
& \sum_{i=1}^{o_v} \bar{w}_{vv}^i y_v^i + \sum_{u\in\delta_G^-(v)}\sum_{i=1}^{o_v} \bar{w}_{uv}^i y_v^i \le \bar{p}_v,\ \ \forall v\in V\\
& y_v^i \in \{0,1\},\ \ \forall v\in V, i\in[o_v]
\end{aligned}
$$

**Lemma 8.** *Any feasible solution of $PG'$ is feasible in $\bar{PG}_1$, and any feasible solution of $\bar{PG}_1$ is feasible for $PG$ except violating each local constraint by a factor of $1+\delta$.*

*Proof.* We only prove local constraints for each direction since the proof of the global constraint is similar. Let $\{y_v^i\}_{v\in V, i\in[o_v]}$ be a feasible solution of $PG'$. We know that

$$\sum_{i=1}^{o_v} w_{vv} c_v^i y_v^i + \sum_{u\in\delta_G^-(v)}\sum_{i=1}^{o_v} w_{uv} c_u^i y_v^i \le p_v, \forall v\in V$$

Then

$$\sum_{i=1}^{o_v} \bar{w}_{vv}^i y_v^i + \sum_{u\in\delta_G^-(v)}\sum_{i=1}^{o_v} \bar{w}_{uv}^i y_v^i \le \frac{2\rho}{p_v\delta}\left(\sum_{i=1}^{o_v} w_{vv} c_v^i y_v^i + \sum_{u\in\delta_G^-(v)}\sum_{i=1}^{o_v} w_{uv} c_u^i y_v^i\right) \le \frac{2\rho}{p_v\delta} p_v \le \bar{p}_v \tag{3.16}$$

as desired. On the other hand, suppose $\{y_v^i\}_{v\in V, i\in[o_v]}$ is a feasible solution of $\bar{PG}_1$. We know that

$$\sum_{i=1}^{o_v} \bar{w}_{vv}^i y_v^i + \sum_{u\in\delta_G^-(v)}\sum_{i=1}^{o_v} \bar{w}_{uv}^i y_v^i \le \bar{p}_v, \forall v\in V$$

Then from (3.16) we have:

$$\sum_{i=1}^{o_v} w_{vv}c_v^i y_v^i + \sum_{u\in\delta_G^-(v)}\sum_{i=1}^{o_v} w_{uv}c_u^i y_v^i \le \frac{p_v\delta}{2\rho}\left(\sum_{i=1}^{o_v}(\bar{w}_{vv}^i+1)y_v^i + \sum_{u\in\delta_G^-(v)}\sum_{i=1}^{o_v}(\bar{w}_{uv}^i+1)y_v^i\right)$$

$$\le \frac{p_v\delta}{2\rho}\sum_{i=1}^{o_v}\bar{w}_{vv}^i y_v^i + \sum_{u\in\delta_G^-(v)}\sum_{i=1}^{o_v}\bar{w}_{uv}^i y_v^i + \frac{p_v\delta\rho}{2\rho} \le \frac{p_v\delta\bar{p}_v}{2\rho} + \frac{p_v\delta}{2} \le \frac{p_v\delta}{2\rho}\left(\frac{2\rho}{\delta}+1\right) + \frac{p_v\delta}{2}$$

$$\le p_v(1+\delta), \forall v \in V$$

□

Note that for each $\ell \in [q_v]$, there is a solution $\{y_v^i\}_{i\in[o_v]}$ such that $\sum_{i=1}^{o_v} c_v^i y_v^i = \ell$. If $\ell \le 2^{o_v-1}-1$, set $y_v^{o_v}=0$ and if $2^{o_v-1}-1<\ell\le q_v$, set $y_v^{o_v} = q_v - 2^{o_v-1}+1$. In both cases there is a unique solution such that $\sum_{i=1}^{o_v} c_v^i y_v^i = \ell$. Hence, there is a one-to-one correspondence from $x_v$ to $\{y_v^i\}_{i\in[o_v]}$. It is not difficult to see that for a given $x_v$, the solution $\{y_v^i\}_{i\in[o_v]}$ defined above is the one such that $\sum_{i=1}^{o_v}\bar{w}_{vv}^i y_v^i + \sum_{u\in\delta_G^-(v)}\sum_{i=1}^{o_v}\bar{w}_{uv}^i y_v^i$ is minimized. Now let $\bar{w}_{vu}(x_v) = \sum_{i=1}^{o_v}\bar{w}_{vu}^i y_v^i$, for any $v,u\in V$, where $\{y_v^i\}_{i\in[o_v]}$ corresponds to the solution of $x_v$. Let $\Lambda_v = [q_v]\cup\{0\}$. Using these notations, we know that $\bar{PG}_1$ (also PG) is equivalent to the following integer linear programming problem (denoted as $\bar{PG}_2$):

$$\begin{aligned}
\max \quad & \sum_{v\in V}\omega_v x_v && (\bar{PG}_2)\\
\text{s.t.} \quad & \sum_{v\in V} x_v \le p\\
& \bar{w}_{vv}(x_v) + \sum_{u\in\delta_G^-(v)}\bar{w}_{uv}(x_u) \le \bar{p}_v, \ \ \forall v\in V\\
& x_v\in\Lambda_v, \ \ \forall v\in V
\end{aligned}$$

**Step 2: Preliminaries of tree decompositions on $k$-outerplanar graphs**

**Definition 16.** *A tree decomposition of an undirected graph $G=(V,E)$ is a pair $(\{X_i|i\in I\}, T=(I,F))$, where $\{X_i|i\in I\}$ is a family of subsets of $V$, one for each node of $T$, and $T$ is a tree such that:*

1. *$\bigcup_{i\in I} X_i = V$,*
2. *for all edges $(v,w)\in E$, there exists an $i\in I$ with $v\in X_i$ and $w\in X_i$,*
3. *for all $i$, $j$, $k\in I$: if $j$ is on the path from $i$ to $k$ in $T$, then $X_i\cap X_k\subseteq X_j$ (running intersection property)*

*The width of a tree decomposition $(\{X_i|i\in I\}, T=(I,F))$ is $\max_{i\in I}|X_i|-1$. The minimum width of all tree decompositions of $G$ is called treewidth.*

**Definition 17.** *A tree decomposition* $(\{X_i|i \in I\}, T = (I, F))$ *of* $G = (V, E)$ *is called a nice tree decomposition if* $T$ *is a rooted binary tree and*

1. *if a node* $i \in I$ *has two children* $j$ *and* $k$*, then* $X_i = X_j = X_k$ *(joint node),*
2. *if a node* $i \in I$ *has one child* $j$*, then either* $X_i \subset X_j$*, and* $|X_i| = |X_j| - 1$ *(forget node), or* $X_j \subset X_i$ *and* $|X_j| = |X_i| - 1$ *(introduce node),*
3. *if node* $i \in I$ *is a leaf of* $T$*, then* $|X_i| = 1$ *(leaf node).*

**Lemma 9** ( [77])**.** *For any* $k$*-outerplanar graph* $G = (V, E)$*, there is an algorithm to compute a tree decomposition* $(\{X_i|i \in I\}, T = (I, F))$ *of* $G$ *with treewidth at most* $3k - 1 = O(k)$*, and* $I = O(|V|)$ *in* $O(k|V|)$ *time.*

Given a tree decomposition $(\{X_i|i \in I\}, T = (I, F))$ for $G = (V, E)$ with treewidth $k$ and $I = O(|V|)$, we can obtain a nice tree decomposition with the same treewidth $k$ and the same number of nodes $O(k|V|)$ in $O(k^2|V|)$ time [84]. Thus, for any $k$-outerplanar graph $G = (V, E)$, we can compute a nice tree decomposition $(\{X_i|i \in I\}, T = (I, F))$ of $G$ with treewidth at most $3k - 1 = O(k)$, and $I = O(k|V|)$ in $O(k^2|V|)$ time. In the following, we will assume there is a nice tree decomposition for any $k$-outerplanar graph.

**Dynamic Programming (DP).** We present a DP approach to solve $\bar{PG}_2$ on a directed $k$-outerplanar graph using a nice tree decomposition of its undirected version. Note that a nice tree decomposition of an undirected version of a directed graph is also a nice tree decomposition of itself. Suppose we have a nice tree decomposition $(\{X_i|i \in I\}, T = (I, F))$ of a directed $k$-outerplanar graph $G = (V, E)$. We will use a bottom-up DP approach for $\bar{PG}_2$. In the following we will present our DP approach to the more general application of the allocation of pollution licences (application 2).

For any node $i \in I$, suppose $X_i = \{v_1^i, v_2^i, \cdots, v_t^i\}$, where $t \leq 3k$. We say that vertex $v_1^i$ belongs to node $X_i$. Similarly we say that a vertex belongs to a subtree of $T$, meaning that this vertex belongs to some node of this subtree. Recall that given any allocation of licences $\{x_v\}_{v \in V}$, the maximum number of cars (and so the maximum number of licences) allowed at any moment in city $v$ is $\bar{w}_{vv}(x_v) + \sum_{u \in \delta_G^-(v)} \bar{w}_{uv}(x_u)$ (the local constraint). Let $\vec{a^i} = (a_1^i, a_2^i, \cdots, a_t^i)$ denote the number of licences allocated to vertices in $X_i$, i.e., $a_s^i$ denotes the number of licences allocated to vertex $v_s^i$, $s \in [t]$. Similarly $\vec{\ell^i}$ denotes the locally maximum number of cars allowed at any moment in vertices of $X_i$. Let $G_i$ denote the subgraph generated by all the vertices belonging to the subtree (node $X_i$) rooted at $X_i$. We use $Q^i$ to denote the total number of licences allocated to $G_i$. Let $\Omega_i(\vec{a^i}, \vec{\ell^i}, Q^i)$ denote the optimal objective value of $\bar{PG}_2$ restricted to the subgraph $G_i$, when the number of licences on $v_s^i$ and the number of allowed cars at any moment on i are respectively $a_s^i$ and $\ell_s^i$, $s \in [t]$, and the total number of licences on $G_i$ is exactly $Q^i$. If there is no feasible solution to $\Omega_i(\vec{a^i}, \vec{\ell^i}, Q^i)$, our DP approach

will automatically set $\Omega_i(\vec{a^i}, \vec{\ell^i}, Q^i)$ to $-\infty$. Let $\bar{w}_{uv}(x_v) \equiv 0$ if $(u, v)$ is not an edge in $G$. Note that the range of $a^i_s$ we need to compute is in $\Lambda_v$, $\ell^i_s$ ranges from 0 to $\bar{p}_{v^i_s}$, $s \in [t]$ and $Q^i$ from 0 to $p$. The DP approach is as follows:

- $X_i$ is a leaf node or a start node, where $t = 1$. $\Omega_i(a^i_1, \ell^i_1, Q^i) = \omega_{v^i_1} a^i_1$ if the triple $(a^i, \ell^i, Q^i)$ is feasible, which can be verified easily e.g. $Q^i = a^i_1$ and $\ell^i_1 = \bar{w}_{v^i_1 v^i_1}(a^i_1)$. Let $\Omega_i(a^i_1, \ell^i_1, Q^i) = -\infty$ if the triple $(a^i, \ell^i, Q^i)$ is not feasible.

- $X_i$ is a forget node and suppose its child is $X_j = X_i \cup \{v^j_{t+1}\}$.
  $\Omega_i(\vec{a^i}, \vec{\ell^i}, Q^i) = \max_{a^j_{t+1}, \ell^j_{t+1}} \Omega_j(\vec{a^i}, a^j_{t+1}, \vec{\ell^i}, \ell^j_{t+1}, Q^i)$

- $X_i$ is an introduce node and suppose its child is $X_j = X_i \setminus \{v^i_t\}$. Let $a^j_s = a^i_s$ and $\ell^j_s = \ell^i_s - \bar{w}_{v^i_t v^i_s}(a^i_t)$, $\forall s \in [t-1]$. $\Omega_i(\vec{a^i}, \vec{\ell^i}, Q^i) = \Omega_j(\vec{a^j}, \vec{\ell^j}, Q^i - a^i_t) + \omega_{v^i_t} a^i_t$ if $\sum_{s \in [t]} \bar{w}_{v^i_s v^i_t}(a^i_s) = \ell^i_t$, and $\Omega_i(\vec{a^i}, \vec{\ell^i}, Q^i) = -\infty$ otherwise.

- $X_i$ is a joint node and suppose its two children $j$ and $k$ are such that $X_j = X_k = X_i$. $\Omega_i(\vec{a^i}, \vec{\ell^i}, Q^i) = \max_A \{\Omega_j(\vec{a^j}, \vec{\ell^j}, Q^j) + \Omega_k(\vec{a^k}, \vec{\ell^k}, Q^k)\}$, where the condition $A = \{(\vec{a^j}, \vec{\ell^j}, Q^j), (\vec{a^k}, \vec{\ell^k}, Q^k) \mid \vec{a^j} + \vec{a^k} = \vec{a^i}, \vec{\ell^j} + \vec{\ell^k} = \vec{\ell^i}, Q^j + Q^k = Q^i\}$.

- $X_i$ is the root of $T$, $OPT(Q^i) = \max_{\vec{a^i}, \vec{\ell^i}} \{\Omega_i(\vec{a^i}, \vec{\ell^i}, Q^i)\}$ is the optimal value (social welfare) of $\bar{PG}_2$ when the total scaled number of licences is exactly $Q^i$, i.e., the global constraint satisfies $\sum_{v \in V} x_v = Q^i$.

**Analysis of running time of DP.** It is not difficult to see that the above DP approach gives the correct solution of $\bar{PG}_2$ on $k$-outerplanar graphs. For each node $X_i$, we need to keep $O(pq^{3k\lceil \frac{2\rho}{\delta} \rceil^{3k}}) = O(|V| q^{3k+1\lceil \frac{2\rho}{\delta} \rceil^{3k}})$ number of $\Omega_i$ values. Each $\Omega_i$ can be computed in $O(|V| q^{3k+1\lceil \frac{2\rho}{\delta} \rceil^{3k}})$ time (this is the worst case running time when $X_i$ is a joint node). There are $O(k|V|)$ nodes in $T$. Therefore, the total running time of the DP approach (by multiplying the above three numbers) is $O(k|V|^3 q^{6k+2\lceil \frac{2\rho}{\delta} \rceil^{6k}})$.

Based on the above DP approach, we can solve $\bar{PG}_2$ on any $k$-outerplanar graph optimally for any fixed $k$ (which includes any directed tree whose treewidth is 2). Therefore, for any $\delta > 0$ and fixed $k$, we can use VCG to get an optimal deterministic truthful mechanism for PG on any directed $k$-outerplanar graph that violates each local constraint by a factor of $\delta$ and runs in $O(k|V|^3 q^{6k+2\lceil \frac{2\rho}{\delta} \rceil^{6k}})$ time (note that Theorem 15 also works for bounded treewidth graphs).

**Theorem 15.** *For any $\delta > 0$ and fixed $k$, there is an optimal deterministic truthful mechanism for PG on any directed $k$-outerplanar graph $G = (V, E)$ that violates each local constraint by a factor of $1 + \delta$ and runs in $O(k|V|^3 q^{6k+2\lceil \frac{2\rho}{\delta} \rceil^{6k}})$ time, where $\rho = |V|(\lfloor \log_2(q) \rfloor + 2)$.*

**Step 3: PTAS for planar graphs** Observe that when there are some boundary conditions on $k$-outerplanar, the above DP approach still works. For example, if the

number of licences of any vertex in any first and last face (level 1 and level $k$ face) of the $k$-outerplanar graph is zero, we just modify the dynamic programming approach in a bottom-up manner to set $\Omega_i = -\infty$ if any vertex $v$ in any first and last face is a parameter of $\Omega_i$ and its number of licences $a_v^i > 0$. Then the modified DP approach is the desired algorithm for $\bar{PG}_2$ on the $k$-outerplanar graph under this boundary condition.

**Proposition 5.** *PG is strongly NP-hard on planar graphs when we allow a $\delta$ violation of local constraints.*

*Proof.* Suppose we restrict PG instances to require that

$$\sum_{u \in \delta_G^-(v)} w_{uv} \leq p_v, \forall v \in V \tag{3.17}$$

Then the maximum independent set problem can be solved as such PG problem with each $p_v = 1$. Further, observe that if $\delta = \min_{u,v}\{w_{uv}\}$, then even if we allow for $(1+\delta')$-violation of the local constraints, where $0 < \delta' < \delta$, the maximum independent set problem can still be solved as such PG problem. Maximum independent set on a planar graph with degree at most 3 is strongly NP-hard [56]. Theorem 17 provides a PTAS for PG with $q = poly(|V|)$ and $(1+\delta')$-violation, giving a tight approximation in this sense. □

**Theorem 16.** *For any fixed $k$ and $\delta > 0$, there is an $O(k^2|V|^3 q^{6k+2\lceil \frac{2\rho}{\delta} \rceil 6k})$ algorithm for PG with integer variables on directed planar graph $G = (V, E)$ that achieves $\rho^{in}$-approximation and violates each local constraint by a factor of $1+\delta$, where $\rho = |V|(\lfloor \log_2(q) \rfloor + 2)$ and $\rho^{in} = \frac{k}{k-2}$).*

*Proof.* We use $OPT(\bar{PG}_2)$ to denote $OPT_G^{in}(\bar{PG}_2)$ omitting the superscript and subscript. By Lemma 8, we know $OPT = OPT(PG) \leq OPT(\bar{PG}_2)$. Let $\bar{PG}_2(i)$ denote the $\bar{PG}_2$ restricted on $G$ when setting $x_v = 0$ for each $v$ that belongs to any face $f \equiv i$ or $i+1$ $(mod\, k)$. Let $\{x_v^*\}_{v \in V}$ be an optimal solution for $\bar{PG}_2$. Then we know

$$\sum_{i \in [k]} \sum_{v \in f : f \equiv i \text{ or } i+1 (mod\, k)} x_v^* = 2OPT(\bar{PG}_2)$$

As a consequence, there exists $i \in [k]$ such that

$$\sum_{v \in f : f \equiv i \text{ or } i+1 (mod\, k)} x_v^* \leq \frac{2OPT(\bar{PG}_2)}{k}$$

Observe that $\{x_v\}_{v \in V}$ is a feasible solution for $\bar{PG}_2(i)$, where $x_v = 0$ if $v$ belongs to any face $f \equiv i$ or $i+1$ $(mod\, k)$ and $x_v = x_v^*$ otherwise. Thus,

$$OPT(\bar{PG}_2(i)) \geq (1 - \frac{2}{k})OPT(\bar{PG}_2) \geq (1 - \frac{2}{k})OPT$$

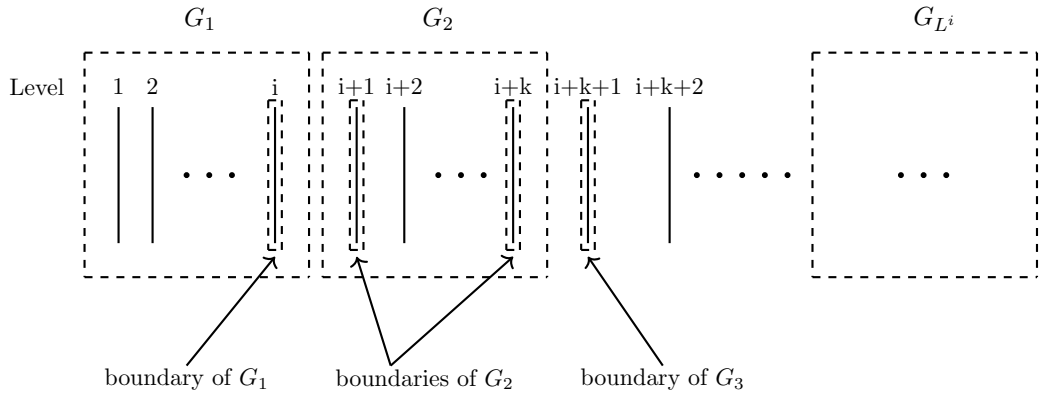



Figure 3.8: An illustration of how to select k-outerplanar graphs

Solving each $\bar{PG}_2(i)$, $i \in [k]$, then choosing $\max_{i\in[k]}\{OPT(\bar{PG}_2(i))\}$ (which is at least $(1-\frac{2}{k})OPT$) gives the desired result. Now let us see how to solve $\bar{PG}_2(i)$. Note that for $\bar{PG}_2(i)$, $x_v = 0$ for any $v$ who belongs to any face $f \equiv i$ or $i+1$ $(mod\, k)$. $\bar{PG}_2(i)$ consists of independent $k'-$outerplanar graphs, each of which has some boundary condition i.e. the emission amount of any vertex in any first and last face is zero and $k' \leq k$. Suppose the number of these independent $k'$-outerplanar graphs is $L^i$. Without loss of generality, suppose these $k'$-outerplanar graphs are ordered from exterior to interior as $G_s = (V_s, E_s)$, $s \in [L^i]$ (e.g. $G_s$ is the subgraph of $G$ constructed by all the vertices of levels from $(s-2)k+i+1$ to $(s-1)k+i$, $s = 2, \cdots, L^i - 1$, with boundary $x_v = 0$ if $v$ is of level $(s-2)k+i+1$ or $(s-1)k+i$, see Figure 3.5.3).

Let $\Omega_s(Q^s)$ denote the optimal value if there is a solution such that the total allocated scaled emission amount to $G_s$ is exactly $Q^s$ with boundary condition and $\Omega_s(Q^s) = 0$ otherwise, which can be solved by the above DP approach on $k'$-outerplanar graphs with boundary conditions. Then, it is not difficult to see the optimal solution for $\bar{PG}_2(i)$ is the optimal solution of the following integer linear program (denoted $SUB$):

$$
\begin{aligned}
\max \quad & \sum_{s\in[L^i]} \sum_{Q^s=0}^{p} \Omega_s(Q^s) y_{sQ^s} && (SUB)\\
\text{s.t.} \quad & \sum_{s\in[L^i]} \sum_{Q^s=0}^{p} Q^s y_{sQ^s} \leq p \\
& \sum_{Q^s=0}^{p} y_{sQ^s} = 1 \\
& y_{sQ^s} \in \{0,1\},\ \forall s \in [L^i],\ Q^s = 0, 1, \ldots, p.
\end{aligned}
$$

Let $g_t(Q)$ denote the optimal integer value of $SUB$ when only $G_s$, $s \in [t]$ is considered and the total emission amount allocated to these graphs is exactly $Q$. Then we

have the following recursion function (which is essentially the same as that in Lemma 1):

$$g_t(Q) = \max_{Q^t=0,1,\cdots,Q} \{g_{t-1}(Q-Q^t) + \Omega_t(Q^t)\}$$

The optimal value of $SUB$ is $\max_{Q=0,1,\cdots,p}\{g_{L^i}(Q)\}$, which gives the optimal solution of $\bar{PG}_2(i)$ by tracking the optimal value of this dynamic programming approach. The running time of this approach is $O(|L^i|p^2)$. Hence, the total running time for obtaining and solving $\bar{PG}_2(i)$ is

$$O(|L^i|p^2) + \sum_{s\in[L^i]} O(k|V_s|^3 q^{6k+2} \lceil \frac{2\rho}{\delta} \rceil^{6k}) = O(k|V|^3 q^{6k+2} \lceil \frac{2\rho}{\delta} \rceil^{6k})$$

We need to solve $\bar{PG}_2(i)$, for each $i \in [k]$ and then get $\max_{i\in[k]}\{OPT(\bar{PG}_2(i))\}$. Therefore, the overall running time is $O(k^2|V|^3 q^{6k+2} \lceil \frac{2\rho}{\delta} \rceil^{6k})$ as desired. □

Let $\frac{2}{k} = \epsilon$ in Theorem 16. Also note that $\rho = |V|(\lfloor \log_2(q) \rfloor + 2)$. We have:

**Theorem 17.** *For fixed $\delta, \epsilon > 0$ there is an*

$$O\left(\frac{1}{\epsilon^2}|V|^{12/\epsilon+3} q^2 \lceil \frac{2(\lfloor \log_2 q \rfloor + 2)q}{\delta} \rceil^{12/\epsilon+1}\right) = \left(\frac{|V|q(\log_2 q + 2)}{\delta}\right)^{O(\frac{1}{\epsilon})} \tag{3.18}$$

*time algorithm for PG on directed planar graph $G = (V, E)$ that achieves social welfare $(1-\epsilon)OPT^{in}(PG)$ and violates each local constraint by a factor of $1+\delta$. This is a PTAS for PG with polynomial size integer variables.*

## 3.6 General objective function for bounded degree graphs

### 3.6.1 Approximation algorithms

If $R(x)$ (recall from (3.4)) is monotone, we present an algorithm with an approximation ratio of $O(\Delta)$ for PG on a graph with maximum degree at most $\Delta$.

**Theorem 18.** *If $R(x)$ with binary variables is monotone increasing, then there is an $(\rho^{fr} = \frac{e\gamma_{\Delta+2}}{e-1} + 1)$-approximation algorithm for PG with integer variables on graph with degree at most $\Delta$.*

*Proof.* If $x_v \in \{0,1\}$, $\forall v \in V$, for any $A \subseteq V$, we define $g(A) = R(x)$ where $x_v = 1$, $\forall v \in A$ and $x_v = 0$, for any $v \notin A$. Observe that $R$ with binary variables is submodular if and only if $g$ satisfies $g(A \cup B) + g(A \cap B) \leq g(A) + g(B)$, for any $A, B \subset V$. For any $A \subseteq V$, and $v \in V$, denote by $A + v$ the set $A \cup \{v\}$. Let $g_v(A) = g(A+v) - g(A)$. Then it is not difficult to see that $g(A \cup B) + g(A \cap B) \leq g(A) + g(B)$, for any $A, B \subset V$ if and only if for any $A \subseteq B \subseteq V$ and $v \in V \setminus B$, $g_v(A) \geq g_v(B)$. Next we will prove that $g_v(A) \geq g_v(B)$, which implies that $R(x)$ is submodular.

