Università del Salento

Dipartimento di Matematica "E. De Giorgi"

Multicast problems in telecommunication networks

Tesi presentata per il conseguimento del titolo di Dottore di Ricerca

Supervisore
Prof. Paolo Nobili

Dottoranda
Dott.ssa Valeria Leggieri

Coordinatore del Dottorato di Ricerca
Prof. Domenico Perrone

Dottorato di Ricerca in Matematica – XIX Ciclo

Settore Disciplinare: Ricerca Operativa (MAT/09)

Riassunto

Questa dissertazione è dedicata principalmente allo studio di uno tipo di problema di Ottimizzazione Combinatoria, il problema di Multicast e di alcune sue varianti. Dato un grafo $G = (V, E)$ e un sottoinsieme R di elementi dell'insieme dei nodi V, il problema di Multicast consiste nel determinare un sottoinsieme connesso T dell'insieme degli archi che ricopra tutti i nodi di R (usando eventulamente anche dei nodi nel complementare di R) e che minimizzi una opportuna funzione obiettivo che rappresenta il costo di connessione.

La maggior parte dei risultati presentati riguarda dei particolari tipi di reti, le reti Ad-Hoc senza fili. I nodi di queste reti sono apparecchi elettronici (sensori, computer, radio trasmettitori etc.) che inviano dei segnali radio senza utilizzare delle infrastrutture fisse e senza avere un'amministrazione centralizzata. Il problema di Multicast, in questo caso, è quello di assegnare una potenza agli apparecchi della rete in modo che gli elementi di un insieme R ricevano i segnali inoltrati da un particolare nodo della rete detto sorgente e che la somma delle potenze assegnate sia minima. Una delle caratteristiche di una trasmissione radio consiste nel fatto che una qualsiasi trasmissione può essere captata da tutti gli apparecchi che si trovano nel raggio di trasmissione dell'emittente, e quindi al contrario delle reti con fili, pagando il costo di un unico arco e dunque di una sola trasmissione è

possibile raggiungere e connettere più nodi nello stesso tempo.

In particolare, i principali contributi di tale dissertazione possono essere sintetizzati come segue:

- Si propone una formulazione di Set Covering per il problema di minima potenza in reti Ad-Hoc senza fili che nel confronto con alcune delle formulazioni presenti nella letteratura risulta avere il migliore rilassamento lineare e si propongono due metodi di risoluzione del problema che sfruttano una possibile riduzione del problema stesso sulla base delle proprietà del modello di Set Covering.

- Si presentano, inoltre, due euristiche per generare delle disuguaglianze valide appartenenti alla prima chiusura di Chvàtal del politopo di Set Covering così da rafforzarne il rilassamento lineare. Nel caso di reti senza fili con un limitato numero di nodi, si confrontano il valore ottimo e i tempi di soluzione del rilassamento lineare del problema con l'aggiunta dei vincoli generati dalle euristiche, con il valore ottimo e i tempi di esecuzione del rilassamento lineare del problema con l'aggiunta dei vincoli della prima chiusura di Chvàtal del politopo di Set Covering.

- Inoltre, viene proposta una originale variante del problema di Multicast in cui agli apparecchi elettronici è assegnata una probabilità di fallimento nella ricezione e trasmissione dei messaggi. Sono infatti presentate nella dissertazione tre formulazioni di programmazione lineare intera mista che modellizzano la richiesta di connessione dell'insieme R, formato da tutti i nodi della rete eccetto la sorgente, con un livello di affidabilità fissato. La soluzione ottima di questo problema non solo fornisce una connessione, ma in realtà permette di individuare una connessione robusta di tutti i nodi della rete con la sorgente.

- Infine, un' altra variante del problema di Multicast, considerata nella dissertazione, è quella in cui non si richiede solo una connessione dell'insieme R con la sorgente con il minimo costo (o peso) totale, ma assegnando a ogni arco del grafo anche dei tempi di percorrenza dell'arco stesso, si affronta il problema di trovare un albero di costo minimo che connetta i nodi di R con la sorgente con l'ulteriore vincolo che i terminali in R siano raggiunti entro un tempo limite prestabilito. Per questo problema, nel caso di reti con fili, sono proposte quattro formulazioni di programmazione lineare intera mista insieme a delle tecniche di preprocessamento del grafo per ridurne il numero sia di nodi che di archi. Le quattro formulazioni sono state utilizzate per risolvere problemi di Steiner Tree proposti nella libreria SteinLib [48] con i tempi di percorrenza sugli archi generati in modo casuale in maniera sia correlata che non correlata con i costi degli archi.

Classificazione AMS 2000: *Primaria:* 90C27, 90C11 *Secondaria:* 90C35, 90B10, 68R10.

Parole chiave: Ottimizzazione Combinatoria; problema di Multicast; Programmazione lineare intera mista; problema di Set Covering; reti Ad-Hoc senza fili; problema di Broadcast probabilistico; problema di Steiner Tree con vincoli di ritardo; Preprocessamento.

Abstract

This dissertation is devoted, mainly, to a specific class of Combinatorial Optimization problems: the Multicast problem and some related variants. Specifically, given a graph $G = (V, E)$ and a subset R of elements of the set of the nodes V, the Multicast problem consists in determining a connected subset T of the set of the edges, whose elements connect all the nodes belonging to R (using possibly some nodes not in R) in such a way that an objective function representing the cost of the connection is minimized.

The major part of the presented results is devoted to a particular type of network, the Ad-Hoc wireless network. The nodes of these networks are electronic devices (sensors, computers, radio transmitters etc.) which transmit radio signals without using a fix infrastructure and without a centralized administration. The Multicast problem, in this case, consists in assigning a power to the devices of the network in such a way that the elements belonging to the set R receive the messages originated from a particular node of the network, called source, and the total amount of assigned power is minimized. One of the peculiarity of a radio transmission is that every signal forwarded by a node can be received by all the nodes placed in the transmission range of the communication and, thus, contrary to the wired network case, performing only one transmission and so paying the cost of a single arc, it is possible to connect several nodes at the same time.

In particular, we can summarize the main contributions of in this dissertation as follows:

- We propose a Set Covering formulation for the Minimum Power Multicast problem in wireless Ad-Hoc networks, which results to be stronger than certain formulations presented in literature and we propose two exact methods for solving the problem making use of a possible reduction of the size of the problem which is based on the properties of the Set Covering problem.

- We present also two heuristics for generating valid inequalities of the first Chvátal closure of the Set Covering polytope and, thus, for strengthening the linear relaxation of the formulation of the Minimum Power Multicast problem. In the case of wireless networks with a small number of nodes, we compare the optimum value and the computational time for solving the linear relaxation of the problems with the addition of the constraints generated by the heuristics with the optimum value and the computational time occuring for solving the linear relaxation of the problems with all the cuts belonging to the first Chvátal closure of the Set Covering polytope.

- Moreover, an innovative variant of the Multicast problem is considered, in which to the devices of a wireless network is assigned a probability of failure in the reception and transmission of the messages. Indeed, we present here three mixed integer programming formulations for the problem of connecting the source with all the other nodes of the network (R is the set of all the nodes of the network except the source) with a reliability threshold. The solution, hence, not only guarantees a connection, but in fact gives a robust connection of the elements of the network with the source.

- Finally, another variant of the Multicast problem, considered in the dissertation, is the problem of finding not only a connection of a subset R with a source with the minimum total cost (or weight) but, assigned to each arc a delay, we deal with the problem of finding a minimum cost arborescence connecting the source with the elements of R with additional time limit constraints. For this problem, in case of wired networks, four mixed integer programming formulations are proposed together with a preprocessing procedure for reducing the size of the problem. The four formulations with the preprocessing procedure have been tested on some Steiner Tree problems proposed in the SteinLib library [48], where the delay on the arcs have been randomly generated in a correlated and non-correlated way with respect to the costs of the arcs.

Mathematics Subject Classification 2000: *Primary:* 90C27, 90C11 *Secondary:* 90C35, 90B10, 68R10.

Keywords: Combinatorial Optimization; Multicast problem; Mixed integer programming formulations; Set Covering problem; Wireless Ad-Hoc networks; Probabilistic Broadcast problem; Delay-constrained Steiner Tree problem; Preprocessing.

Contents

List of Figures

List of Tables

Introduction

The Multicast problem is a Combinatorial Optimization problem whose aim is to connect by wired or wireless links a set of required vertices at the minimum cost. There are several contexts in which such a problem finds its application, one of these is the Multicast routing in communication [66]. The main objective, in this case, is to ensure that an information generated by a node of a network, called source, reaches a multicast group which is a set of selected elements of the network, minimizing the usage of resources, in particular the energy or the power employed in the communication.

The major part of the presented results is devoted to a particular type of network, the Ad-Hoc wireless network (see e.g. [60], [72], [84]). The vertices of these networks are electronic devices (sensors, computers, radio transmitters etc.) which transmit radio signals without using a fix infrastructure and without a centralized administration. This type of network is expected to be used in several fields going from natural disasters to battlefields, where the existing infrastructures are damaged or unusable.

The devices of an Ad-Hoc network are supposed to be stationary and they are equipped with an omnidirectional antenna in such a way that the signal is spread radially from the nodes. A device may communicate with a single–hop, i.e. directly, with any other terminal which is located

within its transmission range. In order to communicate with the terminals placed out of this range a multi–hop communication has to be performed: it simply consists in making use of intermediate devices, called routers, that retransmit the received messages to the directly unreachable terminals ([72], [84]).

A crucial issue in this context consists in assigning a transmission power to each node in order to ensure the connectivity of the network while minimizing the total power expenditure over the network. Determining the optimal transmission power for each node is, indeed, desirable since a high power value will achieve a wide transmission range and, therefore, reach many nodes via a direct link, but at the same time will require higher consumption and will increase the interference level. On the other hand, low energy value may isolate one or more nodes causing the network to be disconnected. Both Cagalj *et al.* and Clementi *et al.* have shown that the Multicast problem in wireless Ad-Hoc networks is an NP-hard problem ([13], [20]).

In case of Multicast problem in wired networks, we take into account a Quality of Service in the routing of the communication. Indeed, in many application [66] there may be the further request of delivering the information generated by a source and directed to a set of destinations within a maximum delay. Naturally, the Quality of Service constraints and in the specific the maximum delay constraints impose a restriction on an acceptable Multicast tree. Only recently the Delay-constrained Steiner Tree problem has been object of study (see [49], [66]), indeed, with the developments of the multimedia technology, the real-time applications need to transmit information within a certain amount of time and so a message generated by one source of the network has to reach a set of target devices for delivering the same information in a fixed delay limit.

The first chapter is a preliminary chapter in which all the concepts: definitions, properties and problems that are used all along the dissertation are presented.

With the second chapter, we begin to consider the Minimum Power Multicast problem in wireless Ad-Hoc networks. We present a Set Covering formulation for the problem and we show that it is better than the formulation proposed in [53] and better than two adaptations to the Multicasting case of formulations proposed in [3] and in [60] for the Broadcasting problem (where the Broadcasting problem is a particular Multicast problem in which all the nodes of the network must be connected to the source). We propose also two exact procedures for solving the problem that use the properties of a Set Covering formulation and we present some computational results on randomly generated graphs with size ranking from 5 to 100 nodes and an increasing number of destinations.

In the third chapter, we study the properties of the Set Covering polytope of the Multicast problem in wireless Ad-Hoc networks. Specifically, we describe two heuristics for finding particular valid inequalities of the first Chvátal closure in order to strengthen the linear relaxation of the formulation. We compare the results on the improvement of the lower bounds obtained by solving the linear relaxation of the formulation with the addition of the constraints generated by these heuristics with the results produced minimizing over the first Chvátal closure polytope [19].

In the fourth chapter, we deal with the Broadcasting problem in which the minimization of the power cost and the achievement of a robust routing are considered. Indeed, we take into account the possibility that the devices may be subject to a temporary damage or a permanent failure and so they are assigned a probability of being active. We propose three mixed integer linear programming formulations whose optimal solution not only minimizes

the total transmission power over the network, but also guarantees a certain reliability level. The optimal solution provides a broadcasting structure robust enough to guarantee, in case of failure of some terminals, a reliable connectivity for the remaining terminals.

The study of a generalization of the Steiner tree problem is, instead, the topic of the fifth chapter. In particular the Delay constrained Steiner Tree problem is analysed, there, in wired networks. We present several valid mixed integer programming formulations that provide a tree spanning the source and the required nodes with the minimum cost and that satisfies a maximum delay threshold. We compare the respective linear relaxations of the formulations and we describe some preprocessing procedures to reduce the size of the problems. We present exact and approximate solution procedures with some computational results.

At the end, there is an appendix in which we briefly define some of the symbols used in the dissertation.

Chapter 1

Preliminaries

In this introductive chapter, we want to recall several basic definitions and properties ([10], [64], [68], [77], [86]) that will be used in the subsequent chapters. A list of further notations can be found at the end of the dissertation.

First of all, a *Linear Programming* problem (LP) consists in minimizing or maximizing a linear function, called objective function, on a feasible region defined by a series of linear constraints. An example of LP problem in standard form looks like the following:

$$\begin{aligned} &\min c^T x \\ &s.t. \\ &\quad A\,x = b \\ &\quad x \geq 0 \end{aligned} \tag{1.1}$$

where A is a $m \times n$ real matrix with rank m, c is an n-dimensional vector, b an m-dimensional vector and x an n-dimensional vector of decision variables.

If the decision variables take only integer values, the problem:

$$\begin{aligned} &\min c^T x \\ &s.t. \\ &\quad A\,x = b \\ &\quad x \geq 0 \\ &\quad x \in \mathbb{Z}^n \end{aligned} \tag{1.2}$$

is an *Integer Linear Programming* (IP) problem. In particular, if all the decision variables are restricted to $0-1$ values, the problem is called *Binary Integer Programming* (BIP).

If some, but not all the decision variables are integer, the problem:

$$\begin{aligned} &\min c^T x + d^T y \\ &s.t. \\ &\quad A\,x + By = b \\ &\quad x \geq 0,\ y \geq 0 \\ &\quad x \in \mathbb{Z}^n \end{aligned} \tag{1.3}$$

is called *Mixed Integer Programming* (MIP) and B is a $m \times p$ matrix, d is a p-dimensional vector and y is a p-dimensional vector of real variables.

1.1 Formulations

Definition 1.1.1. The *feasible region* of an LP problem (1.1) is the set $P = \{x \in \mathbb{R}^n_+ :\ A\,x = b\}$ which is a polyhedron, while the feasible region of an IP problem (1.2) is the set $S := P \cap \mathbb{Z}^n$. If the polyhedron P is bounded, it is called polytope.

Definition 1.1.2 (Relaxation of an IP problem). Given the IP problem (1.2) with feasible region S, a problem of this type: $\min\{c^T x : \ x \in T \subseteq \mathbb{R}^n\}$ is a *relaxation* of it if $S \subseteq T$.

Naturally, the optimal value of a relaxation of an IP problem is lower than the optimal value of the IP problem and so it represents a lower bound for the optimal value of the IP problem.

There are several possible relaxations of an IP problem, but in the following we will consider only the linear relaxation.

Definition 1.1.3 (Linear relaxation). The *linear programming relaxation* of an IP problem: $\min\{c^T x : \ x \in P \cap \mathbb{Z}^n\}$ with formulation $P = \{x \in \mathbb{R}^n : A\, x \geq b\}$ is the LP problem: $\min\{c^T x : \ x \in P\}$.

The linear programming relaxation can be, thus, obtained by eliminating the restriction that the variables x need to be integer. For this reason, again, the optimal value of the linear relaxation of an IP problem is a lower bound of the optimal value of the IP problem itself.

Definition 1.1.4. Given two linear formulations P_1 and P_2 for an integer problem:

(i) the formulation P_1 is *better* than P_2 if and only if $P_1 \subset P_2$,

(ii) the formulation P_1 is *equivalent* to P_2 if and only if $P_1 = P_2$,

(iii) if neither formulation is better than the other they are *incomparable*.

Definition 1.1.5 (Convex hull). Given a set $S \subseteq \mathbb{R}^n$, the *convex hull* of S, denoted by $conv(S)$, is the set of all the possible finite convex combination of elements of S, i.e. $conv(S) := \{x \in \mathbb{R}^n : \ x = \sum_{i=1}^{k} \alpha_i x_i, \ \sum_{i=1}^{k} \alpha_i = 1, \ \alpha_i \geq 0 \ \forall i \in \{1, .., k\}, \ \text{ for all } \{x_1, .., x_k\} \text{ subsets of } S\}$.

Among all the possible linear relaxations of an integer programming problem, the best one is the convex hull of all its feasible points:

$$P_I := conv(P \cap \mathbb{Z}^n) = conv(\{x \in \mathbb{R}^n : \ Ax \geq b, \ x \text{ integer}\}). \tag{1.4}$$

Proposition 1.1.1. *It holds that* $P_I \subseteq P$.

In Figure 1.1, the yellow polytope is the convex hull of a feasible set S of integer points and it represents an ideal formulation for an IP problem with feasible set S, while the polytope which is the union of the yellow and green portions is a possible linear relaxation of the IP formulation.

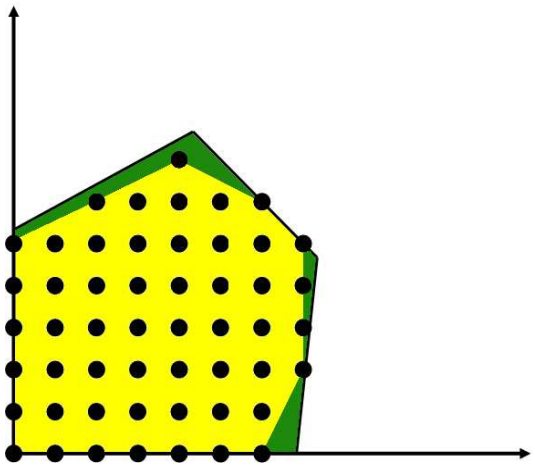

Figure 1.1: The ideal formulation and a possible LP relaxation of an IP problem

Definition 1.1.6 (Full–dimensional polyhedron). A polyhedron $P = \{x \in \mathbb{R}^n : \ A\,x \geq b\}$ is *full-dimensional* if and only if $dim(P) = n$, where $dim(P)$ is the maximum number of affinely independent points of P minus one.

In general, it is not trivial to give a complete description of the polyhedron P_I of an IP or MIP problem, so that it is interesting to strengthen certain inequalities, in particular, to find facet defining inequalities.

Definition 1.1.7 (Valid inequalities). Let $\pi \in R^n$, $\pi_0 \in R$ and let $P \subseteq \mathbb{R}^n$ be a polyhedron; the inequality $\pi^T x \leq \pi_0$ is a *valid inequality* for the polyhedron P if $\pi^T x \leq \pi_0$ for all the points $x \in P$, that is if $P \subseteq \{x \in \mathbb{R}^n : \ \pi^T x \leq \pi_0\}$.

Definition 1.1.8 (Facet defining inequalities). A valid inequality $\pi^T x \leq \pi_0$ is a *facet defining inequality* for a polyhedron P if and only if the equality $\pi^T x = \pi_0$ is verified for $dim(P)$ affinely independent points of P.

Another definition we should give is the definition of the support of a vector:

Definition 1.1.9 (Support). If x^* is an n-dimensional vector its *support* is the set:

$$Supp := \{j \in \{1, 2, .., n\} : \ x_j^* \neq 0\}.$$

1.2 Set Covering problem

The Set Covering problem is a classical Combinatorial Optimization problem of great theoretical and practical interest.

Definition 1.2.1 (Set Covering problem). Given a finite set I and a family $F = \{F_j\}_{j \in J}$ of subsets of I, given a cost $c_j \in \mathbb{R}^+$ associated with each element F_j of the family F. A subset $\overline{J}$ of the set J is a *cover* of I if

- $I = \bigcup_{i \in \overline{J}} F_i$

and it has the minimum cost if

$$\sum_{j \in \overline{J}} c_j \leq \sum_{j \in J'} c_j, \qquad \forall \ J' \subseteq J, J' \text{ cover of } I.$$

The Set Covering problem consists, thus, in finding a subset $\overline{J}$ of J such that

$$I = \bigcup_{j \in \overline{J}} F_j$$

and the cost $\sum_{j \in \overline{J}} c_j$ is the minimum of the costs of all the possible covers of I.

The Set Covering problem has been shown to be NP-complete in 1972 [45]. This type of problem can be formulated as an optimization problem introducing a $0-1$ matrix $A \in \mathbb{R}^{n \times m}$ called incidence matrix whose generic element a_{ij} is defined by:

$$a_{ij} = \begin{cases} 1 & \text{if } i \in F_j, \\ 0 & \text{otherwise.} \end{cases}$$

A formulation of the Set Covering problem can be, thus, the following:

$$\begin{aligned} & \min c^T x \\ & s.t. \\ & \quad Ax \geq \mathbf{1} \\ & \quad x \in \{0,1\}^n \end{aligned} \tag{1.5}$$

where c is a n-dimensional vector of costs.

There are several conditions for reducing the size of the incidence matrix of the Set Covering problem. Indeed, denoting by a_i^T the i^{th} row of A and by A_j the j^{th} column of A, the next proposition states some dominance rules for rows and columns of A.

Proposition 1.2.1 (Dominance of rows and columns).

i) If the i^{th} row is null, then the Set Covering problem is infeasible.

ii) If the i^{th} row has only one element equal to one in the k^{th} column, then set $x_k = 1$ and erase not only the column A_k, but also all the rows j such that $a_{jk} = 1$.

iii) Let A_i and A_j be two columns such that $a_{ki} \geq a_{kj}$ for every row index k. If the corresponding costs are such that $c_i \leq c_j$, then erase the column j.

iv) Let a_i^T and a_j^T be two rows such that $a_{ik} \geq a_{jk}$ for every column index k, then covering the j^{th} row implies the covering of the i^{th} row so that, the i^{th} row can be erased.

We denote by M the set of the row indices of the incidence matrix A and by N the set of the column indices of A. The Set Covering polytope $P_I(A)$ is:

$$P_I(A) := conv\big(\{x \in \mathbb{R}_+^{|N|} : \ A\,x \geq \mathbf{1},\ x \leq \mathbf{1},\ x \text{ integer}\}\big)$$

and the relaxed polytope $P(A)$ is:

$$P(A) := \{x \in \mathbb{R}_+^{|N|} : \ A\,x \geq \mathbf{1},\ x \leq \mathbf{1}\}.$$

For each $i \in M$, we denote by N^i the set of the column indices j such that the value of the element a_{ij} of the matrix A is one, i.e.,

$$N^i := \{j \in N : \ a_{ij} = 1\}.$$

The Set Covering polytope has been widely studied (see e.g. [7], [8], [22], [76]) and here we summarize some of its properties.

Proposition 1.2.2.

- *$P_I(A)$ is full-dimensional if and only if $|N^i| \geq 2$ for all $i \in M$;*
- *if $P_I(A)$ is full-dimensional, then the inequality $x_i \geq 0$ defines a facet of $P_I(A)$ if and only if $|N^i \setminus \{j\}| \geq 2$ for all $i \in M$;*

- *if $P_I(A)$ is full-dimensional, then all the inequalities $x_j \leq 1$ for all $j \in N$ define facets of $P_I(A)$;*
- *if $P_I(A)$ is full-dimensional and $\pi_0 > 0$, then all facet defining inequalities $\pi\, x \geq \pi_0$ for $P_I(A)$ have $\pi_j \geq 0$ for all $j \in N$.*

Remark 1.2.1. The only facet defining inequalities for the Set Covering polytope having right hand side equal to one are among the inequalities of the system $A\, x \geq \mathbf{1}$.