Let $A \subseteq B \subseteq V$ and $v \in V \setminus B$. Denote by $dr_u^{A+v}$ the total welfare change of player $u$ by adding $v$ to set $A$. Observe that

$$g_v(A) = dr_v^{A+v} + \sum_{u \in \delta_G^-(v) \cap A} dr_u^{A+v}.$$

By simple calculations,

$$dr_v^{A+v} = b_v(1) - b_v(0) - d_v(1 + \sum_{u \in \delta_G^-(v) \cap A} w_{uv} x_u) + d_v(\sum_{u \in \delta_G^-(v) \cap A} w_{uv} x_u),$$

$$dr_u^{A+v} = -d_u(\sum_{u' \in \delta_G^-(u) \cap A} (w_{u'u} x_{u'} + w_{vu})) + d_u(\sum_{u' \in \delta_G^-(u) \cap A} w_{u'u} x_{u'}).$$

By convexity of $d_u$, we know that

$$\begin{aligned} dr_v^{A+v} &\geq \Delta r_v^{B+v}, \forall u \in \delta_G(v) \cap A \\ dr_u^{A+v} &\geq \Delta r_u^{B+v}, \forall u \in \delta_G(v) \cap A \\ \Delta r_u^{B+v} &\geq 0, \forall u \in \delta_G^-(v) \cap B \end{aligned} \tag{3.19}$$

Hence, $g_v(A) \geq g_v(B)$ and $R(x)$ with binary variables is submodular.

For the graph with degree $\Delta$, note that PG is $\Delta+2$ column sparse. By Proposition 2, there is a randomized $\rho^{fr} = \frac{e\gamma_{\Delta+2}}{e-1}$-approximate algorithm for PG with binary variables if $R(x)$ is monotone increasing (which can be derandomized to be deterministic with the same approximation ratio). Now observe that for a concave function $G(x)$ from $\mathbb{R}_{\geq 0}$ to $\mathbb{R}_{\geq 0}$, we have $G(x+y) \leq G(x)+G(y)$, for any $x$, $y \in \mathbb{R}_{\geq 0}$ (without loss of generality let $x \geq y > 0$, by concavity of $G$, it holds that $\frac{G(x+y)-G(x)}{x+y-x} \leq G'(x) \leq G'(y) \leq \frac{G(y)-G(0)}{y-0}$). By this property, since $b_v(x)$ and $-d_v(x)$ are concave from $\mathbb{R}_{\geq 0}$ to $\mathbb{R}_{\geq 0}$, for any $v \in V$, for any feasible solution $x = \{x_v\}_v$ and $y = \{y_v\}_v$, we have $R(x+y) \leq R(x) + R(y)$. By ellipsoid algorithm for convex programming problem in [64], we can get an optimal fractional solution of PG denoted as $x^* = \{x_v^*\}_v$. Let $z^* = \{z_v^*\}_v$ where $z_v^* = \lfloor x_v^* \rfloor$, for any $v \in V$ and $x^* = z^* + y^*$. Let $y'$ be an $\frac{e\gamma_{\Delta+2}}{e-1}$-approximate solution for PG with binary variables when $R(x)$ is monotone increasing. Note that $y^*$ is a feasible fractional solution for PG with binary variables. We know $R(y^*) \leq \frac{e\gamma_{\Delta+2}R(y')}{e-1}$. Let $x'$ be an solution of PG with integer variables such that $x' = y'$ if $R(y') \geq R(z^*)$ and $x' = z^*$ otherwise. Therefore, we have $R(x^*) \leq R(z^*) + R(y^*) \leq R(z^*) + \frac{e\gamma_{\Delta+2}R(y')}{e-1} \leq (\frac{e\gamma_{\Delta+2}}{e-1} + 1)R(x')$. □

### 3.6.2 Truthful in expectation mechanisms

In this section, we will prove that there is an $O(\gamma_{\Delta+2})$ (recall the basic definitions in the beginning of the chapter for $\gamma$) truthful in expectation mechanism for PG on any graph with degree $\Delta$ when $b_v$ is linear and $d_v$ is piece-wise linear with one shift point, under a further natural assumption of a relaxation of the condition for the slopes of

functions $c_v(x_v)$ and $d_v(x_v)$ (see inequalities (3.20) and (3.21) later in this section). For each player $v \in V$, let $b_v(x_v) = s_v^0 x_v$, a linear function starting from the origin with slope $s_v^0$. Let also $d_v$ be a piece-wise linear convex function with one shift point where the first part is a linear function starting from the origin with slope $s_v^1$, the shift point is $(y_v, s_v^1 y_v)$ and the second part is a linear function starting from the shift point with slope $s_v^2 \geq s_v^1$ (see Figure 3.9).

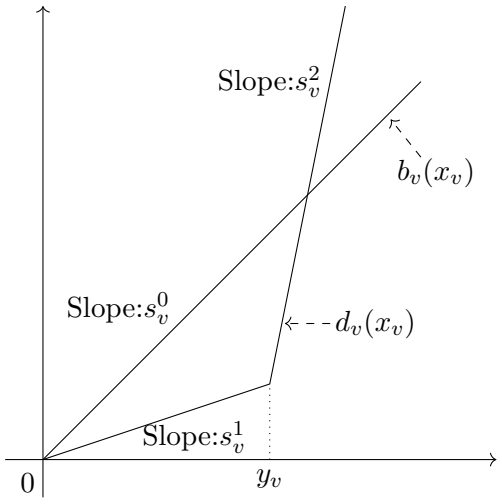



Figure 3.9: An illustration of function $b_v$ and $d_v$.

As we know if the emitted pollution from player $v$ is large enough, the welfare of player $v$ should be negative. The damage function is piece-wise linear with one shift point, which means player $v$'s welfare (valuation minus damage) will decrease after the total pollution in $v$ reaches $y_v$. Precisely, when $x_v + \sum_{u \in \delta_G^-(v)} w_{uv} x_u \geq y_v$, we should have $s_v^0 \leq s_v^2$. However, we can relax this condition to

$$s_v^0 - s_v^2 - \sum_{u \in \delta_G^+(v)} s_u^1 w_{vu} \leq 0. \tag{3.20}$$

The second condition on the slope $s_v^0$ is somehow more subtle. Intuitively, player $v$'s emitted pollution should not affect his neighbour's total pollution too much. This means that if his neighbour $u$'s pollution reaches $y_u$, then the total social welfare $R(x)$ should decrease. That is, for any $v \in V$, and any $u \in \delta_G^+(v)$, if $\sum_{u' \in \delta_G^-(u)} w_{u'u} x_{u'} \geq y_u$, then

$$s_v^0 - s_v^1 - \sum_{u' \in \delta_G^+(v) \setminus \{u\}} s_{u'}^1 w_{vu'} \leq s_u^2 w_{vu}. \tag{3.21}$$

**Lemma 10.** *Let $x^*$ be an optimal fractional solution of PG under the condition that functions $b_v$ and $d_v$ satisfy constraints (3.20) and (3.21), then $x^*$ has the following property: for each $v \in V$, the local level of pollution in $v$ satisfies that $x_v^* + \sum_{u \in \delta_G^-(v)} w_{uv} x_u^* \leq y_v$.*

*Proof.* We prove this lemma by contradiction. Suppose there exists $v \in V$, such that $x_v^* + \sum_{u \in \delta_G^-(v)} w_{uv} x_u^* > y_v$. If $x_v^* > 0$, by constraints (3.20), we can decrease the value of $x_v^*$ by an amount of $\alpha$ such that $x_v^* - \alpha + \sum_{u \in \delta_G^-(v)} w_{uv} x_u^* > y_v$. By simple calculation, the total social welfare increases by an amount of $-(s_v^0 - s_v^2 - \sum_{u \in \delta_G^+(v)} s_u^1 w_{vu})\alpha \geq 0$. Thus, we can do this until either $x_v^* = 0$ or $x_v^* - \alpha + \sum_{u \in \delta_G^-(v)} w_{uv} x_u^* \leq y_v$. If the first case holds and the second case does not hold, then there exists $u \in \delta_G^-(v)$ with $x_u^* > 0$. Note that $v \in \delta_G^+(u)$. Since $\sum_{u \in \delta_G^-(v)} w_{uv} x_u^* > y_v$, if we decrease the value $x_u^*$ by $\alpha$, by simple calculation, the total social welfare increases by at least $-(s_u^0 - s_u^i - \sum_{u' \in \delta_G^+(u) \setminus \{v\}} s_{u'}^i w_{uu'} - s_v^2 w_{uv})\alpha \geq -(s_u^0 - s_u^1 - \sum_{u' \in \delta_G^+(u) \setminus \{v\}} s_{u'}^1 w_{uu'} - s_v^2 w_{uv})\alpha$, which is non-negative by constraints (3.21). Here, the value $i$ is defined as follows, if total pollution in $v$ is below $y_u$ then $s_u^i = s_u^1$, otherwise $s_u^i = s_u^2$, with the same argument for $s_{u'}^i$. By this operation, we can decrease the value of $v$ without loss of total social welfare until the total level of pollution in $v$ does not reach $y_v$. □

**Lemma 11.** *If PG functions $b_v$ and $d_v$ satisfy constraints (3.20) and (3.21) then there is a deterministic polynomial time algorithm with approximation ratio $\rho^{fr} = \gamma_{\Delta+2}$.*

*Proof.* By Lemma 10, we know the optimal fractional solution $x^*$ can satisfy that $x_v^* + \sum_{u \in \delta_G^-(v)} w_{uv} x_u^* \leq y_v$, for any $v \in V$. Hence, we can modify the constraint (3.6) in PG to

$$x_v + \sum_{u \in \delta_G^-(v)} w_{uv} x_u \leq \min\{y_v, p_v\}. \tag{3.22}$$

This modified PG has the same optimal fractional solution as PG. In the modified PG, $R(x) = \sum_{v \in V} \omega_v x_v$, where $\omega_v = s_v^0 - s_v^1 - \sum_{u \in \delta_G^-(v)} s_u^1 w_{vu}$. By Proposition 1, there is a deterministic polynomial time algorithm for the modified PG with approximation ratio $\rho^{fr} = \gamma_{\Delta+2}$. This algorithm is also an algorithm for PG with the same approximation ratio. □

With Lemma 11, there is a truthful in expectation mechanism for PG with approximation ratio $\gamma_{\Delta+2} = (e + o(1))(\Delta + 2)$:

**Theorem 19.** *Suppose the bidding strategy $s_v^0$ of each player $v \in V$ satisfies constraints (3.20) and (3.21). There is a randomized, individually rational, $(\rho^{fr} = \gamma_{\Delta+2})$-approximation mechanism that is truthful in expectation for PG on $G$ with degree at most $\Delta$.*

*Proof.* Since the bidding strategy $s_v^0$ of each player $v$ satisfies constraints (3.20) and (3.21), by Proposition 3 and Lemma 11, there is a randomized, individually rational, $\gamma_{\Delta+2}$-approximation mechanism that is truthful in expectation for the modified PG, which is also a truthful in expectation mechanism for PG with the same approximation ratio. □

**Corollary 2.** *If $b_v$ and $d_v$ are linear functions for any $v$ and the bidding strategy $s_v^0$ of player $v$ is arbitrary, then there is a randomized, individually rational, $(\rho^{fr} = \gamma_{\Delta+2})$-approximation mechanism that is truthful in expectation for PG.*

*Proof.* Since $d_v$ is linear, it is equivalent to the above piece-wise linear function with $s_v^2 = +\infty$ and $y_v = +\infty$, for any $v \in V$. By Theorem 19, there is a randomized, individually rational, $\gamma_{\Delta+2}$-approximation mechanism that is truthful in expectation for PG. □

**Remark:** We cannot anticipate an algorithm with constant approximation ratio for PG on the graph with average degree $\Delta$ (the average degree of a graph $G$ is $\frac{\sum_{v \in V} |\delta_{G^{un}}(v)|}{|V|}$) even if $\Delta = 1$. Consider a graph $G'$ consisting of a complete graph with $n$ vertices and $n^2 - n$ isolated vertices with valuation 0. Note that $G'$'s average degree is $\frac{n^2}{n^2} = 1$. PG on $G$ cannot be approximated within $n^{1-\epsilon}$ unless Unique Game conjecture fails. Thus, PG cannot be approximated within $(n^2)^{\frac{1-\epsilon}{2}}$ on $G'$, where $n^2$ is the number of vertices of $G'$.

## 3.7 Literature overview

An invaluable source of pollution control regulations comes from the managerial approaches in environment policies. The majority of literature in this field deals with symmetric information. This problem however shows a fundamental asymmetry between the regulatory bodies and pollutants. The research contributions considering environmental policy with asymmetric information and the diffusion nature of pollution have been limited until recently.

In order to control pollution, an incentive mechanism that is environmentally friendly and resource efficient needs to be designed and deployed by regulatory authorities. However, it is not obvious how to design such a mechanism in the presence of asymmetric information; just as Hurwicz [69] put it: the firms know that information will be used by the regulator to design a policy which will affect their profits. Hence, they have an incentive to manipulate reported information in order to influence the content of the policy. In this context, Farell [48] discusses the relevance of the Coase Theorem. This theorem basically asserts that bargaining will lead to an efficient outcome regardless of the initial allocation of property if negotiation and trade in presence of externality are possible and the transaction costs are sufficiently low. Considering the problems of incomplete information, that paper shows that voluntary negotiation does not lead to the first-best outcome that maximizes joint surplus in the presence of two-sided private information. That is to say, centralised economic institutions such as government control and intervention, and decentralised institutions such as bargaining and ownership rights, should be viewed as complementary to each other. Therefore, a necessary

condition for the government when designing an optimal pollution control plan is the truthful information about firms.

Kwerel [92], Dasgupta et al. [37] and Spulber [129] have proposed mechanisms that implement truth telling by firms to maintain a mild level of pollution. Under this assumption the firms can communicate with the regulator but not with each other. In Kwerel's scheme [92] firms are informed in advance that their messages will be translated into pollution taxes. The regulator issues a fixed number of transferable pollution licences and offers a subsidy for those licences which firms hold in excess of emission. Both the number of licences to be issued and the subsidy rate offered are calculated on the basis of the cost information provided by firms.

Kim and Chang [83] constructed an optimal incentive tax/subsidy scheme in an oligopoly market with pollution and suggested a differential damages mechanism, which leads to an optimal emission level. McKitrick [103] proposes a Cournot Mechanism for pollution control under asymmetric information, in which a Nash Equilibrium exists, is stable and can be reached by iterative computations. Because firms may attempt to manipulate the pollution level allocation to their own advantage, the adjustment rule is exogenous and depends on the actions of the firms. The approach by Karp and Livernois [75] is related to that in Conrad and Wang [34]. The authors examined the steady-state properties of a tax adjustment mechanism in situations where the government has no information about firms' abatement costs.

These prior studies provide an overall framework in the administrative approach to control pollution. However, those models are only a first level of approximation in characterizing the reality. Although, there is some literature studying an economics environment consisting of firms or countries with geographical distinction, few of them take the diffusion nature of air and water pollution into consideration. For instance, Petrosjan and Zaccour [116] study the problem of allocation over time of total cost incurred by countries in a cooperative game of pollution reduction. Segerson [126] develops a general incentive scheme for controlling nonpoint source pollution[5] that considers the diffusion nature, in which rewards for environmental quality above a given standard are combined with penalties for substandard quality. Based on the work of Petrosjan and Zaccour [116], Belitskaya [18] develops an $n$-person network game model of emission reduction. Dorner et al. [45] create a multi-objective modeling system using Bayesian probability networks to study nonpoint source pollution. Both the work of Belitskaya [18] and Dorner et al. [45] are different from the setting of ours, in either model assumption or function settings. In addition to these works built on network framework, Dong et al. [44] models the water pollution problem as a cost

[5]Nonpoint source (NPS) pollution refers to both water and air pollution from diffuse sources, that is sources without a specified fixed location. For instance, nonpoint source water pollution affects a water body from sources such as polluted runoff from agricultural areas draining into a river, or wind-borne debris blowing out to sea. This work deals mainly with point source pollution.

sharing problem on a tree network. However, none of the literature mentioned above takes into account the role of governments in pollution abatement, more specifically how to make policies assuming information asymmetry. A model that adequately takes both factors into account is what we need to tackle such problems in reality.

Few other papers have studied air pollution in relation to network models. Singh and Datta [128] use artificial neural network method to identify unknown pollution sources in the groundwater. Gianessi et al. [58] analyze the national water pollution control policies. And, finally, Trujillo and Hugh [135] study multi-objective air pollution monitoring network design. These papers use networks in a very different context from ours.

Turning into current practice, emission trading is a market-based approach used to control pollution by providing economic incentives for achieving reductions in the emissions of pollutants. Various countries have adopted emission trading systems as one of the strategies for mitigating climate-change by addressing international greenhouse-gas emission [132].

A central authority (usually a governmental body) sets a limit or cap on the amount of a pollutant that may be emitted. The limit or cap is allocated and/or sold by the central authority to firms in the form of emission permits which represent the right to emit or discharge a specific volume of the specified pollutant [131]. Permits (and possibly also derivatives of permits) can then be traded on secondary markets. For example, the European Union Emissions Trading Scheme (EU ETS) trades primarily in European Union Allowances (EUAs), the Californian scheme in California Carbon Allowances, the New Zealand scheme in New Zealand Units and the Australian scheme in Australian Units [133]. Firms are required to hold a number of permits (or allowances or carbon credits) equivalent to their emissions. The total number of permits cannot exceed the cap, limiting total emissions to that level. Firms that need to increase their volume of emissions must buy permits from those who require fewer permits [131, 132]. Currently a simple auction mechanism for selling EUAs is adopted in Europe, see, e.g., [4]. Furthermore in order to limit the automobile pollution, governments use policies of car taxation [71], [55]. A radical transport policy introduced in the UK and first applied in Central London resulting in 19% reduction of $CO_2$ emissions (see table 2 in [16]).

## 3.8 Open problems

We presented a new network model for the pollution control problem and studied planar and tree networks which model realistic scenarios. These networks can be applied to model air and water pollution from diffuse sources. Our main technical results include a constant approximation algorithm and a PTAS with a small violation in the constraints for the case of planar graphs and an FPTAS which is truthful in expectation and a 3-

approximate deterministic truthful mechanism for the case of trees. We obtained these results by introducing novel algorithmic techniques for planar and tree graphs which could be of independent interest.

Many interesting open problems arise from this new model. Our main open question is to determine whether PG with binary variables on planar graphs admits a PTAS or whether it is APX-hard. Another direction would be to study lower bounds on truthful (deterministic, universal, truthful in expectation) mechanisms for PG. Can externality be used to obtain such lower bounds? Furthermore it would be interesting to generalize our results to other graphs, e.g., Euclidean graphs.

# Chapter 4

# Coverage problems in wireless sensor networks

## 4.1 The problems

Modern technological advances in micro-electronic and mechanical systems, digital electronics, and wireless communications have boosted the low-cost development of multifunctional sensing devices, called sensor nodes, which, despite their relatively small dimensions, have exceptionally superior sensing, processing and communication capabilities. A set of spatially distributed sensor nodes which are wirelessly interconnected constitutes a so called Wireless Sensor Network (WSN) [7, 145].

WSNs thanks to their capability to monitor phenomena taking place in almost every type of environment, are considered to be among the most emerging scientific domains in $21^{st}$ century and have an ever growing variety of applications [3]. However, the utilization of WSNs is limited because of the strict energy limitations of the wireless nodes. Specifically, a sensor node within a WSN dissipates amounts of energy mainly during communication but also during sensing and processing. On the other hand, typical sensor nodes are powered by simple batteries. Thus in WSNs comprising of hundreds or even thousands of randomly deployed nodes, it is impractical to either recharge or replace the node batteries that have run out of energy. As nodes get depleted, the continuous monitoring of the whole network is deteriorated while the energy cost of communication is increased because less multi-hop routing paths remain available. In this way the *network lifetime* is rapidly reduced. That is why, great amount of research in WSNs aims at the maximization of network lifetime [108, 149].

Several definitions of network lifetime use a *coverage* variant i.e. the continuous monitoring of whole areas or discrete targets of interest via the sensing nodes of WSNs. Each sensor node monitors phenomena taking place within its sensing area, which is typically considered as a disk with the sensor being placed at its center. Due to this ability, WSNs have a continuously increasing range of applications [13].

The most common definition for the coverage variant uses 1-coverage to define the

network lifetime as the time that the region of interest is covered by at least one node. However, according to Friedrich and Dressler [39], coverage can be defined in different ways, depending on both the configuration of the region of interest and the accomplished redundancy of the coverage. Specifically, according to the region configuration there are three possible types of coverage. The first one is the so called area or volume coverage, where each point inside a two-dimensional area or a three-dimensional volume must be covered. The second alternative is the so called target coverage case where only a finite set of target points inside an area of interest has to be covered. Barrier coverage is the third type of coverage case, where a mobile target can pass undetected through a barrier of sensor nodes.

Similarly, there are two approaches to describe the degree of coverage redundancy that can be accomplished by a given WSN. The first approach is termed $\alpha$-coverage. It requires that only a given percentage $\alpha$ of the region of interest is covered by at least one sensor node. The second approach is named k-coverage. It requires that each point within the region of interest is covered by at least k sensors.

In the case where sensor nodes are owned by selfish agents who may sell the service of routing data [11] or for the coverage of an area/set of targets we want to derive mechanisms that assign right payments to the agents who own the sensors. In this sense the mechanism encourages them to declare the truth about their battery cost per unit time (truthful mechanisms). At the same time we also want to maximize the lifetime of the network. In particular we consider the following optimization problems:

- Budgeted Maximum Lifetime Coverage (BMLC) in which we want to find a proper schedule of active/sleep sensor nodes such that the total time that the areas of interest are monitored is maximized while a monetary budget is not exceeded.

- Weighted Sensor Cover (WSC) in which we aim at finding a set of weighted sensor nodes that monitor all the target points having the minimum total weight.

We study two approaches (each suitable for a different scenario) for two problems of target coverage. In the former, we propose a combinatorial model in which the targets are covered by the sensor nodes with some probability given on input of the problem. In the latter we consider a geometric model in which the sensor nodes, the targets and the base station are deployed in the plane.

### 4.1.1 Our results

Inspired by applied scenarios in which full sensing is not possible as in the hidden terminal problem [134], we introduce a random instance model for the budgeted maximum lifetime coverage (BMLC) problem and in this model we design a novel truthful approximate mechanism for this problem. Our mechanism is randomized, polynomial time and truthful with high probability with respect to both random instances and

internal randomness of the algorithm. Technically, our approach is based on an interesting reduction from the budgeted maximum lifetime coverage problem to the classic knapsack problem. Then, we utilize the combinatorial structure of the coverage problem to prove that with high probability, i.e., on almost all instances of the problem, this reduction results in a monotone algorithm, that leads to a truthful mechanism under some technical assumptions. Monotonicity, which is an algorithmic property that is sufficient and often also necessary for truthfulness for one parameter agents [14], is usually difficult to achieve and there are only very few approaches known in literature towards this goal. Our approach is an example of a new such technique and can also be of independent interest.

Second, we study the closely related minimum sensor coverage problem in wireless sensor networks on unit disk graphs, generalize and extend the recent PTAS [97] for this problem to obtain a monotone algorithm and therefore a truthful PTAS for the problem where the costs of sensor nodes are the private data of the agents. However, since the algorithm chooses the minimum cost solution out of many possible ones, hence a monotone algorithm for every subproblem does not necessarily lead to a monotone algorithm for the whole problem, as shown in [107]. Thus we modify the algorithm and consider the stronger notion of bitonicity. We first show that the combination of the bitonic algorithms for each subproblem is monotone and therefore leads to a truthful mechanism. To the best of our knowledge this is the first attempt to study strategic issues for target coverage problems.

We finally show that the well known reduction based on the primal-dual Garg-Koenemann framework, first used by Berman et al. [19], leads to the same approximation guarantee as for the classic maximum lifetime coverage problem, even if we have many additional budget constraints. This implies a polynomial time approximation scheme (PTAS) for the budgeted maximum lifetime coverage problem in unit disks graphs (i.e., sensors are unit disks on the plane), and a $O(\log(m))$-approximation for the problem in general graphs, where $m$ is the number of targets.

### 4.1.2 Motivation and related work

The simultaneous maximization of network lifetime as well as the network coverage are considered to be crucially important issues in WSNs and various research approaches have been proposed to deal with them [9]. For instance, some of them use the so called Maximum Lifetime Coverage Problem (MLCP) approach, proposing sleep/activate schedules for sensor nodes in order to maximize the lifetime of target coverage in the field [41]. Actually, Cardei et al. [27] first proposed the appropriate scheduling of the operation of the network nodes in order to achieve the extension of coverage lifetime in a network. Specifically, they defined two types of operational modes of nodes and developed a linear programming algorithm and a greedy algorithm to ar-

range the appropriate alternation of operational modes for each sensor node in order to both conserve energy and retain the service quality of the coverage. Similarly, Zhu and Sivakumar [150] proposed a communication strategy called Communication through Silence (CtS) which primarily uses silence, along with a minimal amount of energy to deliver information between sensor nodes to achieve energy-efficient communication without significant degradation on overall throughput in WSNs.

The maximization of the coverage of an area under cost-efficiency and stability constraints has also been addressed by applying optimization techniques [82]. Berman et al. [19] reduced the area coverage problem to target coverage by adding $O(N^2)$ targets where $N$ is the number of sensor nodes and proposed an $O(\log N)$ approximation algorithm for MLCP. Some other research schemes use clustering methods in order to achieve efficient organization of sensor nodes into clusters [110].

Alternatively many research works pursue to extend the network lifetime through energy efficient routing protocols which ensure that data are relayed from source nodes to destination nodes by using multi-hop routing via the most energy-efficient paths possible [74].

However, energy efficiency may be indeed anticipated from the point of view of the overall network benefit, but not from the point of view of an individual and selfish node profit. Generally, a node is considered to be selfish if it does not agree to provide any of its available resources (battery reserves, CPU cycles, or network bandwidth) to forward data which are not of its direct interest, even though it expects other nodes to relay data on its behalf.

In order to stimulate cooperation among nodes in a network various protocols have been proposed. These protocols may be classified into two main categories. The first of them includes the so called reputation-based protocols, while the other one is the market-based (or credit-based) approach.

In reputation-based approach, the behavior of every single node is dynamically measured by the rest of the nodes that belong to the same network. The incentive for a node is to keep relaying an estimated amount of traffic in order to maintain its reputation to an acceptable level [101]. The relay of traffic from a node with high reputation is facilitated by other nodes by virtue of its past behavior, and on the other hand, nodes with poor reputation are isolated from the network participation. These mechanisms relate the desire of a node to relay traffic with its reputation. In order to achieve the reputation based forwarding methodology, the foremost requirement is the effective implementation of a truthful and cheat proof mechanism.

For example, Marti et al. proposed in [102] a reputation-based system where nodes use one mechanism to detect misbehaving nodes and another mechanism to avoid these nodes during route selection. Michardi and Molva [105] proposed a mechanism which not only makes decisions about cooperation but also identifies and avoids malicious

nodes which on purpose drop the data packages they receive. This mechanism uses a weighted combination of the values of three different reputation measures: subjective reputation (which is mainly based on the past observations), indirect reputation (which includes only positive reports by other nodes) and functional reputation (which evaluates task-specific reputation).

Buchegger and Le Boudec introduce in [22] a protocol, called CONFIDANT, which aims at detecting and isolating misbehaving nodes, thus making it unattractive to deny cooperation. For this reason, it uses experienced, observed, or reported routing and forwarding behavior of other nodes in order to define trust relationships and make routing decisions.

All of the above reputation-based protocols were evaluated purely through numerical case studies. Analytical approaches to study reputation-based systems include the following: Urpi et al. proposed in [136] a model, based on game theory, which is capable of formally studying and analyzing strategies for cooperation and as an example they developed a simple strategy that enforces packet forwarding among nodes.

Srinivasan et al. [130] determined the optimal throughput that each node should receive and proposed a distributed and scalable acceptance algorithm which is used by the nodes to decide whether to accept or reject a relay request.

In the market-based (or credit-based) approach, nodes receive a micro-payment (or credit) for every packet that they forward. In return, nodes can use these payments (credits) to send their own traffic.

Buttyan and Hubaux in [23] proposed the use of a counter, in each node, whose indication decreases when the node sends an own packet, increases when the node forwards a packet, and requires to remain always positive. Besides stimulating packet forwarding, the proposed mechanism encourages the users to keep their nodes turned on and to refrain from sending a large amount of packets to distant destinations.

Irissappane et al in [72] proposed a Partially Observable Markov Decision Process to make routing decisions in an energy-efficient and secure manner, when the information about the sensor nodes is limited.

In [148], Zhong et al. proposed a system to provide incentive to mobile nodes to cooperate, which determines payments and charges from a game-theoretic perspective, and motivates each node to report its behavior honestly, even when a collection of the selfish nodes collude.

## 4.2 Preliminaries and models

In both BMLC and WSC the base station acts as a coordinator of a game aiming at the coverage of the targets of interest. The base station pays the agents in order to incentivize them to dispose their sensors in its service. More precisely the problems are the following:

**Definition 18** (BMLC)**.** *We are given a network consisting of a set of n randomly deployed sensor nodes each owned by a different agent, a set of m target points and a base station of fixed locations in sensors' vicinity. In BMLC we are asked to find a schedule of sensor nodes' activity to maximize the lifetime of the network subject to the constraints that each target is continuously monitored by at least one sensor and that a monetary budget of the base station is not exceeded.*

**Definition 19** (WSC)**.** *We are given a network consisting of a set of n weighted and randomly deployed sensor nodes each owned by a different agent, a set of m target points and a base station of fixed locations in sensors' vicinity. In WSC we are asked to find a set of sensor nodes whose weight is minimum subject to the constraint that all target points are monitored.*

### 4.2.1 Basic definitions

The network consists of the following elements: a base station, a set of sensor nodes $N = \{s_1, \dots s_n\}$, each owned by a different agent, and a set of targets points of interest $\mathcal{T} = \{\tau_1, \dots, \tau_m\}$. The positions of the all the elements are considered fixed. We assume that all sensor nodes have the same sensing and transmission range and that $m = \Theta(n)$. Each sensor node $i$ has a limited battery capacity denoted by $\beta_i$ (measured in energy units). A sensor node consumes energy when it is active i.e. when it monitors its sensing area or when it transmits data. Otherwise it does not consume energy (sleep mode). We assume that all batteries have the same consumption rate (1 unit of energy per 1 unit of time). Thus we can measure the capacity of the batteries in time units as well. We also denote by $w_i = 1/\beta_i$ the weight of sensor $s_i$.

A sensor node $s_i$ *covers* a target $\tau_j$ if the latter is in the sensing range of $s_i$. A set $S$ of sensor nodes that monitor all the target points at the same time will be called a sensor cover or simply a cover. Denote by $r$ the number of possible covers (note that $r$ can be exponentially large in $n$) and the $j$-th cover by $\mathcal{C}_j$, $j = 1, \dots, r$. A cover $\mathcal{C}_j$ is active for $t_j$ time units, if all its sensor nodes are active for $t_j$ time units. We use an $n \times r$ matrix $A$ to represent the set of covers. More specifically, the rows of $A$ represent the sensors and the columns represent the covers. If $s_i$ belongs to $\mathcal{C}_j$ then $A_{ij} = 1$, otherwise $A_{ij} = 0$.

In BMLC the objective is the maximization of the total time for which all the targets are covered i.e. $\max \sum_{j=1}^{r} t_j$ subject to battery and budget constraints. First, the total time that a sensor is active must not exceed its battery capacity:

$$\sum_{j=1}^{r} A_{ij} t_j \leq \beta_i, \quad \forall i = 1, \dots, n \tag{4.1}$$

Furthermore, the base station has a budget capacity of $C$ monetary units to dispose to the agents for monitoring the targets of interest so:

$$\sum_{i=1}^{n}\sum_{j=1}^{r} A_{ij}t_j c_i \leq C \tag{4.2}$$

Combining the above constraints we derive the following LP for BMLC:

$$\begin{aligned}
\max \quad & \sum_{j=1}^{r} t_j && (LP_1)\\
\text{s.t.} \quad & \sum_{i=1}^{n}\sum_{j=1}^{r} A_{ij}t_j c_i \leq C, \\
& \sum_{j=1}^{r} A_{ij}t_j \leq \beta_i, \quad \forall i = 1,\ldots,n \\
& t_j \geq 0, \forall j = 1,\ldots,r
\end{aligned}$$

The non negativity constraints are imposed on $t_j \geq 0, \forall j = 1,\ldots,r$ due to the fact that the variables correspond to time units. The cost per time unit $c_i$ expresses the valuation of agent $i$ and is considered to be his private information. Agent $i$ declares to the mechanism a value (his bid) $b_i$ about the cost/time of activity of $s_i$. An algorithm for BMLC takes as input matrix $A$, the bids of the agents $b = (b_i, b_{-i})$ and the battery capacities of the sensors $\beta_i, \forall i = 1,\ldots,n$. In a feasible solution for BMLC, the algorithm assigns a value to $t_j, \forall j = 1,\ldots,r$. The active time of agent $i$ is the total time its sensor node is active i.e. $\sum_{j:s_i\in\mathcal{C}_j} t_j$ and his profit is $profit_i(c_i, b) = p_i(b) - c_i \sum_{j:i\in j} t_j$, where $p_i$ is the payment defined in (2.1).

In WSC the valuation of $i$ is his weight $w_i$ (measured in monetary units) which corresponds to the amount needed for the disposal of $s_i$ to monitor the targets in sensing area. The profit of agent $i$ is then $profit_i(w_i, b) = p_i(b) - w_i$.

## 4.3 Our mechanism for BMLC

We introduce here a natural model of randomly generated instances of BMLC that in fact was considered in the literature before in context of the closely related minimum cardinality set cover problem, see Vercellis [138]. Suppose the number of targets $m$ is given. Then every sensor node among $n$ sensor nodes independently at random covers each target, again independently at random, with probability $\epsilon > 0$, where $\epsilon$ is a given constant. Thus each sensor node covers $\epsilon m$ targets in expectation. Our analysis carries out even when each sensor node $i$ covers each target $j$ with probability $\epsilon_{ij}$, where $0 < \epsilon_{ij} < 1$ for all $i, j$. In this case we have to replace $\epsilon$ with $\max_{ij} \epsilon_{ij}$ in the analysis.

In this section let us assume that matrix $A$ as defined in the $(LP_1)$ for BMLC is the result of this randomly generated instance of BMLC. That is, each column of $A$ is a feasible cover.

We design a mechanism that is truthful with high probability, that is, on almost all instances in this random instances model. We are only aware of one paper employing a similar approach using VCG-based mechanisms providing average case bounds for mechanism design [141]. In contrast to that paper, we need to design a dedicated mechanism for our problem, which is not VCG. We prove truthfulness of our mechanism by exploiting the combinatorial structure of BMLC. We call this mechanism Random Knapsack Greedy (RKG).

We now present a high level outline of the ideas leading to our new mechanism. Note that BMLC is NP-hard to solve since it is a more general case of the lifetime coverage problem known to be NP-hard when $A$ is not of polynomial size [19]. Thus, since the VCG mechanism requires an optimal solution, it does not admit a polynomial time optimal algorithm unless P=NP. Because we aim at an efficient mechanism, we are not able to use VCG. Then usually it is a complex task to design a monotone, i.e., truthful mechanism. We provide here a new technique which achieves this goal on the majority of the instances of BMLC. We believe that this technique is of independent interest. In addition to being theoretically interesting, this approach also is relevenat to applications, because in practice one usually deploys mechanisms on typical, and rather not worst-case instances.

We first introduce a relaxation of BMLC to an easier problem in such a way that we are able to design a monotone mechanism for that easier problem (it is the fractional maximum weighted knapsack problem, that has known monotone greedy algorithms). Secondly, we need to translate the solution of the knapsack problem back to the original space of variables of BMLC in a way that preserves monotonicity and enables us to obtain a good approximation guarantee. The second step requires much care and it crucially uses combinatorial properties of the random instances of BMLC. We describe now the details.

To reduce BMLC to the knapsack problem our algorithm introduces a new variable $z_i$ for every battery constraint of sensor node $i$ to be

$$z_i = \sum_{j=1}^{r} A_{ij} t_j, \ i = 1, \ldots, n, \tag{4.3}$$

and then we normalize them to the variables $y_i = \frac{z_i}{\beta_i}, \forall i \in [N]$. With the polynomially many new variables, $(LP_1)$ becomes the following fractional knapsack LP

$$\begin{aligned} \max \quad & \sum_{i=1}^{n} \beta_i y_i && (LP_2) \\ \text{s.t.} \quad & \sum_{i=1}^{n} c_i \beta_i y_i \leq C && (4.4) \\ & 0 \leq y_i \leq 1, \forall i = 1, \ldots, n \end{aligned}$$

This new system is now polynomial in size and the problem can be solved optimally in polynomial time by the Greedy algorithm [35]. Greedy sorts the items in decreasing order of the ratio of profit ($\beta_i$ here) per size ($c_i\beta_i$ here), $r_i = \frac{\beta_i}{c_i\beta_i} = \frac{1}{c_i}$. The items are considered by Greedy in this order and added fully (that is if $i$ is added then $y_i := 1$) to the solution one by one so long as the current solution does not exceed the budget $C$. For the first item that violates the budget $C$ only a fractional portion of it is added to the solution to the extent that the constraint (4.4) is fulfilled with equality. Then the Greedy stops. It is clear that Greedy is monotone since if agent $i$ misreports to a value $c_i' < c_i$ this can only lead to higher $r_i$ and thus a higher position in the order. In this case we see that the total time allocated to sensor $i$, that is, the value of $y_i = z_i/\beta_i$, doesn't decrease as needed for monotonicity according to Archer and Tardos [14].

Once we have a solution to ($LP_2$), we need to translate it back to the $r$-dimensional space of ($LP_1$). This is obtained by solving the system of linear equations (4.3). Note, that it is crucial that we insist on the exact solution of the system (4.3) to preserve monotonicity.

As the main technical ingredient of our construction, we will show that it suffices to solve the system (4.3) on a carefully chosen $n \times n$ submatrix $A'$ of matrix $A$. This submatrix has to have full rank to guarantee solvability of the system (4.3), and it has to be column-sparse to guarantee good approximation to BMLC.

The columns of submatrix $A'$ are generated independently at random and each consists of exactly $q = \lfloor \alpha \log m \rfloor$ ones, where $\alpha$ is constant to be defined later. Vercellis [138] considered the randomly generated instances described above for the set cover problem and showed that the optimal, that is minimum cardinality, solution to the set cover instance contains $q$ sets. We prove that any randomly generated column of length $n$ with exactly $q = \lfloor \alpha \log(m) \rfloor$ ones is a feasible cover with high probability (note that in our notation, a one with index $i$ in this column means that sensor node $i$ is present in the cover represented by this column, and zero means that sensor node $i$ is not present).

Recall that matrix $A$ is randomly generated by our random instances process, and we will prove that $A'$ will be its submatrix with high probability.

**Lemma 12.** *A column of matrix $A'$ with $q = \lfloor \alpha \log(m) \rfloor$ ones is not a cover with probability at most $\frac{1}{m^{d-1}}$, where $\alpha = \frac{d}{\log \frac{1}{1-\epsilon}}$ and $d > 1$ is a constant.*

*Proof.* The probability that target $\tau$ is not covered by any of the $\alpha \log(m)$ sensor nodes in that column (we denote this event by $\bar{\tau}$), is

$$\begin{aligned} Pr(\bar{\tau}) &= (1-\epsilon)^{\alpha \log m} = 2^{\alpha \log m \log(1-\epsilon)} \\ &= 2^{-\alpha \log m \log(\frac{1}{1-\epsilon})} = (\frac{1}{m})^{\alpha \log(\frac{1}{1-\epsilon})} \end{aligned}$$

Let $\tau_1, \ldots, \tau_m$ be all the target points. By the union bound, the probability that the

cover represented by our column is infeasible is

$$Pr(\bar{\tau_1} \vee \ldots \vee \bar{\tau_m}) \leq \sum_{i=1}^{m} Pr(\bar{\tau_i}) = \sum_{i=1}^{m} (\frac{1}{m})^{\alpha \log \frac{1}{1-\epsilon}}$$
$$= m(\frac{1}{m})^{\alpha \log \frac{1}{1-\epsilon}} = m^{-\alpha \log \frac{1}{1-\epsilon}+1}$$

□

Using the union bound and Lemma 12 we obtain:

**Corollary 3.** *The probability that one vector out of n randomly chosen columns of $A'$ is not a cover is at most $nm^{-\alpha \log \frac{1}{1-\epsilon}+1} \leq \frac{1}{n}$ for a suitable choice of $\alpha$.*

In order to prove that $n$ distinct vectors are linearly independent we will need the following:

**Lemma 13.** *Suppose we have $k$ distinct 0/1 vectors $v^1, \ldots, v^k$, each of length $n$ and each with exactly $q = \lfloor \alpha \log m \rfloor$ ones and let $\lambda_1, \ldots, \lambda_k$ be $k$ real numbers. Any 0/1 vector $v'$ with exactly $q$ ones that can be expressed as a linear combination $v' = \sum_{i=1}^{k} \lambda_i \cdot v^i$ is one of the vectors $v^1, \ldots, v^k$.*

*Proof.* The proof is by induction on $k$. The case of $k = 1$ is obvious. Now, we will prove the base case of $k = 2$. Let $v' = \lambda_1 v^1 + \lambda_2 v^2$. Clearly, it is not possible that $\lambda_1 = \lambda_2 = 0$, because then vector $v'$ does not contain $q$ ones. Furthermore, since $v^1 \neq v^2$, there is at least one entry $i$ for which $v_i^1 = 1$ and $v_i^2 = 0$ and another entry $j \neq i$ for which $v_j^1 = 0$ and $v_j^2 = 1$. Because $v'$ is a 0/1 vector, we must have that $\lambda_1, \lambda_2 \in \{0, 1\}$. Now if $\lambda_1 = \lambda_2 = 1$, vector $v'$ will have more than $q$ ones.

We now consider the cases: If $v'_j = 1$ we have that $\lambda_1 v_j^1 + \lambda_2 v_j^2 = 1 \Rightarrow \lambda_1 = 0$ and $\lambda_2 = 1$, so $v' = v^2$. Similarly if $v'_i = 1$, we have that $\lambda_1 = 1$ and $\lambda_2 = 0$, thus $v' = v^1$. Then, we have that $v'_j = v'_i = 0$ which is impossible because then $\lambda_1 = \lambda_2 = 0$. This proves the base case.

Suppose now that the inductive hypothesis holds for $k = s$ vectors, $s \geq 3$. We will prove that it also holds for $k = s + 1$. Let $v'$ be a 0/1 vector such that $v' = \sum_{i=1}^{s} \lambda_i v^i = \sum_{i=1}^{s-1} \lambda_i v^i + \lambda_s v^s$. By induction hypothesis the first sum is one of the vectors $v^1, \ldots, v^{s-1}$. Now the sum reduces to the induction base case where $\lambda_1 = 1$ and therefore the claim follows. □

Our (randomized) algorithm to construct an $n \times n$ submartix $A'$ of matrix $A$ simply chooses $n$ distinct columns of $A$, such that each such column has exactly $q = \lfloor \alpha \log m \rfloor$ ones. By Lemma 13 these columns will be linearly independent. Let $A'$ be the resulting matrix formed by these $n$ columns.

Note that in RKG we do not need to explicitly write matrix $A$, so the time remains polynomial. We now prove the correctness of our algorithm. Using Lemma 13, we easily see that:

**Algorithm 3:** The RKG Mechanism

**Input** : Instance of BMLC where each of the targets that each sensor node selects to cover are chosen independently at random with prob. $\epsilon \in (0,1)$.

**Output**: A polynomial sequence of covers of all targets and the duration ($t_j$) of their sensor node being active, and payments – one to each sensor node.

1 Formulate (implicitly) BMLC as ($LP_1$) of possibly exponentially many variables (one per cover).

2 Define the variables $z_i$ for every sensor node $i$ as $z_i = \sum_{j=1}^{r} A_{ij} t_j$ and $y_i = z_i/\beta_i$. Then obtain the fractional knapsack ($LP_2$).

3 Solve ($LP_2$) optimally by Greedy [35].

4 To get a solution for the original variables $t_j$, (independently at random in expected polynomial time) choose $n$ distinct covers (columns) of matrix $A$ to form matrix $A'$ and solve the resulting system (4.3) with $A$ replaced by $A'$ using, e.g., Gaussian elimination. (* Note: $A'$ has full rank with high probability and there is a unique solution for $t_j$. *)

5 Compute payments as in (2.1). (* Note: The algorithm is single parameter ($c_i$) monotone, as needed in [14]. *)

**Lemma 14.** *The $n$ columns of matrix $A'$ as above are linearly independent with probability* 1*, i.e., with certainty.*

Combining Corollary 3 and Lemma 14 we obtain:

**Lemma 15.** *The $n$ columns of matrix $A'$ as above are linearly independent with probability* 1*, and are all feasible covers with probability at least $1 - nm^{-\alpha \log \frac{1}{1-\epsilon}+1} \geq 1 - \frac{1}{n}$, for a suitable choice of $\alpha$ such that $\alpha = \Theta(1/\log(\frac{1}{1-\epsilon}))$.*

The above algorithm is truthful and achieves an approximation of $\alpha \log m$ with high probability:

**Theorem 20.** *Algorithm RKG runs in polynomial time, is truthful and achieves an $O(\alpha \log(m))$-approximation for the randomly generated instances of BMLC with probability at least $1 - nm^{-\alpha \log \frac{1}{1-\epsilon}+1}$ if the system of equations* (4.3) *has a positive solution, where $\alpha = \Theta(1/\log(\frac{1}{1-\epsilon}))$ for any fixed $\epsilon \in (0,1)$.*

*Proof.* Truthfulness follows by monotonicity and the use of payments defined by equation (2.1). The correctness of the algorithm follows by Lemma 15. We will prove now the approximation guarantee. Let $t$ be the optimal solution of BMLC. Each cover contains at least one sensor node so $\sum_{j=1}^{r} t_j \leq \sum_{j=1}^{r} n_j t_j$, where $n_j$ is the number of sensor nodes in cover $j$. Let $\bar{y}$ be the optimal solution to knapsack, that is to ($LP_2$), obtained by Greedy. Let $\bar{t}$ be the corresponding solution of the system (4.3). The value of the objective is

$$\sum_{i=1}^{n} \beta_i \bar{y_i} = \sum_{i=1}^{n} \bar{z_i} = \sum_{i=1}^{n} \sum_{j=1}^{r} A_{ij} \bar{t_j} = \sum_{j=1}^{r} \bar{t_j} (\sum_{i=1}^{n} A_{ij}) = \sum_{j=1}^{r} n_j \bar{t_j}$$

Since this is the optimal fractional solution to the linear program ($LP_2$), we have that $\sum_{j=1}^{r} n_j t_j \leq \sum_{j=1}^{r} n_j \bar{t_j}$. The crucial observation now is that we can express the optimal solution $\bar{y}$ to knapsack by any submatrix of $A$ that is full rank. Let $A'$ be an $n \times n$ matrix with $n$ columns, each of $\lfloor \alpha \log m \rfloor$ sensor nodes as in Lemma 15. That is with probability at least $1 - nm^{-\alpha \log \frac{1}{1-\epsilon}+1}$, the system of equations $\bar{z_i} = \sum_{j=1}^{r} A'_{ij} \tilde{t_j}$ has a solution $\tilde{t}$ that is a feasible solution to BMLC. As above we now have that $\sum_{i=1}^{n} \bar{z_i} = \sum_{j=1}^{r} n'_j \tilde{t_j}$, where $n'_j$ refers to the number of sensor nodes in column $j$ of matrix $A'$ and we know that $n'_j \leq \alpha \log m$. This finally implies that

$$\sum_{j=1}^{r} t_j \leq \sum_{j=1}^{r} n_j t_j \leq \sum_{j=1}^{r} n_j \bar{t_j} = \sum_{j=1}^{r} n'_j \tilde{t_j} \leq \alpha \log m \sum_{j=1}^{r} \tilde{t_j}.$$

□

**Resource augmentation**

We note that the system of equations (4.3) does not always have a positive solution for variables $t_j, \forall j = 1, \ldots, r$. In such cases one possible solution is using mobile sensor nodes by the base station which can cover all the target points while occurring an additional cost. This notion was used in a slightly different way in [47] under the name of resource augmentation.

## 4.4 A truthful PTAS for WSC

Our approach follows and enhances the algorithm by Li and Jin [97]. The authors presented a PTAS that finds a set of disks (sensor nodes with unit radius in the plane) which cover all the target points and have the minimum total weight. Our main result is a modification of the guessing phase of the algorithm in order to obtain a truthful mechanism for WSC. Since we consider single parameter agents (here the private parameter of each agent is the weight), in order to obtain truthfulness, we first need to show that the algorithm is monotone. Then using the payment scheme by [14] we derive a truthful mechanism for WSC. However, monotonicity is not sufficient when combining several monotone subalgorithms. In order to circumvent this we need the stronger notion of bitonicity [107], as the algorithm proceeds in iterations and chooses the minimum cost solution.

### 4.4.1 The algorithm by Li and Jin [97]

Let $\epsilon > 0$ be a fixed error parameter. We only outline their algorithm here (for details see [97]). The algorithm consists of the following phases:

**General description (plane partition):** The algorithm proceeds in iterations choosing the minimum cost solution computed. Let $B$ be the area in the plane in which all

the target points lie in. Define a coordination system of $x$ and $y$ axes. The algorithm proceeds with a constant number of iterations in each of which $B$ is partitioned into squares, called *blocks*, of size $L = \frac{1}{\epsilon}$ aligned to the coordination system. Each block is further partitioned into *squares* of size $\mu = O(\epsilon)$. The authors present an algorithm that computes an approximate $(1+\epsilon)$ solution for each block and then take the union of these solutions for whole $B$. In every iteration each block is shifted two squares to the right and two squares up and a new solution is computed for every block and therefore for $B$. The solution with the smallest weight among those is returned.
The algorithm proceeds with the following three phases in every block:

**Steps for each block:**

1. Guessing: It guesses whether the optimal solution contains more than $K = O(1/\epsilon^5)$ disks. In other words, the algorithm enumerates over all possible combinations of a constant number of disks and returns that combination for which the objective is minimum. If the optimal solution OPT contains at most $K$ disks the algorithm enumerates all possible combinations (in $O(n^K)$ time where $n$ is the number of disks) and chooses the combination of disks that cover all the targets and have the smallest weight. Otherwise it guesses the set $\mathcal{G}$ of the $K$ disks with the largest weight. Assuming that the guess is correct the algorithm discards all the targets that are covered by $\mathcal{G}$ and all the disks with weight larger than the weight of the cheapest disk in $\mathcal{G}$.