1.3 Graphs

We report here several definitions about the graphs.

Definition 1.3.1 (Undirected and directed Graph). An *undirected graph* G is a pair $G = (V, E)$, where V is a finite set of nodes or vertices and E is a family of subsets of V of cardinality two, called edges. Furthermore, a *directed graph* D is a pair $D = (V, A)$ where V is the set of vertices and A is a set of ordered pairs of vertices, called arcs.

Definition 1.3.2 (Path). Given a graph $G = (V, E)$ a *path* is a sequence $[v_1, v_2, ..., v_k]$ of nodes with $k > 1$, such that each pair of consecutive nodes belongs to E and there is no repetition of nodes in the sequence.

Definition 1.3.3 (Cycle). Given a graph $G = (V, E)$ a *cycle* is a sequence $[v_1, v_2, ..., v_k]$ with $k \geq 1$, such that each pair of consecutive nodes belongs to E, the nodes $v_1, v_2, ..., v_{k-1}$ are distinct and $v_1 = v_k$.

Definition 1.3.4 (Tree). A *tree* $T = (V', E')$ is a connected graph with no cycles.

Definition 1.3.5 (Cutset). Let $G = (V, E)$ be an undirected graph, S be a subset of V and S^c its complementary in V, a *cutset* is the set: $\delta(S) :=$

$\{e = \{i, j\} \in E : i \in S, j \in S^c\}$. If the graph $G = (V, A)$ is a directed graph, then for $S \subset V$ two directed cuts can be defined:

$$\delta^+(S) := \{(i, j) \in A : i \in S, j \in S^c\}$$

is the set of the arcs outgoing from S and

$$\delta^-(S) := \{(i, j) \in A : i \in S^c, j \in S\}$$

is the set of the incoming arcs in S.

Definition 1.3.6 (degree). The *degree* of a node $v \in V$ is the cardinality of $\delta(\{v\})$. For simplicity it is common to use $\delta(v)$ instead of $\delta(\{v\})$. In a directed graph, the set of the incoming arcs in v is denoted by $\delta^-(v)$, whereas the set of the outgoing arcs from v is denoted by $\delta^+(v)$.

1.4 Shortest Path, Spanning Tree and Maximum Flow problems

Three well studied problems are defined in this section: the Shortest Path problem, the Minimum Spanning Tree problem and the Maximum Flow problem.

Definition 1.4.1 (The Shortest Path). Given a graph $G = (V, E)$ with nonnegative cost (or length) associated with each edge $e \in E$, the *Shortest Path* (SP) problem consists in finding a path from a source node s to a terminal node t with the minimum total cost (or length).

The Shortest Path problem is polynomially solvable and Dijkstra's algorithm is an efficient algorithm for solving it. This algorithm [27] starts with the node $s \in V$ and a set $L := \{s\}$; at each iteration the algorithm labels

a node $i \in L^c$ with the shortest length of a path from s to i with internal nodes in L, updates the set $L := L \cup \{i\}$ and updates the distances from s to the nodes in L. This process is repeated until $t \in L$.

Definition 1.4.2 (The Minimum Spanning tree). Let $G = (V, E)$ be a graph with nonnegative cost (or weight) associated with each edge $e \in E$, the *Minimum Spanning Tree* problem consists in finding a tree with the minimum total cost (or weight) that spans all the nodes of G.

The greedy process that underlies Dijkstra's algorithm is similar to the process used in Prim's algorithm. Prim's algorithm [70] is used to find the Minimum Spanning Tree in a graph $G = (V, E)$. Starting with a node $s \in V$ and a set $L := \{s\}$, at each iteration the algorithm chooses a minimum-cost edge $e = \{u, v\} \in E$, connecting a node $u \in L$ to a node $v \in L^c$ and updates the set $L := L \cup \{v\}$. This process is repeated until $L = V$.

Definition 1.4.3 (Maximum Flow problem in capacitated graph). Given a directed graph $G = (V, A)$, two different nodes s and t belonging to V and a nonnegative capacity u_{ij} for each arc $(i, j) \in A$, the *Maximum Flow* problem consists in finding the maximum value of f such that a $|A|$-dimensional nonnegative vector x satisfies the flow conservation constraints

$$\sum_{(i,j)\in A} x_{ij} - \sum_{(j,i)\in A} x_{ji} = \begin{cases} f & \text{if } i = s, \\ 0 & \forall\, i \in V \setminus \{s, t\}, \\ -f & \text{if } i = t, \end{cases}$$

not exceeding the capacities on the arcs ($0 \leq x_{ij} \leq u_{ij},\ \forall (i, j) \in A$).

Definition 1.4.4 (Cut and capacity of a cut). Given a directed graph $G = (V, A)$ with a nonnegative capacity u_{ij} for each arc $(i, j) \in A$ and given two different nodes s and t, an $s - t$ *cut* is a partition (S, S^c) of the set V

such that $s \in S$ and $t \in S^c$. The *capacity* of this $s-t$ cut is

$$C(S, S^c) := \sum_{\substack{(i,j) \in A \\ i \in S,\ j \in S^c}} u_{ij}$$

Remark 1.4.1. The maximum flow value equals the total net flow across any $s-t$ cut (S, S^c):

$$f = \sum_{\substack{(i,j) \in A \\ i \in S,\ j \in S^c}} x_{ij} - \sum_{\substack{(j,i) \in A \\ j \in S^c,\ i \in S}} x_{ji}$$

Proposition 1.4.1 (Max-flow–Min-cut). *The value of a Maximum Flow problem equals the capacity of a Minimum cut [33].*

1.5 Steiner Tree problem

The Steiner Tree problem in a network is the problem of connecting a set of required vertices with the minimum cost.

Definition 1.5.1 (The Steiner Tree Problem (ST)). Given an undirected graph $G = (V, E)$ with a cost (or weight) c_e on each edge $e \in E$ and given a subset of the nodes R, called required nodes; the *Steiner Tree* problem consists in finding a minimum cost subtree of G that spans all the nodes in R with the possibility of including or not the nodes in $V \setminus R$, which are called Steiner nodes.

In general, the Steiner Tree problem is an NP-complete problem. Two special versions of the problem are polynomially solvable: if $|R| = 2$, then the problem reduces to the Shortest Path problem and if $R = V$, then the problem is the minimum Spanning Tree problem.

Definition 1.5.2 (Steiner Arborescence problem). The *Steiner Arborescence* problem is the directed version of the ST problem; the graph G is a directed weighted graph, a root node s, called source, is given and it is required to find a directed path from s to every terminal nodes in R with the minimum cost.

The cost or weight of a Steiner Tree T is indicated by $c(T)$ and it is defined as follows:

$$c(T) := \sum_{e \in T} c_e.$$

1.5.1 Preprocessing

Preprocessing the graph is an important factor for solving the ST problem in a reasonable time. It is applied on the undirected graph $G = (V, E)$ and the goal of this process is to reduce the size of the problem contracting or deleting nodes or edges in order to obtain an equivalent but reduced graph $G' = (V', E')$ ([6], [9], [16], [47], [81]).

Definition 1.5.3 (Feasible reduction). Given a Steiner Tree problem on the graph $G = (V, E)$ with terminal set R and costs c, a *feasible reduction* is a transformation of the problem into a Steiner Tree problem on the graph $G' = (V', E')$, with terminal set R', costs c' and constant cost $c_r \in \mathbb{R}_+$ with the properties that:

(i) $|V'| \leq |V|$,

(ii) $|E'| \leq |E|$,

(iii) $|R'| \leq |R|$,

(iv) if S is a feasible solution for the original problem, then there exists a feasible solution S' for the reduced problem with $c(S) = c'(S') + c_r$.

Quite simple reduction tests for the Minimum Steiner Tree are the degree tests applied recursively to each reduced graph until no more reduction can be performed.

Proposition 1.5.1 (Degree Reductions). *Given a Steiner Tree problem on the graph $G = (V, E)$, with terminal set R and vector of costs c:*

(i) *A Steiner node with degree less than or equal to one can be eliminated;*

(ii) *If a node i in R has degree one, its incident edge $\{i, j\}$ is contained in every feasible solution and can be contracted;*

(iii) *If a Steiner node i has degree two and $\{i, j\}$ and $\{i, k\}$ are its adiacent edges, then these edges can be replaced by the edge $\{j, k\}$ whose associated cost is $c_{(j,k)} = c_{(i,j)} + c_{(i,k)}$.*

Remark 1.5.1. Contracting an edge $\{i, j\}$ incident to a node $i \in R$ means:

- if $j \in R$, identify node i with j, eliminate the edge $\{i, j\}$, reduce the cardinality of R and store the cost $c_{(i,j)}$, that is, the costant cost c_r of the definition above is updated, i.e. $c_r := c_r + c_{(i,j)}$;
- if $j \in V \setminus (R \cup \{s\})$, identify nodes i with j (that becomes a required node) and update c_r.

1.5.2 Reduced costs fixing

Definition 1.5.4 (Reduced costs). Given an LP problem of the form (1.1), let B be an $m \times m$ nonsingular submatrix of A, x be a basic solution

and c_B be the vector of costs of the basic variables. For each $j \in \{1, .., n\}$ the *reduced cost* $\overline{c}_j$ of the variable x_j is defined according to the formula:

$$\overline{c}_j = c_j - c_B^T B^{-1} A_j.$$

Let z_{LP} be the optimal value of the linear relaxation of an IP problem (see the problem (1.2)) and let z_{UB} be the value of the best feasible solution known for the problem (an upper bound for the optimal value of the problem).

Proposition 1.5.2 (Reduced costs fixing). *[64] If a nonbasic variable x_j at its lower bound in the optimal solution of the linear relaxation of an IP is such that $z_{LP} + \overline{c}_j \geq z_{UB}$, then there exists an optimal solution of the IP with x_j at its lower bound. Similarly, if a nonbasic variable x_k at its upper bound in the optimal solution of the linear relaxation of an IP is such that $z_{LP} - \overline{c}_k \geq z_{UB}$, then there exists an optimal solution of the IP with x_k at its upper bound.*

Chapter 2

Minimum Power Multicast problem

In this chapter, we take into account the Minimum Power Multicast problem (MPM) in wireless Ad-Hoc networks [52]. The chapter is organized as follows: an introduction to the problem is given in section 2.1 and related works are presented in section 2.2. A formal description of the modelling aspects of the problem can be found in section 2.3, while the mathematical formulation of the MPM problem expressed in terms of a Set Covering problem is discussed in section 2.4 together with its comparison with some of the formulations that have been proposed in the literature. In section 2.5, we show some logic inequalities, whereas in 2.6, we report how to modify the graph associated with the Multicasting problem in wireless networks in order to model it as a Steiner Arborescence problem in a wired network. Section 2.7 is devoted to the description of two exact procedures for solving the problem that include the reduction technique for the Set Covering problem to reduce the huge number of the model's constraints. Finally, some computational results are illustrated in section 2.8 and some concluding

remarks are summarized in 2.9.

2.1 Introduction

Ad-Hoc networks are composed of a set of mobile devices with limited resources, that communicate with each other by transmitting a radio signal without using any fixed infrastructure or centralized administration. Nowadays, this kind of networks find their applications in several fields such as exchanging messages in an area where natural disasters have destroyed the existing infrastructure or in a battlefield. They are also used, for example, to allow internet access or simply to exchange information in buildings or in trains or to enable video-conferencing, etc. (see e.g. [66], [84]). The devices of an Ad-Hoc network, called also nodes, are arbitrarily located in an area where they are able to move, but at the time of the transmission all the nodes are supposed to be stationary; all along this dissertation, we will consider only static networks. Every terminal of the network is equipped with an omnidirectional antenna in such a way that the signal is spread radially from the nodes. A device may communicate with a single–hop, i.e. directly, with any other terminal which is located within its transmission range. In order to communicate with the terminals placed out of this range a multi–hop communication has to be performed: it simply consists in making use of intermediate devices, called routers, that retransmit the received message to the directly unreachable terminals ([72], [84]). Those nodes that are not reached by any signal are called isolated nodes.

The Multicast problem consists in connecting a specified device, called "source", with a set of target terminals, called "destinations", with the possibility of using any other device of the network as router. Since the resources of the devices are limited (for example nodes are equipped with

batteries) the source–destination connections should be obtained using the minimum amount of power. This objective would also have the advantage of reducing the interferences within the network and, consequently, of improving the signal quality.

The Minimum Power Multicast problem consists in assigning a transmission power to each node of the network in such a way that the source is connected to all the destinations with the minimum total transmitting power. We omit to consider interference problem in the model and we suppose that there is no constraint on the maximum transmission power of the nodes. Finally, we assume that the topology of the network and hence the exact position of all the terminals is known in advance.

2.2 Related works

The MPM problem represents a generalization of the very well known Minimum Power Broadcasting (MPB) problem. Indeed, if the set of destinations coincides with all the nodes of the network, except the source, the MPM problem reduces to the MPB problem (see e.g. Althaus *et al.* [1], Altinkemer *et al.* [3], Das *et al.* [25], Montemanni *et al.* [60], Wieselthier *et al.* [85], Yuang [88]). The MPM problem has been proved to be NP-complete (Cagalj *et al.* [13], Clementi et al. [20], [21]) and thus difficult to solve to optimality. Moreover, it is not simply a minimum Steiner Arborescence ([25], [57], [84]) connecting the source with the destinations because of the so called "broadcast property". Indeed a transmitting node reaches all the nodes of the network placed within its transmission range without any additional power, so that the amount of power in the solution of the MPM problem is not worse than the amount of power in the solution of the minimum Steiner Arborescence on the same but wired network.

While the MPB problem has attracted a wide attention in the scientific literature, the MPM problem has been scarcely studied despite its applicative importance. In fact, nowadays most of the MPM formulations available represent somehow an adaptation of the MPB models to the multicasting case. Interesting approaches to the problem are due to Wieselthier *et al.* [84] and to Das *et al.* [25]. The first authors describe an algorithm, called the Broadcast Incremental Power (BIP), and three greedy heuristics for the Multicast Power problem. The Broadcast Incremental Power (BIP) [84] is a modification of the Prim's algorithm [70]. Indeed, starting with a node $s \in V$ source of the communication and a set $L := \{s\}$, at each iteration the algorithm chooses a minimum-incremental power edge $e = (u, v) \in E$, connecting a node $u \in L$ to a node $v \in L^c$ and updates the set $L := L \cup \{v\}$. This process is repeated until $L = V$. The increment of power is the difference between the power that has to be used by a node $u \in L$ to reach a node $v \in L^c$ and the power already assigned to u.

Three different integer programming models have been proposed in [25] by Das *et al.*; these formulations for the MPM problem have been obtained as a generalization of those constructed for the MPB problem. Some specific studies for the multicast case have been considered in Guo *et al.* [36] and in Leino [53]. In particular, a linear integer formulation for the MPM problem has been presented in Leino [53] and a general scheme of a cutting plane algorithm has been used for its solution, whereas a flow-based formulation expressed in terms of a mixed integer programming has been suggested in Guo *et al.* [36].

2.3 Mathematical Models for the MPM

We shall model the MPM problem in terms of a graph, by considering the devices of the network as nodes and the transmission links as arcs like in Figure 2.1.

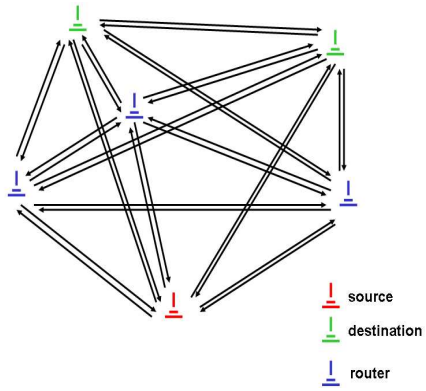

Figure 2.1: Example of a Multicast problem in a complete graph with 6 nodes

Let $G = (V, A)$ be a directed complete graph, where V represents the set of the terminals of the network and A is the set of directed arcs which connect all the possible pairs (i, j), with $i, j \in V$ and $j \neq i$. Each node $i \in V$ can receive data from any other node of the network and send data to any node in its transmission range, which is not a priori constrained to assume any fixed value. We select a particular node $s \in V$ as the source of the messages (the red antenna in Figure 2.1), and a subset of nodes $R \subset V$ whose elements are the destinations of the communication (the green antennae in Figure 2.1). Nodes belonging to $V \setminus (R \cup \{s\})$ may act either as routers, i.e., they can be involved in forwarding the messages or they may remain isolated without receiving or transmitting any signal (the blue antennae in Figure 2.1).

Let n and m be two integer numbers representing respectively the cardinality of set V and that of R, with $1 \leq m < n$. We note that if $m = 1$ the problem reduces to finding the Shortest Path from the source to the destination and if $m = n - 1$ the Multicasting problem reduces to a Broadcasting problem. Despite some analogies with the Minimum Spanning Arborescence problem, the MPB problem in wireless networks has been proved to be NP-complete ([13], [20], [21]). We assume that the nodes are fixed since we are considering static networks and, thus, all the distances d_{ij} between each pair of nodes i and j in V are known in advance. This is an approximation of the real world applications, but it is not too restrictive, as one may think, especially, if we consider optimization over short time intervals and assume that the devices move slowly in the area.

For simplicity, we consider here the case in which for any distinct nodes $i, k, l \in V$, it holds: $d_{ik} \neq d_{il}$.

With each arc (i, j) it is associated a cost p_{ij} that represents the minimum amount of power required to establish a direct connection from node i to node j. As usually assumed in literature in a simple signal propagation model [72], the power p_{ij} is considered to be proportional to the power of the distance d_{ij} with an environment-dependent exponent κ whose value is typically in the interval [2,5]; therefore, $p_{ij} := (d_{ij})^{\kappa}$. Notice that the results presented in this dissertation remain valid also in case more complex signal propagation models are considered.

Most of the already defined formulations of the problem ([53], [60], [84]) use, instead of the costs p_{ij} for the arcs, an incremental cost c_{ij} defined as follows:

$$c_{ij} = p_{ij} - p_{ia^i_j} \qquad \forall (i, j) \in A,$$

where, according to the definition given in [60], the node a^i_j is the "ancestor" of j with respect to i:

$$a_j^i := \begin{cases} i & \text{if } p_{ij} = \min\limits_{k \in V}\{p_{ik}\}, \\ \arg\max\limits_{k \in V}\{p_{ik} | p_{ik} < p_{ij}\} & \text{otherwise.} \end{cases} \tag{2.1}$$

By introducing the so called *range assignment* function, which assigns to each node $i \in V$ its transmitting power $r(i)$:

$$r : V \to \mathbb{R}^+, \qquad i \mapsto r(i),$$

the MPM problem can be equivalently formulated defining such a function in order to minimize the quantity $\sum_{i \in V} r(i)$, while guaranteeing the connection among the source and all the destinations. Obviously, in any efficient solution, $r(i)$ must be zero or equal to p_{ij} for some j (i.e., node i does not transmit or uses exactly the amount of power necessary to reach a target node j), so we shall assume this to be the case. We want to stress here that when we talk about connection among the source and all the destinations in this chapter and in chapter 4 we do not mean necessarily a direct connection, but we do not also mean the existence of a path in the traditional sense (see Definition 1.3.2) from the source to each destination. In fact, since the nodes are equipped with omnidirectional antennae and the communication is a radio transmission, any signal forwarded by node $i \in V$ and directed to node $j \in V$ is also received by all the nodes that are not more distant than j from i, i.e., if $r(i) = p_{ij}$, then every node $k \in V$ such that $p_{ik} \leq p_{ij}$ receives the signal (see Figure 2.2). This is the so called "broadcast property" ([60], [84]) which is a peculiarity of this kind of networks. Several nodes can be, therefore, covered and reached with a single transmission and, hence, using a single transmission power.

Even though the MPM problem consists in assigning the transmission power to the nodes, as suggested before, it is convenient to consider the decision variables associated with the arcs ([25], [60]) in order to model the

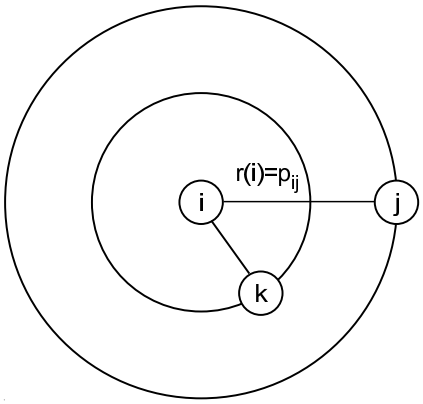

Figure 2.2: Broadcast property

link states. In particular, we want to model: (i) the event that node i is transmitting to a target node j (that is, i uses exactly an amount p_{ij} of power); (ii) the event that the transmission of node i is received by node j (that is, the power assigned to node i is not smaller than p_{ij}); and (iii) the event that arc (i, j) belongs to the underlying Steiner Arborescence which connects s with every node in R. We introduce, thus, three sets of variables, x, y and z to characterize each of the three above events.

The set of variables x describes which node transmits to whom; formally, using the *range assignment* function:

$$x_{ij} := \begin{cases} 1 & \text{if } r(i) = p_{ij}, \\ 0 & \text{otherwise.} \end{cases}$$

The set of variables y determines which nodes are in the transmission range of other nodes, i.e. for all $(i, j) \in A$, $y_{ij} = 1$, if the node i transmits and reaches node j, otherwise $y_{ij} = 0$. By expressing y variables using the definition of the function r, we can write for all $(i, j) \in A$:

$$y_{ij} := \begin{cases} 1 & \text{if } r(i) \geq p_{ij}, \\ 0 & \text{otherwise.} \end{cases}$$

Finally, the variables z define a Steiner Arborescence T, connecting s with all the destinations in R: for all $(i,j) \in A$, if $(i,j) \in T$, then $z_{ij} = 1$ (that is the node i is transmitting and the node j is reached by it), otherwise $z_{ij} = 0$.

The "broadcasting property" makes the difference between the Minimum Steiner Arborescence problem and the Minimum Power Multicast problem ([25], [84]), indeed, if the objective function of the first problem in a wired network can be expressed in this way:

$$\min \sum_{(i,j)\in A} p_{ij} z_{ij},$$

the objective function for the Multicasting problem in a wireless network is the following:

$$\min \sum_{i\in V} \max_{j\in V\setminus\{i\}} p_{ij} z_{ij}.$$

For this reason, the cost of an optimal solution of the Multicasting problem is a lower bound for the optimal Steiner Arborescence solution in the same but wired graph.

Since we assign only one power value to each node $i \in V$, there will be at most one intended target node j for i. Thus, as in [25]:

Remark 2.3.1. For any node $i \in V$ the following relation holds

$$\sum_{j\in V\setminus\{i\}} x_{ij} \leq 1.$$

Furthermore, using the inequalities of the Remark 2.3.1, it is possible to express a relation between variables y and x. Indeed, if variable $x_{ik} = 1$, it means that node i transmits with the power necessary to reach k. Any other node j which is not farther than k from i also receives the transmission, therefore, $y_{ij} = 1$. We can, thus, derive:

Remark 2.3.2. For all $(i,j) \in A$ the following relation binds the y and x variables:

$$y_{ij} = \sum_{k \in V \setminus \{i\},\ d_{ij} \leq d_{ik}} x_{ik}.$$

Moreover, we can notice that in any efficient solution, if variable $x_{ij} = 1$, then also variable $z_{ij} = 1$, since the link (i,j) belongs to the underlying Steiner Arborescence connecting the source to the destinations; on the other hand, an arc (i,j) might belong to the Steiner Arborescence even if j is not the target node of i, i.e., $r(i) = p_{ik} > p_{ij}$, with $k \in V \setminus \{i\}$ and $x_{ij} = 0$ but $z_{ij} = 1$.