2. Construction of set $\mathcal{H}$: Next, the algorithm chooses a set $\mathcal{H}$ of at most $\epsilon K$ disks from the remaining ones. The algorithm first adds in $\mathcal{H}$ the furthest pair of disks in every square (independently of their weights). Next, a constant number of disks are added to $\mathcal{H}$. This is done by a careful way to break the uncovered regions into two subregions which fulfill certain geometric properties that are related to the geometric structure of disks and targets in the plane. The criteria for adding a disk to cover these subregions depend either on the distance between pairs of disks or on the intersections of disks. However these criteria are independent of disks' weights.

3. Dynamic Programming. Once $\mathcal{G}$ and $\mathcal{H}$ are found the remaining small pieces of uncovered targets of the instance are solved optimally by dynamic programming.

### 4.4.2 Our truthful mechanism

We will describe now how to modify their algorithm to achieve monotonicity. We note that in order to prove that the algorithm is bitonic it is enough to prove bitonicity for one of the iterations. Furthermore, observe that the phase of construction of $\mathcal{H}$ and the phase of dynamic programming are bitonic and therefore monotone because the choice

of the disks is independent of the weights for the former and because we compute the optimal solution for the latter.

The procedure of standard guessing as shown by Li and Jin is not monotone and hence not bitonic. As an example consider the case where sensor node $s_i$ is in the guessed solution with weight $w_i$ and all other sensor nodes with weight larger than $w_i$ are removed. Then if the weight of $s_i$ decreases too much, this will result in an infeasible solution since all non guessed sensor nodes will be removed.

In order to achieve bitonicity and therefore monotonicity we use the approach proposed by Grandoni et al. [63] to guess not only the subset of largest weight disks but also to guess their weights. This step is crucial for bitonicity and does not affect significantly the approximation ratio and the running time of the algorithm.

Let $w_{min}$ and $w_{max}$ be the smallest and largest guessed weights of the disks, respectively. Starting from a fixed high weight we keep decreasing its value using binary search, until all disks are discarded, except from those that were guessed, leading to an infeasible solution. At this point this value is $w_{min}$. Similarly if we keep increasing the value until no disk is discarded leading to the same solution after some value, we have found the value of $w_{max}$.

The next step is to guess the weights of the disks in a guessed subset. Let $T$ be the set of all the integer powers of $(1+\epsilon)$ between $w_{min}$ and $w_{max}$. For a fixed guessed subset of the $K$ disks we guess the weights in the set $T$. We then consider all the possible combinations of weights with values in $T$. Under each different assignment of weights to disks we have a different subproblem. Each subproblem is formed of the $K$ guessed disks together with one of the possible assignments of the guessed weights of the disks. The combinations of the $K$ guessed disks is $n^K$ and the possible combinations of the guessed weights to disks is $(\log_{1+\epsilon} \frac{w_{max}}{w_{min}})^K$ so in total the number of subproblems is $n^K(\log_{1+\epsilon} \frac{w_{max}}{w_{min}})^K$ which is polynomial, for a fixed constant $K$. Note that the approximation guarantee with additional guess of weights follows the arguments of [63]. We next prove that this procedure is bitonic.

**Lemma 16.** *The procedure of guessing the disks and their weights is bitonic.*

*Proof.* First fix a set $\mathcal{G}$ of the $K$ guessed disks and fix an assignment of weights. Let $w_D \in T$ denote the guessed weight of disk $D$. It is not difficult to see that the procedure of guessing is monotone. Consider a disk $D$ with original weight $w_D$ and suppose it is in the set $\mathcal{G}$ of the guessed disks. If we consider the same instance where $D$ now has weight $w'_D < w_D$ then $D \in \mathcal{G}$ due to the deterministic procedure of choosing disks. We will now show how the objective of the problem changes according to the change in the weight of disk $D$.

Suppose that $D \notin \mathcal{G}$ with weight $w_D$. Then by monotonicity of guessing $D \notin \mathcal{G}$ for any weight $w'_D > w_D$ due to the deterministic procedure of guessing. Thus the value of the objective will not change.

Consider now the case where $D \in \mathcal{G}$ with weight $w_D$. By monotonicity $D \in \mathcal{G}$ for any weight $w'_D < w_D$. Let $S$ denote the fixed set of disks $\mathcal{G}$ when $D \in \mathcal{G}$ has original weight $w_D$ and let $w(S)$ denote its weight. Recall that the algorithm after the guessing constructs the set $\mathcal{H}$, it then computes the optimal solution on the remaining instance and among all the possible guesses it chooses that one with the smallest total weight. Consider now all the possible $q$ solutions by the algorithm $S_1, \ldots S_q$ in a non-decreasing order of their weights i.e. $w(S_1) \leq \ldots w(S_q)$. Let $S'$ be the solution when $w'_D < w_D$. If $w(S') < w(S_1)$ then the output solution has the smallest cost among all the possible ones and the lemma holds. If $w(S_1) < w(S')$ then the algorithm will output $S_1$ which has the smallest weight and the lemma holds. □

Combining the bitonicity of construction of $\mathcal{H}$, the bitonicity of the dynamic programming procedure and Lemma 16, we have that every iteration of the modified algorithm is bitonic and therefore the algorithm is monotone. The payments can be computed in polynomial time by binary search in order to find the critical value. Combining the monotone algorithm with this payment scheme we have the following:

**Theorem 21.** *The modified algorithm described above leads to a truthful PTAS.*

The overal algorithm is presented below. We denote by $S^i_j$ the value returned by the algorithm Guessing Sensors and Weights (GSW) on shift $i$ for block $B^i_j$.

**Algorithm 4:** The PTAS for WSC

**Input** : A plane $B$ and $\epsilon$
**Output**: A monotone PTAS for WSC on $B$

```
Partition B in blocks B_1, ..., B_p
for i ← to p do
    for j ← 1 to p do
        Partition B_j in q squares
        S^i_j = GSW((B^i_j))
        Shift each block by two squares to the right and to squares to the top
    end
end
```

**Algorithm 5:** Guessing Sensors and Weights (GSW)

**Input** : A block $B_i$ and $\epsilon$
**Output**: A monotone PTAS for WSC on $B_i$

```
Guess a constant number of the most expensive sensors in OPT
Guess the weights of the guessed sensors
Construct H of O(1/ε^4) sensors and include it in the solution
Solve by Dynamic Programming the remaining instance
```

## 4.5 An extension of the Garg-Konemann algorithm for BMLC

We prove that the framework of Garg-Konemann [57] used by Berman et al. [19] retains the approximation guarantee when we have many additional budget constraints (e.g. a budget on the delay related to the Quality of Service of the transmitted data). The algorithm reduces BMLC to WSC by losing an approximation factor of $(1+\epsilon)$. Their algorithm is based on a primal dual approach using $LP_1$ and its dual which we denote by $LP_3$:

$$\begin{aligned} \min \quad & \sum_{i=1}^{n} \beta_i \omega_i + C\omega_0 \\ \text{s.t.} \quad & \sum_{i=1}^{n} A_{ij}(\omega_i + c_i\omega_0) \geq 1, \quad \forall j = 1, \dots r \\ & \omega_i \geq 0, \forall i = 1, \dots, n \end{aligned} \tag{$LP_3$}$$

The $\omega_0$ variable in the above LP corresponds to the budget constraint of $LP_1$. The authors use an $f$ approximation algorithm for the WSC as a subroutine to find an $f$ approximate minimum weight column on matrix A. We provide below the proof which essentially follows the proof by Garg and Könemann [57] for completeness.

The $\omega_i$ variables of the dual correspond to the weights of the sensor nodes for WSC. As can be seen from $LP_3$ the additional budget constraint only changes the weight of sensor node $s_i$ from $\omega_i$ to $\omega_i + c_i\omega_0$. We can then consider their algorithm having as weights to the sensor nodes the new ones and solve BMLC.

**Theorem 22.** *The reduction of BMLC to set cover by Garg-Konemann is an $(1+\epsilon)f$ approximation, where $f$ is the approximation factor of set cover.*

*Proof.* The algorithm proceeds in iterations. Let $D(k) \equiv D(\omega^k) = b^T\omega^k$ be the objective of the dual and $g^{k-1}$ be the value of the primal variables at the beginning of the $k$-th iteration. The algorithm calls an $f$ approximation algorithm for set cover as a subroutine to choose the minimum weighted column of $A$. Let $q$ be the minimum weighted column at $k$-th iteration and $\alpha(\omega) = \min_j \sum_i A_{ij}\omega_i$ be its value. The algorithm finds the row $p$ for which the value $\beta_i/A_{ij}$ is minimum. The primal variables $t_q$ are increased by $b_p/A_{p,q}$ and thus $g^k = g^{k-1} + b_p/A_{p,q}$. The duals change as follows $\omega_i^k = \omega_i^{k-1}(1 + \epsilon\frac{b_p A_{i,q}}{\beta_i A_{p,q}})$, where $\epsilon > 0$. Furthermore the dual variables are initialized as $\omega_i^0 = \delta/\beta_i$ and the algorithm stops at the iteration $t$ for which $D(t) \geq 1$. For iteration $k \geq 1$ we have:

$$
\begin{aligned}
D(k) &= \sum_i \beta_i \cdot \omega_i^k \\
&= \sum_i \beta_i \cdot \omega_i^{k-1} + \epsilon \frac{b_p}{A_{p,q}} \sum_i A_{i,q} \cdot \omega_i^{k-1} \\
&= D(k-1) + \epsilon(g^k - g^{k-1}) \cdot \alpha(k-1) \\
&\leq D(k-1) + \epsilon(g^k - g^{k-1}) \cdot f \cdot \alpha(k-1)
\end{aligned}
$$

So recursively we have that :

$$
D(k) \leq D(0) + \epsilon \sum_{l=1}^{k} (g^l - g^{l-1}) \cdot D(l-1) \cdot f
$$

If we let $\beta = \min_\omega D(\omega)/\alpha(\omega)$, then $\beta \leq D(l-1)/\alpha(l-1)$ so

$$
D(k) \leq m\delta + \frac{\epsilon}{\beta} \sum_{l=1}^{k} (g^l - g^{l-1}) \cdot D(l-1) \cdot f
$$

Define next

$$
x(i) m\delta + \frac{\epsilon}{\beta} \sum_{l=1}^{k} (g^l - g^{l-1}) x(l-1)
$$

for all $i \geq 0$ in order to solve the recurrence. We then have:

$$
\begin{aligned}
x(k) &= m\delta + \frac{\epsilon}{\beta} \sum_{l=1}^{k-1} (g^l - g^{l-1}) x(l-1) + \frac{\epsilon}{\beta} (g^l - g^{l-1}) x(k-1) \\
&= (1 + \frac{\epsilon}{\beta}(g^k - g^{k-1})) x(k-1) \\
&\leq e^{\epsilon(g^k - gk-1)/\beta} x(k-1) \\
&\leq e^{\epsilon g^k/\beta} x(0) = m\delta \cdot e^{\epsilon g^k/\beta}
\end{aligned}
$$

and since $D(k \leq x(k))$ we get

$$
D(k) \leq m\delta e^{f \epsilon g^k/\beta}
$$

We then obtain

$$
\frac{\beta}{g^t} \leq \frac{\epsilon \cdot f}{\ln(m\delta)^{-1}}
$$

by the stopping condition $D(t) \geq 1$. □

We note that the proof can be extended when multiple budgets are considered, preserving the approximation ratio.

## 4.6 Open problems

We provided here efficient and truthful mechanisms for coverage problems in sensor networks. An interesting extension of our work is to combine issues of strategic coverage by sensors (i.e. truthful payment to battery usage) with issues of incentive based routing. Another extension would be to add the quality of service (e.g. delays) as a second parameter of every agent to our basic BMLC problem. We note that the design of truthful mechanisms for multidimensional problems (even for 2 dimensions) differs radically than for one parameter.

# Chapter 5

# The primal dual method for network design problems

## 5.1 The problem

Network design is a topic which includes many problems with numerous applications including telecommunications, transportation planning and electronics [70]. In the field of Electronics, a classical objective when optimizing the area of Very Large Scale Integration (VLSI) layouts is the minimum total interconnection on the circuits. Although there are several other criteria dominating the routing objective like the reliability and manufacturability issues and noise, minimizing the length of the wires is of huge importance in order to avoid high temperatures of the circuit [29, 86]. Given a number of pins, called *terminal* pins we are asked to interconnect them using as less wire as possible. The circuit might include other junctions which do not necessarily need to be connected called *Steiner* pins. In the minimum Steiner tree problem we seek to find the set of Steiner pins in order to minimize the total length of wires to connect the terminal pins.

Similarly, in communication networks we are given pairs of nodes and again we are asked to connect them minimizing the total length between them. However, parts of the network are susceptible to failure or can be destructed for various reasons, thus the existence of redundant paths between the pairs of nodes is necessary to increase the survivability [100]. In the *Survivable Network Design Problem (SNDP)* we are given pairs of network nodes and a number of required paths for the connection of each pair. We seek to connect the pairs by the given number of alternative paths having the minimum total length.

In large networks the components are heterogeneous and thus can be owned by different entities, the agents. Every agent provides services to the central authority of the network and gets a reward for this service. A task can be for example the forwarding of messages through his component. Consider the representation of the network by a graph where each edge represents a component of an agent. We need to connect pairs

of nodes in the network with the minimum total cost by at least a number of edge disjoint paths for every pair, where now every edge belongs to a different agent. The input to the problem i.e. the costs of the edges is now given by the agents. We assume that the costs of the edges are the private information of the agents and that each agent owns exactly one edge. The problem now becomes more complicated since we have to cope with the incentives of those agents. An agent declares a price for his service which however might not correspond to his true cost for the service. Hence, we need to find a solution to the optimization problem which will incentivize all the agents to declare their true cost, or simply we need a truthful mechanism for the problem.

In this chapter we draw our attention to the network design problems described above. More precisely we study the *Survivable Network Design Problem (SNDP)* and its special cases of *Steiner forests* and *Steiner trees*. We survey approximation algorithms that are based on the primal dual method, some of which achieve the best to date approximation e.g. the algorithm by Goemans and Williamson [62] for Steiner forests. We then prove that all these algorithms are monotone and thus we can obtain truthful mechanisms with an appropriate payment scheme similarly to [14]. Because of monotonicity the payment to agent $i, \forall i \in E$ can be computed using binary search until we find the *critical value* $\bar{c}_i$ i.e. the value for which $i$ will be in the solution for any declared cost $c'_i \leq \bar{c}_i$ and out of the solution for any $c'_i > \bar{c}_i$. We note that we are not aware of any explicitly written such proofs of monotonicity of these algorithms in the literature.

## 5.2 Further related work

There has been a lot of work in the area of approximation algorithms for network design problems. Especially for the Steiner tree problem the bibliography on approximation algorithms is vast [24, 61, 76, 119, 122, 146].

A very simple algorithm which is not difficult to prove that it is also monotone, achieves an approximation of 2 for the Steiner tree problem. The *Minimum Spanning Tree (MST) heuristic* computes the minimum spanning tree on the metric closure of a graph. Almost all the algorithms that followed are based on an initial computation of the MST heuristic and then proceed with local improvements.

Until recently no LP relaxation was known with integrality gap smaller than 2. Byrka et al. [24] presented an algorithm based on iterative randomized rounding with approximation of $\ln(4)+\epsilon < 1.39$ on general graphs and also proved that the integrality gap is at most $1 + \ln(3)/2 < 1.55$. Following the results of [24], Goemans et al. [61] later considered the hypergraphic LP relaxations presented a deterministic $\ln(4 + \epsilon)$ algorithm for general graphs and proved an $\ln(4)$ upper bound on the integrality gap. We note that there is no primal dual algorithm known for the Steiner tree problem on general graphs with approximation ratio better than 2.

SNDP is known to be NP-hard even when the requirements are equal to 1 for every pair of nodes and even on the special case of quasi bipartite graphs. On general graphs it is also known to be NP-hard when the cost of the edges are 1 or 2 [20]. Furthermore, on quasi bipartite graphs it is hard to approximate within $\frac{96}{95}$ of the optimal solution [32] and thus there is no PTAS unless P=NP.

The first approximation algorithm for SNDP was given by Williamson et al. [143] who obtained a ratio of $2k$, where $k$ is the maximum requirement of a set. Goemans et al. [60] improved this ratio to $2H_k$ presenting a primal dual algorithm, where $H_k = 1+\frac{1}{2}+\ldots+\frac{1}{k}$ is the harmonic function. The approximation ratio for both was obtained by primal-dual algorithms. Jain [73] further improved this factor to 2 introducing the iterative rounding method. Agrawal, Klein, & Ravi [6] obtain a 2-approximation algorithm using the primal dual method for SNDP where the edge requirements for each pair of terminal nodes are in $\{0, 1\}$. They also obtained a $2\log f_{max}$-approximation algorithm for SNDP when multiple copies are allowed, where $f_{max}$ is the maximum requirement of crossing between cuts of the graph.

## 5.3 The primal dual method

The Primal Dual Method constitutes a very important tool in combinatorial optimization and can be used as a different means to solving linear programs. Kuhn first proposed the Hungarian method for the assignment problem [89] which later inspired Dantzig, Ford and Fulkerson [36] to propose the primal dual method.

For many network design combinatorial problems which are NP-hard, the best to date Linear Programming (LP) based approximation algorithms use the iterative rounding [73], the iterative randomized rounding [24] and some algorithms use the Minimum Spanning Tree (MST) as a subroutine by an iterative primal dual approach [85]. However the Primal Dual method possesses features that make it valuable to study. Primal dual algorithms do not require the solution of the LP making them faster to implement. Furthermore, these algorithms can be customized easily for a specific problem and are often monotone, thus leading to truthful mechanisms.

Many fundamental problems in combinatorial optimization such as flows on networks, computation of shortest paths and matching either use the primal dual method [115] or can be described in terms of this method e.g. Dijkstra's algorithm for shortest paths [40]. Many of the most fundamental combinatorial problems in **P** are solved optimally by the primal dual method [115].

This method uses the LP formulation of two problems, the primal and the dual. Initially a primal dual algorithm starts with a feasible solution to the dual and an infeasible solution to the primal. It then gradually improves the objective of the dual while also improving the feasibility of the primal, assuming that the complementary slackness conditions are all the time fulfilled. The algorithm stops when a feasible

solution to the primal is found. In this chapter we draw our attention on approximation algorithms for NP-hard problems that produce a solution based on the primal dual method. A more detailed terminology of the primal dual method can be found in [67, 115].

## 5.4 Preliminaries and models

In the network design problems we are given a graph $G = (V, E)$ representing the network, a cost function on the edge set $c : E \to \mathbb{Q}_+$ and a set of requirements. In the Steiner tree problem the requirement is given set of nodes $S \subseteq V$ called *terminals*. More precisely we have the following:

**Definition 20** (Steiner tree)**.** *Given a graph $G = (V, E)$, a cost function $c : E \to \mathbb{Q}_+$ and a set of terminal nodes $S \subseteq V$, find a minimum cost tree that spans the terminal nodes.*

A more general case of the problem is that of Steiner forests, where we are given subsets of terminal nodes and are asked to connect all the nodes in each of the subsets. If we also require that every pair of nodes $u, v \in V$ are connected by $r(u, v)$ edge disjoint paths, then we have the generalized Steiner tree problem also known as the Survivable Network Design Problem (SNDP). This connectivity requirement can be formulated by a cut requirement function $f : 2^V \to \mathbb{Z}^+$ which specifies the number of edges that need to cross every cut $(S, V \setminus S)$ in a feasible solution, namely $f(S) = \max\{r(u, v) | u \in S$ and $v \in V \setminus S\}$. More formally we have the following:

**Definition 21** (SNDP)**.** *Given a graph $G = (V, E)$, a cost function $c : E \to \mathbb{Q}_+$ and a cut requirement function $f : 2^V \to \mathbb{Z}^+$, find a minimum cost set of edges that connects each pair $u, v \in V$ satisfying the connectivity requirement of $f$.*

The ILP formulation of SNDP is the following:

$$\begin{aligned} \min \quad & \sum_{e \in E} c_e x_e & (5.1) \\ \text{s.t.} \quad & \sum_{e \in \delta(S)} x_e \geq f(S), \quad \forall S \subseteq V \\ & x_e \in \{0, 1\}, \qquad \forall e \in E \end{aligned}$$

where the variable $x_e$ denotes whether edge $e$ is taken in the solution or not according to its value 1 or 0 respectively. We denote by $\delta(S)$ the set of edges which have one endpoint in $S$ and one in $V \setminus S$. Relaxing the integrality constraints and dropping the redundant constraints $x_e \leq 1$, we get the following LP:

$$\begin{aligned} \min \quad & \sum_{e \in E} c_e x_e \\ \text{s.t.} \quad & \sum_{e \in \delta(S)} x_e \geq f(S), \quad \forall S \subseteq V \\ & x_e \geq 0, \qquad \forall e \in E \end{aligned} \tag{5.2}$$

and its dual LP is the following:

$$\begin{aligned} \max \quad & \sum_{S \subseteq V} f(S) y_S \\ \text{s.t.} \quad & \sum_{S: e \in \delta(S)} y_S \leq c_e, \quad \forall e \in E \\ & y_S \geq 0, \qquad \forall S \subseteq V \end{aligned} \tag{5.3}$$

Note that the number cuts and hence the number of constraints in (5.10) can be exponentially large on the input graph. In the game theoretic setting of the problems, each edge is owned by a selfish agent. The agent incurs a cost $c : E \to \mathbb{Q}_+$ when providing his edge in the service of the network, and this cost is considered to be his private information.

## 5.5 Approximation algorithms in network design

We study three approximation algorithms for SNDP and its special cases, all based in the primal dual method. We first study the Steiner tree problem on quasi bipartite graphs. We continue with the study of Steiner forests on general graphs which are a special case when function $f$ is proper. We conclude the chapter with the more general case of general functions $f$. The three primal dual algorithms that are presented are based on the general framework of three phases: initialization, edge augmentation and pruning (see Algorithm 6 below). In the edge augmentation phase, the dual variables of every *unsatisfied* set are raised until an edge $e$ becomes *tight* i.e. the constraint of the dual LP for $e$ is fulfilled with equality namely $\sum_{S: e \in \delta(S)} y_S = c_e$. A set $S$ is called unsatisfied if $\delta(S) < f(S)$.

In order to break possible ties we consider that initially the edges are numbered with distinct numbers. If there exists a tie, the edge with the smallest number is added first in the solution. An example of how the algorithm proceeds can be seen in Figures 5.1,5.2,5.3,5.4,5.5 below, reproduced from [137].

Since the agents are single parameter, monotonicity is a sufficient condition for truthfulness. Recall that monotonicity in this problem means that if an agent owning edge $e$ wins with a cost $c_e$ (i.e. is part of the final solution), he will keep winning if he declares a cost $c'_e < c_e$.

**Algorithm 6:** Primal Dual framework

**INITIALIZATION:**
Set $A \leftarrow \emptyset$ and $y_S \leftarrow 0$ for every set $S \subseteq V$
**EDGE AUGMENTATION:**
**while** $\exists$ *an unsatisfied set* **do**
    Uniformly increase $y_S$ for every $S$ until an edge (or arc) $e$ becomes tight
    Add $e$ to $A$
**end**
**PRUNING:**
**return** $A' \leftarrow \{e \in A | A \setminus \{e\}$ is infeasible for primal$\}$

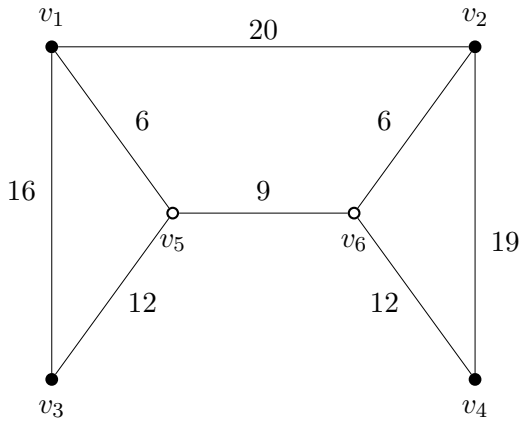



Figure 5.1: Initial instance at the beginning of the primal dual algorithm where the black circles represent the required nodes and white circles the Steiner ones

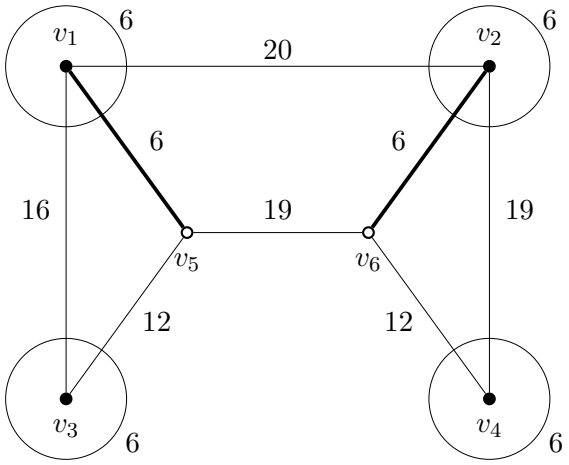



Figure 5.2: First and second iteration of the primal dual algorithm where the cicles represent the values of the duals at the current iteration. The first edges that become tight are $(v_1, v_5)$ and $(v_2, v_6)$ with cost 6, however they are added in to the solution in different iterations.