On the basis of the definition of the variables and the above observations, we have:

Remark 2.3.3. For all $(i,j) \in A$ the following relations must hold

$$x_{ij} \leq z_{ij} \leq y_{ij}.$$

We describe now three formulations presented in literature. The first one is a slight modification in terms of notation of the model proposed by Leino [53]:

$$\min \sum_{(i,j) \in A} c_{ij} y_{ij} \tag{2.2}$$

$s.t.$

$$\sum_{i \in S, j \in S^c} y_{ij} \geq 1 \qquad \forall\, S \subset V,\ s \in S,\ R \cap S^c \neq \emptyset \tag{2.3}$$

$$y_{ij} \leq y_{ia_j^i} \qquad \forall\, (i,j) \in A,\ a_j^i \neq i \tag{2.4}$$

$$y_{ij} \in \{0,1\} \qquad \forall\, (i,j) \in A. \tag{2.5}$$

The second one is an adaptation to the Multicasting problem of the MPB formulation defined in Montemanni et al [60] (by omitting the symmetric connectivity condition):

$$\min \sum_{(i,j)\in A} c_{ij} y_{ij} \tag{2.6}$$

$s.t.$

$$\sum_{i\in S, j\in S^c} z_{ij} \geq 1 \qquad \forall\, S \subset V,\ s \in S,\ R \cap S^c \neq \emptyset \tag{2.7}$$

$$y_{ij} \leq y_{ia_j^i} \qquad \forall\, (i,j) \in A,\ \ a_j^i \neq i \tag{2.8}$$

$$z_{ij} \leq y_{ij} \qquad \forall\, (i,j) \in A \tag{2.9}$$

$$y_{ij}, z_{ij} \in \{0,1\} \qquad \forall\, (i,j) \in A. \tag{2.10}$$

Observe that, since variables z_{ij} do not appear in the objective function, we can strengthen formulation (2.7) − (2.10) by substituting inequalities (2.9) with the equations $z_{ij} = y_{ij}$ without losing any optimal solution. By doing so, it is easy to see that formulation (2.7) − (2.10) is, in fact, a relaxation of formulation (2.3) − (2.5).

Finally, the last formulation is the multicasting version of the MPB formulation presented in Altinkemer et al [3]. While the first two formulations minimize the incremental cost, this model minimizes directly the power to

be assigned to each arc:

$$\min \sum_{(i,j)\in A} p_{ij}x_{ij} \tag{2.11}$$

$s.t.$

$$\sum_{i\in S, j\in S^c} z_{ij} \geq 1 \qquad \forall S \subset V,\ s \in S,\ R \cap S^c \neq \emptyset \tag{2.12}$$

$$z_{ij} \leq \sum_{k\in V\setminus\{i\}, d_{ij}\leq d_{ik}} x_{ik} \qquad \forall\,(i,j) \in A \tag{2.13}$$

$$x_{ij}, z_{ij} \in \{0,1\} \qquad \forall (i,j) \in A. \tag{2.14}$$

Constraints (2.3), (2.7) and (2.12) are the "connectivity constraints", that is, for each cut (S, S^c) with $s \in S$ and $S^c \cap R \neq \emptyset$, these constraints enforce the existence of at least one arc outgoing from a node belonging to S and incoming in a node of S^c; constraints (2.4) and (2.8) are the "broadcast constraints", enforcing the "broadcast property"; constraints (2.9) and (2.13) represent the variable relations described in Remarks 2.3.2 and 2.3.3; and constraints (2.5), (2.10) and (2.14) are the domain definition constraints.

2.4 The Set Covering Formulation

In this section, we will define our Set Covering–based model for the MPM problem. We start by proposing a first formulation that we prove to be at least as strong as the formulation (2.2) − (2.5). Then by exploiting the topological properties of the problem, we introduce our Set Covering model.

For convenience, we shall use the following notation: for each node $i \in V$, let v^i be an array whose components are the nodes of the network ordered

with respect to an increasing distance from node i. In other words, if j and k are two indices in $\{1, \ldots, n\}$ with $j \leq k$, then v^i_j and v^i_k are two nodes in V whose distances from i are related by

$$d_{iv^i_j} \leq d_{iv^i_k}.$$

We refer to v^i as a *distance array.*

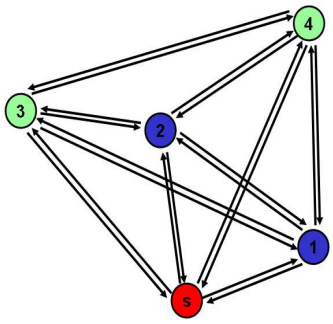

Figure 2.3: Example for the distance arrays

Example 2.4.1. For the network in Figure 2.3 the distance arrays are the following: $v^s = (s, 1, 2, 3, 4), v^1 = (1, s, 2, 4, 3), v^3 = (3, 2, s, 4, 1)$, $v^4 = (4, 2, 1, 3, s)$.

By Remark 2.3.2, we have:

Remark 2.4.1. For all $i \in V$ and $j \in \{2, \ldots, n-1\}$ the following relations must hold

$$x_{iv^i_j} = y_{iv^i_j} - y_{iv^i_{j+1}}$$

and for $j = n$:

$$x_{iv^i_n} = y_{iv^i_n}.$$

We propose now a first formulation which uses only the variables x:

$$\min \sum_{(i,j)\in A} p_{ij}x_{ij} \tag{2.15}$$

s.t.

$$\sum_{i\in S, j\in S^c} \sum_{k\in V\setminus\{i\},\, d_{ij}\le d_{ik}} x_{ik} \ge 1 \qquad \forall\, S\subset V,\ s\in S,\ R\cap S^c\neq\emptyset \tag{2.16}$$

$$\sum_{j\in V\setminus\{i\}} x_{ij} \le 1 \qquad \forall\, i\in V \tag{2.17}$$

$$x_{ij}\in\{0,1\} \qquad \forall\,(i,j)\in A. \tag{2.18}$$

We notice that it is possible to use Remarks 2.3.2 and 2.4.1 to augment formulation (2.2) – (2.5) with variables x_{ij} and formulation (2.15) – (2.18) with variables y_{ij}, so that their linear relaxations can be compared. By doing so, we can derive the following result.

Proposition 2.4.1. *The linear relaxation of formulation* (2.15) – (2.18) *is equivalent to the linear relaxation of formulation* (2.2) – (2.5)*.*

Proof. First of all, observe that, since vectors x and y are related as in Remarks 2.3.2 and 2.4.1 the objective functions (2.2) and (2.15) express the same quantity. In fact, by the definition of incremental costs, for any $i\in V$ and $j\in\{2,\dots,n\}$ we have

$$p_{iv^i_j} = \sum_{k=2}^{j} c_{iv^i_k}.$$

Hence, by using Remark 2.4.1, we have

$$\sum_{j=2}^{n} p_{iv^i_j}x_{iv^i_j} = \sum_{j=2}^{n-1}\sum_{k=2}^{j} c_{iv^i_k}(y_{iv^i_j} - y_{iv^i_{j+1}}) + \sum_{k=2}^{n} c_{iv^i_k}y_{iv^i_n} =$$

$$\sum_{k=2}^{n} c_{iv_k^i} \sum_{j=k}^{n} y_{iv_j^i} - \sum_{k=2}^{n-1} c_{iv_k^i} \sum_{j=k+1}^{n} y_{iv_j^i} = \sum_{k=2}^{n} c_{iv_k^i} y_{iv_k^i}.$$

Consequently, we have

$$\sum_{(i,j)\in A} p_{ij} x_{ij} = \sum_{i\in V} \sum_{j=2}^{n} p_{iv_j^i} x_{iv_j^i} = \sum_{i\in V} \sum_{k=2}^{n} c_{iv_k^i} y_{iv_k^i} = \sum_{(i,j)\in A} c_{ij} y_{ij}.$$

Assume now that $\overline{x}$ is a feasible solution of the relaxation of (2.15) − (2.18), and that $\overline{y}$ is the corresponding vector of variables obtained in Remark 2.3.2. We have to show that $\overline{y}$ is a feasible solution for the linear relaxation of (2.2) − (2.5). Indeed, we have:

$$\sum_{i\in S, j\in S^c} \overline{y}_{ij} = \sum_{i\in S, j\in S^c} \sum_{k\in V\setminus\{i\},\ d_{ij}\le d_{ik}} \overline{x}_{ik} \ge 1.$$

Moreover, for any $(i,j) \in A$ such that $a_j^i \neq i$, since variables $\overline{x}_{ij}$ are not negative, we have:

$$\overline{y}_{ij} = \sum_{k\in V\setminus\{i\},\ d_{ij}\le d_{ik}} \overline{x}_{ik} \le \overline{x}_{ia_j^i} + \sum_{k\in V\setminus\{i\},\ d_{ij}\le d_{ik}} \overline{x}_{ik} = \sum_{k\in V\setminus\{i\},\ d_{ia_j^i}\le d_{ik}} \overline{x}_{ik} = \overline{y}_{ia_j^i}$$

and, for any $(i,j) \in A$,

$$0 \le \overline{y}_{ij} = \sum_{k\in V\setminus\{i\},\ d_{ij}\le d_{ik}} \overline{x}_{ik} \le \sum_{j\in V\setminus\{i\}} \overline{x}_{ij} \le 1.$$

On the other hand, let $\overline{y}$ be a feasible solution for the linear relaxation of formulation (2.2) − (2.5) and let $\overline{x}$ be the corresponding vector of variables obtained by Remark 2.4.1. We can show that $\overline{x}$ is a feasible solution for the linear relaxation of (2.15) − (2.18). Indeed, by using Remark 2.3.2,

constraints (2.16) are easily seen to be satisfied. Moreover, for any $i \in V$, by Remark 2.4.1 we have:

$$\sum_{j \in V \setminus \{i\}} x_{ij} = \sum_{j=2}^{n} x_{iv_j^i} = \sum_{j=2}^{n-1} (y_{iv_j^i} - y_{iv_{j+1}^i}) + y_{iv_n^i} = y_{iv_2^i} \leq 1,$$

which means that constraints (2.17) are also satisfied. Finally, by using (2.4), we have:

$$0 \leq \overline{y}_{ia_j^i} - \overline{y}_{ij} = \overline{x}_{ia_j^i} \leq 1.$$

□

By using similar arguments as those in the proof of Proposition 2.4.1 and letting variables x and y be related according to Remarks 2.3.2 and 2.4.1, it is easy to prove the following:

Remark 2.4.2. Any feasible solution to the linear relaxation of formulation $(2.6) - (2.5)$ is also feasible for the linear relaxation of formulation $(2.15) - (2.18)$.

We can notice that in constraints (2.16) the coefficients of some variables x_{ij} could be greater than one. This suggests to strengthen the formulation by reducing to one all the left-hand-side coefficients of constraints (2.16). In order to describe the resulting constraints, we introduce the following notation.

Let S be any proper subset of V. For every $i \in S$, we label with $v^i_{k(S)}$ the first component in the distance array v^i which is not an element of S. Furthermore, we denote by $K^i(S)$ the subset of $V \setminus \{s\}$ whose elements are all the nodes of the network different from the source and having distance from i greater than or equal to $d_{iv^i_{k(S)}}$. For a better understanding of this notation, we give an example.

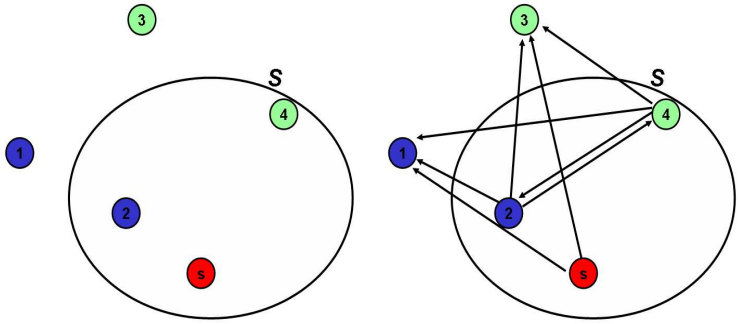

Figure 2.4: Example for constraints (2.20)

Example 2.4.2. Looking at Figure 2.4, $V := \{s, 1, 2, 3, 4\}$, $R := \{3, 4\}$ and $S := \{s, 2, 4\}$. The distance arrays are: $v^s = (s, 2, 4, 1, 3)$, $v^1 = (1, 2, 3, s, 4)$, $v^2 = (2, s, 1, 4, 3)$, $v^3 = (3, 4, 1, 2, s)$, $v^4 = (4, 3, s, 2, 1)$; thus $v^s_{k(S)}$ and $v^2_{k(S)}$ are node 1, while $v^4_{(k(S))}$ is node 3 and $K^s(S) := \{1, 3\}$, $K^2(S) := \{1, 3, 4\}$ and $K^4(S) := \{1, 2, 3\}$.

Now we are able to present the strengthened formulation of the MPM problem:

$$\min \sum_{(i,j)\in A} p_{ij} x_{ij} \tag{2.19}$$

$s.t.$

$$\sum_{i\in S} \sum_{j\in K^i(S)} x_{ij} \geq 1 \qquad \forall\, S \subset V,\ s \in S,\ R \cap S^c \neq \emptyset \tag{2.20}$$

$$\sum_{j\in V\setminus\{i\}} x_{ij} \leq 1 \qquad \forall\, i \in V \tag{2.21}$$

$$x_{ij} \in \{0, 1\} \qquad \forall\, (i, j) \in A. \tag{2.22}$$

The set of constraints (2.20) represents the connectivity requirements; for every cut (S, S^c) with $s \in S$ and $R \cap S^c \neq \emptyset$ there should be a node

i in S that transmits with a power sufficient to reach at least one node in S^c. We remark that the "target" node j of node i (that is, the one such that $x_{ij} = 1$) does not need to be in S^c, indeed, j can belong to S, but the distance between i and j must be greater than the distance from i to a node in S^c. For example, the presence of one of the arcs in Figure 2.4 would satisfy the constraint (2.20) relative to the choice of $S = \{s, 1, 4\}$. Constraints (2.21) ensure that at most one power value is assigned to each node and, finally, (2.22) are the binary restrictions on the variables.

We now show that constraints (2.21) in the last formulation are redundant for defining any optimal solution of the linear relaxation of the formulation as the objective value coefficients are non negative.

Proposition 2.4.2. *Let $\overline{x}$ be an optimal solution of* (2.19) *satisfying constraints* (2.20) *and the linear relaxation of constraints* (2.22)*. Then we have:*

$$\sum_{j \in V \setminus \{i\}} \overline{x}_{ij} \leq 1 \qquad \forall i \in V.$$

Proof. Assume that there exists $h \in V$ such that

$$\sum_{j \in V \setminus \{h\}} \overline{x}_{hj} > 1. \tag{2.23}$$

Let $l \in \{1, ..., n\}$ be the smallest index such that:

$$\sum_{j=l+1}^{n} \bar{x}_{hv_j^h} \leq 1,$$

let R denote the set $\{v_l^h, v_{l+1}^h, \ldots, v_n^h\}$ and $r = v_l^h$. By setting, for all $j \in V \setminus \{h\}$,

$$x^*_{hj} = \begin{cases} \overline{x}_{hj} & \text{if } j \in R \setminus \{r\}, \\ 1 - \sum_{j \in R \setminus \{r\}} \overline{x}_{hj} & \text{if } j = r, \\ 0 & \text{otherwise,} \end{cases}$$

we have that: $x^*_{hr} = 1 - \sum_{j \in R \setminus \{r\}} \overline{x}_{hj} < \overline{x}_{hr}$ and, thus,

$$\sum_{j \in V \setminus \{h\}} p_{hj} x^*_{hj} < \sum_{j \in V \setminus \{h\}} p_{hj} \overline{x}_{hj}.$$

Let, for any node $i \in V \setminus \{h\}$ and for any node $j \in V \setminus \{i\}$, $x^*_{ij} = \overline{x}_{ij}$. Then, the new solution x^* is feasible, since constraints (19) are still satisfied. Moreover, we have that:

$$\sum_{(i,j) \in A} p_{ij} x^*_{ij} < \sum_{(i,j) \in A} p_{ij} \overline{x}_{ij}.$$

This leads to a contradiction, since $\overline{x}$ is by assumption an optimal solution. □

By the above Proposition, we can remove constraints (2.21) from the formulation. Moreover, since all the powers are positive values, we notice that, in any optimal solution, no node is assigned the power to reach exactly the source, so that all the incoming arcs of A in the source s can be eliminated from the graph:

$$A := A \setminus \{(i,j) \in A : \ i \in V, j = s\}.$$

The final formulation of the problem, that we propose is a Set Covering formulation:

$$\min \sum_{(i,j) \in A} p_{ij} x_{ij} \tag{2.24}$$

s.t.

$$\sum_{i \in S} \sum_{j \in K^i(S)} x_{ij} \geq 1 \qquad \forall\, S \subset V,\ s \in S,\ R \cap S^c \neq \emptyset \tag{2.25}$$

$$x_{ij} \in \{0,1\} \qquad \forall\, (i,j) \in A. \tag{2.26}$$

Table 2.1: Average gap for (2.3)-(2.5) and for (2.25)-(2.26)

n	m	*(2.3)-(2.5)* gap	*(2.25)-(2.26)* gap	n	m	*(2.3)-(2.5)* gap	*(2.25)-(2.26)* gap
5	1	0.21183	0	15	1	0.48164	0
5	2	0.27884	0	15	2	0.49797	0
5	3	0.19820	0	15	3	0.44208	0
5	4	0.17085	0	15	4	0.40148	0.00002
				15	5	0.38226	0.00002
10	1	0.36262	0	15	6	0.35043	0.00708
10	2	0.41995	0	15	7	0.33496	0.00952
10	3	0.34237	0	15	8	0.28470	0.01015
10	4	0.35768	0.00009	15	9	0.29569	0.01280
10	5	0.32836	0.00028	15	10	0.28654	0.01123
10	6	0.32093	0.00390	15	11	0.27004	0.01793
10	7	0.30090	0.00626	15	12	0.26053	0.01835
10	8	0.29403	0.00971	15	13	0.24193	0.01835
10	9	0.24807	0.00666	15	14	0.23624	0.02104

Constraints (2.25) are the connectivity constraints and constraints (2.26) are the domain definition constraints.

Since the number of constraints (2.25) is $2^{n-1} - 2^{n-m-1}$, the main difficulty of this problem, beyond the fact that it is an integer problem, is caused by the huge number of such constraints. Moreover, it is evident that the broadcasting version of this problem has the maximum number of constraints of type (2.25). Notice, however, that in general many of the constraints (2.25) are redundant and can be removed from the formulation because they are dominated by other constraints in (2.25).

Remark 2.4.3. The optimal solution of the linear relaxation of the Set Covering formulation provides a lower bound that is more effective than the lower bound produced by the optimal solution of the linear relaxation of the formulation (2.2) – (2.5).

In order to compare the two formulations we have done several experi-

ments. In Table 2.1 each column reports the average value of the gap between the optimal value OPT of the integer problem and the optimal value LB of the linear relaxation of the two formulations for 20 randomly generated problems for each combination of the number of nodes/destinations. We indicate with *gap* the value $(OPT-LB)/LB$. From the results reported in Table 2.1, it is highlighted firstly that the lower bound of the Set Covering formulation is much better than the lower bound of the formulation $(2.2)-(2.5)$, secondly that for problems with few nodes and few destinations the optimal solution of the linear relaxation of our proposed formulation is already an integer solution.

2.5 Logic inequalities

We present some inequalities that can be added to the problem and that can be found just considering logic properties of the MPM problem.

Remark 2.5.1. The following inequalities:

$$\text{(i)} \qquad x_{ij} + x_{ji} \leq 1 \qquad \forall\, i \in V,\ j \in \delta^+(i);$$

$$\text{(ii)} \qquad \sum_{i \in V \setminus \{j\}} x_{ij} \leq 1 \qquad \forall\, j \in V;$$

are inequalities that reduce the feasible region of the MPM problem but they do not cut off any fractional optimal solution of the linear relaxation of $(2.24) - (2.26)$.

Remark 2.5.2. The number of the arcs of an optimal integer solution of the MPM problem (that is the number of the transmissions in an optimal solution) should be at most the number of arcs in an acyclic graph spanning all the nodes of the network and hence $\sum_{(i,j)\in A} x_{ij} \leq n-1$. We can notice that if the power assigned to the source is exactly the power necessary to

reach its most distant destination, placed in the k^{th} position of the array v^s, then all the destinations are reached by the signal generated by the source and no other transmission must be performed in order to create the connection. This remark can be expressed with the constraint:

$$\sum_{(i,j)\in A\setminus\{(s,v_k^s)\}} x_{ij} \leq (n-1)(1-x_{sv_k^s}). \tag{2.27}$$

In an optimal solution, if the source s transmit to the node v_k^s then the right hand side of (2.27) is zero and this force all the other variable x_{ij} to be zero otherwise it holds: $\sum_{(i,j)\in A\setminus\{(s,v_k^s)\}} x_{ij} \leq \sum_{(i,j)\in A} x_{ij} \leq n-1$ and the constraint (2.27) is fulfilled.

Remark 2.5.3. The inequalities

$$\sum_{j\in\delta^-(i)} x_{ji} \leq \sum_{j\in\delta^+(i)} x_{ij} \qquad \forall\, i \in V \setminus (R\cup\{s\}) \tag{2.28}$$

are the flow-balance constraints (see e.g. [47]). If i is a router and i is directly reached by a communication originated by a node j in the network, constraint (2.28) forces node i to transmit. In no optimal integer solution a router i is a leaf of the arborescence, indeed, if it exists $j \in \delta^-(i)$ such that $x_{ji} = 1$ and for each $k \in \delta^+(i)$ the variables x_{ik} are all equal to zero, the cost p_{ji} paid for this type of solution can be reduced making j transmit to a node h closer to j than i without disconnecting any destination.

2.6 Multicasting problem and Minimum Steiner Arborescence

Minimum Power Multicast problem on the directed graph $G = (V, A)$ can be reduced into a Minimum Steiner Arborescence problem ([14], [55]) on a

directed graph $G' = (V', A')$. The graph $G' = (V', A')$ can be constructed as follows: for each node $i \in V$, consider the set of the outgoing arcs from i (see Definition 1.3.5), $\delta^+(i)$. For each arc $(i, j) \in \delta^+(i) \setminus \{(i, v_2^i)\}$ a node u should be inserted into the graph and the arc (i, j) should be split into the arcs (i, u) and (u, j). The cost of the arc (i, j) is assigned to the arc (i, u), whereas a zero cost is assigned to (u, j). Furthermore, all the arcs (u, k) with $p_{ik} \leq p_{ij}$ should be added to the graph with a zero cost.

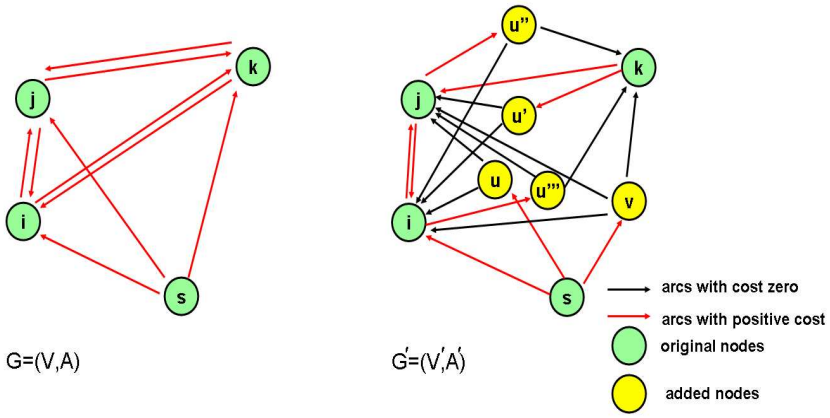

Figure 2.5: The graph for a Multicast problem in wireless network and the graph for the equivalent Steiner Arborescence problem

With this transformation $(n-2) + (n-1)(n-3)$ new nodes are added to the original graph so that in total $|V'| = (n-1)^2$, whereas the $(n-1)^2$ arcs of G are substituted by $(n^3 - n^2 - 2n)/2$, i.e. $|A'| = (n^3 - n^2 - 2n)/2$. The cardinality of V' is $O(n^2)$ and the cardinality of A' is $O(n^3)$; the size of the problem, thus, grows very rapidly as the size of the original problem increases.

Example 2.6.1. Figure 2.5 is a little example of a graph $G = (V, A)$ for the Multicasting problem with 4 nodes and of the graph $G' = (V', A')$ on which

the Steiner Arborescence problem has the same optimal solution value as the optimal solution value of the Multicast problem. All the arcs in red are arcs with strictly positive costs, while the arcs in black have costs zero.

2.7 Solution Methods

As discussed before, the main difficulty for the solution of the Set Covering formulation is represented by the set of constraints (2.25), but a considerable help may be given by the structure of the formulation. Here, we propose two solution methods that exploit such structure.

In the first procedure, we generate the whole constraint matrix, but we take into account only a subset of its rows. Indeed, initially, we create a submatrix by selecting $n-1$ rows and we perform a preprocessing on this submatrix in order to erase dominated rows and columns, then we solve the integer problem and finally, we check whether violated constraints exist. If all the constraints (2.25) of the problem are satisfied, the procedure is interrupted since the optimal solution has been found, otherwise, we add at most n^2 violated rows at a time and we repeat the iterative process for the new submatrix until an optimal solution is found.

We specify that among the first $n-1$ rows of the initial submatrix, we select the row corresponding to the inequality relative to the subset $S = \{s\}$ and all the rows corresponding to the inequalities relative the subsets S such that $|S^c| = 1$. Moreover, whenever we find a row which is dominated in the current submatrix, we label it and we do not admit the possibility of reintroducing it in any subsequent matrix; only at the end of the procedure, before electing the optimal solution we check whether all the erased constraints are satisfied, otherwise we add the violated ones and the

whole process is repeated.

In our second method, violated constraints are generated iteratively on the basis of the current solution looking at its support (see Definition 1.1.9). We start with the inequalities (2.25) generated by the sets $S := \{s\}$ and $S := \{s, v_2^s\}$ and we solve the resulting linear relaxation of the problem. On the basis of the optimal solution, we define the related variables y using the equality in the Remark 2.3.2 and we construct the connected component of the network starting with the source. The connected component of the source is the set of the nodes of the graph such that there exists a directed path from the source to these nodes using the arcs in which the values of the variables y are not zero. While at least one destination is not connected to the source, the cut (2.25), generated by the set S of the nodes belonging to the connected component of the source, is added to the formulation and the linear relaxation of the problem is solved again until all the destinations are in the connected component of the source. At this point, if the current solution is integer, then the procedure is interrupted, otherwise a maximum flow problem from the source to each destination with the current y values as capacities is solved (see Definition 1.4.3). If all the maximum flow values are at least one and the current optimal solution is fractional, then the current integer problem is solved and if all the destinations are connected to the source the procedure is interrupted, otherwise the cut (2.25) generated by the set S of the nodes connected to the source is generated and the integer problem is solved again. If at least one maximum flow value is less than one, then we define the set S corresponding to the cuts with minimum capacity (see Proposition 1.4.1), we add these constraints to the current formulation and we solve again the linear relaxation of the current problem. Every time a set of rows is added to the current submatrix, we perform the preprocessing (see Proposition 1.2.1). The procedure sketched above can be formalized by means of the following procedure:

Step 0: Let F be a formulation for problem MPM with only the constraints generated by $S = \{s\}$ and $S = \{s, v_2^s\}$ among the constraints (2.25);

Step 1: Solve the linear relaxation of F, and let $\overline{x}$ be the optimal solution;

Step 2: Define variable y as in Remark 2.3.2 and find the connected component of the source;

Step 3: If there is at least one destination that is not connected to the source, define S, the set of the nodes connected to the source, add the constraint (2.25) relative to S to the current formulation, perform the preprocessing of the constraint matrix and go to Step 1;

Step 4: If all the destinations are connected to the source and the current solution is integer; Stop.

Step 5: If all the destinations are connected to the source and the current solution is fractional go to Step 6;

Step 6: For each source-destination pair, solve the maximum flow problem with the current y as capacities;

Step 7: If all the values of the maximum flow problems are greater than or equal to 1, solve the integer problem, $\overline{x}$ is the optimal solution and go to Step 2;

Step 8: If at least one value of the maximum flow problems is lower than 1; define S corresponding to the minimum capacity cut; add the constraints (2.25) relative to S to the current formulation, perform the preprocessing of the constraint matrix, solve the linear relaxation of the problem and go to Step 6.

The preprocessing of the matrix, used in both methods, consists in finding and erasing the dominated columns and rows. We take advantage of the

fact that the matrix is composed by only ones and zeros and we use the common preprocessing techniques for the Set Covering problem (see Proposition 1.2.1). A dominated column is either a null column or a column whose cost (power) is not smaller than that of another column which is, component-wise, not greater, while a row is dominated if there exists another row of the matrix which is, component-wise, not greater. The convergence of both the procedures is guaranteed because the number of inequalities (2.25) is, albeit huge, finite.

2.8 Experimental Results

We have implemented the solution algorithms in C and we have run the codes on a Dual Intel Xeon 3.2GHz machine with 4 GB RAM memory using the version 9.1 of Cplex as solver.

The experiments have been performed on a set of test problems with increasing number of nodes and of possible destinations; for each problem size, 20 different instances are generated. The nodes of the networks have been uniformly generated on a grid of size 10000×10000 and the source and the destinations have been randomly selected among the generated nodes as well. To obtain the power values from the distances we have set the coefficient κ to 2, while we have set to 3600 seconds the maximum resolution time, after which the solution process is interrupted.

Our computational results have been summarized in Tables 2.2, 2.3 and 2.4 in which we indicate with *Cplex* 9.1 the solution by the integer cplex solver of the entire problem (including all the constraints), with *method I* the method of choosing violated inequalities among all the generated constraints and with *method II* the method in which we generate violated constraints

Table 2.2: Average computational times for randomly generated problems with up to 15 nodes

		Cplex 9.1		***method I***			***method II***		
n	m	T	σ	T	σ	It	T	σ	It
5	1	**0.0000**	0.000	0.0005	0.000	2.1	0.001	0.002	2.8
5	2	**0.0000**	0.000	0.0002	0.000	2.2	0.002	0.004	3.6
5	3	**0.0000**	0.000	0.0002	0.000	2.4	0.001	0.003	4.1
5	4	**0.0000**	0.000	0.0002	0.000	2.6	0.002	0.004	4.5
10	1	0.010	0.005	**0.000**	0.000	2.7	0.003	0.006	5.5
10	2	0.016	0.005	**0.003**	0.004	2.8	0.008	0.009	8.0
10	5	0.025	0.004	**0.002**	0.012	2.9	0.015	0.718	12.3
10	9	0.022	0.004	**0.004**	0.005	3.0	0.024	0.014	15.3
15	1	1.207	0.171	0.073	0.047	3.4	**0.015**	0.022	10.1
15	5	3.849	0.522	0.127	0.046	4.1	**0.079**	0.054	28.5
15	10	4.859	2.217	0.134	0.077	3.6	**0.127**	0.054	36.7
15	14	5.171	2.615	**0.115**	0.061	5.7	0.143	0.058	38.5

on the basis of the nodes reachable by the signal spread by the source. All the methods use Cplex to solve the resulting LP or IP problems.

In the Tables 2.2, 2.3 and 2.4, we report the number of nodes of the network n, the number of destinations m, the average execution time T, its standard deviation σ and the average number of iterations It required to solve the problem. Moreover, in Table 2.4 we report the percentage $NS\%$ of the not solved instances within the time limit.

The best solution average time among the solving procedures is highlighted with a bold character. The results in Table 2.2 are related to networks with 5, 10 and 15 nodes combined with all the possible numbers of destinations. It is clear that for networks with 5 and 10 nodes, all the procedures solve the MPM problem quite quickly; Cplex seems to be more efficient only when $n = 5$, whereas the first method works better when $n = 10$. When we increase the value of n the second method has the best

Table 2.3: Average computational times for randomly generated problems with 20 nodes

		method I			***method II***		
n	m	T	σ	It	T	σ	It
20	1	2.628	1.606	5.8	**0.057**	0.059	19.1
20	5	4.923	2.030	6.4	**0.306**	0.228	45.4
20	10	4.828	2.086	5.4	**0.694**	0.392	62.0
20	15	4.207	1.684	4.9	**0.779**	0.412	65.0
20	19	4.034	1.328	4.1	**0.904**	0.678	66.6

Table 2.4: Average computational times for randomly generated problems with 30, 50 and 100 nodes

		method II			
n	m	T	σ	It	$NS\%$
30	1	1.288	1.315	61.4	
30	10	8.930	6.086	111.7	
30	15	7.789	4.609	108.4	
30	29	9.077	5.325	106.4	
50	1	6.647	7.588	74.7	
50	10	512.223	401.593	294.2	10
50	25	640.236	889.187	248.0	30
50	49	712.714	646.270	214.5	10
100	1	348.916	375.378	143.0	
100	5	927.537	606.565	212.8	60

performance. For networks with 15 nodes, the first method is the most efficient when the number of destination is greater than 10 and so for the broadcasting version of the problem.

In Table 2.3, we present the results for the MPM problem on networks with 20 nodes; while it is not possible to solve any of these problems generating the whole matrix of constraints, the second method outperforms the first method even when $m = n - 1$.

A different situation is shown in Table 2.4. For the MPM problems on networks with more than 30 nodes, the first method fails to solve the problem because of the memory required to generate the whole constraint matrix. On the contrary, the second method is still able to solve the MPM problem on networks with up to 50 nodes, but presently there are still some instances not solved within the time limit of an hour. Instances with 100 nodes have been solved, by now, for just a limited number of destination.

2.9 Concluding Remarks

We have proposed a Set Covering–based formulation for the Minimum Power Multicasting problem in Ad-Hoc networks, and we presented two possible algorithms for its solution. We carried out an experimental study by using a set of test problems randomly generated having a number of nodes ranging from 5 to 100. While we think that the presented formulation represents an original and effective approach to the problem, we are conscious that some improvements should be done. The theoretical and polyhedral properties of the model may be investigated together with a better way of generating violated constraints. In this direction goes the following chapter.

Chapter 3

Chvátal-Gomory cuts for the Multicast polytope

In this chapter, we want to highlight some properties of the polytope of the Set Covering formulation (see Proposition 1.2.2) for the Multicasting problem in the wireless Ad-Hoc networks. The inequalities in section 2.5 can be added to the problem to reduce the feasible region of the MPM problem, but in general they are not able to cut off any optimal fractional solution of the linear relaxation of the problem. The purpose here is to propose heuristics that generate valid inequalities for the Set Covering polytope that cut off fractional optimal solutions of the linear relaxation of the MPM problem. In particular, in section 3.2 we propose two heuristics that find violated inequalities with right hand side two belonging to the first Chvátal closure of the MPM problem's polytope. The optimal value of the linear relaxation of the problems with the cuts generated with the heuristics is compared in section 3.4 with the optimal value obtained by solving the problems over the first Chvátal closure polytope (see section 3.3).

3.1 Introduction

First of all, we give here the definition of a Chvátal-Gomory cut and of the first Chvátal closure polyhedron for a general IP problem. Given the Integer Programming problem:

$$\begin{aligned} &\min c^T x \\ &s.t. \\ &\quad Ax \geq b \\ &\quad x \geq 0, \\ &\quad x \text{ integer} \end{aligned} \tag{3.1}$$

where A is a $m \times n$ real matrix, c and b are a n-dimensional and a m-dimensional vectors respectively and x is a n-dimensional vector of variables that take integer values, a Chvátal-Gomory cut, indicated by CG cut, is defined as follows ([19], [35]):

Definition 3.1.1 (Chvátal-Gomory cut). A *Chvátal-Gomory cut* is a valid inequality for $P_I(A)$ of the form:

$$\lceil u^T A \rceil \, x \geq \lceil u^T b \rceil$$

where $u \in R_+^m$ is the CG multiplier vector and $\lceil .. \rceil$ is the upper integer part.

The first Chvátal closure polyhedron is the polyhedron obtained by intersecting the relaxed polyhedron P(A) with all the CG cuts.

Definition 3.1.2 (First Chvátal closure). The *first Chvátal closure* of $P(A)$ is the polyhedron $P_1(A)$ defined as follows [19]:

$$P_1(A) := \{x \in \mathbb{R}_+^n : \; Ax \geq b, \lceil u^T A \rceil \, x \geq \lceil u^T b \rceil \quad \forall u \in \mathbb{R}_+^m\}.$$

The three polyhedrons are related by the relations

$$P_I(A) \subseteq P_1(A) \subseteq P(A)$$

therefore, $P_1(A)$ is a better approximation of $P_I(A)$ than $P(A)$.

For this reason, we try to find violated CG cuts to cut off fractional solution of the linear relaxation of the Set Covering formulation for the MPM problem.

The Minimum Power Multicast problem can be expressed in a general form:

$$\begin{aligned} &\min p^T x \\ &s.t. \\ &\qquad Bx \geq \mathbf{1} \\ &\qquad x \in \{0,1\}^{|A|} \end{aligned} \tag{3.2}$$

where $B = (b_{ij})_{i \in M, j \in N}$ is a $0-1$ matrix, $p \in R^{|A|}$ is the array of the powers and A is the set of the arcs of the network and M and N are the index sets of the rows and the columns respectively of the matrix B. The Set Covering polytope is denoted by $P_I(B)$ and the relaxed polytope by $P(B)$. We denote once more by n the number of nodes of the wireless network and m the number of destinations.

For the results ([7], [8], [22]) reported in the first introductive chapter (see Proposition 1.2.2), we can make here some remarks about the polytope of the Minimum Power Multicasting problem.

Remark 3.1.1. The polytope $P_I(B)$ is always nonempty (if $n \geq 2$ and $m \geq 1$, then $|N_i| \geq 1$ for all $i \in M$) and it is full-dimensional if $n \geq 3$.

Indeed, in this case, $|M| \geq 2$ and for each $i \in M$ the cardinality of N^i is at least equal to two.

Remark 3.1.2. If $n \geq 4$, then for each $j \in N$ the inequality $x_j \geq 0$ is a facet of $P_I(B)$. In fact, for each $i \in M$ and $j \in N$ the cardinality of $N^i \setminus \{j\}$ is at least equal to two. Furthermore all the inequalities $x_j \leq 1$ with $j \in N$ are facets of $P_I(B)$.

The heuristics that we propose, generate valid inequalities with right hand side equal to two and the principle of construction of these inequalities is the following method proposed in ([7], [8]).

Chvátal Gomory cuts can be generated considering positive linear combination of the rows of the matrix and rounding up to the nearest integer all the coefficients of the obtained inequality. In particular, positive linear combinations can be built selecting a subset U of the set of the row indices M, adding all the inequalities of the problem with index in U, then dividing all the coefficients by $|U| - \epsilon$ for a certain positive small enough ϵ and finally rounding all the coefficients up.

Remark 3.1.3. The CG cut relative to a selected $U \subseteq M$ can be obtained by adding all the inequalities $b_i^T x \geq 1$ with $i \in U$ and dividing the resulting inequality by $|U| - \epsilon$:

$$\frac{1}{|U| - \epsilon} \sum_{i \in S} b_i^T x \geq \frac{|U|}{|U| - \epsilon}$$

and finally rounding both members of the inequality up:

$$\left\lceil \frac{1}{|U| - \epsilon} \sum_{i \in S} b_i^T \right\rceil x \geq 2$$

for $0 < \epsilon < 1$.

Looking at the columns of the submatrix of B constituted by all the rows whose index belong to U, it is easy to give a value to the coefficients of the new inequality, indeed, we have ([7], [8]):

Remark 3.1.4. For each $U \subseteq M$ the coefficients of a CG cut can be obtained in this manner:

$$\pi_j^U = \begin{cases} 0 & \text{if } b_{ij} = 0 \text{ for all } i \in U, \\ 2 & \text{if } b_{ij} = 1 \text{ for all } i \in U, \\ 1 & \text{otherwise,} \end{cases} \tag{3.3}$$

so that the inequality $\pi^U x \geq 2$ is the CG cut relative to the choice of U.

Remark 3.1.5.

(i) If $U = \{i\}$, then the inequality $\pi^U x \geq 2$ reduces to the original row $b_i^T x \geq 1$.

(ii) If $U = M$ and the Multicast problem is a Broadcast problem ($m = n-1$), then the inequality generated by the previous method becomes:

$$\sum_{(i,j) \in A \setminus \{(s, v_n^s)\}} x_{ij} + 2x_{sv_n^s} \geq 2.$$

This inequality means that either the source communicates with its most distant node v_n^s or, in order to satisfy the "wireless" connection with all the other destinations, there must be at least another transmitting node in the network in addition to the source.

(iii) If $U = M$ and $m < n-1$ and k is the position of the most distant destination with respect to the source in the array v^s, then the inequality generated by the previous method becomes:

$$\sum_{(i,j) \in A \setminus \{(s, v_j^s) : 1 \leq j < k\}} x_{ij} + 2 \sum_{j=k}^{n} x_{sv_j^s} \geq 2$$

that means that either the source is assigned the power to reach v_k^s or at least there are two hops in the network.

Before going on, we want to insert here two valid inequalities, one for the Broadcast problem and one for the more general Multicast problem in wireless networks. These inequalities have both right hand side equal to two.

The first inequality is for the Broadcast problem. We recall that v_2^s and v_n^s represent respectively the closest and the most distant node with respect to the source and that $v_n^{v_2^s}$ is the most distant node with respect to the node which is the closest to the source. In this section, we indicate with w the node v_2^s. Two sets $\mathcal{A}$ and $\mathcal{B}$ must be introduced. $\mathcal{A}$ is the set of all the arcs of A outgoing from a node i, different from the source s and the node w and incoming in a node j which is different from w and furthermore, which is more distant with respect to i than the node v_n^w, i.e.

$$\mathcal{A} := \{(i,j) \in A : \ i \in V \setminus \{s,w\},\ j \in V \setminus \{w\},\ d_{ij} \geq d_{i\,v_n^w}\}.$$

Analogously $\mathcal{B}$ is the set of all the arcs of A outgoing from a node i which is different from the source s and the node w and incoming in a node j which is more distant with respect to i than the node v_n^s, i.e.

$$\mathcal{B} := \{(i,j) \in A : \ i \in V \setminus \{s,w\},\ j \in V,\ d_{ij} \geq d_{iv_n^s}\}.$$

Proposition 3.1.1. *The following inequality:*

$$\sum_{i \in V \setminus \{v_1^s, v_2^s, v_n^s\}} x_{si} + 2x_{s\,v_n^s} + \sum_{i \in V \setminus \{v_1^w, v_n^w\}} x_{w\,i} + 2x_{w\,v_n^w} + \\ + \sum_{(i,j) \in \mathcal{A}} x_{ij} + \sum_{(i,j) \in \mathcal{B} \setminus \mathcal{A}} x_{ij} \geq 2 \tag{3.4}$$

is a valid inequality for $P_I(B)$.

In the multicasting case, denoting by v_k^s the most distant destination from the source and by v_h^w the most distant destination with respect to w,

$\mathcal{A}$ is the set of all the arcs of A outgoing from a node i which is different from the source s and the node w and incoming in a node j, different from w, which is more distant with respect to i than the node v_h^w, i.e.

$$\mathcal{A} := \{(i,j) \in A : \ i \in V \setminus \{s,w\},\ j \in V \setminus \{w\},\ d_{ij} \geq d_{iv_h^w}\},$$

and $\mathcal{B}$ is the set of all the arcs of A outgoing from a node i which is different from the source s and the node w and incoming in a node j, which is more distant with respect to i than the node v_k^s, i.e.

$$\mathcal{B} := \{(i,j) \in A : \ i \in V \setminus \{s,w\},\ j \in V \setminus \{s\},\ d_{ij} \geq d_{iv_k^s}\}.$$

Proposition 3.1.2. *The inequality:*

$$\sum_{2<i<k} x_{s\,v_i^s} + 2\sum_{i=k}^{n} x_{s\,v_i^s} + \sum_{2<i<h} x_{w\,v_i^w} + 2\sum_{i=h}^{n} x_{w\,v_i^w} + \sum_{(i,j)\in\mathcal{A}} x_{ij} + \sum_{(i,j)\in\mathcal{B}\setminus\mathcal{A}} x_{ij} \geq 2 \tag{3.5}$$

is a valid inequality for $P_I(B)$.

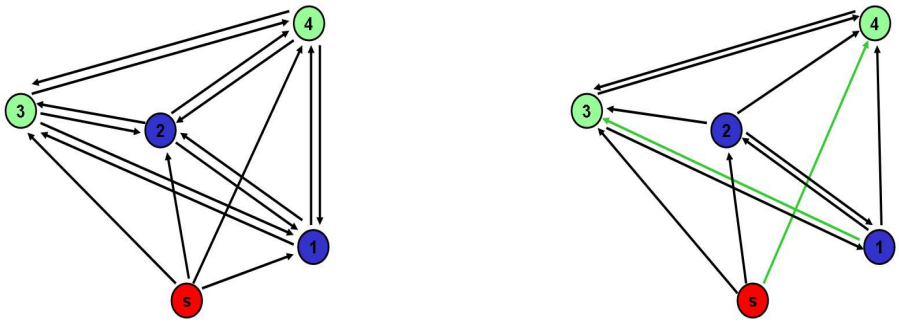

Figure 3.1: An inequality with right hand side two

Naturally, inequality (3.4) is a particular case of inequality (3.5); we give here a simple example for explaining how to construct inequality (3.5).

Example 3.1.1. For the network in Figure 3.1 the distance arrays are the following: $v^s = (s, 1, 2, 3, 4)$, $v^1 = (1, s, 2, 4, 3)$, $v^2 = (2, 3, s, 4, 1)$, $v^3 = (3, 2, s, 4, 1)$, $v^4 = (4, 2, 1, 3, s)$, hence, the set $\mathcal{A} =: \{(2, 3), (2, 4), (2, 1), (4, 3)\}$ and $\mathcal{B} := \{(2, 4), (2, 1), (3, 4), (3, 1)\}$, and the inequality (3.5) is:

$$x_{s2} + x_{s3} + 2x_{s4} + x_{12} + x_{14} + 2x_{13} + x_{23} + x_{24} + x_{21} + x_{43} + x_{34} + x_{31} \geq 2$$

In fact inequality (3.5) forces the source either to reach directly its most distant destination 4 (the green arc in Figure 3.1) or to communicate with a node placed between 1 and 4 and at this point, it is required another transmission to cover node 4. If the source transmits toward its closest node 1, the latter is forced to reach directly its most distant destination 3 (the green arc in Figure 3.1) or to communicate with another node and, in this case, the constraint forces another communication to cover node 3.