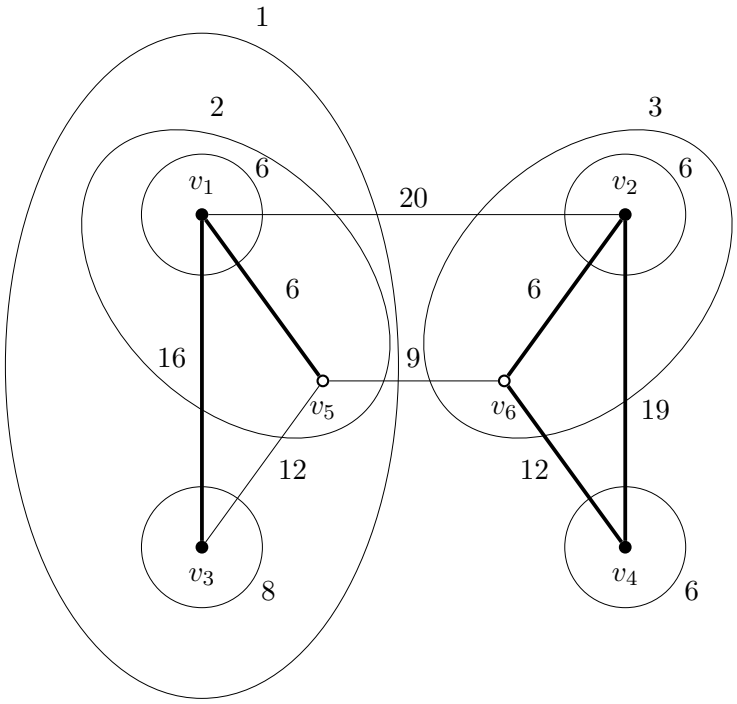



Figure 5.3: At the third and fourth iteration of the algorithm the edges $(v_1, v_3)$ and $(v_4, v_6)$ become tight respectively

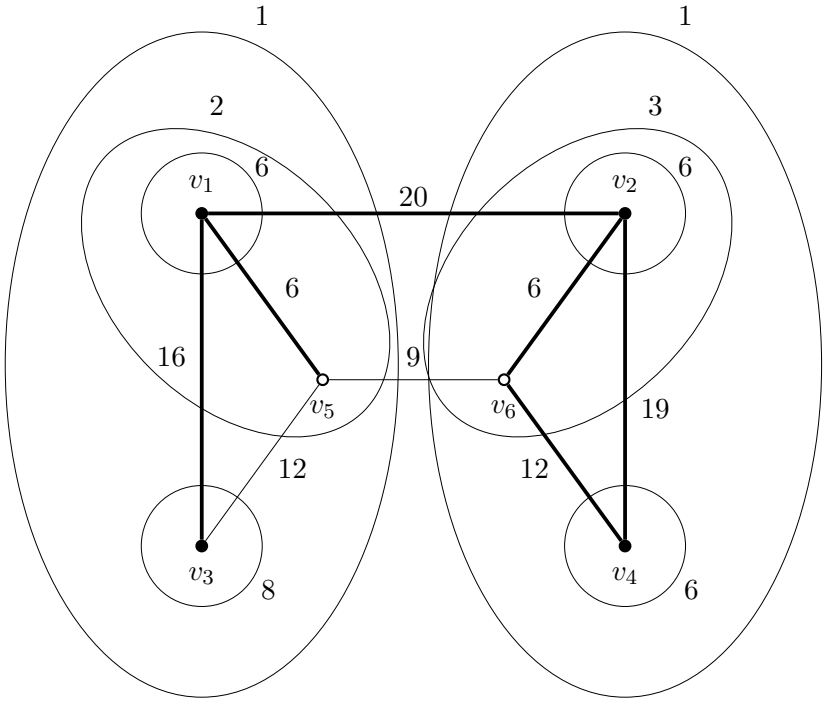



Figure 5.4: In the fifth iteration the edge $(v_1, v_2)$ becomes tight and we get a connected tree

### 5.5.1 Steiner trees on quasi bipartite graphs

In the Steiner tree problem we are given on input two sets of nodes $\mathcal{R} \subseteq V$ called the set of *required* or *terminal* nodes and $\mathcal{S} \subseteq V$ called the set of *Steiner* nodes. We are

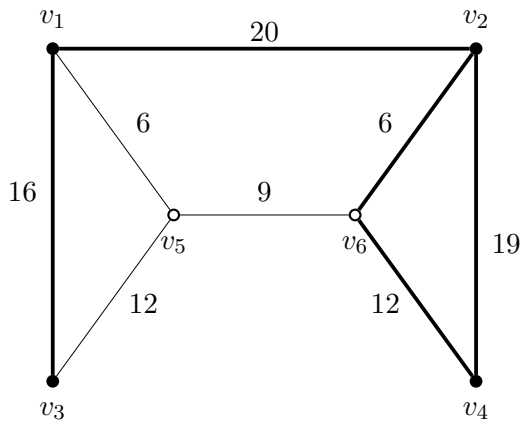



Figure 5.5: The returned solution of the primal dual algorithm is represented by the thick edges

asked to find a minimum cost tree that spans all the terminal nodes by possibly using some of the Steiner nodes. In the special case of *quasi bipartite* graphs there is no edge between any pair of Steiner nodes. The goal is to find the optimal set of Steiner nodes $\mathcal{I} \subseteq \mathcal{S}$. Computing then the Minimum Spanning Tree on $\mathcal{R} \cup \mathcal{I}$ we give us the minimum cost solution.

The best to date approximation for Steiner tree on quasi bipartite graphs is $\frac{73}{60} \approx 1.217$ due to Goemans et al. [61]. However on quasi bipartite graphs the primal-dual algorithm by Rajagopalan and Vazirani [120] achieves an approximation of $\frac{3}{2}$ based on the bidirected cut relaxation described below.

In the bidirected cut relaxation the directed version of the graph is considered first, where each edge $e = (u, v)$ is replaced by two directed edges $e' = (u \to v)$ and $e'' = (v \to u)$ with costs $c_{e'} = c_{e''} = c_e$. An arbitrary node $r \in \mathcal{R}$ is then chosen as a *root* and according to this node a set of nodes $S \subseteq V$ will be called *valid* if $S$ contains at least one terminal and $V \setminus S$ contains $r$. A set is called *unsatisfied* if $f(S) = 1$, but there is no edge crossing the cut $(S, \bar{S})$. A minimally unsatisfied set is the smallest (with respect to inclusion) unsatisfied set.

Let $\vec{E}$ be the set of directed edges. Then the ILP formulation of the problem is shown below:

$$\begin{aligned} \min \quad & \sum_{e \in \vec{E}} c_e x_e \\ \text{s.t.} \quad & \sum_{e \in \delta(S)} x_e \geq 1, \quad \forall \text{ valid set } S \\ & x_e \in \{0, 1\}, \qquad \forall e \in \vec{E} \end{aligned} \tag{5.4}$$

This ILP can be relaxed to the following LP after dropping the redundant constraints $x_e \leq 1$:

$$\begin{aligned} \min \quad & \sum_{e \in \vec{E}} c_e x_e \\ \text{s.t.} \quad & \sum_{e \in \delta(S)} x_e \geq 1, \quad \forall \text{ valid set } S \\ & x_e \geq 0, \qquad\qquad \forall e \in \vec{E} \end{aligned} \tag{5.5}$$

and the dual LP is the following:

$$\begin{aligned} \max \quad & \sum_{\text{valid set } S} y_S \\ \text{s.t.} \quad & \sum_{S: e \in \delta(S)} y_S \leq c_e, \quad \forall e \in \vec{E} \\ & y_S \geq 0, \qquad\qquad \forall \text{ valid set } S \end{aligned} \tag{5.6}$$

The optimal solution to the above ILP contains the edges that have the minimum total cost such that there is at least one edge crossing every valid set. The authors separate the algorithm into two parts called *symmetric* and *asymmetric*.

In the symmetric part each dual variable $y_S$ corresponds to a *proper* set $S$ i.e. both $S$ and $V \setminus S$ contain terminal nodes. Initially the dual variables of all the proper sets are set to 0 and the set of edges in the solution $A$ is empty. In every iteration the algorithm raises uniformly the dual variables of all minimally unsatisfied sets (with respect to inclusion) until a directed edge $e = (u, v)$ becomes tight. If $u$ is a Steiner node the algorithm halts. In this case $u$ is added to set $\mathcal{I}$ and the algorithm proceeds to the next phase. Otherwise $e$ is added to $A$. In case there are more than one tight edges the edge that is added to the solution is chosen in an arbitrary way. Thus in case of ties the tight edge with the smallest number will be added to the solution. The symmetric part stops when there is a path consisting only of edges in $A$ that connects any pair of nodes in $\mathcal{R}$ i.e. when there is no unsatisfied set.

In the asymmetric part the edges in $A$ are taken in reverse order to which they were added in the symmetric part. Set arbitrarily a node $r \in \mathcal{R}$ to be the root. Let $S$ be a valid set and $u \in S$ be a terminal node. If there is a path connecting $u$ and $r$ using edges in $A \setminus \{e\}$ for every $u \in \mathcal{R} \setminus \{r\}$ then $e$ is deleted. This means that there is an edge crossing every valid set $S$. Thus the edges of $A$ constitute a directed tree with $r$ as a root. The algorithm returns the corresponding undirected edges of $A$. The algorithm is described fully below.

**Definition 22.** *We say that two sets $A, B \subseteq 2^V$ cross if $A \cap B \neq \emptyset$ and $A \not\subseteq B$ and $B \not\subseteq A$.*

**Definition 23.** *A family of sets $\mathcal{F} \subseteq 2^V$ is called laminar if no two sets $A, B \in \mathcal{F}$ are crossing.*

**Algorithm 7:** Primal Dual algorithm for Steiner tree on quasi-bipartite graphs

**Input** : An edge weighted quasi bipartite graph $G = (V, E)$, a set of terminals $\mathcal{R}$ and a set $\mathcal{I} = \emptyset$

**Output**: A Steiner tree on $\mathcal{R} \cup \mathcal{I}$ of cost at most $(3/2 + \epsilon) \cdot OPT$

**INITIALIZATION:**
Let $\mathcal{I} = \emptyset$ and $y_S \leftarrow 0$ for every proper set $S$
**SYMMETRIC PART:**
**while** *there exists an unsatisfied set* **do**
    $i \leftarrow i + 1$
    Uniformly raise $y_S$ for every minimally unsatisfied set $S$ until some edge $e_i = (u \rightarrow v)$ becomes tight
    **if** $u \in \mathcal{S}$ **then**
        $\mathcal{I} =\leftarrow \mathcal{I} \cup \{e_i\}$ and $i \leftarrow 0$
    **end**
    **else**
        Compute MST($\mathcal{R} \cup \mathcal{I}$)
    **end**
**end**
**ASYMMETRIC PART:**
Pick as a root an arbitrary node $r \in R$
**for** *j=i down to 0* **do**
    **if** $\mathcal{R} \cup (\mathcal{I} \setminus \{e_j\})$ *is feasible for primal* **then**
        $\mathcal{I} \leftarrow \mathcal{I} \setminus \{e_j\}$
    **end**
    **return** $B \leftarrow$ Undirected($\mathcal{R} \cup \mathcal{I}$)
**end**

**Lemma 17.** *The family of minimally unsatisfied sets $\cup S$ is laminar.*

*Proof.* For the purpose of contradiction suppose there are two minimally unsatisfied sets $S$ and $S'$ which have a crossing such that $S$ is unsatisfied in iteration $i$ and $S'$ gets unsatisfied in iteration $i'$ with $i' > i$. However $S'$ should have been unsatisfied in $i$, but this contradict the minimality assumption for $S$. □

As a corollary from Lemma 17 we get that if on some iteration $i$ of the symmetric part an edge $e$ on the boundary of an unsatisfied set $S$ becomes tight and chosen in the solution, then in iteration $i+1$ (if any) the new unsatisfied set that is formed must contain $S$ and the endnode of $e$ that is outside of $S$. If in iteration $i$ there is some other (at most one) unsatisfied set $S'$ that may contain $e$ in its boundary then in any new iterations any unsatisfied set that contains $S$ must also contain $S'$.

**Lemma 18.** *There are no cycles by the edges of $A$ after the symmetric part.*

*Proof.* We prove the lemma by induction. In the initialization part the number of minimally unsatisfied sets is $|\mathcal{R}|$. Clearly each set corresponds to a trivial tree. Suppose that at iteration $i$ we have a collection of trees. In the next iteration, $i+1$, when an edge becomes tight, either an unsatisfied set will be extended by one edge or two unsatisfied sets which do not have common nodes will be joined into a larger set containing the edge. In any of the two cases the tree structure is maintained. □

Observe that by Lemma 18 the only edges that will be removed in the asymmetric part are those which are adjacent to one terminal and one Steiner node, since removing an edge between two terminal nodes would make the problem infeasible. We can now prove the monotonicity of the algorithm. Based on the above observations we get the following:

**Theorem 23.** *Algorithm 7 is monotone.*

*Proof.* Consider an instance $I$ in which arc $e$ has cost $c_e$ and suppose that $e \in A$ after the deletion step. Now consider the instance $I'$ where the only difference from $I$ is that $e$ has cost $c'_e < c_e$. For the purpose of contradiction suppose that $e \notin A$ in $I'$.

First observe that if $e$ has cost $c'_e$ it will become tight either at an earlier iteration or at the same iteration as when it has cost $c_e$. Thus $e \in A$ on $I'$ after the symmetric part. As a result $e$ is deleted in the asymmetric part when it has smaller cost.

Let $u_1, v_1$ be two terminal nodes which are connected by a path $p$ on $I$ after the asymmetric part and suppose $e \in p$. Let $\bar{e}$ be the last arc that becomes tight and closes the path $p$ on $I$. Since $e$ is deleted on $I'$ there is another path $p'$ connecting $u_1$ and $v_1$ such that $e \notin p'$. Let $\tilde{e} \in p'$ be the last arc which becomes tight on $I'$ before $\bar{e}$ and closes the path $p'$. Figure 5.6 depicts the case of the two paths $p$ and $p'$. The black circles represent terminal nodes an white circles the Steiner nodes. The dashed lines represent

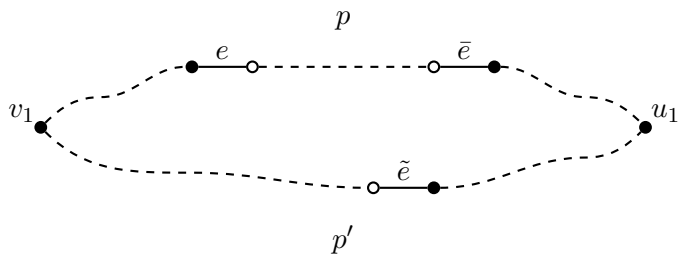



Figure 5.6: Instance of two paths closing a cycle between the terminal nodes $u_1$ and $v_1$. It is also possible that $p'$ might share edges with $p$ but $e \notin p'$

a path between two nodes. We consider two cases according to the time instance that $\bar{e}$ becomes tight.

If $\bar{e}$ becomes tight on $I$ after $e$ then it must also become tight on $I'$ either at the same or at an earlier iteration since the duals that have $\bar{e}$ as a boundary can only reach it earlier. But then we have two paths $p$ and $p'$ closing a cycle, a contradiction.

Consider now the case where $\bar{e}$ becomes tight on $I$ before $e$. If $c_{\bar{e}} < c'_e$ then $\bar{e}$ will be come tight at the same iteration on $I'$ as on $I$. If $c'_e \leq c_{\bar{e}} \leq c_e$ then $e$ will become tight before $\bar{e}$ on $I'$. However the value of the duals around $\bar{e}$ does not change thus if $\bar{e}$ becomes tight on iteration $i$ on $I$ it will become tight on $i+1$ on $I$ in this case. But again we have a cycle by $p$ and $p'$ which is a contradiction. □

### 5.5.2 The Steiner forest problem

We will consider the more general class of *proper* functions:

**Definition 24.** *A function $f : 2^V \to \mathbb{N}$ is proper if $f(V) = 0$ and the following conditions hold:*

1. *$f$ is symmetric that is $f(S) = f(V \setminus S)$*
2. *$f$ satisfies the maximality property that is $f(A \cup B) \leq \max(f(A), f(B))$, for any disjoint sets $A, B \subseteq V$.*

The class of 0-1 proper functions (i.e. $2^V \to \{0, 1\}$) includes many network design problems that have been extensively studied in the literature, such as finding the shortest path, the minimum spanning tree and the minimum Steiner forest in general graphs. We will draw our attention to the more general one, the problem of finding the minimum Steiner forest in general graphs.

In the Steiner forest problem we are given an edge weighted graph $G = (V, E)$ and $k$ pairs of terminal nodes $\{u_1, v_1\}, \ldots \{u_k, v_k\}$. We are asked to find a set of edges of minimum total cost such that $u_i$ is connected to $v_i$ $\forall i \in [k]$. The primal dual algorithm for this problem due to Goemans and Williamson [62], achieves an approximation ratio of 2 which is the best known to date for this problem.

Algorithm 8 shown below consists of the following phases. Initially the algorithm begins with an empty forest, $A = \emptyset$ and the dual variables for all sets $S \subseteq V$ set to 0.

In the first phase the dual variables are uniformly raised until a dual constraint, which corresponds to a tight edge, becomes tight. This edge is then added to the solution and the algorithm proceeds in the next iteration raising the dual variables of the remaining *unsatisfied* sets. Following this procedure the primal feasibility is improved. When $f(S) = 0$ for connected components of $A$ a feasible solution for the primal has been found. We note that we can break ties in a similar way as explained in Algorithm 7 in the case of quasi bipartite graphs.

In the second phase the edges in $A$ are considered in the order they were added to the solution. For each edge $e$, if the solution $A$ remains feasible without $e$, then $e$ is removed from $A$. The general form of the Steiner forest problem can be written the following integer program:

$$\begin{aligned} \min \quad & \sum_{e \in E} c_e x_e \\ \text{s.t.} \quad & \sum_{e \in \delta(S)} x_e \geq 1, \quad \forall \text{ unsatisfied set } S \subseteq V \\ & x_e \in \{0,1\}, \qquad \forall e \in E \end{aligned} \tag{5.7}$$

Relaxing the integrality constraints and after dropping the redundant constraints $x_e \leq 1$, we get the following LP:

$$\begin{aligned} \min \quad & \sum_{e \in E} c_e x_e \\ \text{s.t.} \quad & \sum_{e \in \delta(S)} x_e \geq 1, \quad \forall \text{ unsatisfied set } S \subseteq V \\ & x_e \geq 0, \qquad \forall e \in E \end{aligned} \tag{5.8}$$

and the dual LP is the following:

$$\begin{aligned} \max \quad & \sum_{S \subseteq V} y_S \\ \text{s.t.} \quad & \sum_{S : e \in \delta(S)} y_S \leq c_e, \quad \forall e \in E \\ & y_S \geq 0, \qquad \forall S \subseteq V \end{aligned} \tag{5.9}$$

Let $y_S$ be the dual variable of set $S$ and $\delta(S)$ be the set of edges that cross the cut $(S, V \setminus S)$. We say that an edge $e \in E$ is *overtight* if $\sum_{S : e \in \delta(S)} y_S > c_e$. Let $T$ denote the set of terminal nodes. The algorithm starts with an initially feasible solution to

the dual ($y = 0$) and an infeasible solution to the primal ($A = \emptyset$). Initially the active sets are singletons each for every terminal node. It gradually increases the duals until a feasible solution to the primal is found.

**Algorithm 8:** Primal Dual algorithm for Steiner forests and 0-1 proper functions

**Input** : An edge weighted graph $G = (V, E)$ and a proper function $f : 2^V \to \{0, 1\}$
**Output**: A set of edges $A$ connecting all the pairs constituting a Steiner forest of cost at most $2 \cdot OPT$

```
1  INITIALIZATION:
2  Let A ← ∅, i = 0 and y_S ← 0 for every S ⊆ V
3  EDGE ADDITION:
4  while there exists an unsatisfied set S with f(S) = 1 do
5  |  i ← i + 1
6  |  Uniformly increase y_S for every active set S ∈ V until some e_i ∈ E becomes
   |  tight
7  |  A ← A ∪ {e_i}
8  end
9  EDGE DELETION:
10 for j=0 up to i do
11 |  if A \ {e_j} is feasible for primal then
12 |  |  A ← A \ {e_j}
13 |  end
14 |  return A
15 end
```

In similar way as for Algorithm 7 we can prove the following:

**Lemma 19.** *There are no cycles after the EDGE ADDITION part of Algorithm 8.*

**Theorem 24.** *Algorithm 8 is monotone.*

*Proof.* First observe that the family of unsatisfied sets is laminar similarly to the case of the primal dual algorithm by Algorithm 7 (see Lemma 17). Furthermore notice that the edges are considered for deletion in order in which they were added and not in reverse order as in Algorithm 7. It is not difficult to see that Algorithm 8 does not halt as Algorithm 7. Thus if we follow the proof of Theorem 23 we derive the claimed result. □

### 5.5.3 General proper functions

We will now consider the case of general proper functions where the range of $f$ can be any nonnegative integer i.e. $f : 2^V \to \mathbb{Z}$. The algorithm *UNCROSSABLE* presented by Williamson et al. [143], which will be described later, achieves an approximation of $2 \cdot f_{max}$, where $f_{max}$ is the maximum requirement of $f$ i.e. $f_{max} = \max_S f(S)$.

The algorithm by Williamson et al. [143] proceeds in *phases*. It starts with an empty set $A = \emptyset$ and gradually increases the set of edges. Let $A_p$ be the set of edges that have been chosen at the end of phase $p$ which satisfy the function $f_p(S) = min\{p, f(S)\}$ i.e. $A_p$ is a feasible solution to $IP$ 5.7 for function $f_p$. If we let $f_{max} = \max_{S \subseteq V} f(S)$, then the algorithm will terminate after phase $f_{max}$ producing a feasible solution to the problem.

The set $A_{p-1}$ is augmented to $A_p$ by finding a feasible solution to the following ILP:

$$\begin{aligned} \min \quad & \sum_{e \in E} c_e x_e \\ \text{s.t.} \quad & \sum_{e \in \delta(S)} x_e \geq h(S), \quad \forall S \subseteq V \\ & x_e \in \{0,1\}, \qquad \forall e \in E_p \end{aligned} \tag{5.10}$$

where

$$h(S) = \begin{cases} 1 & \text{if } f_p(S) = p \text{ and } |\delta(S) \cap A_{p-1}| < p, \\ 0 & \text{otherwise} \end{cases}$$

and $E_p = E - A_{p-1}$ is the set of edges considered in phase $p$. The function $h$ is called *uncrossable* if $f$ is a proper function. More formally:

**Definition 25.** *A function $h : 2^V \to \{0,1\}$ is uncrossable if $h(V) = 0$ and for two sets $A$ and $B$ for which $h(A) = h(B) = 1$ then either $h(A \cup B) = h(A \cap B)$ or $h(A \setminus B) = h(B \setminus A) = 1$.*

In order to solve the above minimization problem, Williamon et al. [143] suggested the UNCROSSABLE algorithm as shown below that gives a solution with approximation ratio 2 for any uncrossable function $h$:

There are two main differences between UNCROSSABLE and Algorithm 8 for 0-1 proper functions. In the first part of Algorithm 8, we increase the variables of the duals of all the active sets $S$ for which $f(S) = 1$. However, in UNCROSSABLE the active sets are computed by the $Max\text{-}Violated(h, A)$ oracle. $Max\text{-}Violated(h, A)$ takes as input the currently computed solution $A$ and an uncrossable function $h$ and returns a set of minimally unsatisfied sets $S$ (according to inclusion) for which $h(S) - |\delta_A(S)| = \max_X h(X) - |\delta_A(X)| > 0$, if there exists such a set. These sets are called the maximally violated sets. Furthermore, UNCROSSABLE differs from Algorithm 8 in the deletion step too. In UNCROSSABLE the edges are deleted in reverse order to which they were added in $A$. We now have the following:

**Lemma 20.** *UNCROSSABLE is monotone.*

**Algorithm 9:** UNCROSSABLE

**Input** : An edge weighted graph $G = (V, E)$ and an uncrossable function $h$
**Output**: A Steiner forest $A$ of cost at most $2 \cdot OPT$ connecting the pair of nodes $(u_i, v_i)$, $\forall i = 1, \ldots, k$

$A \leftarrow \emptyset$ $i = 0$ and $y_S \leftarrow 0$ for every $S \subseteq V$
$\mathcal{C} \leftarrow Max\text{-}Violated(h, \emptyset)$
**while** *there exists a unsatisfied set* $S \in \mathcal{C}$ **do**
  $i \leftarrow i + 1$
  Uniformly increase $y_S$ for every unsatisfied set $S \in \mathcal{C}$ until some $e \in E$ becomes tight
  $A \leftarrow A \cup \{e_i\}$ (added edge)
  $\mathcal{C} \leftarrow Max\text{-}Violated(h, A)$
**end**
**for** *j=i down to 0* **do**
  **if** $A \setminus \{e_j\}$ *is feasible for primal* **then**
    $A \leftarrow A \setminus \{e_j\}$ (deleted edge)
  **end**
  **return** $A$
**end**

*Proof.* Recall that the $Max\text{-}Violated(h, A)$ oracle returns the set of minimally unsatisfied sets $S$ for which $h(S) - |\delta_A(S)| = \max_X h(X) - |\delta_A(X)| > 0$, if there is such a set. Since $h : 2^V \rightarrow \{0, 1\}$, the above equality has a positive value when $h(S) = 1$ and $|\delta_A(S)| = 0$. Furthermore the components of any minimally unsatisfied set constitute a laminar family [143]. Consider now the following:

**Claim 3.** *A minimally unsatisfied set is connected.*

*Proof.* Consider a minimally unsatisfied set $S$ with $h(S) = 1$ and for the purpose of contradiction suppose that it consists of two (or more) connected components $X_1$ and $X_2$. If an edge becomes tight and it connects $X_1$ with $X_2$ then $S$ will remain unsatisfied. Thus the edge must cross $S$ so it will either cross $X_1$ or $X_2$. Either way one of the two components will become satisfied which means that our assumption of $S$ being the minimally unsatisfied set was wrong. s a result $X_1$ and $X_2$ must be connected. □

We will now show that if an edge $e$ is not deleted in the reverse deletion step with original cost $c_e$, it will not be deleted with cost $c'_e < c_e$ either. We will consider the possible cases for an edge $e$.