3.2 Heuristics for generating a CG cut with right hand side two

The aim of the heuristics is to find CG cuts with right hand side equal to two that cut off fractional optimal solutions of the linear relaxation of the Multicasting problem. Starting with the support of the optimal solution for the LP problem two propositions can be useful. According to Definition 1.1.9, if x^* is an optimal solution of the linear relaxation of the Multicasting problem, its support is the set $Supp := \{j \in N : \ x_j^* > 0\}$, moreover, the set of the column indices j such that $x_j^* = 1$ can be denoted by I , i.e. $I := \{j \in N : \ x_j^* = 1\}$.

Proposition 3.2.1. *([7], [8]) Let $\pi^T x \geq 2$ be an inequality that cuts off the fractional optimal solution x^*, then $\pi_j^U = 0$ for all $j \in I$.*

The above proposition suggests a first criterion for selecting the subset U of M, indeed, we have:

Remark 3.2.1. The set U does not contain any row i of the matrix B such that exists at least a $j \in I$ with $b_{ij} = 1$.

The second proposition is the following:

Proposition 3.2.2. *Let $\pi^U x \geq 2$ be an inequality that cuts off x^*, then for all $i \in U$ it holds that $b_i^T x^* < 2$.*

Hence another rule for selecting the subset U is:

Remark 3.2.2. The set U does not contain any row i of the matrix B such that $\sum_{j \in M} b_{ij}\, x_j^* \geq 2$.

The inputs of the heuristics are a current fractional solution x^* of the linear relaxation of the problem (see 3.2) and the constraint matrix B. The goal is to find a subset $U \subset M$ such that $\pi^U\, x^* < 2$ and, initially, U is set to be equal to M. Using Propositions 3.2.2 and 3.2.1, the heuristics eliminate from U, first of all, all the row indices i such that $b_i^T\, x^* \geq 2$ and then all the row indices i such that $b_{ij} = 1$ and $x_j^* = 1$.

Given a subset U of M, we denote by *value* the quantity:

$$value(U) := \sum_{j \in Supp} \pi_j^U\, x_j^*,$$

where the coefficients π^U are computed using the definition (3.3).

3.2.1 Row-criterion

The elements of the support are ordered in an increasing way with respect to the x_j^*'s value, and, then, if there exist j and k in $Supp$ such that $x_j^* = x_k^*$ the elements of the support are ordered in an increasing way with respect to the number of ones present in the corresponding column in the submatrix whose row indices are in U.

Until a cut is found or all the rows whose indices are in U have been explored,

Step 0: We select a row $i \in U$ and we set $W := \emptyset$;

Step 1: While $value(U \setminus W) \geq 2$ and $|W| < |U| - 1$, iteratively we select a column j in the ordered support such that $b_{ij} = 0$ and we update W, $W := W \cup \{k \in U : \ b_{kj} = 1\}$;

Step 2: If $value(U \setminus W) < 2$ we have found a cut that cuts off the current fractional solution x^* and we add it to the MPM formulation, if, otherwise, $|W| = |U| - 1$ we select a new row $h \in U$ setting again $W := \emptyset$ and we come back to Step 1.

3.2.2 Greedy-criterion

The column j corresponding to the greatest value of x_j^* is selected and the element j is eliminated from the set $Supp$ (that is $Supp := Supp \setminus \{j\}$). All the indices i of the current U such that $b_{ij} = 1$ are eliminated from U, U is updated ($U = U \setminus \{i \in U : \ b_{ij} = 1\}$) and $value(U)$ is computed. While $value(U) \geq 2$ and $|U| > 1$, we choose the column $k \in Supp$ such that the coefficients π^U relative to $U \setminus \{i \in U : \ b_{ik} = 1\}$ give the smallest value of

value among all the possible choices of an element in the current set *Supp*. We updated *Supp* and U, $Supp := Supp \backslash \{k\}$ and $U := U \backslash \{i \in U : \ b_{ik} = 1\}$ respectively and we check again the value of *value* and the cardinality of U. If $value(U) < 2$, the cut whose coefficients are π^U has been found and we add it to the Set Covering formulation for the MPM problem; if $value(U) \geq 2$ and $|U| \leq 1$ with this heuristic no more cuts can be added.

3.3 Most violated inequality over the first Chvátal closure

The heurists find a violated inequality with right hand side equal to two. If one wants to find the most violated inequality over the first Chvátal closure, then a MIP problem which has been proved to be an $NP-$ hard problem [30] must be solved.

Formally the Multicasting problem is:

$$\begin{aligned} &\min p^T x \\ &s.t. \\ &\qquad Bx \geq \mathbf{1} \\ &\qquad -Ix \geq -\mathbf{1} \\ &\qquad x \geq 0, \ x \text{ integer.} \end{aligned} \tag{3.6}$$

If x^* is the optimal solution for the linear relaxation of this problem, then the separation problem, is the problem of finding $u \in \mathbb{R}_+^{|M|}$ and $v \in \mathbb{R}_+^{|N|}$ such that $\left\lceil u^T B - v^T I \right\rceil x < \left\lceil u^T \mathbf{1} - v^T \mathbf{1} \right\rceil$ or proving that no cut is violated, that is, no such u and v exist. If a cut can be found, minimizing the difference: $\left\lceil u^T B - v^T I \right\rceil x - \left\lceil u^T \mathbf{1} - v^T \mathbf{1} \right\rceil$ produces the most violated CG cut.

Remark 3.3.1. The vectors u and v can be assumed to have each component less than one [31] as each coefficient of the problem is integer. In fact, suppose for axample that $u_i \geq 1$ for an $i \in M$. The CG cut associated with u_i is dominated, since it can be obtained as the sum of $\lfloor u_i \rfloor$ times the constraint $b_i^T x \geq 1$ and the CG cut associated with the fractional part of u_i.

Denoted by $\pi := \lceil u^T B - v^T I \rceil$ and by $\pi_0 := \lceil u^T \mathbf{1} - v^T \mathbf{1} \rceil$ for a certain $u \in \mathbb{R}_+^{|M|}$ and $v \in \mathbb{R}_+^{|N|}$, the separation model [31] can be formulated as follows:

$$
\begin{aligned}
&\min \pi^T x^* - \pi_0 \\
&s.t. \\
&\quad \pi_j \geq u^T B_j - v_j \qquad \forall j \in \{1, .., |N|\} \\
&\quad \pi_0 < u^T \mathbf{1} - v^T \mathbf{1} + 1 \\
&\quad 0 \leq u_i \leq 1 - \epsilon \qquad \forall i \in \{1, .., |M|\} \\
&\quad 0 \leq v_k \leq 1 - \epsilon \qquad \forall k \in \{1, .., |N|\} \\
&\quad \pi, \pi_0 \text{ integer}
\end{aligned}
\tag{3.7}
$$

Naturally, even in this case ϵ is a positive, but small enough, real number that has been set to 0.01 as recommended in [31]. To reduce the number of integer variables π one can observe that all the variables x_i with $x_i^* = 0$ do not give any contribution to the objective function value of the separation problem and so, the separation problem itself can be constructed only on the support of the solution x^*. Indeed, for any $j \in N \setminus Supp$ the value of the corresponding π_j can be computed using the optimal value of the variables u and v, that is $\pi_j = \lceil u^T B_j - v_j \rceil$.

The separation problem can be, thus, reduced to the following MIP problem [31]:

$$
\begin{aligned}
&\min \sum_{j \in Supp} \pi_j x_j^* - \pi_0 \\
&s.t. \\
&\quad s_j + \pi_j - u^T B_j + v_j = 0 \quad && \forall j \in Supp \\
&\quad s_0 + \pi_0 - u^T \mathbf{1} + v^T \mathbf{1} = 0 && \\
&\quad 0 \leq u_i \leq 1 - \epsilon && \forall i \in \{1, .., |M|\} \\
&\quad 0 \leq v_k \leq 1 - \epsilon && \forall k \in \{1, .., |N|\} \\
&\quad 0 \leq s_j \leq 1 - \epsilon && j \in Supp \cup \{0\} \\
&\quad \pi_j \text{ integer} && j \in Supp \cup \{0\}
\end{aligned}
\tag{3.8}
$$

where the variables $s_j = \left\lceil u^T B_j - v_j \right\rceil - u^T B_j + v_j$ are slack variables.

3.4 Preliminary computational results

The two heuristics and the exact separation problem have been implemented in C and the codes have run on a Opteron 246 machine with 2 GB RAM memory using the version 9.1 of Cplex as solver.

The experiments have been performed on the set of test problems with increasing number of nodes and of possible destinations generated in chapter 2, whose linear relaxation do not provide an integer solution. While the linear relaxation of the MPM problem provides a fractional solution and a CG cut can be found using the heuristic processes in sections 3.2.1 or 3.2.2 or solving the separation problem (3.8) it is added to the current formulation and the problem is solved again. In the Table 3.1, we want to present the preliminary results obtained with networks with up to 15 nodes. We report there the number of nodes n, the number of destinations m and the seed from which the problem has been generated *seed*.

Table 3.1: Heuristics-Exact problem of generating CG cuts

				3.2.1			*3.2.2*			*3.8*		
n	m	$seed$	$\frac{OPT-LP}{LP}$	CG	Gap	T	CG	Gap	T	CG	Gap	T
10	7	1	0.007	3	0	0.01	1	0.007	0.01	5	0	4
10	8	1	0.007	3	0	0.03	1	0.007	0	5	0	3
10	9	1	0.012	4	0	0.01	2	0.005	0.02	2	0	0.8
15	5	14	0.002	1	0	0.03	1	0	0.04	1	0	0.03
15	6	14	0.002	1	0	0.05	1	0	0.04	1	0	0.03
15	7	14	0.002	1	0	0.03	1	0	0.04	1	0	0.04
15	8	20	0.000	1	0	0.08	1	0	0.06	1	0	0.03
15	9	10	0.139	23	0.024	1.76	9	0.093	0.84	-	-	> 600
15	9	20	0.000	1	0	0.09	1	0	0.07	1	0	0.04
15	10	2	0.005	4	0	1.11	1	0.005	0.15	2	0	123
15	10	10	0.139	23	0.024	1.77	9	0.093	0.83	-	-	> 600
15	10	20	0.033	16	0	0.75	1	0.033	0.19	-	-	> 600
15	11	2	0.005	4	0	1.12	1	0.005	0.13	2	0	125
15	11	10	0.139	33	0.006	3.01	1	0.139	0.17	-	-	> 600
15	11	20	0.034	19	0	0.96	1	0.034	0.19	-	-	> 600
15	12	2	0.005	4	0	1.12	1	0.005	0.14	2	0	425.18
15	12	10	0.152	31	0.001	3.04	1	0.152	0.18	-	-	> 600
15	12	20	0.032	11	0	0.6	1	0.032	0.24	-	-	> 600
15	13	2	0.005	4	0	1.12	1	0.005	0.14	2	0	425.91
15	13	3	0.019	4	0	0.35	1	0.019	0.08	6	0	146.03
15	13	10	0.152	32	0.036	4.86	4	0.128	0.32	-	-	> 600
15	13	18	0.010	7	0	0.42	1	0.010	0.07	-	-	> 600
15	13	20	0.032	11	0	0.61	1	0.032	0.24	-	-	> 600
15	14	2	0.005	4	0	1.11	1	0.005	0.13	2	0	442.75
15	14	3	0.019	6	0	0.36	1	0.019	0.08	6	0	145.65
15	14	10	0.152	34	0.020	2.05	4	0.128	0.31	-	-	> 600
15	14	18	0.004	2	0	0.28	1	0.004	0.11	4	0	86.10
15	14	20	0.032	11	0	0.61	1	0.032	0.23	-	-	> 600

The column $(OPT - LP)/LP$ reports the gap between the optimal solution OPT of the integer problems and the optimal value of their linear relaxations LP. *Gap* is the ratio $(OPT - LP)/LP$ where OPT is the optimal value of the integer problem (3.6) while LP is the optimal value of the linear relaxation of the problem with the addition of the CG cuts that can be generated with the different methods. For each problem, we report the number CG of the CG cut generated, the *Gap* and the computational time T. The computational time T does not include the preprocessing time of the matrix but only the time for solving the linear relaxations of the problems and the time for generating the cuts. If T is greater than 600 seconds, then the computation is interrupted.

Obviously, finding the most violated inequalities in the first Chvátal closure on the basis of the current fractional solutions and inserting them to the formulation, gives the best value of the lower bounds but it is also true that it is too time consuming even for small networks (15 nodes); there are several cases in which the whole problem is not solved within the time limit.

The heuristic of section 3.2.1 provides cuts that reduce strongly the gap and, in most of the considered cases, the optimal solution of the linear relaxation of the problems with the generated CG cuts is integer. However, it generates more cuts than the other approaches and it is not as fast as the procedure with the heuristic in section 3.2.2.

The heuristic in section 3.2.2 is the fastest and it provides few cuts that reduce the gap but not so strongly as for the cuts found with the procedure in section 3.2.1 or solving the problem (3.8); in many cases, also with graphs with 10 nodes, inserting the CG cuts of heuristic in 3.2.2 to the linear relaxation of the problems does not reduce to zero the value *Gap*.

3.5 Concluding remarks

The row-based heuristic is able to generate CG cuts that improve the lower bounds of the linear relaxation of the Set Covering polytope for the Multicasting problems in wireless networks in a reasonable time, but two steps can still be done: the first is to find facet defining inequalities not necessarily belonging to the first closure and the second is to generate facet defining inequalities without scanning, in the worst case, all the rows of the current matrix U (as in the heuristic procedure in section 3.2.1).

The programs Porta [17] (POlyhedron representation transformation algorithm) and cdd [34] have been run on the randomly generated MPM problem in order to obtain an explicit description of the Set Covering polytope. Unfortunately, it is not possible to terminate the programs for the problems whose linear relaxations have a fractional optimal solution, because neither of them is able to provide (in days of computation) the description of the polytope for networks with more than 5 nodes. At present, all the generated graphs with 5 nodes (more than 500 problems have been generated) can be solved just with the linear relaxation of the problem and no more constraints than those that are in the formulations are required in the description of their polytopes.

The effectiveness of the Set Covering formulation (2.25)-(2.26) for the Minimum Power Multicast problems, has been also checked using the tool in [5]. No coefficient of the constraint matrix is strengthened by the code that Andersen *et al.* propose.

Chapter 4

MIP formulations for a probabilistic Broadcasting Minimum Power problem

In this chapter, we consider a new variant of the Minimum Energy Broadcast (MEB) problem: the Probabilistic MEB (PMEB) [63]. As seen in chapter 2, the objective of the classic MEB problem is to assign transmission powers to the nodes of a wireless network is such a way that the total energy used in the transmission is minimized, while a connected broadcasting structure is guaranteed. In the new variant of the problem presented in section 4.1, we take into account a concept of reliability for the nodes with the goal of guaranteeing the broadcasting structure satisfying a chosen reliability level. Three mixed integer linear programming formulations for the new problem are presented in section 4.4, whereas efficient formulation-dependent methods for the solution of the different formulations are described in section 4.5. Computational results, aiming at ranking the proposed approaches, depending on the characteristics of the problems

under investigation, are proposed in section 4.6.

4.1 Introduction

We recall that in Ad-Hoc wireless networks, one terminal can communicate through wireless channels with another terminal using a single hop if the second terminal is within the transmission range of the first one, otherwise a multi-hop communication is required.

A crucial issue in this context consists in assigning a transmission power to each node in order to ensure connectivity of the network, while minimizing the total power expenditure over the network. We consider in this chapter the case of the Broadcast problem, in which a designated source terminal has to communicate with all the other nodes, and we assume to operate on a static network, i.e. distances among terminals are known in advance, together with the characteristics of the environment in which the terminals are operating. However, even if many contributions have been given to the deterministic models for the MEB problem none has considered nodes' reliability. The deterministic assumption represents a poor approximation of the reality; the terminals are, indeed, electronic devices that may be subject to a temporary damage or a permanent failure. This remark suggests the appropriateness of solving the problem as an optimization problem that takes into account the uncertain nature of nodes availability. This is a salient characteristic that makes the problem much more complex to solve than its classic, fully deterministic counterpart. To the best of our knowledge, no mathematical models explicitly incorporating the uncertain availability of the nodes have been proposed so far. We want to provide an original contribution in this direction. More specifically, we present three mixed integer linear programming formulations for a variant of the MEB

problem in which nodes failure is taken into account, and the optimal solution not only minimizes the total transmitting power over the network, but also guarantees a certain reliability level for the whole network, based on assumption about the reliability of the single terminals. The rationale is that in the reality one implicitly accepts that failures will happen in the devices and, therefore, the goal of the PMEB problem is to provide broadcasting structures robust enough to guarantee, in case of failure of some terminals, a reliable connectivity for the remaining terminals.

4.2 Related works

The Minimum Energy Broadcast (MEB) problem and its variants have already been the subject of many works. Both Cagalj *et al.* and Clementi *et al.* have shown its NP-hardness in [13] and [20], respectively. Althaus *et al.* have proposed a mixed integer linear programming model and have developed an exact approach, based on branch and bound, for its solution [1]. Alternative formulations have been suggested and solved to optimality by Das *et al.* in [25]. Montemanni *et al.* have proposed in [60] two mixed integer programming formulations together with a preprocessing rule and some valid inequalities [62]. Several heuristic methods have been also proposed in the literature. Wieselthier *et al.* have developed in [4] the well known BIP (Broadcast Incremental Power) algorithm. Metaheuristic approaches have been suggested by Marks *et al.* in [42] and by Das *et al.* in [23]. More recently, Lagrangian relaxation procedures have been proposed by Altinkemer *et al.* in [3] and by Yuan in [88]. Montemanni *et al.* have used the simulated annealing paradigm to find a near-optimal solution [61]. The cross-entropy metaheuristic has been also employed by Li *et al.* in [54] to define a new probabilistic approach called the RTO (Randomized Tree Op-

timization) algorithm. Another method has been proposed in [42] in which the initial solution is determined by means of a random tree generation within an evolutionary approach.

4.3 Network Model

The mathematical formulation of the MEB problem can be given considering the network as a directed complete graph $G = (V, A)$ where V represents the set of nodes corresponding to the terminals of the network and A is the set of arcs. As in chapter 2, a cost p_{ij} that corresponds to the power required to establish a link from node i to j is associated with each arc $(i, j) \in A$.

The MEB problem consists, therefore, in defining a range assignment r minimizing $\sum_{i \in V} r(i)$, subject to the constraints that a directed path exists from a source node s to all the other nodes in the network.

Another definition of the MEB problem can be given in terms of the optimal arborescence rooted at node s: for a node i and an arborescence T of G, let (i, i_T) be the maximum cost arc originated from i in T, i.e. $(i, i_T) \in T$ and $p_{ii_T} \geq p_{ij}$, for all $(i, j) \in T$. Due to the broadcasting property, the power cost of an arborescence T is then $c(T) = \sum_{i \in V} p_{ii_T}$. It is now easy to observe that an arborescence rooted at s is contained (not necessarily strictly contained) in any valid broadcasting structure. The MEB problem can, therefore, be described as the problem of finding the arborescence T with the minimum power cost $c(T)$.

In reality, some nodes of the network may fail due to technical problems or battery draining. This important aspect is neglected in the models

presented so far in the literature. We aim at starting to close this gap by presenting a model where a concept of reliability, connected with node failures, is taken into account.

In order to consider node failures, we associate with each node i of the network a value $q_i \in]0,1]$ representing the probability that node i will remain active (i.e. it will not fail) for the whole operating time of the network. The value of q_i has to be assigned by the decision makers, and reflects the reliability of each node. Typically it will depend on the physical characteristics of the area where each node is deployed. For example, in military applications a node i close to the enemy will have a high probability to be destroyed, and consequently a small value for q_i. Based on the same idea, a node i deployed in an impervious territory will have again a small value for q_i.

We can now formally define the Probabilistic Minimum Energy Broadcast (PMEB) problem as a MEB problem where a given minimum reliability level $\alpha \in]0,1]$ has to be achieved. Specifically, the reliability level of the paths from s to each other node i of the network will have to be at least α. A more formal definition of the PMEB problem will be given in the remainder of this section, after some important remarks.

The uncertain events characterizing our problem (i.e. node failures) are independent from each other, that is, if a node happens to fail, this does not affect the correct functioning of the other terminals of the network. It is also possible to observe that if nodes i and j have a probability values of functioning q_i and q_j respectively, then link (i,j) has a probability value of being available equal to the product $q_i q_j$. The same reasoning can be extended to paths: the probability of a multi-hop transmission path from node i to node j is equal to the product of the probabilities q_k associated

with the nodes involved in the path. In mathematical terms:

$$\mathcal{P}(P_{ij}) := \prod_{v \in P_{ij}} q_v,$$

where P_{ij} represents the path connecting i to j under investigation, and $\mathcal{P}$ is the probability function.

Finally, we would like to observe that, since $\mathcal{P}(P_{sj}) \leq q_s q_j$ for each $j \in V \setminus \{s\}$ (because s and j will be the extremes of each path from s to j), a feasible solution to the PMEB problem can exist if and only if $q_s q_j \geq \alpha$ for each $j \in V \setminus \{s\}$. We suppose again that there are no limits in the transmission power that can be assigned to the nodes, so that the arcs (s, j) are always elements of A.

4.4 Mixed integer linear programming formulations

For the PMEB problem, the decision variables are a set of continuous variables y representing the transmission power of each node, i.e. $y_i := r(i)$ for each $i \in V$, and a second set of integer variables z, that describe the optimal arborescence structure, and that are defined as follows:

$$z_{ij} := \begin{cases} 1 & \text{if } (i,j) \in T, \\ 0 & \text{otherwise,} \end{cases}$$

where T represents the arborescence connecting the source s with all the other nodes of the network.

4.4.1 F_1: Path-Based formulation

Let $\mathcal{U}$ represent the set of all infeasible paths originated in s. The generic element P of $\mathcal{U}$ verifies the condition that the product of the probabilities of the nodes involved in path P is less than the reliability level α, i.e.

$$\mathcal{U} := \{P : P \text{ is an } s-k \text{ path for } k \in V \setminus \{s\}, \text{ such that } \prod_{i \in P} q_i < \alpha\}. \quad (4.1)$$

Notice that the set $\mathcal{U}$ potentially has a huge cardinality and for this reason, it will be used in an implicit way in the method we propose, as described in the following sections.