Before we proceed the following observation will be helpful. Suppose an edge $e$ is added in one of the iterations of the algorithm connecting two sets $X$ and $Y$ and suppose that it is later deleted in reverse deletion. Then the possible combinations for the value of $h$ on sets $X$ and $Y$ are $h(X) = 0, h(Y) = 1$, $h(X) = 1, h(Y) = 0$ and $h(X) = 1, h(Y) = 1$, omitting the case where $h(X) = 0, h(Y) = 0$ as not possible. The The first two cases are symmetric so we will consider next one of them.

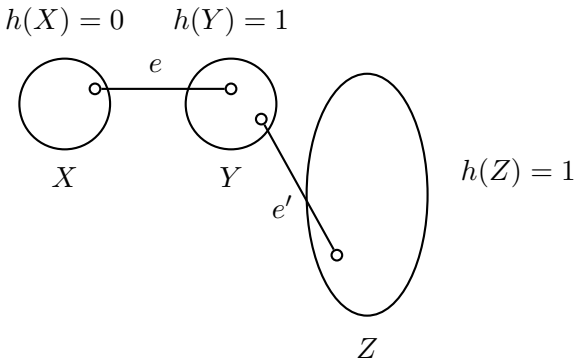



Figure 5.7: Instance of two sets $X, Y$ with $h(X) = 0$ and $h(Y) = 1$

Consider first the case where there are 3 sets $X, Y$ and $Z$ (the current minimally unsatisfied sets) with $h(X) = 0, h(Y) = 1$ and $h(Z) = 1$ as shown in Figure 5.7. Suppose first that $e$ is added before $e'$. Then by the reverse deletion step it will be considered for deletion after $e'$. As can be seen $e$ is redundant as by deleting it the solution remains feasible. Furthermore if $e$ is added after $e'$ then it will be considered for deletion first. Again since the solution remains feasible without $e$ it will be deleted.

Consider now the case depicted in Figure 5.8 where $h(X) = 1, h(Y) = 1$ and $h(Z) = 1$. Similarly as above, we consider the order to which the edges are added in the solution. If $e$ is added before $e'$ and $e''$ then it will be considered last for deletion. In this case both $e'$ and $e''$ will not be deleted so that sets $Z$ and $W$ remain satisfied and $e$ will be deleted as redundant. Using similar argument if $e$ is added before $e''$ and after $e'$ or before $e'$ and after $e''$ it will be redundant and thus deleted.

With this observation let us now consider an edge $e$ having cost initially $c_e$ and suppose that it is added and not deleted in the solution by UNCROSSABLE. Consider now the same instance with the only difference that $e$ has now cost $c'_e < c_e$. Clearly raising the duals uniformly, $e$ will either become tight at an earlier iteration or at the same as with the original cost. Suppose now that $e$ is deleted with cost $c'_e$. Then this must be one of the cases discussed above and depicted in figures 5.7 and 5.8. But in those cases $e$ was deleted as redundant and independently of its cost. As a result $e$ should be deleted with cost $c_e$, a contradiction.

□

The overall algorithm for general proper functions achieves an approximation of $2f_{max}$ since the algorithm proceeds in $f_{max}$ phases, each calling UNCROSSABLE as a subroutine.

Goemans et al. [60] improved the approximation of the above algorithm to $2H(f_{max})$, where $H(k) = 1 + \frac{1}{2} + \ldots + \frac{1}{k} = O(\log k)$ is the harmonic function, by also considering the broader class of *weakly supermodular* functions:

**Definition 26.** *A function $f : 2^V \to \mathbb{Z}$ is weakly supermodular if it satisfies the*

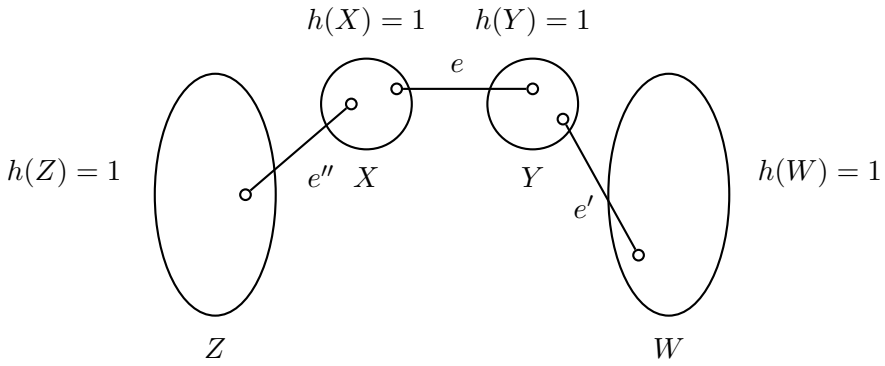



Figure 5.8: Instance of two sets $X, Y$ with $h(X) = 1$ and $h(Y) = 1$

*following:*

1. $f(V) = 0$
2. *For any two sets* $A, B \subset V$ *at least one of the following holds:*
   - $f(A) + f(B) \leq f(A \setminus B) + f(B \setminus A)$
   - $f(A) + f(B) \leq f(A \cap B) + f(A \cup B)$

The algorithm follows the general framework of raising the duals at every iteration until an edge becomes tight and when a primal feasible solution is found the redundant edges are deleted. The basic change in their algorithm is the selection process of the edges in phase $p$. The set of edges $A_p$ must now satisfy $f_p = f(S) - f_{max} + p$. As the previous algorithm, the improved one terminates after $f_{max}$ phases and the solution $A_{f_{max}}$ is feasible. Furthermore if $f$ is weakly supermodular, so is $f_p$.

The improved algorithm calls UNCROSSABLE as a subroutine in every phase $p = 1, \ldots, f_{max}$, which uses an uncrossable function $h_p(S)$ in order to augment the set of edges from $A_{p-1}$ to $A_p$. The uncrossable function is now defined as

$$h_p(S) = \begin{cases} 1 & \text{if } \Delta_p(S) = \max_S \Delta_p(S) = f_{max} - p + 1, \\ 0 & \text{otherwise} \end{cases}$$

where $\Delta_p(S) = f(S) - |\delta_{A_{p-1}}(S)|$ is the *deficiency* of the set $S$. Deficiency shows the number of edges we need to add to the current solution $A_{p-1}$ from $\delta(S)$ in order to have $f(S)$ and thus a feasible solution.

**Theorem 25.** *Algorithm 10 is monotone.*

*Proof.* Consider two instances of the problem which only differ in the cost of an edge $e$. Let its cost be $c_e$ in instance $I$ and $c'_e < c_e$ in instance $I'$. First observe that if an edge becomes tight on some phase $p$ it can only be deleted in $p$ and not at any later phase. This is because the set of edges considered in every phase does not contain any of the edges added in previous iterations since $E_p = E \setminus A_{p-1}$. Suppose that $e$ becomes tight

**Algorithm 10:** Primal Dual algorithm for SNDP and weakly supermodular functions

**Input** : An edge weighted graph $G = (V, E)$ and a weakly supermodular function f for the requirements of every pair of nodes

**Output**: A set of edges $A$ which constitute a feasible solution to SNDP of cost at most $2H(f_{max}) \cdot OPT$ connecting the pair of nodes $(u_i, v_i)$ with requirements $r_{u_i v_i}, \forall i = 1, \ldots, k$

**INITIALIZATION PHASE:**
$A \leftarrow \emptyset$
**for** $p \leftarrow 1$ *to* $f_{max}$ **do**
  **PHASE** $p$**:**
  $\Delta_p(S) = f_p(S) - |\delta_{A_{p-1}}(S)|, \forall S \subset V$
  $h_p(S) \leftarrow \begin{cases} 1 & \text{if } \Delta_p(S) = \max_S \Delta_p(S) = f_{max} - p + 1, \\ 0 & \text{otherwise} \end{cases}$
  $E_p \leftarrow E \setminus A_{p-1}$
  $A' =$UNCROSSABLE$(V, E_p, c, h_p)$
  $A_p \leftarrow A_{p-1} \cup A'$
  **return** $A_{f_{max}}$
**end**

in phase $p$ on instance $I$. We only need to show that if $e$ becomes tight and deleted in any phase $1 \leq i < p$ then the solution on $I'$ will be the same to that of instance $I$. If $e$ is considered in phase $p$ is it will become tight and will not be deleted since UNCROSSABLE is monotone.

Let $p' < p$ be the phase in which $e$ becomes tight and then deleted in instance $I'$. Recall the possible cases of $e$ from Figures 5.7 and 5.8. Suppose that the presence of $e$ had as result the addition of a new edge $\bar{e}$ in phase $p'$. This means that $\bar{e}$ was necessary for an unsatisfied set. However $\bar{e}$ becomes tight independently of $e$ so it should have been added in instance $I$ as well. But this contradicts our assumption. Similarly we can prove that no other edge was deleted because of the presence of $e$. □

# Chapter 6

# Maxmin Heterogeneous Facility Location Games on the Line

## 6.1 The problem

Facility location games lie in the intersection of AI, game theory and social choice theory and have been studied extensively over the past years. In the basic version of the problem, first studied by Procaccia and Tennenholtz [118], a central authority has to locate a facility on a real line based on the reported preferences of selfish agents. Its goal is to locate the facility in such a way that the sum of the utilities of the agents is maximized. [1] However, the agents might misreport their preference and manipulate the authority if the new location results in a better outcome for them.

In the more general case of the problem there are $k$ facilities to be placed on the line and each agent reports to the authority his preferred location and his preference for each of the facilities. In what follows, we focus on the case where the locations of the agents are publicly known and their preferences are their private knowledge.

One main objective for the planner is to design rules to locate the facility, or mechanisms, that are truthful, i.e. no agent has incentives to misreport his preferences about the locations of the facilities. The term *approximate mechanism design without money*, introduced in [118], is usually deployed for problems like the one described above. Since it is not always possible to design truthful mechanisms that find an optimal solution, the goal is to design mechanisms that approximately maximize an objective function under the constraint that they are truthful. In [118] *homogeneous* facility location games were studied, where one or two *identical* facilities have to be placed on a real line and every agent wants to be as close as possible to any of them. In this setting, the agents were reporting to the planner a point on the line representing their location. The objectives in that setting were the maximization of the social welfare, i.e. the sum of the utilities and the maximization of the minimum utility of all the agents.

[1] Note that in [118], the objective was to minimize the social cost, which is equivalent to maximizing the social welfare.

However, in many real life scenarios the facilities and the agents' preferences are *heterogeneous*, i.e. each facility serves a different need and each agent has potentially different needs from the others. In general, an agent might want to: be close to a facility, be away from a facility, or be indifferent about its presence. Feigenbaum and Sethuraman [49] studied one facility heterogeneous games where each agent reports his preferred location on the line while the planner knows whether he wants to be close to or away from the facility. Zou and Li [151] extended the model of [49] for heterogeneous 2-facility games and studied the social welfare objective for several different scenarios of the information the planner knows. We note that none of [49, 151] studied the case where some agents were indifferent for some of the facilities. Serafino and Ventre [127] studied heterogeneous 2-facility games on discrete networks. In their setting, each agent is located on a node of a graph and either is indifferent or wants to be close to each facility. The authors studied the case where the planner knows the location of every agent but not their preferences for the facilities.

We extend the models of [49, 127, 151] and study heterogeneous $k$-facility location games (simply $k$-facility games) on the real line for *maxmin* objectives, which are important to measure the *fairness* of allocation. This class of objectives captures the well studied minimum utility objective, termed UTILITY, and the recently proposed minimum *happiness* objective, termed HAPPINESS. HAPPINESS is a novel fairness criterion introduced in [104]. The happiness of an agent is defined to be the ratio between the utility he gets under the locations of the facilities returned by the mechanism over the maximum utility the agent could get under any possible locations of the facilities. To the best of our knowledge, there is no prior work on this model. We note that while our model is a natural extension of the aforementioned models almost none of those results apply to ours.

We also note that facility location problem on the line can be seen from the networks perspective. Consider the problem of distributing the data in the databases of a company to its data network as mentioned in [127]. The offices are located in specific positions, however if we consider reducing their intermediate distance, then the problem can be modeled as the facility location game on the real line where an office can be located in any position of the line.

### 6.1.1 Our contributions

We prove that there is no optimal deterministic or randomized truthful mechanism for $k$-facility games even for instances with $k = 2$, two agents and known locations of the agents. We furthermore derive inapproximability bounds for deterministic and randomized truthful mechanisms. We note that the techniques we use are fundamentally different from [127], since in our model the facilities can be located anywhere on the line without any constraint, making the analysis more complex.

We then propose truthful mechanisms for $k$-facility games. For $k = 2$ we propose a general technique that produces approximately truthful mechanisms and can be used for any maxmin objective. We derive explicit approximation ratios for UTILITY and HAPPINESS. More specifically, we prove that our mechanism is $(1 - \frac{\sqrt{2}}{2})$-approximate for both objectives even if both agents' locations and preferences for the facilities are not known to the mechanism. If all the agents are indifferent or want to be close to the facilities we prove that the mechanism that locates every facility in the middle of the line is $\frac{1}{2}$-approximate for both objectives and any $k \geq 2$. If the agents' locations are known to the mechanism, then we show how we can utilize the optimal mechanism for the 1-facility game and get a $\frac{3}{4}$-approximate truthful mechanism for UTILITY when $k = 2$. In the case where all the agents are indifferent or want to be away from the facilities, we show that the mechanism that locates $\lfloor \frac{k}{2} \rfloor$ facilities at one end of the line and the rest at the other end is $\lfloor \frac{k}{2} \rfloor / k$-approximate and truthful for both objectives and any $k \geq 2$. Finally, we provide a $\frac{1}{2}$-approximate randomized universally truthful mechanism, for both objectives and any $k \geq 2$. We note that the majority of our mechanisms satisfy stronger notions of truthfulness like *group strategy proofness*.

As a byproduct we show that some of our mechanisms achieve the same approximation guarantee for the social welfare objective, thus we establish a lower bound that was not known before and we complement the results of [151].

### 6.1.2 Further related work

There is a long line of work on homogeneous facility location games [8, 43, 53, 54, 98, 99, 118, 147]. Different objectives and utility functions have been studied as well. In [50] the objective that the authors studied was the sum of $L_p$ norms of agent's utilities, while in [51] it was the sum of least squares. In [52] double peak utility functions were introduced. The obnoxious facility game on the line i.e. the case where every agent wants to be away from the facilities, was introduced in [30] and the model was later extended to trees and cycles in [31]. In [144] the objective of least squares was studied for obnoxious agents. The objective of maximum envy was recently introduced for facility location games in [25].

## 6.2 Preliminaries and model

In a *k-facility location game* (simply $k$-facility game), there is a set $\mathcal{N} = \{1, \ldots, n\}$ of agents located on the line $[0, \ell]$ and a set of $k$ distinct facilities $F = \{1, \ldots, k\}$ that need to be located on the line. Each agent $i$ is associated with a location $x_i \in [0, \ell]$ (we consider a restricted interval in contrast to [118] where $x_i \in \mathbb{R}$) and a vector $t_i \in \{-1, 0, 1\}^k$ that represents his preferences for the facilities.

If agent $i$ wants to be *far* from the facility $j$, then $t_{ij} = -1$, if he is *indifferent*, then

$t_{ij} = 0$ and if he wants to be *close* to $j$ then $t_{ij} = 1$. We will use $y = (y_1, \ldots, y_k)$ to denote the locations of the facilities and $s = (s_1, \ldots, s_n)$ to denote the profile of the agents, i.e. their declared tuples $s_i = (x_i, t_i), \forall i \in \mathcal{N}$. A vector $s_{-i} = (s_1, \ldots, s_{i-1}, s_{i+1}, \ldots, s_n)$ is the vector of tuples excluding $s_i$ thus we can denote a profile as $s = (s_i, s_{-i})$. A *mechanism* $M$ is an algorithm that takes as input a profile $s$ and outputs the locations of the facilities, i.e. $y = M(s)$. A mechanism is *deterministic* if it chooses the locations of the facilities $y$ deterministically and *randomized* if $y$ is chosen according to a probability distribution.

We study the two objectives Utility and Happiness. Utility that agent $i$ gets from facility $j$, denoted as $u_{ij}$, depends on the distance $|x_i - y_j|$ and on the agent's preference for that facility. Formally, $u_{ij}(x_i, t_{ij}, y_j) = g_{ij}(|x_i - y_j|)$ where $g_{ij}$: is an *increasing* function if $t_{ij} = -1$, is a *decreasing* function if $t_{ij} = 1$ and is constant if $t_{ij} = 0$. For normalization purposes we assume that $g_{ij}(\cdot) \leq \ell$ for every $i \in \mathcal{N}$ and $j \in F$ and $g_{ij}(\cdot) = \ell$, if $t_{ij} = 0$. If $y_j$ is chosen from a probability distribution with density function $p(y_j)$, then the expected utility is equal to $\int_0^\ell p(y_j) \cdot g_{ij}(|x_i - y_j|)dy_j$. The total (expected) utility that agent $i$ gets under $y$ is defined as the sum of the utilities he gets for each of the facilities, i.e. $u_i(x_i, t_i, y) = \sum_{j \in [k]} u_{ij}(x_i, t_{ij}, y_j)$. Happiness of an agent for given locations of the facilities is the utility the agent gets under these locations over the maximum utility the agent could get. Let $u_i^*(x_i, t_i) = \max_y u_i(x_i, t_i, y)$. Then, the happiness of agent $i$ under the locations $y$ is the ratio $\frac{u_i(x_i,t_i,y)}{u_i^*(x_i,t_i)}$. Thus, Happiness is $\max_y \min_i \frac{u_i(x_i,t_i,y)}{u_i^*(x_i,t_i)}$.

For every agent $i \in [n]$ we define a function $h_i(u_i)$ that is increasing with $u_i$, incorporating this way all the different objective functions. The function $h_i(u_i)$ is considered public knowledge for all $i \in [n]$. We aim at designing mechanisms that locate the facilities in such a way that the minimum of $h_i(u_i)$ is maximized. Formally, we study objectives of the form $\max_y \min_i h_i(u_i(x_i, t_i, y))$. Throughout this chapter we assume that the functions $g_{ij}$ and the locations $x_i$ of every $i \in \mathcal{N}$ and $j \in F$ are public knowledge, whereas the preferences of the agents are considered to be their private knowledge, unless stated otherwise.

Let $\mathrm{OPT}(s)$ and $M(s)$ denote the optimal value and the value of mechanism $M$ for the objective function under the profile $s$. A mechanism $M$ achieves approximation ratio $\alpha < 1$, or it is $\alpha$-approximate, if $M(s) \geq \alpha \cdot \mathrm{OPT}(s)$ for any type profile $s$.

A mechanism is called truthful if no agent can benefit by misreporting his preferences. Formally, let $(s_i, s_{-i})$ be a true profile for which the returned locations by the mechanism are $y$ and let $(s_i', s_{-i})$ be any misreported profile with returned locations $y'$. A mechanism $M$ is then truthful if $u_i(x_i, t_i, y) \geq u_i(x_i, t_i, y')$. A randomized mechanism is universally truthful if it is a probability distribution over deterministic truthful mechanisms and truthful in expectation if no agent can increase his *expected* utility by misreporting his type. A mechanism is called *group strategy proof* if for any coalition

of the agents no one can benefit by jointly misreporting their profiles. Furthermore a mechanism is called *false-name proof* if no agent can benefit by using multiple and different identities in the game.

We note that the solution to the optimization problem can be solved efficiently for both objectives by the following LP (assuming that the objective functions are linear):

$$\begin{aligned} \max \quad & \epsilon \\ \text{s.t.} \quad & h_i(u_i(x_i, t_i, y)) \geq \epsilon, \ \ \forall i \\ & y \in [0, \ell]^k \end{aligned}$$

where we are searching the values of the locations (variables $y$) and the largest value of variable $\epsilon$ such that $h_i(\cdot)$ is larger than $\epsilon$.

## 6.3 Inapproximability results

In this section we provide inapproximability results for truthful mechanisms for 2-facility games considering $h_i(u_i) = u_i$. We study two prominent objective functions, namely the UTILITY and the HAPPINESS objectives. We prove that placing the facilities on the locations that maximize the objective under the declared preferences of the agents, is not truthful even on instances with two agents. Furthermore, we provide inapproximability results for truthful mechanisms. In the rest of the chapter we follow the literature assuming that the utility of agent $i \in \mathcal{N}$ from facility $j \in \{1, 2\}$ is defined as:

$$u_{ij}(x_i, t_{ij}, y_j) = \begin{cases} |x_i - y_j|, & \text{if } t_{ij} = -1 \\ \ell, & \text{if } t_{ij} = 0 \\ \ell - |x_i - y_j|, & \text{if } t_{ij} = 1. \end{cases} \tag{6.1}$$

Again, the utility of agent $i$ from the facilities is the sum of the utilities over all the facilities, i.e. $u_i(x_i, t_i, y) = u_{i1}(x_i, t_{i1}, y_1) + u_{i1}(x_i, t_{i2}, y_2)$.

### 6.3.1 Utility

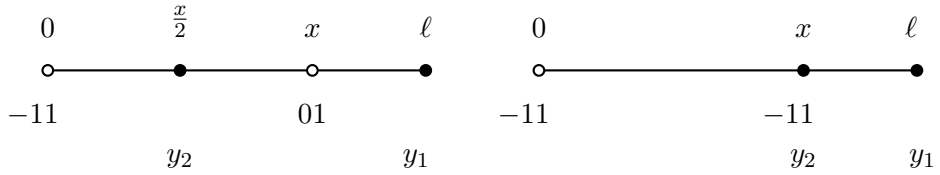
We first study the UTILITY objective, defined as $\max_y \min_i u_i(x_i, t_i, y)$ and prove that there is no 0.851-approximate deterministic or randomized truthful mechanism.

**Theorem 26.** *There is no $\alpha$-approximate deterministic truthful mechanism for the 2-facility game with $\alpha \geq 0.851$.*

*Proof.* Let us consider the instances $I$ and $I'$ depicted in Figure 6.1, where the white circles correspond to the two agents. Agent $a_1$ is located in 0 and agent $a_2$ in $x > 0$, where $x$ will be specified later in the proof. Without loss of generality we assume that $\ell = 1$. On instance $I$ the preferences of $a_1$ are $t_1 = (-1, 1)$, while $a_2$ has preferences

$t_2 = (0, 1)$. It is not hard to see that the optimal locations for the facilities are $y_1 = 1$ and $y_2 = \frac{x}{2}$ where each agent gets utility $2 - \frac{x}{2}$. The optimal locations are depicted by black circles in the figure.

On instance $I'$ agent $a_1$ has the same preferences as on instance $I$ while the preferences of agent $a_2$ are $t'_2 = (-1, 1)$. The optimal locations for the facilities in this instance are $y_1 = 1$ and $y_2 = x$ where each agent gets utility $2 - x$.




(a) Instance I (b) Instance $I'$

Figure 6.1: Example for preferences in $\{-1, 0, 1\}^2$

Instances $I$ and $I'$ show that the mechanism which locates the facilities on the optimal locations is not truthful. On instance $I$ agent $a_2$ can declare $t'_2 = (-1, 1)$ and increase his utility from $2 - \frac{x}{2}$ to 2.

Let $M$ be a truthful mechanism. First observe that $M$ will locate facility $f_1$ on 1 on instance $I$, since any other location decreases the utility of agent $a_1$ and thus does not achieve the maximum approximation guarantee on this instance e.g. if $f_1 = 1 - \epsilon$ then $u_{11} = 1 - \epsilon$ which results in smaller utility for $a_1$ than when we place $f_1$ on 1. Suppose that $M$ locates facility $f_2$ on $y_2 \leq x$ on instance $I$. If $x < y_2$, then the approximation of $M$ on instance $I$ is not optimized, since this approximation guarantee could be increased by setting $y_2 = x$.