The first MIP formulation F_1 that we propose for the PMEB is as follows:

$$\min \sum_{i \in V} y_i \quad (4.2)$$

$$s.t.$$

$$y_i \geq p_{ij} z_{ij} \qquad \forall\, (i,j) \in A \quad (4.3)$$

$$\sum_{\substack{(i,j) \in A, \\ i \in S,\, j \in V \setminus S}} z_{ij} \geq 1 \qquad \forall\, S \subset V,\ s \in S \quad (4.4)$$

$$\sum_{(i,j) \in P} z_{ij} \leq |P| - 1 \qquad \forall\, P \in \mathcal{U} \quad (4.5)$$

$$z_{ij} \in \{0,1\} \qquad \forall\, (i,j) \in A \quad (4.6)$$

$$y_i \in I\!R^+ \qquad \forall\, i \in V. \quad (4.7)$$

Constraints (4.3) establish the relation between variables z and y. Constraints (4.4) represent the connectivity requirements: for each partition

(S, S^c) such that $s \in S$ and $S^c \neq \emptyset$, there must be at least an arc outgoing from S and incoming in S^c. Inequalities (4.5), ensure the reliability constraints across all the source-destination paths: if the source s and a destination t are connected by the path $P \in \mathcal{U}$, and, hence, by a path that do not respect the reliability level, then constraint (4.5) excludes the path P from any feasible solution. Finally, constraints (4.6) are the binary restrictions on the variables and constraints (4.7) define the domain definition for the continuous y variables.

Since the cardinality of $\mathcal{U}$ will be large already for small values of $|V|$, handling $\mathcal{U}$ efficiently becomes a critical issue. For this reason in our method for solving F_1, we will initially omit constraints (4.5), and we will generate them in a dynamic way only when they are violated. An analogous reasoning can be applied also to constraints (4.4), that are present again in a huge number. The procedure will be explained in details in section 4.5.1.

4.4.2 F_2: Cumulative Probability formulation

The idea behind our second PMEB model is to get rid of set $\mathcal{U}$ used in formulation F_1 and to introduce a new variable associated with each node k of the network expressing the probability value accumulated till that node along the arborescence. Such a variable can be defined as the product of the probability values of the nodes along the $s - k$ path. Instead, we will use here a continuous variable τ_k, for $k \in V$ equivalently defined as the sum of the logarithm of the probability values of the nodes along the $s - k$ path. The use of the logarithm will be clarified in a formal way in the next section 4.4.3.

It is worth noting that variables τ_k are, indeed, state variables since they depend on the values assumed by variables z, that are still present in this

formulation with the same meaning as in section 4.4.1. Also variables y have the same meaning as in 4.4.1.

The model F_2 can be, thus, formulated as follows:

$$\min \sum_{i \in V} y_i \tag{4.8}$$

$s.t.$

$$y_i \geq p_{ij} z_{ij} \qquad \forall\, (i,j) \in A \tag{4.9}$$

$$\sum_{\substack{(i,j) \in A, \\ i \in S,\, j \in V \setminus S}} z_{ij} \geq 1 \qquad \forall\, S \subset V,\ s \in S \tag{4.10}$$

$$\tau_i \leq \tau_j + \log q_i + M(1 - z_{ji}) \qquad \forall\, (i,j) \in A \tag{4.11}$$

$$\tau_s = \log q_s \tag{4.12}$$

$$\tau_i \geq \log \alpha \qquad \forall\, i \in V \tag{4.13}$$

$$\tau_i \leq 0 \qquad \forall\, i \in V \tag{4.14}$$

$$y_i \in I\!R^+ \qquad \forall\, i \in V \tag{4.15}$$

$$z_{ij} \in \{0,1\} \qquad \forall\, (i,j) \in A. \tag{4.16}$$

While most of the constraints are common with the model presented in subsection 4.4.1, some others are specific for the Cumulative Probability model and deserve some description. For each arc $(i,j) \in A$ constraint (4.11) updates, through a recursive process, the value of τ_i whenever node i is reached directly from node j. Clearly, such a constraint should be meaningful only if arc (i,j) belongs to the arborescence, otherwise it should become redundant. This is guaranteed by means of the term $M(1-z_{ji})$, that appears in the right hand side of the constraint. It dominates the inequality whenever $z_{ji} = 0$ for a big enough coefficient M (it suffices for M to take the value in (4.17)), and vanishes otherwise. Constraint (4.12) initializes

the recursive process by assigning $\log q_s$ to τ_s. The set of constraints (4.13) imposes the reliability requirement on each terminal of the network. Finally, constraints (4.14), (4.15) and (4.16) define variables domains. We notice that variables τ_i take nonpositive values since they are sums of logarithms of values belonging to the $]0, 1]$ interval.

This formulation uses, within constraints (4.11), a constant M whose value must be sufficiently big. In this specific context, it is possible to show that M can be set, for example, to

$$M := -(n+1) \min_{i \in V} \log q_i \tag{4.17}$$

in order to guarantee the reliability level satisfaction.

It is possible, however, to strengthen constraints (4.11) by defining a specific constant M for each node i, in such a way that constraints (4.11) become redundant when arcs (j, i) do not belong to the arborescence T. A choice for these constants, using constraints (4.13), can be the following:

$$M_i := -\log \alpha - \log q_i \qquad \forall i \in V. \tag{4.18}$$

By setting these constants to the previous values, constraints (4.11) of the Cumulative Probability formulation can be replaced by the constraints:

$$\tau_i \leq \tau_j + \log q_i + M_i(1 - z_{ji}) \qquad \forall\, (i, j) \in A. \tag{4.19}$$

This latter strengthened version of the constraints will be, thus, used in the formulation and for the experiments presented in section 4.6.

4.4.3 F_3: Multicommodity Flow formulation

The formulation presented in this section is based on a Multicommodity Flow model as described, for example, in [57]. It includes into the model an

explicit representation of all the paths connecting the source s to each node $d \in V$. For this goal, we do not use the spanning arborescence variables z and we introduce, for each node $d \in V$ and each arc $(i,j) \in A$, a new variable denoted by t_{ij}^d that takes value 1 if arc (i,j) is on the path from s to d, and 0 otherwise, in fact, it represents the value of the commodity d flowing through the arc (i,j). Variables y remains the same as before, and have the same meaning as in sections 4.4.1 and 4.4.2.

The model F_3 can be thus summarized as:

$$\min \sum_{i \in V} y_i \tag{4.20}$$

s.t.

$$y_i \geq p_{ij} t_{ij}^d \qquad \forall\,(i,j) \in A, \forall\, d \in V \setminus \{s\} \tag{4.21}$$

$$\sum_{j \in V \setminus \{s\}} t_{sj}^d = 1 \qquad \forall\, d \in V \setminus \{s\} \tag{4.22}$$

$$\sum_{i \in V \setminus \{d\}} t_{id}^d = 1 \qquad \forall\, d \in V \setminus \{s\} \tag{4.23}$$

$$\sum_{i \in V \setminus \{j\}} t_{ij}^d - \sum_{i \in V \setminus \{j\}} t_{ji}^d = 0 \qquad \forall\, d \in V \setminus \{s\},\ \forall\, j \in V \setminus \{s,d\} \tag{4.24}$$

$$q_d \prod_{i \in V} q_i^{\sum_{j \in V \setminus \{i\}} t_{ij}^d} \geq \alpha \qquad \forall\, d \in V \setminus \{s\} \tag{4.25}$$

$$t_{ij}^d \in \{0,1\} \qquad \forall\,(i,j) \in A,\ \forall\, d \in V \setminus \{s\} \tag{4.26}$$

$$y_i \in I\!R^+ \qquad \forall\, i \in V \tag{4.27}$$

The objective function (4.20) of this model remains unchanged with respect to the other formulations F_1 and F_2. Constraints (4.21) regulates the power emitted by node i based on the value of variables t. The sets of constraints (4.22)–(4.24) are the usual multicommodity flow equations

that guarantee, for each possible source-destination pair, the flow conservation on the source node, on the destination node, and on any intermediate node, respectively. We remark that $\sum_{j \in V \setminus \{i\}} t_{ij}^d = 1$ if node i is on the active path from s to d. Constraints (4.25) are the reliability requirements, and finally, constraints (4.26) and (4.27) are the limitations on the decision variables. The Multicommodity Flow formulation F_3 is a non-linear programming model because of the presence of the set of reliability constraints (4.25). Such constraints could be, however, linearized by making use of the logarithmic properties, as follows:

$$\begin{aligned} \log\left(q_d \prod_{i \in V} q_i^{\sum_{j \in V \setminus \{i\}} t_{ij}^d}\right) &= \log q_d + \sum_{i \in V} \log\left(q_i^{\sum_{j \in V \setminus \{i\}} t_{ij}^d}\right) \\ &= \log q_d + \sum_{i,j \in V,\, j \neq i} t_{ij}^d \log q_i. \end{aligned} \tag{4.28}$$

Constraints (4.25) can be, thus, replaced by the following linear constraints:

$$\sum_{i,j \in V,\, j \neq i} t_{ij}^d \log q_i + \log q_d \geq \log \alpha \qquad \forall d \in V \setminus \{s\} \tag{4.29}$$

These considerations above are the motivations on the use of the cumulation of the logarithms of the probability values for the nodes in formulation F_2 instead of the product of the probability values accumulated along the paths.

4.5 Algorithms for the MIP formulations

Here we present the methods for solving the different formulations presented in section 4.4.

4.5.1 Algorithm for F_1

The drawback of formulation F_1 is represented by the sets of constraints (4.4) and (4.5) that are in an intractable number, from a practical point of view. However, since only a small fraction of these constraints is saturated at optimality, we choose to solve the problem by means of an iterative approach. Namely, constraints (4.4) and (4.5) are initially not considered, and a subset of them will be inserted into the formulation only in case the current optimal solution violates them. This iterative mechanism will be repeated until a solution that respects all constraints (4.4) and (4.5) (both those explicitly added to the formulation and those implicitly checked) is found.

The procedure sketched above can be formalized by means of the following procedure:

Step 0: Let F_1' be formulation F_1 for problem $PMEB$ without constraints (4.4) and (4.5);

Step 1: Solve F_1', and let $(\overline{y}, \overline{z})$ be the optimal solution;

Step 2: If $\overline{z}$ violates a constraint ctr_4 of type (4.4) (the separation routine will be described later), then add ctr_4 to F' and go to Step 1;

Step 3: If $\overline{z}$ violates a constraint ctr_5 of type (4.5) (the separation routine will be described later), then add ctr_5 to F' and go to Step 1;

Step 4: $(\overline{y}, \overline{z})$ is the optimal solution of F_1 (and not only of F_1').

Notice that the procedure converges after a limited number of iterations since the number of inequalities (4.4) and (4.5) is, albeit significant, finite.

It is important to observe that a speed-up may be obtained by first considering the linear relaxation of F_1' in Step 2, and adding the corresponding violated constraints of type (4.4). In this way, many of the constraints might be added before considering the (more time consuming) integer program F_1'. In section 4.6 some results that confirm the correctness of this idea will be presented.

We however did not implement the speed-up since the computation times reported in section 4.6.3 indicate that the method based on F_1 is already the fastest one for some types of problems (without considering the linear relaxation first). On the other hand, the method is far from being the best one on problems with different characteristics.

Separation of inequalities (4.4) Once a solution $(\overline{y}, \overline{z})$ of F_1' is available, the presence of violated inequalities of type (4.4) of F_1 not inserted into F_1' can be easily detected. We use a set L containing all the nodes of the connected component of the source, that is for each node $i \in L$ the exists a directed path from the source to i using the arcs in which the values of the variables z are equal to 1. Two situations are possible at this point:

(i) if $|L| = |V|$, then no violated constraint of type (4.4) exists in the current solution $(\overline{y}, \overline{z})$;

(ii) if $|L| < |V|$, then a violated constraint of type (4.4) has been identified. Therefore, we can add the following violated inequality to F_1:

$$\sum_{i \in L, j \in V \setminus L} z_{ij} \geq 1.$$

Separation of inequalities (4.5) Once a solution $(\overline{y}, \overline{z})$ of F_1' is available, the presence of violated inequalities of type (4.5) of F_1 not inserted into

F_1' can be detected as follows. Since variables $\overline{z}$ define an arborescence (no violated constraint of type (4.4) exists because of the structure of the algorithm), it is enough to calculate, for each $k \in V \setminus \{s\}$, the following value:

$$R_{sk}^{\overline{z}} := \prod_{i \in P_{sk}^{\overline{z}}} q_i$$

where $P_{sk}^{\overline{z}}$ is the set of nodes encountered along the (unique) path from s to k on the arborescence defined by variables $\overline{z}$. In our current implementation of the algorithm, we visit the arborescence defined by variables $\overline{z}$, and as soon as we identify a path from s to k (with k possibly not a leaf) with $R_{sk}^{\overline{z}} < \alpha$, we add the constraint of type (4.5) corresponding to $P_{sk}^{\overline{z}}$ to model F_1'. After a constraint has been added, we do not stop the separation procedure, but we seek for other violated constraints, i.e. more than one constraint can be added at each invocation of the separation routine.

4.5.2 Algorithm for F_2

Similarly to what happens in the Path-Based formulation (see section 4.5.1), subtour elimination constraints (4.10) are in a very large number, too. Therefore, in order to solve the problem F_2, we need to run an iterative approach, starting with a relaxation of this formulation. The procedure we use, which is formally defined in the reminder of this section, is very similar to that described in section 4.5.1 for the Path-Based formulation. The main difference between the two solution approaches is that in this case we have only one set of critical inequalities to be added whenever violated (instead of the double set in case of the Path-Based formulation). The procedure can be formalized by means of the following procedure:

Step 0: Let F_2' be formulation F_2 for problem $PMEB$ without constraints

(4.10);

Step 1: Solve F_2', and let $(\overline{y}, \overline{z}, \overline{\tau})$ be the optimal solution;

Step 2: If $\overline{z}$ violates a constraint ctr_{10} of type (4.10) (the separation routine is analogous to that described in section 4.5.1 for the separation of inequalities (4.4)), then add ctr_{10} to F_2' and go to Step 1;

Step 3: $(\overline{y}, \overline{z}, \overline{\tau})$ is the optimal solution of F_2 (and not only of F_2').

Notice that the procedure converges after a limited number of iterations since inequalities (4.10) are in finite, although often huge, number.

An observation analogous to that reported in section 4.5.1 for the method based on formulation F_1 can be done here. In particular, a theoretical speed-up for the method might be obtained by considering first the linear relaxation of F_2' in step 3, for the generation of violated constraints (4.10). However, the results we will report in section 4.6, clearly indicate that this is not the case for the method based on formulation F_2.

4.5.3 Algorithm for F_3

The Multicommodity Flow formulation may have a large number of variables but it does not have critical constraints (like (4.4) and (4.5) in F_1 and (4.10) in F_2) that impose the development of a specific solution technique. Formulation F_3 can be, thus, directly solved by any mixed integer linear programming solver.

4.6 Experimental Results

This section presents the computational experience carried out with the exact methods described in section 4.5. Two different types of experiments will be discussed, covering the following aspects:

- how many constraints of type (4.4) and (4.5) (respectively (4.10)) are generated during the execution of the method based on formulation F_1 (respectively F_2);
- computation times of the three methods: we want to estimate the largest problem which is possible to solve with the methods we propose, and at the same time understand which is the most promising approach, depending on the characteristics of the problem under investigation.

First of all, we describe the characteristics of the benchmarks used for the experiments.

4.6.1 Benchmark description

No benchmark is available from the literature, being the problem treated here for the first time. We have, therefore, generated a set of random instances, trying to produce realistic scenarios.

The nodes have been chosen uniformly in a 5000×5000 grid and the probability that any of the nodes is functioning is assumed to be uniformly distributed in the interval [0.85, 0.95]. These values should be reasonable for real-life applications. Moreover, the value of the coefficient κ, which models signal propagation, has been set to 2.

The three methods described in section 4.5 have been implemented in C and the experiments have been carried out on an Intel Celeron 1.3 GHz / 256 MB machine. The callable library version of CPLEX 9.0 has been used as mixed integer programming solver. Ten random instances have been generated for each problem considered, and a maximum computation time of 3600 seconds has been allowed for each instance.

Table 4.1: Average number of constraints generated while solving the Path-Based formulation F_1 and the Cumulative Probability formulation F_2.

		F_1		F_2
$\|V\|$	α	(4.4)	(4.5)	(4.10)
10	0.50	2.75	0.50	0.00
10	0.60	7.75	4.10	0.00
10	0.70	28.50	39.60	0.00
10	0.80	94.00	46.60	0.00
15	0.50	11.50	0.60	0.00
15	0.60	42.75	44.60	0.00
20	0.50	17.00	23.00	0.00
20	0.60	43.75	82.40	0.00

4.6.2 Number of constraints added

In Table 4.1, we present, for a subset of the problems we will consider in section 4.6.3, the number of constraints (4.4) and (4.5) generated while solving the Path-Based formulation F_1 as described in section 4.5.1, and the number of constraints (4.10) generated while solving the Cumulative Probability formulation F_2 as described in section 4.5.2. In Table 4.1, we report, for each problem considered, the average number of constraints generated.

From Table 4.1, it can be observed how, during the solution of formulation F_1, a considerable number of constraints (4.4) and (4.5) are generated. Moreover, a weak correlation seems to exist among the number of constraints generated for the two families. This result suggests that a speed-up for the solution method described in section 4.5.1 may be obtained by considering the linear relaxation of F_1 for the generation of constraints (4.4) (as suggested in section 4.5.1).

Table 4.2: Computational results for the methods in section 4.5.

		Path–Based F_1			***Cumulative Probability*** F_2			***Multicommodity Flow*** F_3		
$\|V\|$	α	T (sec)	σ (sec)	OOT	T (sec)	σ (sec)	OOT	T (sec)	σ (sec)	OOT
10	0.50	**0.58**	0.71	-	4.67	10.92	-	1.56	1.54	-
10	0.60	**1.55**	2.14	-	3.11	5.71	-	2.26	2.08	-
10	0.70	41.38	52.21	-	14.67	18.71	-	**0.46**	0.70	-
10	0.80	309.84	536.32	-	54.55	51.43	-	**0.05**	0.04	-
15	0.50	**4.64**	3.02	-	65.68	91.15	-	58.10	28.78	-
15	0.60	237.11	540.40	-	459.35	593.40	-	**109.66**	80.38	-
15	0.70	2338.25	1558.93	5	2097.16	1580.71	4	**4.02**	5.56	-
15	0.80	-	-	10	2935.27	1330.43	8	**0.074**	0.01	-
20	0.50	**365.95**	626.78	-	2017.29	1630.16	5	2863.30	952.29	5
20	0.60	**2032.40**	1555.29	5	2710.02	1367.56	7	2267.56	1370.81	5
20	0.70	3364.93	964.74	9	3269.61	991.38	9	**93,72**	200.91	-
20	0.80	-	-	10	-	-	10	**0.21**	0.01	-
25	0.70	-	-	10	-	-	10	**949.33**	1240.36	1
25	0.80	-	-	10	-	-	10	**0.42**	0.02	-
30	0.70	-	-	10	-	-	10	**1809.67**	1791	5
30	0.80	-	-	10	-	-	10	**0.78**	0.05	-

Even more interesting is the situation for constraints (4.10), generated while solving formulation F_2: none of these constraints is generated during the experiments summarized in Table 4.1. The results suggest that considering the linear relaxation of F_2 first, to generate constraints (4.10) in the algorithm discussed in section 4.4.2, would not improve the overall

computation times of the method.

4.6.3 Computation times

Computational results for the algorithms discussed in section 4.5 are summarized in Table 4.2. For each method and for each problem considered we report the average T and standard deviation σ for the execution time (in seconds) and the number of instances not solved to optimality in the given time limit (OOT, out of time). When not all the problems are solved to optimality, only the instances solved to optimality concur to the computation of T and σ. Different values for the reliability threshold of the network α are finally considered. For each problem considered, the best value for T is in bold.

From the results reported in Table 4.2, the exact method based on the Path-Based formulation F_1 appears to be the most efficient approach for small networks (i.e. with at most 15 nodes) and for low values of the reliability threshold α. On the other hand, as the value of α increases, the approach based on the Multicommodity Flow formulation F_3 outperforms by far the other methods, reaching the point of becoming the only method able to solve many of the problems in the given time limit.

It is also interesting to observe how, for most of the problems, the average computational time required to solve the Multicommodity Flow model F_3 decreases as the value of α increases. When α increases, several paths are preliminarily discarded because the product of the probabilities associated with their nodes does not reach the threshold.

A final remark is about the potential speed-up for the method based on model F_1, achievable by considering the linear relaxation of the formulation

Table 4.3: Additional computational results for the Multicommodity Flow formulation F_3.

$\|V\|$	α	T	OOT
25	0.75	2.923	-
30	0.75	47.47	-
35	0.75	90.59	-
40	0.75	936.22	2
45	0.75	1810.50	3
50	0.75	2788.13	5

first while generating violated constraints (4.4). Even if such a speed-up is likely to exist (see section 4.6.2), it would definitely not close the gap between the performance of the methods based on F_1 and F_3 for the problems where the latter is the fastest method.

This attractive performance of the Multicommodity Flow model F_3 suggests to solve larger problems. Indeed, Table 4.3 summarizes the average computational times (and number of instances not solved to optimality) for test problems with up to 50 nodes by setting a constant value of 0.75 for α. The results show how both the computational times T and the number of instances not solved within the required amount of time OOT increase quite drastically as the number of nodes increases. This is related to the explosion in size of formulation F_3. Nevertheless, the method based on model F_3 remains the only one, among those considered, which is able to handle problems with up to 50 nodes in the given time.

4.7 Conclusions

In this chapter we have studied the Minimum Broadcast problem for Ad-Hoc wireless and sensor networks in probabilistic settings. The possible failure of any node in the network is considered explicitly within the mathematical representation of the problem, in order to provide more robust solutions with a given level of reliability. We proposed three different mixed integer linear programming formulations for the problem, and we developed an efficient solution approach for each of them.

Experimental results, aiming at understanding how the different methods perform, have finally been presented. These experiments, carried out on instances with up to 50 nodes, suggest that one method dominates the other two, when reasonable reliability levels are considered.

Chapter 5

Delay-constrained Steiner Tree problem

The problem we want to deal with in this chapter is the minimum Steiner Tree with Delay constraints which has been proved to be an NP–Complete problem. In Multicast problems, indeed, one of the crucial aspects can be the Quality of Service requirement, in particular in communications not only the costs should be minimized but a time limit warranty in the reception of the forwarded messages should be considered. For this reason, we address here a Delay-constrained version of the Steiner Tree problem [51] that may find immediate application in Ad-Hoc wireless networks introducing, also in this context, the Quality of Service requirements ([37], [43]). We present several valid MIP formulations in section 5.2 comparing the respective LP relaxation (in section 5.5). In section 5.6 we describe some preprocessing procedures to reduce the size of the problems. We present exact procedures for solving the problems and some computational results in section 5.7 and 5.9 respectively.

5.1 Introduction and Related works

The Steiner Tree problem is an NP-Hard problem with a long history ([29], [41]) and in the last 20 years it has been well studied and solved ([2], [47], [57], [69]), since several practical problems can be modelled as a Steiner Tree problem. Recently some variants of the classical Steiner Tree problem have been taken into account on the influence of new problems in communications with the introduction of the Quality of Service (QoS) requirement or with the restriction on the maximum degree of the nodes (Degree-constrained Steiner Tree problem). The pure Steiner Tree problem (see Definition 1.5.1) on the graph $G = (V, E)$ is the problem of finding a tree with the minimum total cost connecting a required set of nodes R, subset of V, making possibly use of the other nodes of the graph. The Steiner Tree problem can be extended taking into account the concept of the QoS requirement. Indeed, it could be useful and appreciable in practice to guarantee the connection of a source with the nodes in R within a time limit. In particular in communication networks, messages sent by a source towards all the members of a multicast group can be required to be delivered within a maximum delay ([49], [66]). Naturally, the QoS constraints and, specifically, the maximum delay constraints impose a restriction on an acceptable multicast tree. Only recently, the Delay-constrained Steiner Tree problem has been object of study, specially, with the developments of the multimedia technology. In fact, real-time applications need to transmit information within a certain amount of time and so a message generated by one source of the network has to reach a set of target devices for delivering the same information in a fixed delay limit.