Since $M$ is truthful, facility $f_2$ cannot be located on any $y'_2 > y_2$ on instance $I'$. If $y'_2 > y_2$, then agent $a_2$ from $I$ could declare preferences $t'_2 = (-1, 1)$ and increase his utility. We consider the following two cases concerning the location $y'_1$ in which $M$ locates facility $f_1$ on $I'$:

- $y'_1 \geq x$. Then, obviously $y_1 = 1$ since otherwise the utility of both agents in $I'$ is decreasing and thus $M$ does not achieve the maximum approximation. As a result, under $M$ agent $a_2$ gets utility at most $2 - 2x + y_2$ and thus $M$ achieves approximation $\frac{2-2x+y_2}{2-x}$ on instance $I'$. Furthermore, on instance $I$, agent $a_1$ gets utility $2 - y_2$, since as explained earlier, $M$ locates $f_1$ on 1. Thus, on this instance the approximation of $M$ is $\frac{4-2y_2}{4-x}$. Observe that the approximation guarantee of $M$ on $I$ is decreasing with $y_2$ while on $I'$ it is increasing with $y_2$. If we optimize the approximation guarantee and solve for $y_2$ we get that $y_2 = \frac{6x-2x^2}{8-3x}$. If $y'_1 > x$, then the approximation of $M$ is at most

$$\frac{4 - 2 \cdot \frac{6x-2x^2}{8-3x}}{4 - x} = \frac{4x^2 - 24x + 32}{3x^2 - 20x + 32}. \tag{6.2}$$

- If $M$ locates $f_1$ on $y'_1 < x$ on instance $I'$, then observe that there is no location $y'_2$ for $f_2$ such that both agents get utility strictly larger than 1. In this case $M$ achieves an approximation of at most

$$\frac{1}{2-x}. \tag{6.3}$$

Observe that the approximation guarantee in (6.2) increases with $x$ while in (6.3) it decreases with $x$. If we optimize on the approximation guarantee of $M$, we have to solve for $x$ the equation $-4x^3 + 29x^2 - 60x + 32 = 0$. The unique solution in $[0, 1]$ is $x = \frac{13-\sqrt{41}}{8}$. Using this value in (6.2) and (6.3) we get that any deterministic truthful mechanism on instances $I$ and $I'$ achieves approximation less than 0.851. □

The above inapproximability bound can be extended to randomized mechanisms too.

**Theorem 27.** *There is no $\alpha$-approximate randomized truthful mechanism for the 2-facility game with $\alpha \geq 0.851$.*

*Proof.* We will use again the instances from Figure 1 to prove the claim setting $x = \frac{13-\sqrt{41}}{8}$. Recall that the optimal utility is $\frac{4-x}{2}$ on instance $I$ and $2 - x$ on instance $I'$.

Let $M$ be a randomized truthful mechanism. Observe that the mechanism should locate facility $f_1$ on 1 on instance $I$ in the same way as in proof of Theorem 26; every other location for $f_1$ decreases the approximation guarantee of $M$. Suppose now that $M$ locates $f_2$ on $y \in [0, 1]$ according to the probability distribution $p(y)$. Without loss of generality we can assume that $p(y) = 0$ for every $y > x$. This is because the approximation guarantee of $M$ can be increased if we locate the facility on $x$ instead of some $y > x$. Hence, under $M$ agent $a_1$ gets utility 1 from $f_1$ and utility $\int_0^x p(y)(1-y)dy = 1 - \int_0^x p(y)ydy$ from facility $f_2$, so

$$u_1 = 2 - \int_0^x p(y)ydy$$

in total, on instance $I$. Similarly, agent $a_2$ gets utility 1 from $f_1$ and utility $1 - x + \int_0^x p(y)ydy$ from facility $f_2$ so in total

$$u_2 = 2 - x + \int_0^x p(y)ydy$$

On instance $I'$ we have to consider two cases according to the location in which $M$ places facility $f_1$. If $M$ locates $f_1$ on $y'_1 \geq x$, then without loss of generality we can assume that $f_1$ is placed on 1 since every other location decreases the utility of both agents. Suppose that $M$ places $f_1$ on 1 with some probability and $f_2$ on $y$ according to the probability distribution $\pi(y)$. We can assume that $M$ does not locate $f_2$ on $y > x$, since the utility of both agents could increase by placing it on $x$ instead. Thus agent

$a_2$ gets utility $1-x$ from facility $f_1$ and utility $1-x+\int_0^x \pi(y)ydy$ from facility $f_2$ on instance $I'$, thus in total

$$u_2' = 2 - 2x + \int_0^x p(y)ydy$$

Since $M$ is truthful we have that $\int_0^x \pi(y)ydy \leq \int_0^x p(y)ydy$. If this was not the case, agent $a_2$ could declare preferences $(-1, 1)$ and increase its utility on instance $I$. As a result the approximation guarantee of $M$ on $I'$ is at most

$$\frac{1}{2-x} \cdot \left(2 - 2x + \int_0^x p(y)ydy\right) \tag{6.4}$$

The approximation guarantee on instance $I$ will be $\frac{2}{4-x} \cdot (\min\{u_1, u_2\})$ and since $u_2' < u_2$ the best approximation guarantee of $M$ on $I$ is at most

$$\frac{2}{4-x} \cdot \left(2 - \int_0^x p(y)ydy\right) \tag{6.5}$$

$M$ achieves the best approximation on both instances when the quantities from (6.5) and (6.4) are equal. Hence, if we equalize them and solve for the integral we get that $\int_0^x p(y)ydy = \frac{6x-2x^2}{8-3x}$ and the approximation guarantee is less than 0.851 on both instances for the chosen $x$.

If the mechanism locates $f_1$ on $y_1' < x$, then on any location for $f_2$ there will be an agent with utility at most 1 and the approximation guarantee of the mechanism will be at most $\frac{1}{2-x} < 0.851$. Thus, in all possible cases the approximation of $M$ is bounded by 0.851. □

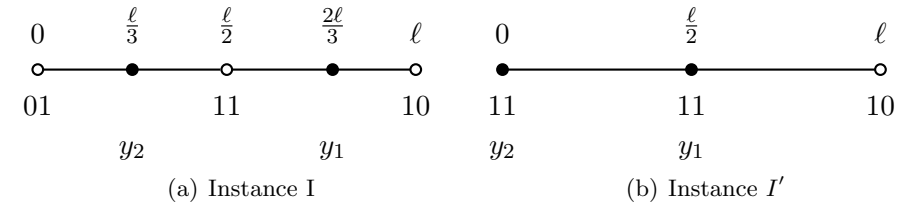
The non existence of optimal deterministic truthful mechanisms can be extended even on instances with three agents where no agent has preference $-1$ for any facility (Fig. 6.2). The same holds if no agent has preference 1 (or 0) for any facility (Fig. 6.3-Fig. 6.4). We call these instances "two-preference instances".

**Theorem 28.** *For any $k \geq 2$, there is no optimal deterministic truthful mechanism for* Utility *for the $k$-facility game even on two-preference instances with three agents and known locations.*

### 6.3.2 Happiness

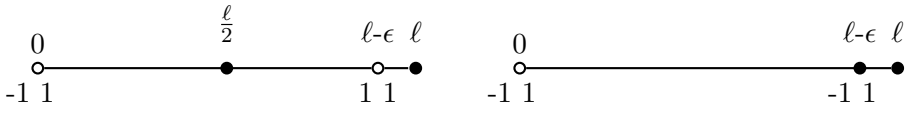
The second objective that we study is Happiness. Using similar arguments as in the Utility objective we get the following:

**Theorem 29.** *For any $k \geq 2$, there is no optimal deterministic truthful mechanism for* Happiness *for the $k$-facility game even on two-preference instances with three agents and known locations.*



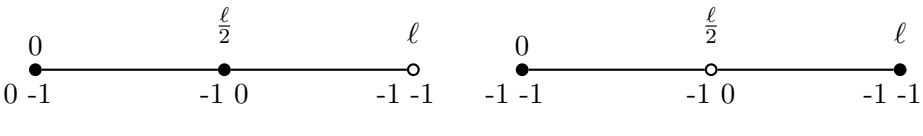

(a) Instance I (b) Instance $I'$

Figure 6.2: Example for preferences in $\{0,1\}^2$. The agent located on 0 in the instance $I$ can declare preferences $(1,1)$ and increase his utility by moving the facility $f_2$ closer to 0.




(a) Instance I (b) Instance $I'$

Figure 6.3: Example for preferences in $\{-1,1\}^2$. The agent located on $\ell-\epsilon$ in the instance $I$ can declare preferences $(-1,1)$ and increase his utility by moving the facility $f_2$ closer to $\ell-\epsilon$.




(a) Instance I (b) Instance $I'$

Figure 6.4: Example for preferences in $\{-1,0\}^2$. The agent located on 0 in the instance $I$ can declare preferences $(-1,-1)$ and increase his utility by moving the facility $f_2$ away from 0. Observe that in Instance $I'$ there are two optimal solutions ($y_1=0, y_2=\ell$ and $y_1=\ell, y_2=0$). However, this does not affect the correctness of our example assuming that the mechanism chooses a solution *deterministically*.

## 6.4 Deterministic mechanisms

In this section we propose deterministic truthful mechanisms. An initial approach would be to consider each facility independently and place it to its optimal location. As we already proved, placing one facility on its optimal position is a truthful mechanism. Furthermore, since we locate the facilities *independently* no agent has an incentive to lie. However, this mechanism achieves poor approximation if for example the agents want to be away from the facilities. Consider the case where there are $n$ agents on locations $0, \frac{2\ell}{n}, \frac{3\ell}{n}, \ldots, \frac{(n-1)\ell}{n}, \ell$ each of whom has preferences $(-1,-1)$. Observe that the optimal location for one facility is to be placed on $\frac{\ell}{n}$ since this location maximizes

the minimum distance between any agent and the facility. Thus, both facilities will be placed on the same location $\frac{\ell}{n}$. Then the agent located in 0 has utility $\frac{2\ell}{n}$, the minimum over all the agents. It is not hard to see that an optimal solution is to locate facility $f_1$ on 0 and facility $f_2$ on $\ell$ where each agent gets utility $\ell$. Hence, the mechanism that locates the facilities independently in their optimal locations is $\frac{2}{n}$-approximate.

The example above provides evidence that a mechanism with good approximation ratio should not put both facilities on the same location if there are agents who have preference $-1$ for the facilities. In the worst case the agent that is closest to the facilities might have preference $-1$ for both of them and thus get low utility. On the other hand, the facilities should not be far away from each other. This is because, in the worst case again, an agent might have preference $-1$ for the facility that is close and preference 1 for the facility that is far from his location.

Using the intuition gained from the discussion above we propose a mechanism for the 2-facility game that combines these ideas and places the facilities symmetrically away from the endpoints of the line.

**Algorithm 11:** Mechanism 1

**Input** : Utility functions $u_i(x_i, t_i, y)$ for each $i \in \mathcal{N}$ and objective function $\max_y \min_i h_i(u_i(x_i, t_i, y))$

**Output**: Locations $y = (y_1, y_2)$

1 Maximize $\min_i h_i(u_i(x_i, t_{i_1}, z) + u_i(x_i, t_{i_2}, \ell - z))$ with respect to $\ell$, for all possible $x_i \in \ell$ and all possible combinations of $(t_{i_1}, t_{i_2})$

2 Set $y_1 = z$ and $y_2 = \ell - z$

Mechanism 1 searches for the optimal value of variable $z$, under the constraint that $0 \leq z \leq \ell$, such that the two facilities are placed in $y_1 = z$ and $y_2 = \ell - z$. It solves the problem of facility location when all the information of the agents (i.e. their location and their preferences) is kept private. Since it does not use any information of the agents and thus it is truthful. It does not even require the locations of the agents since it optimizes over all possible locations. We could apply the same mechanism having the locations as input but this would not change the worst case approximation guarantee. We next use Mechanism 1 to derive approximate truthful mechanisms for Utility and Happiness.

**Case $t_i \in \{-1, 0, 1\}^2$.** Under the Happiness objective, Mechanism 1 returns the locations $y = (z, \ell - z)$ where $z = (1 - \frac{\sqrt{2}}{2})\ell$.

**Lemma 21.** *$\forall i$ the Happiness of agent $i$ is $\frac{u_i(x_i,t_i,y)}{u_i^*(x_i,t_i)} \geq 1 - \frac{\sqrt{2}}{2}$.*

*Proof.* We have to prove the claim for every possible $t_i$ and every $x_i$. We will show that the chosen $y$ is optimal for Mechanism 2 and that it achieves the desired guarantee.

Notice that if an agent has type 0 for some facility, then he gets utility at least $\ell$ no matter where the facilities are located. Moreover, we note that the maximum utility an agent can get is bounded by $2\ell$. Hence, any agent that has preference 0 for some facility gets at least 1/2 of the maximum utility.

Let $x_i$ be the location of agent $i$ and let $t_i$ be his preferences. Without loss of generality we can assume that $x_i \leq \frac{\ell}{2}$, since similar analysis can be applied for $x_i > \frac{\ell}{2}$. Furthermore, let $y = (z, \ell - z)$ be the locations of the facilities. We will consider the cases where $x_i \leq z$ and $x_i > z$. The following two tables show the utility that agent $i$ gets under $y$ when located on $x_i$ and the corresponding ratio for every case.

| $t_i$ | $u_i(x_i, t_i, y)$ | $u_i^*(x_i, t_i)$ | Ratio |
|---|---|---|---|
| 1, 1 | $\ell + 2x_i$ | $2\ell$ | $\geq 1/2$ |
| -1, 1 | $2z$ | $2\ell - x_i$ | $\geq z/\ell$ |
| 1, -1 | $2\ell - 2z$ | $2\ell - x_i$ | $\geq 1/2$ |
| -1, -1 | $\ell - 2x_i$ | $2\ell - 2x_i$ | $\geq (\ell - 2z)/(2\ell - 2z)$ |

Table 6.1: Case analysis when $x_i \leq z$.

| $t_i$ | $u_i(x_i, t_i, y)$ | $u_i^*(x_i, t_i)$ | Ratio |
|---|---|---|---|
| 1, 1 | $\ell + 2z$ | $2\ell$ | $\geq 1/2$ |
| -1, 1 | $2x_i$ | $2\ell - x_i$ | $\geq 2z/(2\ell - z)$ |
| 1, -1 | $2\ell - 2x_i$ | $2\ell - x_i$ | $\geq 2/3$ |
| -1, -1 | $\ell - 2z$ | $2\ell - 2x_i$ | $\geq (\ell - 2z)/(2\ell - 2z)$ |

Table 6.2: Case analysis when $x_i > z$.

Our goal is to find a $z \in [0, \ell]$ that maximizes the minimum ratio. Notice that $\frac{z}{\ell} \leq \frac{2z}{2\ell - z}$ for every $z \in [0, \ell/2]$. The optimal guarantee of Mechanism 2 is achieved when $\frac{z}{\ell} = \frac{\ell - 2z}{2\ell - 2z}$. If we solve for $z$, the feasible solution is $z = (1 - \frac{\sqrt{2}}{2})\ell$ and the approximation guarantee follows. □

Observe that if at least two facilities need to be located, then $\max_y \min_i u_i(x_i, t_i, y) \geq \max_y \min_i \frac{u_i(x_i, t_i, y)}{u_i^*(x_i, t_i)}$, since $u_i^*(x_i, t_i) \geq \ell$. Thus, Mechanism 1 can be used for both Utility and Happiness. Furthermore, since Mechanism 1 does not use any information of the agents, it possesses all the desirable properties like group strategy proofness and false name proofness.

**Theorem 30.** *For 2-facility games Mechanism 1 is* $(1 - \frac{\sqrt{2}}{2})$*- approximate group strategy proof and false name proof for* Utility *and* Happiness *even when both the locations and preferences are not known.*

Theorem 30 shows the sharp contrast between 1-facility and 2-facility games where both locations and preferences are private. Recall that for 1-facility games, when the

locations of the agents are private, there is no deterministic truthful mechanism with bounded approximation guarantee as shown in [49].

**Case $t_i \in \{0,1\}^2$.** In this case, under the HAPPINESS objective, Mechanism 1 returns the locations $y = (\ell/2, \ell/2)$. Observe that $u_i(x_i, t_i, y) \geq 2(\ell - |x_i - \frac{\ell}{2}|) = 2\ell - |2x_i - \ell| \geq \ell$ for every possible combination of $x_i$ and $t_i$. Notice also that $u_i^*(x_i, t_i) = 2\ell$ for every possible $(x_i, t_i)$. As a result Mechanism 1 is $\frac{1}{2}$-approximate for the 2-facility game. It is not hard to see that if there are $k$ facilities, all located on $\frac{\ell}{2}$, then $u_i(x_i, t_i, y) \geq \frac{k}{2}$, while $u_i^*(x_i, t_i) = k$. Furthermore, observe that $\sum_i u_i(x_i, t_i, y) \geq \frac{1}{2} \sum_i u_i^*(x_i, t_i)$ and thus the mechanism achieves the same approximation for social welfare too and the theorem follows:

**Theorem 31.** *If $t_i \in \{0,1\}^k$ for every $i \in \mathcal{N}$, then the Mechanism that locates every facility on $\frac{\ell}{2}$ is $\frac{1}{2}$-approximate for* UTILITY, HAPPINESS *and social welfare.*

We next show that if every agent has preferences in $\{0,1\}^2$ and the agents' locations are known, then for the mechanism that places each facility independently on its optimal location, denoted as $\text{OPT}^2$ is $\frac{3}{4}$-approximate for the 2-facility game with the UTILITY objective.

**Theorem 32.** $\text{OPT}^2$ *is $\frac{3}{4}$-approximate for* UTILITY.

*Proof.* Before we analyze the approximation guarantee of the mechanism, let us first study the locations in which the mechanism places the facilities. Since the preferences of each agent are in $\{0,1\}^2$, it is not hard to see that the optimal location for each facility is the median point between the locations of the leftmost and the rightmost agents that want to be close to the facility.

Without loss of generality we can assume that the agent with the minimum utility under $\text{OPT}^2$, denoted as $a_1$, has preferences $(1,1)$. If $t_i = (1,0)$, then the agent would have utility at least $\frac{3}{2}\ell$ since any other agent who wants to be close to the first facility is located in distance at most $\ell$ from $a_1$'s location. The maximum utility the agent can get is $2\ell$, so the mechanism is $\frac{3}{4}$-approximate.

Suppose that $a_1$ is located on $x \leq \frac{\ell}{2}$. Without loss of generality we can assume that he is located on 0, since for any other location the agent would be closer to the facilities and thus his utility would have increased. Observe that agent $a_1$ will define the locations of the facilities along with the rightmost agents. Clearly if the rightmost agent has preferences $(1,1)$, then $\text{OPT}^2$ is optimal. We can then assume that the rightmost agent, denoted as $a_{r_1}$, has preferences $(0,1)$. Observe that in the worst case $a_{r_1}$ will be located on $\ell$, since on any other location the utility of agent $a_1$ will be lower. We have to consider the two possible preferences for the second rightmost agent with preference 1 for the first facility and prove that $\text{OPT}^2$ achieves the desired approximation. We will use $a_i$ to denote this agent and $x_i$ to denote his location.

We first consider the case where agent $a_i$ has preferences $(1,1)$ and $x_i \geq \frac{\ell}{2}$. The utilities of the agents for the facilities under the locations $(y_1, y_2)$, where $y_2 \leq x_i$, are $u_1 = 2 - y_1 - y_2$, $u_i = 2 - 2x_i + y_1 + y_2$ and $u_{r_1} = 1 + y_2$. OPT$^2$ will locate the facilities on $y_1 = \frac{x_i}{2}$ and $y_2 = \frac{\ell}{2}$ and the utility of agent $a_1$ will be $u_1 = \frac{3-x_i}{2}$. The locations of the facilities that make the utilities of these three agents equal provide an upper bound on the utility that agent $a_1$ gets under the optimal solution, since any other solution would yield lower utility for at least one of these agents. The locations of the facilities that equalize the utilities of the agents are $y_1 = 2x_i - \ell$ and $y_2 = \ell - x_i$ and thus the optimal utility of agent $a_1$ is bounded by $2\ell - x_i$. Hence, OPT$^2$ is $\alpha = \frac{3-x_i}{4-2x_i} \geq \frac{3}{4}$-approximate.

In the case where $x_i < \frac{\ell}{2}$, it is not difficult to see that agent $a_1$ gets utility at least $\frac{5}{4}\ell$ under OPT$^2$. Under the optimal solution the utility of the agents is bounded by $\frac{3}{2}\ell$, since there are no locations for the facilities where both $a_1$ and $a_{r1}$ get more than $\frac{3}{2}\ell$. In this case the mechanism is $\frac{5}{6}$-approximate.

If the preferences of $a_i$ are $(1,0)$, then similar analysis can be applied. □

**Case $t_i \in \{-1,0\}^2$.** Under HAPPINESS, Mechanism 1 returns the locations $y = (0, \ell)$. For every possible combination $(x_i, t_i)$ we get that $u_i(x_i, t_i, y) \geq \ell$ and that $u_i^*(x_i, t_i) \leq 2\ell$. Again, the mechanism can be generalized for the $k$-facilities game by locating $\lceil \frac{k}{2} \rceil$ facilities on 0 and facilities $\lfloor \frac{k}{2} \rfloor$ on $\ell$.

**Theorem 33.** *If $t_i \in \{-1,0\}^k$ for every $i \in \mathcal{N}$, then Mechanism 1 is $\lfloor \frac{k}{2} \rfloor / k$-approximate for* UTILITY, HAPPINESS *and social utility.*

## 6.5 A Randomized Mechanism

We next provide a randomized universally truthful mechanism for $k$-facility games.

**Algorithm 12:** Mechanism 2

1 With probability $\frac{1}{2}$ set $y_j = 0$ for every $j \in F$
2 With probability $\frac{1}{2}$ set $y_j = \ell$ for every $j \in F$

**Theorem 34.** *Mechanism 2 is $\frac{1}{2}$-approximate for the* UTILITY, HAPPINESS *and social welfare.*

*Proof.* It is easy to see that the mechanism is universally truthful since in each case it chooses a fixed location. We will prove that every agent gets utility at least $\frac{\ell}{2}$ in expectation from every facility. Suppose that the agent $i \in \mathcal{N}$ is located on $x_i$ and has preferences $t_i$. Let us study the expected utility the agent gets from the facility $f_j$. If $t_{ij} = 1$, then the agent's utility is $\ell - x_i$ when $y_j = 0$ and $x_i$ when $y_j = \ell$. If $t_{ij} = -1$, then the agent gets utility $x_i$ if $y_j = 0$ and $\ell - x_i$ if $y_j = \ell$. If $t_{ij} = 0$, then the agent gets utility $\ell$ irrespectively to $y_j$. The agent gets utility at least $\frac{\ell}{2}$ in expectation from each

facility. As a result the agent gets utility at least $\frac{k \cdot \ell}{2}$ in expectation. The maximum utility the agent can get is $k \cdot \ell$. The happiness of the agent is at least $\frac{1}{2}$. Since the HAPPINESS objective is a lower bound on the UTILITY objective the claim follows for these two objectives. Furthermore, the expected social welfare is at least $n \cdot \frac{k \cdot \ell}{2}$ while the maximum social welfare is trivially bounded by $n \cdot k \cdot \ell$. Thus the claim follows for the social welfare too. □

## 6.6 Open problems

We studied heterogeneous $k$-facility location games on the real line for maxmin objectives. For $k \geq 2$, we derived inapproximability bounds for deterministic and randomized truthful mechanisms for the UTILITY and HAPPINESS objectives. We provided deterministic truthful mechanisms and a universally truthful mechanism that achieve constant approximation.

Many questions arise from this study. The most obvious is to derive tight bounds for truthful mechanisms. First, we leave as an open problem whether the optimal algorithm to the optimization problem is truthful. Interestingly, apart from $\text{OPT}^2$ truthful mechanisms that use the agents' locations and beat the approximation guarantee of the optimal algorithm do not seem easy to design. We conjecture that there is no truthful mechanism that locates the facilities sequentially and uses the locations of the facilities already placed in a non trivial way. Another interesting question is to study other objectives, like the envy for heterogeneous facility location games. Finally, another line of research is to extend our model on cycles, trees, or higher dimensions.

# Conclusions

In this thesis we studied approximation algorithms and truthful mechanisms for optimization problems in networks having applications in smart cities and urban planning. We presented new mechanism design models and new techniques which could be of independent interest.

More precisely our techniques in order to obtain truthfulness of the mechanisms vary depending on the problem. As we have seen for single parameter agents monotonicity is a sufficient condition for truthfulness. However this property is algorithm specific and thus we do not have a general way of proving whether an algorithm is monotone or not. It is well known that the classes of greedy algorithms (Chapter 3) and primal dual algorithms (Chapter 4) can be made monotone in general. Our novel approach was to restrict the data instances of the problem and obtain monotonicity and thus truthfulness with high probability on most of the instances. We are not aware of a similar approach in the literature and we believe that it could be of independent interest.

A general question is whether the condition of monotonicity can be added as an additional constraint in an LP and whether this way we could have monotone LP's i.e. a solution to the LP that would directly imply monotonicity.

# Bibliography


[1] Air quality in Europe - 2014 Report. European Environment Agency Report No. 5/2014 http://www.eea.europa.eu/publications/air-quality-in-europe-2014.

[2] EU Emissions Auctions. https://www.theice.com/publicdocs/How_the_Auctions_Operate.pdf.

[3] MIT Technology review. 10 breakthrough technologies that will change the world. http://www2.technologyreview.com/news/401775/10-emerging-technologies-that-will-change-the/, 2003.

[4] Emissions auctions. https://www.theice.com/emissions/auctions, 2014.

[5] State of the Air 2014 Report. *American Lung Association*, 30 April, 2014. `http://www.stateoftheair.org/2014/key-findings/`.

[6] Ajit Agrawal, Philip Klein, and R. Ravi. When trees collide: An approximation algorithm for the generalized steiner problem on networks. *SIAM Journal on Computing*, 24(3):440–456, 1995.

[7] Ian F. Akyildiz, Weilian Su, Yogesh Sankarasubramaniam, and Erdal Cayirci. Wireless sensor networks: a survey. *Computer networks*, 38(4):393–422, 2002.

[8] Noga Alon, Michal Feldman, Ariel D Procaccia, and Moshe Tennenholtz. Strategyproof approximation of the minimax on networks. *Mathematics of Operations Research*, 35(3):513–526, 2010.

[9] Giuseppe Anastasi, Marco Conti, Mario Di Francesco, and Andrea Passarella. Energy conservation in wireless sensor networks: A survey. *Ad hoc networks*, 7(3):537–568, 2009.

[10] Eleftherios Anastasiadis, Xiaotie Deng, Piotr Krysta, Minming Li, Han Qiao, and Jinshan Zhang. Network pollution games. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, pages 23–31. International Foundation for Autonomous Agents and Multiagent Systems, 2016.

[11] Luzi Anderegg and Stephan Eidenbenz. Ad hoc-vcg: a truthful and cost-efficient routing protocol for mobile ad hoc networks with selfish agents. In *Proceedings of the 9th annual international conference on Mobile computing and networking*, pages 245–259. ACM, 2003.

[12] K. I. Appel and W. Haken. *Every planar map is four colorable*, volume 98. American mathematical society Providence, RI, 1989.

[13] Th. Arampatzis, John Lygeros, and Stamatis Manesis. A survey of applications of wireless sensors and wireless sensor networks. In *Proceedings of the 2005 IEEE International Symposium on, Mediterrean Conference on Control and Automation Intelligent Control, 2005.*, pages 719–724. IEEE, 2005.

[14] Aaron Archer and Éva Tardos. Truthful mechanisms for one-parameter agents. In *Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on*, pages 482–491. IEEE, 2001.

[15] Brenda S. Baker. Approximation algorithms for np-complete problems on planar graphs. *Journal of the ACM (JACM)*, 41(1):153–180, 1994.

[16] David Banister. The sustainable mobility paradigm. *Transport policy*, 15(2):73–80, 2008.

[17] Nikhil Bansal, Nitish Korula, Viswanath Nagarajan, and Aravind Srinivasan. Solving packing integer programs via randomized rounding with alterations. *Theory of Computing*, 8(1):533–565, 2012.

[18] Anna V Belitskaya. Network game of pollution cost reduction. *Contributions to Game Theory and Management*, 6(0):24–34, 2013.

[19] Piotr Berman, Gruia Calinescu, Chintan Shah, and Alexander Zelikovsky. Power efficient monitoring management in sensor networks. In *Wireless Communications and Networking Conference, 2004. WCNC. 2004 IEEE*, volume 4, pages 2329–2334. IEEE, 2004.

[20] Marshall Bern and Paul Plassmann. The steiner problem with edge lengths 1 and 2. *Information Processing Letters*, 32(4):171–176, 1989.

[21] P. Briest, P. Krysta, and B. Vöcking. Approximation techniques for utilitarian mechanism design. *SIAM J. Comput.*, 40(6):1587–1622, 2011.

[22] Sonja Buchegger and Jean-Yves Le Boudec. Performance analysis of the confidant protocol. In *Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing*, pages 226–236. ACM, 2002.

[23] Levente Buttyán and Jean-Pierre Hubaux. Stimulating cooperation in self-organizing mobile ad hoc networks. *Mobile Networks and Applications*, 8(5):579–592, 2003.

[24] Jarosław Byrka, Fabrizio Grandoni, Thomas Rothvoss, and Laura Sanità. Steiner tree approximation via iterative randomized rounding. *Journal of the ACM (JACM)*, 60(1):6, 2013.

[25] Qingpeng Cai, Aris Filos-Ratsikas, and Pingzhong Tang. Facility location with minimax envy. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pages 137–143, 2016.

[26] Alberto Caprara, Hans Kellerer, Ulrich Pferschy, and David Pisinger. Approximation algorithms for knapsack problems with cardinality constraints. *European Journal of Operational Research*, 123(2):333–345, 2000.

[27] Mihaela Cardei, My T. Thai, Yingshu Li, and Weili Wu. Energy-efficient target coverage in wireless sensor networks. In *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, volume 3, pages 1976–1984. IEEE, 2005.

[28] Chi-Kin Chau, Khaled Elbassioni, and Majid Khonji. Truthful mechanisms for combinatorial ac electric power allocation. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, pages 1005–1012. International Foundation for Autonomous Agents and Multiagent Systems, 2014.

[29] Xiuzhen Cheng and Ding-Zhu Du. *Steiner trees in industry*, volume 11. Springer Science & Business Media, 2013.

[30] Yukun Cheng, Wei Yu, and Guochuan Zhang. Mechanisms for obnoxious facility game on a path. In *Combinatorial Optimization and Applications*, pages 262–271. Springer, 2011.

[31] Yukun Cheng, Wei Yu, and Guochuan Zhang. Strategy-proof approximation mechanisms for an obnoxious facility game on networks. *Theoretical Computer Science*, 497:154–163, 2013.

[32] Miroslav Chlebík and Janka Chlebíková. The steiner tree problem on graphs: Inapproximability results. *Theoretical Computer Science*, 406(3):207–214, 2008.

[33] Edward H Clarke. Multipart pricing of public goods. *Public choice*, 11(1):17–33, 1971.

[34] Klaus Conrad and Jianmin Wang. On the design of incentive mechanisms in environmental policy. *Environmental and Resource Economics*, 3(3):245–262, 1993.

[35] George B. Dantzig. Discrete-variable extremum problems. *Operations research*, 5(2):266–288, 1957.

[36] George Bernard Dantzig, Lester Randolph Ford Jr, and Delbert Ray Fulkerson. A primal–dual algorithm for linear programs. Technical report, DTIC Document, 1956.

[37] Partha Dasgupta, Peter Hammond, and Eric Maskin. On imperfect information and optimal pollution control. *The Review of Economic Studies*, 47(5):857–860, 1980.

[38] Sanjoy Dasgupta, Christos H. Papadimitriou, and Umesh Vazirani. *Algorithms*. McGraw-Hill, Inc., 2006.

[39] Isabel Dietrich and Falko Dressler. On the lifetime of wireless sensor networks. *ACM Transactions on Sensor Networks (TOSN)*, 5(1):5, 2009.

[40] Edsger W. Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.

[41] Ling Ding, Weili Wu, James Willson, Lidong Wu, Zaixin Lu, and Wonjun Lee. Constant-approximation for target coverage problem in wireless sensor networks. In *INFOCOM, 2012 Proceedings IEEE*, pages 1584–1592. IEEE, 2012.

[42] S. Dobzinski and N. Nisan. Mechanisms for multi-unit auctions. In *Proc. of the 8th ACM conference on Electronic commerce*, pages 346–351. ACM, 2007.

[43] Elad Dokow, Michal Feldman, Reshef Meir, and Ilan Nehama. Mechanism design on discrete lines and cycles. In *Proceedings of the 13th ACM Conference on Electronic Commerce*, pages 423–440. ACM, 2012.

[44] Baomin Dong, Debing Ni, and Yuntong Wang. Sharing a polluted river network. *Environmental and Resource Economics*, 53(3):367–387, 2012.

[45] Sarah Dorner, Jie Shi, and David Swayne. Multi-objective modelling and decision support using a bayesian network approximation to a non-point source pollution model. *Environmental Modelling & Software*, 22(2):211–222, 2007.

[46] Shaddin Dughmi and Tim Roughgarden. Black-box randomized reductions in algorithmic mechanism design. *SIAM Journal on Computing*, 43(1):312–336, 2014.

[47] Rafael Falcon, Xu Li, and Amiya Nayak. Carrier-based coverage augmentation in wireless sensor and robot networks. In *2010 IEEE 30th International Conference on Distributed Computing Systems Workshops*, pages 234–239. IEEE, 2010.

[48] Joseph Farrell. Information and the coase theorem. *The Journal of Economic Perspectives*, 1(2):113–129, 1987.

[49] Itai Feigenbaum and Jay Sethuraman. Strategyproof mechanisms for one-dimensional hybrid and obnoxious facility location models. In *Workshops at the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[50] Itai Feigenbaum, Jay Sethuraman, and Chun Ye. Approximately optimal mechanisms for strategyproof facility location: Minimizing $l_p$ norm of costs. *CoRR*, abs/1305.2446, 2013.

[51] Michal Feldman and Yoav Wilf. Strategyproof facility location and the least squares objective. In *Proceedings of the fourteenth ACM conference on Electronic commerce*, pages 873–890. ACM, 2013.

[52] Aris Filos-Ratsikas, Minming Li, Jie Zhang, and Qiang Zhang. Facility location with double-peaked preferences. In Blai Bonet and Sven Koenig, editors, *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA.*, pages 893–899. AAAI Press, 2015.

[53] Dimitris Fotakis and Christos Tzamos. Winner-imposing strategyproof mechanisms for multiple facility location games. In *Internet and Network Economics*, pages 234–245. Springer, 2010.

[54] Dimitris Fotakis and Christos Tzamos. On the power of deterministic mechanisms for facility location games. *ACM Transactions on Economics and Computation*, 2(4):15, 2014.

[55] Don Fullerton and Sarah E. West. Can taxes on cars and on gasoline mimic an unavailable tax on emissions? *Journal of Environmental Economics and Management*, 43(1):135–157, 2002.

[56] M. R. Garey and D. S. Johnson. The rectilinear steiner tree problem is np-complete. *SIAM Journal on Applied Mathematics*, 32(4):826–834, 1977.

[57] Naveen Garg and Jochen Koenemann. Faster and simpler algorithms for multicommodity flow and other fractional packing problems. *SIAM Journal on Computing*, 37(2):630–652, 2007.

[58] Leonard P. Gianessi, Henry M. Peskin, and GK Young. Analysis of national water pollution control policies: 1. a national network model. *Water Resources Research*, 17(4):796–802, 1981.

[59] Allan Gibbard. Manipulation of voting schemes: a general result. *Econometrica: journal of the Econometric Society*, pages 587–601, 1973.

[60] Michel X. Goemans, Andrew V. Goldberg, Serge A. Plotkin, David B Shmoys, Eva Tardos, and David P. Williamson. Improved approximation algorithms for network design problems. In *SODA*, volume 94, pages 223–232, 1994.

[61] Michel X Goemans, Neil Olver, Thomas Rothvoß, and Rico Zenklusen. Matroids and integrality gaps for hypergraphic steiner tree relaxations. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 1161–1176. ACM, 2012.

[62] Michel X. Goemans and David P. Williamson. A general approximation technique for constrained forest problems. *SIAM Journal on Computing*, 24(2):296–317, 1995.

[63] Fabrizio Grandoni, Piotr Krysta, Stefano Leonardi, and Carmine Ventre. Utilitarian mechanism design for multiobjective optimization. *SIAM Journal on Computing*, 43(4):1263–1290, 2014.

[64] M. Grötschel, L. Lovász, and A. Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1(2):169–197, 1981.

[65] Theodore Groves. Incentives in teams. *Econometrica: Journal of the Econometric Society*, pages 617–631, 1973.

[66] Johan Hastad. Clique is hard to approximate within n 1-&epsiv. In *Foundations of Computer Science, 1996. Proceedings., 37th Annual Symposium on*, pages 627–636. IEEE, 1996.

[67] Dorit S. Hochbaum. *Approximation algorithms for NP-hard problems*. PWS Publishing Co., 1996.

[68] Ron Holzman, Noa Kfir-Dahav, Dov Monderer, and Moshe Tennenholtz. Bundling equilibrium in combinatorial auctions. *Games and Economic Behavior*, 47(1):104–123, 2004.

[69] Leonid Hurwicz. The design of mechanisms for resource allocation. *The American Economic Review*, 63(2):1–30, 1973.

[70] Frank K. Hwang, Dana S. Richards, and Pawel Winter. *The Steiner tree problem*, volume 53. Elsevier, 1992.

[71] Robert Innes. Regulating automobile pollution under certainty, competition, and imperfect information. *Journal of Environmental Economics and Management*, 31(2):219–239, 1996.

[72] Athirai Aravazhi Irissappane, Jie Zhang, Frans A. Oliehoek, and Partha Sarathi Dutta. Secure routing in wireless sensor networks via pomdps. In *IJCAI*, pages 2617–2623, 2015.

[73] Kamal Jain. A factor 2 approximation algorithm for the generalized steiner network problem. *Combinatorica*, 21(1):39–60, 2001.

[74] Dionisis Kandris, Panagiotis Tsioumas, Anthony Tzes, George Nikolakopoulos, and Dimitrios D. Vergados. Power conservation through energy efficient routing in wireless sensor networks. *Sensors*, 9(9):7320–7342, 2009.

[75] Larry Karp and John Livernois. Using automatic tax changes to control pollution emissions. *Journal of Environmental Economics and Management*, 27(1):38–48, 1994.

[76] Marek Karpinski and Alexander Zelikovsky. New approximation algorithms for the steiner tree problems. *Journal of Combinatorial Optimization*, 1(1):47–65, 1997.

[77] I. Katsikarelis. Computing bounded-width tree and branch decompositions of k-outerplanar graphs. *arXiv preprint arXiv:1301.5896*, 2013.

[78] James Keirstead and Nilay Shah. The changing role of optimization in urban planning. In *Optimization, simulation, and control*, pages 175–193. Springer, 2013.

[79] Subhash Khot. On the power of unique 2-prover 1-round games. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 767–775. ACM, 2002.

[80] Subhash Khot and Oded Regev. Vertex cover might be hard to approximate to within 2- $\varepsilon$. *Journal of Computer and System Sciences*, 74(3):335–349, 2008.

[81] Subhash Khot and Nisheeth K. Vishnoi. On the unique games conjecture. In *Annual Symposium on Foundations of Computer Science*, volume 46, page 3. IEEE COMPUTER SOCIETY PRESS, 2005.

[82] Deniz Kilinc, Mustafa Ozger, and Ozgur B. Akan. On the maximum coverage area of wireless networked control systems under stability and cost-efficiency constraints. In *2013 IEEE Global Communications Conference (GLOBECOM)*, pages 274–279. IEEE, 2013.

[83] Jae-Cheol Kim and Ki-Bok Chang. An optimal tax/subsidy for output and pollution control under asymmetric information in oligopoly markets. *Journal of Regulatory Economics*, 5(2):183–197, 1993.

[84] Ton Kloks. *Treewidth: computations and approximations*, volume 842. Springer Science & Business Media, 1994.

[85] Jochen Könemann, David Pritchard, and Kunlun Tan. A partition-based relaxation for steiner trees. *Mathematical Programming*, 127(2):345–370, 2011.

[86] Bernhard Korte, HJ Prömel, and Angelika Steger. Steiner trees in vlsi-layout. *Paths, Flows, and VLSI-layout*, 9:185–214, 1990.

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

[88] Piotr Krysta, Orestis Telelis, and Carmine Ventre. Mechanisms for multi-unit combinatorial auctions with a few distinct goods. *Journal of Artificial Intelligence Research*, 53:721–744, 2015.

[89] Harold W. Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

[90] Ariel Kulik and Hadas Shachnai. On lagrangian relaxation and subset selection problems. In *International Workshop on Approximation and Online Algorithms*, pages 160–173. Springer, 2008.

[91] Ariel Kulik and Hadas Shachnai. There is no EPTAS for two-dimensional knapsack. *Information Processing Letters*, 110(16):707–710, 2010.

[92] Evan Kwerel. To tell the truth: Imperfect information and optimal pollution control. *The Review of Economic Studies*, pages 595–601, 1977.

[93] Ron Lavi, Ahuva Mu'Alem, and Noam Nisan. Towards a characterization of truthful combinatorial auctions. In *Foundations of Computer Science, 2003. Proceedings. 44th Annual IEEE Symposium on*, pages 574–583. IEEE, 2003.

[94] Ron Lavi and Chaitanya Swamy. Truthful mechanism design for multi-dimensional scheduling via cycle monotonicity. In *Proceedings of the 8th ACM conference on Electronic commerce*, pages 252–261. ACM, 2007.

[95] Ron Lavi and Chaitanya Swamy. Truthful and near-optimal mechanism design via linear programming. *Journal of the ACM (JACM)*, 58(6):25, 2011.

[96] Daniel Lehmann, Liadan Ita Oćallaghan, and Yoav Shoham. Truth revelation in approximately efficient combinatorial auctions. *Journal of the ACM (JACM)*, 49(5):577–602, 2002.

[97] Jian Li and Yifei Jin. A ptas for the weighted unit disk cover problem. In *International Colloquium on Automata, Languages, and Programming*, pages 898–909. Springer, 2015.

[98] Pinyan Lu, Xiaorui Sun, Yajun Wang, and Zeyuan Allen Zhu. Asymptotically optimal strategy-proof mechanisms for two-facility games. In *Proceedings of the 11th ACM conference on Electronic commerce*, pages 315–324. ACM, 2010.

[99] Pinyan Lu, Yajun Wang, and Yuan Zhou. Tighter bounds for facility games. In *Internet and Network Economics*, pages 137–148. Springer, 2009.

[100] Thomas L. Magnanti and Richard T. Wong. Network design and transportation planning: Models and algorithms. *Transportation science*, 18(1):1–55, 1984.

[101] Sultan Javed Majeed and Hasan Mahmood. Topological importance as an incentive to cooperate in mobile ad hoc networks: A game theoretic analysis. In *Advanced Communication Technology (ICACT), 2012 14th International Conference on*, pages 46–51. IEEE, 2012.

[102] Sergio Marti, Thomas J. Giuli, Kevin Lai, and Mary Baker. Mitigating routing misbehavior in mobile ad hoc networks. In *Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 255–265. ACM, 2000.

[103] Ross McKitrick. A cournot mechanism for pollution control under asymmetric information. *Environmental and Resource Economics*, 14(3):353–363, 1999.

[104] Lili Mei, Minming Li, Deshi Ye, and Guochuan Zhang. Strategy-proof mechanism design for facility location games: Revisited (extended abstract). In *AAMAS*, pages 1463–1464, 2016.

[105] Pietro Michiardi and Refik Molva. Core: a collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks. In *Advanced communications and multimedia security*, pages 107–121. Springer, 2002.

[106] W David Montgomery. Markets in licenses and efficient pollution control programs. *Journal of economic theory*, 5(3):395–418, 1972.

[107] Ahuva Mu'Alem and Noam Nisan. Truthful approximation mechanisms for restricted combinatorial auctions. *Games and Economic Behavior*, 64(2):612–631, 2008.

[108] Raymond Mulligan and Habib M Ammari. Coverage in wireless sensor networks: a survey. *Network Protocols and Algorithms*, 2(2):27–53, 2010.

[109] Debing Ni and Yuntong Wang. Sharing a polluted river. *Games and Economic Behavior*, 60(1):176–186, 2007.

[110] Stefanos A. Nikolidakis, Dionisis Kandris, Dimitrios D. Vergados, and Christos Douligeris. Energy efficient routing in wireless sensor networks through balanced clustering. *Algorithms*, 6(1):29–42, 2013.

[111] Noam Nisan and Amir Ronen. Algorithmic mechanism design. In *Proceedings of the thirty-first annual ACM symposium on Theory of computing*, pages 129–140. ACM, 1999.

[112] Noam Nisan and Amir Ronen. Computationally feasible VCG mechanisms. *J. Artif. Intell. Res.(JAIR)*, 29:19–47, 2007.

[113] Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay V Vazirani. *Algorithmic game theory*, volume 1. Cambridge University Press Cambridge, 2007.

[114] Christos Papadimitriou. Algorithms, games, and the internet. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 749–753. ACM, 2001.

[115] Christos H. Papadimitriou and Kenneth Steiglitz. *Combinatorial optimization: algorithms and complexity*. Courier Corporation, 1982.

[116] Leon Petrosjan and Georges Zaccour. Time-consistent shapley value allocation of pollution cost reduction. *Journal of economic dynamics and control*, 27(3):381–398, 2003.

[117] David Pritchard and Deeparnab Chakrabarty. Approximability of sparse integer programs. *Algorithmica*, 61(1):75–93, 2011.

[118] Ariel D. Procaccia and Moshe Tennenholtz. Approximate mechanism design without money. In *Proceedings of the 10th ACM conference on Electronic commerce*, pages 177–186. ACM, 2009.

[119] Hans Jürgen Prömel and Angelika Steger. A new approximation algorithm for the steiner tree problem with performance ratio 5/3. *Journal of Algorithms*, 36(1):89–101, 2000.

[120] Sridhar Rajagopalan and Vijay V. Vazirani. On the bidirected cut relaxation for the metric steiner tree problem. In *SODA*, volume 99, pages 742–751, 1999.

[121] Neil Robertson, Daniel Sanders, Paul Seymour, and Robin Thomas. A new proof of the four-colour theorem. *Electronic Research Announcements of the American Mathematical Society*, 2(1):17–25, 1996.

[122] Gabriel Robins and Alexander Zelikovsky. Tighter bounds for graph steiner tree approximation. *SIAM Journal on Discrete Mathematics*, 19(1):122–134, 2005.

[123] Tim Roughgarden and Éva Tardos. How bad is selfish routing? *Journal of the ACM (JACM)*, 49(2):236–259, 2002.

[124] Mark Allen Satterthwaite. Strategy-proofness and arrow's conditions: Existence and correspondence theorems for voting procedures and social welfare functions. *Journal of economic theory*, 10(2):187–217, 1975.

[125] Thomas J. Schaefer. The complexity of satisfiability problems. In *Proceedings of the tenth annual ACM symposium on Theory of computing*, pages 216–226. ACM, 1978.

[126] Kathleen Segerson. Uncertainty and incentives for nonpoint pollution control. *Journal of environmental economics and management*, 15(1):87–98, 1988.

[127] Paolo Serafino and Carmine Ventre. Heterogeneous facility location without money. *Theoretical Computer Science*, 636:27–46, 2016.

[128] Raj Mohan Singh and Bithin Datta. Artificial neural network modeling for identification of unknown pollution sources in groundwater with partially missing concentration observation data. *Water resources management*, 21(3):557–572, 2007.

[129] Daniel F. Spulber. Optimal environmental regulation under asymmetric information. *Journal of Public Economics*, 35(2):163–181, 1988.

[130] Vikram Srinivasan, Pavan Nuggehalli, Carla-Fabiana Chiasserini, and Ramesh R. Rao. Cooperation in wireless ad hoc networks. In *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, volume 2, pages 808–817. IEEE, 2003.

[131] Robert N. Stavins. Policy instruments for climate change: how can national governments address a global problem. *U. Chi. Legal F.*, page 293, 1997.

[132] Robert N. Stavins. Experience with market-based environmental policy instruments. *Handbook of environmental economics*, 1:355–435, 2003.

[133] Anita Talberg and Kai Swoboda. Emissions trading schemes around the world. *Parliament of Australia, viewed*, 2013.

[134] Fouad Tobagi and Leonard Kleinrock. Packet switching in radio channels: Part ii-the hidden terminal problem in carrier sense multiple-access and the busy-tone solution. *IEEE Transactions on communications*, 23(12):1417–1433, 1975.

[135] Arturo Trujillo-Ventura and J. Hugh Ellis. Multiobjective air pollution monitoring network design. *Atmospheric Environment. Part A. General Topics*, 25(2):469–479, 1991.

[136] A. Urpi, M. Bonuccelli, and Silvia Giordano. Modelling cooperation in mobile ad hoc networks: a formal description of selfishness. In *WiOpt'03: Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, pages 10–pages, 2003.

[137] Vijay V. Vazirani. *Approximation algorithms*. Springer Science & Business Media, 2013.

[138] Carlo Vercellis. A probabilistic analysis of the set covering problem. *Annals of Operations Research*, 1(3):255–271, 1984.

[139] Matheus P. Viana, Emanuele Strano, Patricia Bordin, and Marc Barthelemy. The simplicity of planar networks. *Scientific reports*, 3, Article number 3495, 2013.

[140] William Vickrey. Counterspeculation, auctions, and competitive sealed tenders. *The Journal of finance*, 16(1):8–37, 1961.

[141] Yevgeniy Vorobeychik and Yagil Engel. Average-case analysis of mechanism design with approximate resource allocation algorithms. In *International Workshop on Internet and Network Economics*, pages 571–578. Springer, 2010.

[142] Charles J. Vörösmarty, Peter B. McIntyre, Mark O. Gessner, David Dudgeon, Alexander Prusevich, Pamela Green, Stanley Glidden, Stuart E Bunn, Caroline A. Sullivan, C Reidy Liermann, et al. Global threats to human water security and river biodiversity. *Nature*, 467(7315):555–561, 2010.

[143] David P. Williamson, Michel X. Goemans, Milena Mihail, and Vijay V. Vazirani. A primal-dual approximation algorithm for generalized steiner network problems. *Combinatorica*, 15(3):435–454, 1995.

[144] Deshi Ye, Lili Mei, and Yong Zhang. Strategy-proof mechanism for obnoxious facility location on a line. In *Computing and Combinatorics*, pages 45–56. Springer, 2015.

[145] Jennifer Yick, Biswanath Mukherjee, and Dipak Ghosal. Wireless sensor network survey. *Computer networks*, 52(12):2292–2330, 2008.

[146] Alexander Z. Zelikovsky. An 11/6-approximation algorithm for the network steiner problem. *Algorithmica*, 9(5):463–470, 1993.

[147] Qiang Zhang and Minming Li. Strategyproof mechanism design for facility location games with weighted agents on a line. *Journal of Combinatorial Optimization*, 28(4):756–773, 2014.

[148] Sheng Zhong, Jiang Chen, and Yang Richard Yang. Sprite: A simple, cheat-proof, credit-based system for mobile ad-hoc networks. In *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, volume 3, pages 1987–1997. IEEE, 2003.

[149] Chuan Zhu, Chunlin Zheng, Lei Shu, and Guangjie Han. A survey on coverage and connectivity issues in wireless sensor networks. *Journal of Network and Computer Applications*, 35(2):619–632, 2012.

[150] Yujie Zhu and Raghupathy Sivakumar. Challenges: communication through silence in wireless sensor networks. In *Proceedings of the 11th annual international conference on Mobile computing and networking*, pages 140–147. ACM, 2005.

[151] Shaokun Zou and Minming Li. Facility location games with dual preference. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, pages 615–623. International Foundation for Autonomous Agents and Multiagent Systems, 2015.