Many heuristics for solving the problem have been proposed for both static and dynamic networks ([49], [50], [79], [80]). Kompella *et al.* in [49] present greedy heuristics where they find a spanning tree of the closure

graph of the constrained shortest path between the source and the required nodes, while Sriram *et al.* in [79] propose two algorithms for sparse and static communication groups divided into two phases: the first computes all the possible shortest paths from the source to each terminal respecting the maximum delay requirement and the second uses these paths for constructing the multicast tree. Zhu *et al.* [89] propose a heuristic based on a feasible search optimization method that starts with the minimum delay tree and then decrease the costs of the delay-bounded tree. An integer programming formulation together with an exact solution technique can be found in [65] by Noronha *et al.*. In Tseng *et al.* [80], a genetic algorithm and a mixed integer formulation for the Delay and Degree-constrained Broadcast problem is presented, whereas a simulated annealing method is proposed in [50] for a distributed multicast routing in Delay-constrained Steiner Tree problem. The problem of the QoS in a Minimum Energy Multicast problem in wireless Ad-Hoc networks has been already considered and mixed integer programming formulations for the QoS-MPM problem have been proposed in [37] and [43].

5.2 Mixed integer programming formulations

Let $G = (V, E)$ be an undirected graph. With each edge $e = \{i, j\} \in E$, two nonnegative real numbers are associated: the cost c_e and a delay del_e which represents the time needed to run along the edge e. The directed graph associated with $G = (V, E)$ is denoted by $G = (V, A)$, where the set A is the set of the directed arcs (i, j) and (j, i) corresponding to the undirected edge $e = \{i, j\} \in E$. We suppose that both the costs and the delays are symmetric, i.e. for every (i, j) and (j, i) in A we have $c_{(i,j)} = c_{(j,i)}$ and $del_{(i,j)} = del_{(j,i)}$. For simplifying the notation we write c_{ij} and del_{ij}

instead of $c_{(i,j)}$ and $del_{(i,j)}$, respectively. A source node s and a set R of destinations are selected among the elements of V; all the other nodes of the network (different from the source an not belonging to R) are the Steiner nodes.

The Delay–constrained Minimum Steiner Tree problem consists in finding a tree T connecting the source s with every terminal node in R (possibly making use of the Steiner nodes) with the minimum total cost $c(T)$, while respecting a fixed maximum delay $\Delta \in \mathbb{R}^+$. For each $t \in R$, if $P_{(s,t)}$ is a feasible path connecting the source s to the terminal t, then it must hold:

$$\sum_{(i,j)\in P_{(s,t)}} del_{ij} \leq \Delta.$$

Given a path $P_{(i,j)}$ from i to j, we denote by $Del(P_{(i,j)})$ the sum of the delays of the arcs of $P_{(i,j)}$:

$$Del(P_{(i,j)}) := \sum_{(k,h)\in P_{(i,j)}} del_{kh}.$$

In order to model the problem the state link variables y are introduced. For each arc $(i,j) \in A$, the boolean variable y_{ij} indicates whether or not the arc (i,j) belongs to the arborescence T connecting the source with the destinations, i.e.

$$y_{ij} := \begin{cases} 1 & \text{if } (i,j) \in T, \\ 0 & \text{otherwise.} \end{cases}$$

In the following subsections, we present four different mixed integer programming formulations for the minimum Steiner Tree problem with Delay constraints.

5.2.1 F_1: Degree-constrained Minimum Spanning Tree formulation with Delay constraints

As done in [57] for the Steiner Tree problem, the first formulation finds a Degree-constrained Minimum Spanning Tree T_0 respecting the Delay constraints on a modified network $G_0 = (V_0, A_0)$ obtained introducing another node 0 in the graph $G = (V, A)$. The set V_0 is the set of all the elements of V with the addition of the node 0, that is, $V_0 := V \cup \{0\}$ and the set A_0 is the set of all the arcs in A and of all the arcs $(0, i)$ with $i \in V \setminus R$, that is, $A_0 := A \cup \{(0, i) : i \in V \setminus R\}$. All the new directed arcs $(0, i) \in A_0 \setminus A$ have costs c_{0i} and delays del_{0i} equal to zero. On the graph $G_0 = (V_0, A_0)$, we want to find the Degree-Delay-constrained Minimum Spanning Arborescence T_0 such that the new node 0 is directly connected to the source and all the Steiner nodes $i \in V \setminus (R \cup \{s\})$ adjacent to 0 have degree 1 (i.e. if the arc $(0, i) \in T_0$, then for every (j, i) or (i, k) belonging to A neither (j, i) nor (i, k) are in the arborescence T_0) and all the required nodes are reached within the maximum time limit Δ.

Moreover, with each node of the graph $i \in V$ is associated a continuous variable t_i which represents the time when the node i is reached in the arborescence from s to each terminal in R. These variables are bounded to take positive values not greater than Δ i.e.:

$$t_i \in [0, \Delta] \qquad \forall i \in V \setminus \{s\},$$

and naturally $t_s := 0$.

The formulation, that we refer to as F_1, can be expressed as follows:

$$\min \sum_{(i,j)\in A} c_{ij} y_{ij} \tag{5.1}$$

$s.t.$

$$\sum_{(i,j)\in\delta^-(j)} y_{ij} = 1 \qquad \forall\, j \in V \tag{5.2}$$

$$\sum_{(i,j)\in\delta^+(i)} y_{ij} \geq 1 - y_{0i} \qquad \forall\, i \in R^c \tag{5.3}$$

$$y_{0j} + y_{ij} + y_{ji} \leq 1 \qquad \forall\, j \in R^c \setminus \{s\}, (j,i) \in \delta^+(j) \tag{5.4}$$

$$y_{0s} = 1 \tag{5.5}$$

$$t_i - t_j + M_{ij} y_{ij} + \alpha_{ji} y_{ji} \leq M_{ij} - del_{ij} \qquad \forall\, (i,j) \in A \tag{5.6}$$

$$0 \leq t_i \leq \Delta \qquad \forall\, i \in V \setminus \{s\} \tag{5.7}$$

$$t_s = 0 \tag{5.8}$$

$$y_{ij} \in \{0,1\} \qquad \forall\, (i,j) \in A_0. \tag{5.9}$$

Constraints (5.2) together with constraints (5.9) build a spanning arborescence rooted at 0 in G_0: in every feasible solution there is exactly one arc of the graph incoming in each node of V. Constraints (5.3) together with (5.4) are the requirements on the degree of the Steiner nodes and constraint (5.5) forces the new node 0 to be connected to the source in G_0. Finally, constraints (5.7) and (5.8) are the time limitation constraints for the time variable t_i. For each $(i,j) \in A$, M_{ij} and α_{ji} are suitable parameters that will be defined in section 5.3 where constraints (5.6) will be analysed.

5.2.2 F_2: Delay-constrained Steiner Tree formulation with directed cuts

The following formulation is a directed cut formulation for the Steiner Tree problem [87] with the addition of the delay constraints. Even in this formulation, with each node of the graph $i \in V \setminus \{s\}$ is associated a continuous variable $t_i \in [0, \Delta]$ and t_s is set to zero. We refer to the formulation as F_2:

$$\min \sum_{(ij)\in A} c_{ij} y_{ij} \tag{5.10}$$

$s.t.$

$$\sum_{(i,j)\in\delta^+(S)} y_{ij} \geq 1 \qquad \forall S \subset V, s \in S, R \cap S^c \neq \emptyset \tag{5.11}$$

$$\sum_{(j,i)\in\delta^-(i)} y_{ji} \leq \sum_{(i,j)\in\delta^+(i)} y_{ij} \qquad \forall\, i \in V \setminus (R \cup \{s\}) \tag{5.12}$$

$$y_{ij} + y_{ji} \leq 1 \qquad \forall\, (i,j) \in A \tag{5.13}$$

$$t_i - t_j + M_{ij} y_{ij} + \alpha_{ji} y_{ji} \leq M_{ij} - del_{ij} \qquad \forall\, (i,j) \in A \tag{5.14}$$

$$0 \leq t_i \leq \Delta \qquad \forall\, i \in V \setminus \{s\} \tag{5.15}$$

$$t_s = 0 \tag{5.16}$$

$$y_{ij} \in \{0,1\} \qquad \forall\, (i,j) \in A. \tag{5.17}$$

Constraints (5.11) are the directed cut constraints, for each cutset S separating the source from some required nodes in R, there should be at least one outgoing arc. The classical directed cut formulation did not consider the Flow–Balance constraints (5.12) introduced in [47] in order to strengthen the original formulation. These constraints force each Steiner node with one incoming arc to have at least one outgoing arc. Moreover, constraints (5.15),

(5.16) and (5.17) are the variable domain restrictions and again constraints (5.14) will be considered in section 5.3.

5.2.3 F_3: Multicommodity Flow formulation

The following formulation F_3 is a generalization of the Multicommodity Flow formulation for the minimum Steiner Tree problem [87] including the delay constraints. For each required node $k \in R$ and arc $(i,j) \in A$, the variable x_{ij}^k takes value one if the arc (i,j) is in the directed path connecting the source to k, zero otherwise.

$$\min \sum_{(i,j)\in A} c_{ij} y_{ij} \tag{5.18}$$

$s.t.$

$$\sum_{(i,s)\in A} x_{is}^k - \sum_{(s,i)\in A} x_{si}^k = -1 \qquad \forall\, k \in R \tag{5.19}$$

$$\sum_{(i,j)\in A} x_{ij}^k - \sum_{(j,i)\in A} x_{ji}^k = 0 \qquad \forall k \in R,\, \forall\, j \in V \setminus \{k,s\} \tag{5.20}$$

$$\sum_{(i,k)\in A} x_{ik}^k - \sum_{(k,i)\in A} x_{ki}^k = 1 \qquad \forall\, k \in R \tag{5.21}$$

$$0 \le x_{ij}^k \le y_{ij} \qquad \forall (i,j) \in A,\ \forall\, k \in R \tag{5.22}$$

$$\sum_{(i,j)\in A} del_{ij}\, x_{ij}^k \le \Delta \qquad \forall\, k \in R \tag{5.23}$$

$$y_{ij} \in \{0,1\} \qquad \forall\, (i,j) \in A. \tag{5.24}$$

The variable x_{ij}^k represents the quantity of commodity k flowing through the arc (i,j). Constraints (5.19), (5.20) and (5.21) are the flow conservation constraints that guarantee that there is a flow of one unit outgoing from the

source and incoming in each node of R. Constraints (5.23) are the delay constraints, whereas constraints (5.22) are the relation between the x and y variables.

5.2.4 F_4: Multi-cut formulation

The following formulation F_4 is a multi-cut formulation with delay constraints. Even in this formulation, we introduce variables x_{ij}^k that are defined as in formulation F_3.

$$\min \sum_{(ij)\in A} c_{ij} y_{ij} \tag{5.25}$$

$s.t.$

$$\sum_{(i,j)\in\delta^+(S)} x_{ij}^k \geq 1 \qquad \forall\, k \in R,\, \forall\, S \subset V,\, s \in S,\, k \in S^c \tag{5.26}$$

$$0 \leq x_{ij}^k \leq y_{ij} \qquad \forall\, (i,j) \in A,\ \forall\, k \in R \tag{5.27}$$

$$\sum_{(i,j)\in A} del_{ij}\, x_{ij}^k \leq \Delta \qquad \forall\, k \in R \tag{5.28}$$

$$y_{ij} \in \{0,1\} \qquad \forall\, (i,j) \in A. \tag{5.29}$$

Constraints (5.26) force the existence of an arc for each cut (S, S^c) separating the source from each element of R. The remaining constraints have the same meaning of formulation F_3: (5.28) are the delay constraints, (5.27) are the relation between the x and y variables and (5.29) are the variable domain constraints.

5.3 Cumulative-delay constraints

Constraints (5.6) and (5.14) of formulations F_1 and F_2, respectively, are at the same time subtour–elimination constraints and cumulative-delay constraints.

The classical Miller-Tucker-Zemlin constraints (MTZ, see e.g. [58]) have been introduced for providing a polynomial formulation for the Traveling Salesman problem (TSP). In our case, these constraints that include the cumulative delays can be expressed as:

$$t_i - t_j + del_{ij} \leq M_{ij}(1 - y_{ij}) \qquad \forall (i,j) \in A. \tag{5.30}$$

For these constraints, if the variable y_{ij} takes value one, then the value of t_j is forced to the value of t_i plus the delay value on the arc (i,j), if $y_{ij} = 0$, then constraints (5.30) are fulfilled just defining a sufficiently big value of M_{ij}. This value has to make the inequality $t_i - t_j \leq M_{ij} - del_{ij}$ redundant whenever $y_{ij} = 0$ and so it suffices to set $M_{ij} := \Delta + del_{ij}$.

A possible improvement that can be performed is to lift [26] the constraints (5.30) adding a nonnegative term $\alpha_{ji} y_{ji}$, with a sufficiently big value of α_{ji}, namely $\alpha_{ji} := \Delta - del_{ji}$, so that constraints (5.30) become:

$$t_i - t_j + M_{ij} y_{ij} + \alpha_{ji} y_{ji} \leq M_{ij} - del_{ij} \qquad \forall (i,j) \in A. \tag{5.31}$$

If variable $y_{ji} = 0$, then the added term does not give any contribution, if the variable y_{ji} takes value one, then $y_{ij} = 0$ in view of (5.4) or (5.13). Using the inequality (5.31) applied to the arc $(j,i) \in A$ and setting y_{ji} to 1 in (5.31), it is easy to see that the value of t_i is forced to the value of t_j plus the delay value on the arc (j,i).

5.4 Improved cumulative delay constraints

It is possible to strengthen the coefficients M_{ij} and α_{ji} of constraints (5.31) and the lower and upper bounds for (5.7) and (5.15).

The delays on the arcs can define, for every nodes $i \in V \setminus \{s\}$, a time window during which the communication should be received and forwarded by the nodes in order to respect the maximum delay Δ on the nodes of R. A message forwarded by the source s can not reach any node i of the network in a time that is lower than the shortest path value considering the delays as costs. For every node $i \in V$, we denote by $\lambda_i \in \mathbb{R}^+$ the value of the shortest path between s and i with the delays as costs: $\lambda_i = \min\{Del(P) : P \text{ is an } s-i \text{ path}\}$. The cumulated delay t_i at the node i should be greater than or equal to λ_i and obviously, if $\lambda_i > \Delta$ for a required node $i \in R$, the Delay-constrained Steiner Tree problem is infeasible.

Moreover, we can reduce the upper bound for the variables t_i associated with a Steiner node i. Indeed, a Steiner node i is in any feasible solution and, hence, in a feasible arborescence T only if there exists a destination $t \in R$ such that $t_i + Del(P_{(i,t)}) \leq \Delta$, where $P_{(i,t)}$ is the path from i to t in the arborescence T. For this reason, if we denote by ζ_i the value of the Shortest Path from i to the nearest destination in R with the delays as costs, the variables t_i must be at most equal to μ_i, where $\mu_i := \Delta - \zeta_i$. If $i \in R$, then obviously $\mu_i = \Delta$.

Constraints (5.7) and (5.15) become, thus:

$$\lambda_i \leq t_i \leq \mu_i \qquad \forall i \in V \setminus \{s\} \tag{5.32}$$

Naturally, these new extremes of the time window $[\lambda_i, \mu_i]$ can be used to

perform a first delay-based preprocessing (see section 5.6) so that all the Steiner nodes with an empty time window can be eliminated from the graph since they will never be in a feasible solution respecting the maximum delay Δ (see Proposition 5.6.2).

Furthermore, with the introduced limitations on the values of the variables t_i, and after eliminating the Steiner nodes with an empty time window, the coefficients M_{ij} and α_{ji} of constraints (5.31) can be lowered.

Remark 5.4.1. For each $(i,j) \in A$ in constraints (5.31) the coefficients M_{ij} and α_{ji} can be set to $M_{ij} := \mu_i - \lambda_j + del_{ij}$ and $\alpha_{ji} := \mu_i - \lambda_j - del_{ji}$ respectively. Indeed, let $(i,j) \in A$. If $y_{ij} = y_{ji} = 0$, then constraint (5.31) becomes $t_i - t_j \leq M_{ij} - del_{ij}$ which is easy to see that is always fulfilled. If $y_{ij} = 1$, then in view of constraints (5.4) or (5.13), it holds that $y_{ji} = 0$ and so constraint (5.31) is:

$$t_i - t_j + M_{ij} \leq M_{ij} - del_{ij},$$

so that $t_j \geq t_i + del_{ij}$. If $y_{ji} = 1$, then $y_{ij} = 0$ and so constraint (5.31) becomes:

$$t_i - t_j + \alpha_{ji} \leq M_{ij} - del_{ij}.$$

Substituting the value of α_{ji} and M_{ij}, we have:

$$t_i - t_j + \mu_i - \lambda_j - del_{ji} \leq \mu_i - \lambda_j.$$

This last constraint with the addition of constraint (5.31) for the arc $(j,i) \in A$ force t_i to assume the value $t_j + del_{ji}$.

5.5 Comparison of LP relaxations

In this section, given a set $S \subset V$, we denote by $\delta_G^-(S)$ and by $\delta_{G_0}^-(S)$, respectively, the set of the arcs of the graph $G = (V, A)$ and of the graph

$G_0 = (V_0, A_0)$ incoming in S. In an analogous way, $\delta^+_G(S)$ and $\delta^+_{G_0}(S)$ are, respectively, the set of the arcs of the graph $G = (V, A)$ and of the graph $G_0 = (V_0, A_0)$ outgoing from S.

Proposition 5.5.1. *The value of an optimal solution of the linear relaxation of F_2 is not smaller than the value of an optimal solution of the linear relaxation of formulation F_1.*

Proof. First of all we need to augment formulation F_2 with the variables associated with the arcs $(0, i)$ with $i \in R^c$ in order to compare the optimal values of the linear relaxations of the two formulations. Given an optimal solution (y^*, t^*) for the linear relaxation of formulation F_2, as in [69], we define $\overline{y}_{ij} := y^*_{ij}$ for each $(i, j) \in A$ and for the arcs $(0, j)$ with $j \in R^c$ we set $\overline{y}_{0j} := 1 - \sum_{(i,j)\in\delta^-_G(j)} y^*_{ij}$. The solution $(\overline{y}, t^*)$ is still an optimal solution for the linear relaxation of F_2 since the costs associated with the arcs $(0, j)$ with $j \in R^c$ are zero. We should show that the augmented optimal solution $(\overline{y}, t^*)$ is a feasible solution for the linear relaxation of F_1.

Since (y^*, t^*) is an optimal solution for the linear reaxation of F_2 and the costs of the arcs are nonnegative, it follows that:

$$\sum_{(j,s)\in\delta^-_G(s)} y^*_{js} = 0.$$

Constraint (5.5) is fulfilled by $\overline{y}$, since:

$$\overline{y}_{0s} = 1 - \sum_{(j,s)\in\delta^-_G(s)} y^*_{is} = 1.$$

In [69] it is shown that the variables y^* verify the following two inequalities:

$$\sum_{(k,j)\in\delta^-_G(j),k\neq i} y^*_{kj} \geq y^*_{ji} \quad \forall j \in V \setminus \{s\},\ (j, i) \in A, \tag{5.33}$$

and

$$\sum_{(i,j)\in\delta_G^-(j)} y_{ij}^* \leq 1 \quad \forall j \in V \setminus \{s\}. \tag{5.34}$$

Let $j \in R^c \setminus \{s\}$ and $(j,i) \in \delta_{G_0}^+(j)$, in view of constraints (5.33) it follows that:

$$\overline{y}_{0j} + \overline{y}_{ij} + \overline{y}_{ji} = 1 - \sum_{(k,j)\in\delta_G^-(j)} y_{kj}^* + y_{ij}^* + y_{ji}^* = 1 - \sum_{(k,j)\in\delta_G^-(j),\ k\neq i} y_{kj}^* + y_{ji}^* \leq 1,$$

and, hence, constraints (5.4) are fulfilled.

Furthermore, let $k \in R$, in view of constraints (5.11) with $S = V \setminus \{k\}$ and of constraints (5.34), it holds that:

$$1 \leq \sum_{(i,k)\in\delta_G^+(S)} y_{ik}^* = \sum_{(i,k)\in\delta_G^-(k)} y_{ik}^* = \sum_{(i,k)\in\delta_{G_0}^-(k)} \overline{y}_{ik} \leq 1$$

and, hence, constraints (5.2) with $k \in R$ are fulfilled. Let now $k \in V \setminus R$, then

$$\sum_{(i,k)\in\delta_{G_0}^-(k)} \overline{y}_{ik} = \sum_{(i,k)\in\delta_G^-(k)} y_{ik}^* + \overline{y}_{0k} = \sum_{(i,k)\in\delta_G^-(k)} y_{ik}^* + 1 - \sum_{(i,k)\in\delta_G^-(k)} y_{ik}^* = 1.$$

Finally we have to prove that

$$\sum_{(i,j)\in\delta_{G_0}^+(i)} \overline{y}_{ij} \geq 1 - \overline{y}_{0i} \quad \forall i \in R^c.$$

Let $i \in R^c$, for the constraints (5.12) it holds:

$$\sum_{(i,j)\in\delta_{G_0}^+(i)} \overline{y}_{ij} = \sum_{(i,j)\in\delta_G^+(i)} y_{ij}^* \geq \sum_{(j,i)\in\delta_G^-(i)} y_{ji}^* = 1 - \overline{y}_{0i}.$$

The other constraints of formulation F_1 are obviously verified and, therefore, $(\overline{y}, t^*)$ is feasible for the linear relaxation of F_1. □

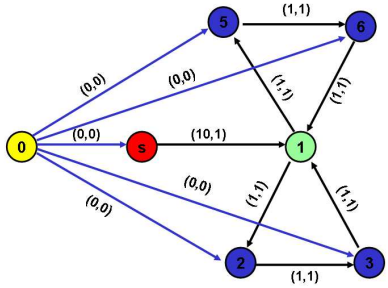

Figure 5.1: Example of an optimal solution of the linear relaxation of F_1 which is infeasible for the linear relaxation of F_2

The example used in [69] and reported in Figure 5.1, can be used to show that there exist Delay-constrained Steiner Tree problems in which the optimal solution of the linear relaxation of F_1 is not feasible for the linear realaxation of F_2.

Example 5.5.1. Consider the graph in Figure 5.1, where $R = \{1\}$ and $\Delta = 10$. The delay constraints in this case are redundant for defining any optimal solution. The solution in the variables y: $y_{02} = y_{03} = y_{15} = y_{56} = y_{61} = \frac{1}{3}$, $y_{12} = y_{23} = y_{31} = y_{05} = y_{06} = \frac{2}{3}$, $y_{0s} = 1$ is an optimal solution for the linear relaxation of the formulation of F_1, but it is not a feasible solution for the linear relaxation of the formulation F_2, since if $S = \{s\}$, then $\sum_{(i,j)\in\delta^+(S)} y_{ij} = 0$.

Proposition 5.5.2. *Formulation F_3 is better than formulation F_4.*

Proof. We have to prove that every feasible solution for the linear relaxation of formulation F_3 is feasible for the linear relaxation of F_4 and that there exist Delay-constrained Steiner Tree problems in which a feasible solution for the linear relaxation of F_4 is not feasible for the linear relaxation of F_3 (see Definition 1.1.4). Let (y^*, x^*) be a feasible solution for the linear

relaxation of formulation F_3. The only constraints that should be checked are constraints (5.26). Let $k \in R$, $S \subset V$ such that $s \in S$ and $k \in S^c$, the value of the flow from s to k is 1, so that:

$$\sum_{(i,j)\in\delta^+(S)} x_{ij}^k - \sum_{(i,j)\in\delta^-(S)} x_{ij}^k = 1;$$

thus

$$\sum_{(i,j)\in\delta^+(S)} x_{ij}^k = 1 + \sum_{(i,j)\in\delta^-(S)} x_{ij}^k \geq 1.$$

A case in which a feasible solution for the linear relaxation of F_4 is not feasible for F_3 in given in the Example 5.5.2. □

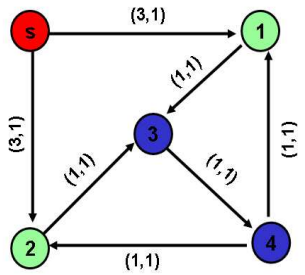

Figure 5.2: Example of a feasible solution for the linear relaxation of F_4 that is not feasible for the linear relaxation of F_3

Example 5.5.2. Consider the graph in Figure 5.2 which is another example proposed in [69]. Suppose $R := \{1, 2\}$ and $\Delta := 10$. The solution $y_{s1} = y_{s2} = y_{13} = y_{23} = y_{34} = y_{41} = y_{42} = x_{s1}^1 = x_{s2}^1 = x_{13}^1 = x_{23}^1 = x_{34}^1 = x_{41}^1 = x_{42}^1 = x_{s1}^2 = x_{s2}^2 = x_{13}^2 = x_{23}^2 = x_{34}^2 = x_{41}^2 = x_{42}^2 = \frac{1}{2}$, is a feasible solution for the linear relaxation of the formulation of F_4, but it is not feasible for the linear relaxation of the formulation F_3 since the constraint (5.20) relative to node 3 is not fulfilled.

5.6 Preprocessing

Preprocessing plays a very useful role in solving combinatorial and integer programming problems. This technique, indeed, reduces the size of the problems by means of logical implications, producing equivalent problems. The preprocessing performed in our problem is based on the fulfilment of the time windows request and on an adaptation of the known preprocessing techniques (see Proposition 1.5.1) used to reduce the size of the graph in the pure Steiner Tree problem; because of the presence of the delay on the arcs, if we want to contract certain edges, we need to store the delays. For this reason, we introduce $m_i \in \mathbb{R}$ for each $i \in V$ and initially we set m_i to zero for all $i \in V$.

5.6.1 Degree-delay preprocessing

Until no more reduction can be performed in the graph, the following tests for reducing the size of the problem are executed:

Proposition 5.6.1 (Degree one test). *For every node $i \in V$*

(i) *if i is a Steiner node and $|\delta(i)| = 1$, then i is eliminated from the graph together with the edge incident in i;*

(ii) *if $|\delta(i)| = 1$, $i \in R$ and $\delta(i) = \{\{i, j\}\}$, then $\{i, j\}$ is contracted, the cost c_{ij} is stored to be added to the optimal solution and if $del_{ij} > m_j$, the values of μ_j and m_j are updated: $\mu_j := \mu_i + m_j - del_{ij}$ and $m_j := del_{ij}$, respectively.*

For every node $i \in V$, the time windows $[\lambda_i, \mu_i]$ is empty if the time

required to reach the node i from the source s is greater than the residual time to reach the nearest (in terms of delays) required node.

Proposition 5.6.2 (Non–empty time windows). *For every node $i \in V$*

(i) *If $\lambda_i > \mu_i$ and i is a Steiner node, then i can be removed from the graph together with all its incident edges.*

(ii) *If $\lambda_i > \mu_i$ and i is a required node, then the Minimum Steiner Tree problem with the delay constraints is infeasible.*

Proof. (i) Suppose on the contrary that an optimal solution contains a Steiner node i with $\lambda_i > \mu_i$ and let t be the nearest terminal from i using the delays as cost. As i is a node belonging to the optimal solution, there exists on the support of this solution at least a path $P_{s,\bar{t}}$ from the source to a terminal $\bar{t} \in R$ passing through i. The total delay along the path $P_{(s,\bar{t})}$ is such that:

$$Del(P_{(s,\bar{t})}) = \sum_{(i,j)\in P_{(s,\bar{t})}} del_{ij} \geq \lambda_i + Del(P_{(i,\bar{t})}) \geq \lambda_i + \zeta_i \geq \lambda_i - \mu_i + \Delta > \Delta,$$

which is a contradiction.

(i) Suppose on the contrary that there exists a feasible solution, since the shortest path value from the source to the required node i with the delays as costs is greater than Δ for each path $P_{(s,i)}$ in the graph it holds: $Del(P_{(s,i)}) \geq \lambda_i > \Delta$ which is a contradiction.

□

Proposition 5.6.3 (Adjacent time request). *For every edge $\{i, j\} \in E$, if $\lambda_i + del_{ij} > \mu_j$ and $\lambda_j + del_{ji} > \mu_i$, then the edge $\{i, j\}$ can be eliminated from the graph.*

Proof. Suppose on the contrary that an optimal solution contains the arc (i,j) with $\lambda_i + del_{ij} > \mu_j$ (the same holds for (j,i) with $\lambda_j + del_{ji} > \mu_i$). Let t be the nearest required node from j using the delays as costs. As j is a node belonging to the optimal solution, there exists on the support of this solution at least a path $P_{(s,\overline{t})}$ from the source to a terminal $\overline{t}$ passing through j. The total delay along the path $P_{(s,\overline{t})}$ is such that

$$\begin{aligned} Del(P_{(s,\overline{t})}) = \sum_{(i,j)\in P_{s,\overline{t}}} del_{ij} &\geq \lambda_i + del_{ij} + Del(P_{(j,\overline{t})}) \geq \lambda_i + del_{ij} + \zeta_i \\ &\geq \lambda_i + del_{ij} - \mu_i + \Delta > \Delta, \end{aligned}$$

which is a contradiction. □

The degree two test is analogous to the test of the Steiner Tree problem (see Proposition 1.5.1), but a further condition on the delays must be inserted in order to respect the maximum delay at the required nodes.

Proposition 5.6.4 (Degree two test). *If $i \in V$ is a Steiner node with $\delta(i) = \{\{i,k\},\{j,i\}\}$,*

(i) *if $\{k,j\} \notin E$, then the edges $\{k,i\}$ and $\{i,j\}$ are substituted by a new edge $\{k,j\}$ with cost $c_{kj} = c_{ki} + c_{ij}$ and delay $del_{kj} = del_{ki} + del_{ij}$ and i can be eliminated.*

(ii) *if $\{k,j\} \in E$, if $c_{ki} + c_{ij} > c_{kj}$ and $del_{ki} + del_{ij} > del_{kj}$, then i can be eliminated from the graph together with the edges $\{i,k\}$ and $\{j,i\}$.*

(ii) *if $\{k,j\} \in E$, $c_{ki} + c_{ij} \leq c_{kj}$ and $del_{ki} + del_{ij} \leq del_{kj}$, then node i is removed from the graph together with its incident edges and the edge $\{k,j\}$ is given the cost $c_{kj} = c_{ki} + c_{ij}$ and the delay $del_{kj} = del_{ki} + del_{ij}$.*

All the formulations are defined on directed graphs, so that, another reduction can be done considering the orientation of the arcs.

Proposition 5.6.5 (Direct arcs test). *Every arc incoming in the source $(i, s) \in A$ and all the directed arcs (i, j) such that $\lambda_i + del_{ij} > \mu_j$ can be eliminated from the directed graph.*

Proof. Because of the nonnegativity of the costs all the arcs $(i, s) \in A$ can be removed from the graph and the rest follows as in Proposition 5.6.3. □

The delay-degree preprocessing consists in summary in these steps:

Step 1: Degree one test;

Step 2: Non–empty time windows test;

Step 3: Adjacent time request test;

Step 4: Degree two test;

Step 5: If at least one contraction or elimination has been executed go to Step 1 else go to Step 6;

Step 6: Consider the directed graph and perform the Direct arc elimination.

5.6.2 LP preprocessing

The LP preprocessing is based on Proposition 1.5.2, in fact, if we denote by z_{LP} the optimal value of the linear relaxation of the problem and by z_{UB} the value of the best known feasible solution of the problem, that is an upper bound for the solution, then Proposition 1.5.2 can be applied to fix the value of certain nonbasic variables.

If y^* is an optimal solution of the linear relaxation of the problem, then if $y^*_{ij} = 0$ its reduced cost $\overline{c}_{ij}$ is nonnegative. Using Proposition 1.5.2, if

$z_{LP} + \overline{c}_{ij} > z_{UB}$, then fixing the variable y^*_{ij} to one does not produce any improvement in the optimal value of the objective function, hence, the value of the variable y^*_{ij} is fixed to zero, which means that it is possible to eliminate the arc (i, j) from the graph.

Moreover, if $y^*_{ij} = 1$ in the optimal solution, the reduced cost $\overline{c}_{ij}$ is nonpositive and, using again Proposition 1.5.2, if $z_{LP} - \overline{c}_{ij} > z_{UB}$, then even in this case reducing to zero the value of y^*_{ij} does not make any improvement and so the variable y^*_{ij} is fixed to take value 1, thus, the arc (i, j) is always in an optimal solution of the IP problem.

5.7 Exact Solution strategies

In this section, we present the methods for solving the different formulations presented in section 5.2.

5.7.1 Algorithm for F_1

The Degree-constrained Minimum Spanning Tree formulation with Delay constraints has a polynomial number of constraints and can be directly solved by any mixed integer linear programming solver. The algorithm for its solution can be summarized as follows:

Step 0: Perform the Degree-delay preprocessing;

Step 1: Solve the linear relaxation of formulation F_1;

Step 2: Perform the LP preprocessing; if an edge is eliminated go to Step 0 else go to Step 3;

Step 3: Solve the MIP formulation F_1.

5.7.2 Algorithm for F_2

The drawback of formulation F_2 is represented by the sets of constraints (5.11) that are in an exponential number, but since only a small fraction of these constraints is saturated at optimality, we choose to solve the problem with an iterative approach. Namely, the initial constraint matrix is constituted by the constraints (5.12) and (5.13), by the Delay-constraints (5.14), by constraints (5.15) and (5.16) and by the cuts (5.11) generated by the subset $S := \{s\}$ and by the subsets S such that $|S^c| = 1$. For speeding the generation of constraints (5.11) up, we solve the linear relaxation of formulation F_1 with all the costs equal to 1, whose optimum value is indicated by β. An approximation of the minimum number of arcs in the solution of the Delay-constrained problem is given by $\lceil \beta \rceil$, hence we add to the initial constraint system the inequality:

$$\sum_{(i,j)\in A} y_{ij} \geq \lceil \beta \rceil \,. \tag{5.35}$$

The procedure of the algorithm can be formalized as follows:

Step 0: Perform the Degree-delay preprocessing;

Step 1: Let F_2' be the formulation F_2 with only the constraints (5.11) corresponding to $S = \{s\}$, and S such that $|S^c| = 1$ and including the new constraint (5.35);

Step 2: Solve F_2', and let $(\overline{y}, \overline{t})$ be the optimal solution;

Step 3: If $\overline{y}$ violates a constraint ctr_{12} of type (5.11) (the separation routine will be described later), then add ctr_{12} to F_2' and go to Step 2;

Step 4: Perform the LP preprocessing, if an edge is eliminated, then go to Step 0 else go to Step 5;

Step 5: Solve the MIP problem; let $(\overline{y}, \overline{t})$ be the optimal solution of F_2';

Step 3: If $\overline{y}$ violates a constraint ctr_{12} of type (5.11), then add ctr_{12} to F_2' and go to Step 5 otherwise the optimal solution has been found.

Notice that the procedure converges after a limited number of iterations.

Separation problem

Once a solution $(\overline{y}, \overline{t})$ of F_2' is available, the presence of violated inequalities of type (5.11) of F_2 not inserted into F_2' can be detected as follows. For each source-destination pair the maximum flow problem with y as capacities is solved. If a maximum flow value is less than 1, then the minimum capacity cut (S, S^c) is indentified and the corresponding constraint (5.11) is generated.

5.7.3 Algorithm for F_3

The Multicommodity Flow formulation F_3 may have a large number of variables but it does not have critical constraints that impose the use of a specific solution technique. Formulation F_3 is, thus, directly solved by any mixed integer linear programming solver. The pseudocode of the solution algorithm for F_3 is the same as in subsection 5.7.1.

5.7.4 Algorithm for F_4

An iterative approach is used for solving the problem with formulation F_4.

The initial constraint matrix is constituted by the constraints of F_4 except constraints (5.26), indeed, among constraints (5.26) only those generated by the subset $S := \{s\}$ and by the subsets S such that $|S^c| = 1$ are considered initially. For speeding the generation of constraints (5.26) up, we have solved the shortest path problem connecting the source to each destination $k \in R$ and we have computed β_k which represents the number of arcs of each s-k path. In order to make the generation of constraints (5.26) faster, for each node $k \in R$ we add to the initial constraint system the inequalities:

$$\sum_{(i,j)\in A} x_{ij}^k \geq \beta_k \qquad \forall k \in R. \tag{5.36}$$

The algorithm is the same as for formulation F_2 (see subsection 5.7.2), the only difference is in the separation procedure. Indeed, when a cut (S, S^c) is found, then all the constraints (5.26) for each $k \in S^c$ are generated at the same time, instead of the unique constraint generated for F_2.

5.8 Heuristic Solution

In order to make the LP preprocessing effective, a good heuristic that provides a feasible solution with a tight upper bound z_{UB} in a reasonable time should be considered. We compute the shortest paths that fulfil the delay constraints between the source and all required nodes and we select the path $\overline{P}(s, \bar{t})$ with the highest length. Till all the required nodes are connected to the source, at each step, the heuristic H_1 adds a new path

that fulfils the maximum delay constraint with the lowest total cost from one of the nodes of the current tree (initially constituted by $\overline{P}(s,\overline{t})$) to one of the required nodes not yet connected to the source. This heuristic is fast but does not provide a tight upper bound. For the sake of reducing the gap between the value of the optimal integer solution and z_{UB}, we propose the heuristic H_1' in which we repeat K times the following procedure: we perturb the costs associated with the arcs, we perform the heuristic procedure H_1 and we consider the best obtained value z_{UB}.

The problem of finding the Shortest Path with capacity constraints has been proved to be NP-Hard in [38]. This type of problem has been widely studied and the case in which the capacity constraints are the delay constraints has been considered in [39]. The Delay-constrained Shortest Path problem can be solved in an exact way with a dynamic approach based on a generalization of Ford-Fulkerson and of Dijkstra algorithms ([44], [59]). An exact solution based on the Lagrangian relaxation has been proposed in [38]. Since we aim at using an efficient and fast heuristic, like in [59], we find an approximate solution of the Lagrangian relaxation of the Delay-constrained Shortest Path problem where the delay constraints are relaxed so that the relaxed problem can be solved by Dijkstra's algorithm.

5.8.1 Heuristic H_1

Given the graph $G = (V, A)$, we indicate by C the set of the required nodes connected to the source. All the Delay-constrained Shortest Paths $\overline{P}_{(s,t)}$ that connect the source to each $t \in R$ are computed, and it is selected the path $\overline{P}(s,\overline{t})$ with the greatest cost (length) whose cost is assigned to z_{UB}. The set C becomes, thus, $C := \{\overline{t}\}$ and we assign a zero cost to all the arcs of the path $\overline{P}_{(s,\overline{t})}$. Unless the set C coincides with R, we add a new

node f to the graph G and we define the set A' of all the arcs of A with the addition of all the arcs (i, f) for each $i \in R \setminus C$, whose is associated a zero cost and a zero delay (the current graph is, thus, $G' = (V', A')$ with $V' = V \cup \{f\}$ and $A' = A \cup \{(i, f) : \ \forall i \in V \setminus C\}$); we solve the Delay-constrained Shortest Path problem between the source and the node f finding the path $P_{(s,f)}$; we update C adding the required node t such that $(t, f) \in P_{(s,f)}$ and we set to zero the costs of the arcs of $P_{(s,f)}$ that belong to A; finally we update the value z_{UB} adding the cost of the path $P_{(s,f)}$, that is, $z_{UB} := z_{UB} + c(P_{(s,f)})$ and we repeat the process. If C coincides with R the current value z_{UB} is the required upper bound.

The algorithm can be summarized as follows:

(Step 0:) Set $C := \emptyset$.

(Step 1:) Compute the approximated Delay-constrained Shortest Paths between the source and all the required nodes. Select $\overline{P}_{(s,\bar{t})}$ the path with the maximum cost (length).

(Step 2:) Set $z_{UB} := c(\overline{P}_{(s,\bar{t})})$, $C := C \cup \{\bar{t}\}$ and $c_{ij} := 0$ for all $(i, j) \in \overline{P}_{(s,\bar{t})}$.

(Step 3:) Add a node f to the graph $G = (V, A)$; define $V' = V \cup \{f\}$ and $A' := A \cup \{(i, f) : \ \forall i \in R \setminus C\}$; set $c_{if} = del_{if} = 0$ for all $(i, f) \in A'$.

(Step 4:) Compute the approximated Delay-constrained Shortest Path $P_{(s,f)}$ on the graph $G' = (V'A')$, find $\bar{t} \in R$ such that $(\bar{t}, f) \in P_{(s,f)}$.

(Step 5:) Set $z_{UB} := z_{UB} + c(P_{(s,f)})$, $C := C \cup \{\bar{t}\}$ and $c_{ij} := 0$ for all $(i, j) \in P_{(s,f)} \cap A$. If $C \subset R$, then go to step 3 else Stop.

5.8.2 Heuristic H_1'

In this heuristic, we perturb the cost associated with each arc (i, j) of the graph, that is, we generate a random number ϵ_{ij} in the interval $[0.5, 1.5]$ and we assign to the arc (i, j) the cost $\epsilon_{ij} c_{ij}$; we solve the problem of finding a feasible solution for the Delay constrained Steiner Tree problem with the perturbed costs with the procedure H_1 and we store the best obtained value of z_{UB}. We have seen on the basis of the experimental results that we can find the best gap between the optimal value of the Delay constrained Steiner Tree problem and the value z_{UB}, if we perturb the costs and solve the problem for $K = 500$ times.

5.9 Computational results

All the instances of the Delay-constrained Steiner Tree problem has been solved on an Opteron 246 machine with 2 GB RAM memory using the version 9.1 of Cplex as solver. We have set to 30 minutes the computational time limit. By NS we indicate the number of instances not solved within the time limit when the solution process is interrupted.

5.9.1 Description of the problem instances

To the best of our knowledge, no benchmark is available for the Delay constrained Steiner Tree problem in literature. We have, therefore, considered the problems proposed in the SteinLib library [48] for the pure Steiner Tree problem, in particular the problems of the class B and the first 10 instances of the class C. The instances of class B and C are randomly gen-

erated sparse graphs with edge weights between 1 and 10; for the class B, the size of the problems goes from graphs with $|V| = 50, |R| = 9, |E| = 63$ to graphs with $|V| = 100, |R| = 50, |E| = 200$, whereas for the considered instances of the class C the size of the problems goes from $|V| = 500, |R| = 5, |E| = 625$ to $|V| = 500, |R| = 250, |E| = 1000$. For the classical Steiner Tree problem these instances can be solved in few seconds with the local preprocessing or by efficient known algorithms. We have generated randomly the delays on the edges in such a way that they result correlated and non-correlated to the costs. In the first case a random number r is generated in the interval $[0.8, 1.2]$ and for each edge $\{i, j\}$ we set $del_{ij} = r * c_{ij}$, in the second case the delays are simply random values belonging to the interval $[1, 100]$. On the basis of the generated delays, we have computed the value MP which is the maximum among the shortest paths with the delays as costs between the source and each required node, then in the problems indicated with 0.1 we have set Δ to the value $\Delta := 1.1 * MP$ and in the problems indicated with 0.5 we have set Δ to $\Delta := 1.5 * MP$. With these choices none of the problems is infeasible. In the following tables, we indicate for example by B Ran 0.1 the set of the instances of the class B with delays non-correlated with the costs and with $\Delta = 1.1 * MP$ and with C Cor 0.5 the set of the instances of the class C with delays correlated with the costs and with $\Delta = 1.5 * MP$.

In columns *Gap*, we report the mean of the ratios $(OPT - LP)/OPT$ where OPT is the optimum value of the integer problem and LP is the optimum value of the linear relaxation of the problem. For each class of problems, we indicate with T the mean of the resolution times in seconds for the instances solved within the time limit and with T_{max} the maximum computational time. If certain instances in a class are not solved within 30 minutes, then T_{max} reports the number of not solved problems.

5.9.2 Performance of the different formulations

In Table 5.1, we report the gap between the value of the optimal integer solution of the instances and the value of an upper bound provided by the heuristic H_1'; *gap* is, indeed, the mean of the values $(z_{UB} - OPT)/OPT$.

Table 5.1: Gap for the heuristic H_1'

Problem	$gap \times 100$	*Problem*	$gap \times 100$
B Ran 0.1	1.28	C Ran 0.1	3.06
B Ran 0.5	0.66	C Ran 0.5	1.24
B Cor 0.1	0.28	C Cor 0.1	2.26
B Cor 0.5	0.20	C Cor 0.5	2.01

We use the value z_{UB} of the heuristic H_1' to perform the LP proprocessing of the problem.

In Table 5.2, we present the average gap and the computational time for the different algorithms of section 5.7. All the instances of the class B have been solved within the required time limit, whereas there are certain instances of the class C that are unsolved. Formulation F_1 is the fastest among all the other formulations even if the optimal value of the linear relaxation of the problems are always the worst with respect to the lower bounds provided by the other formulations. Moreover, F_3 is the formulation with the closest optimal value of the linear relaxation to the optimal integer value, but for example 6 over the 10 instances of the different problems of the class C are not solved in the time limit.

Regarding to formulations F_2 and F_4, one provides a better gap (F_4), but the other solve the problems in a lower time (F_2), but just using the MIP

solver for solving the instances none of the two's has interesting behaviours if compared with F_1 and F_3.

Table 5.2: Average gap and computational times for the Delay-constrained Steiner Tree problem

	F_1			F_2		
Problem	$Gap \times 100$	T	T_{max}	$Gap \times 100$	T	T_{max}
B Ran 0.1	9.5	0.14	1.35	7.59	6.75	114.06
B Ran 0.5	6.54	0.73	3.81	3.85	4.77	45.24
B Cor 0.1	5.44	0.14	0.71	2.83	0.82	5.11
B Cor 0.5	3.81	0.42	2.14	1.02	1.55	18.17
C Ran 0.1	7.30	196.00	2NS	5.45	121.23	5NS
C Ran 0.5	5.84	82.52	3NS	2.72	38.89	5NS
C Cor 0.1	6.28	210.50	2NS	2.40	22.25	5NS
C Cor 0.5	2.47	94.73	1NS	0.04	88.80	6NS

	F_3			F_4		
Problem	$Gap \times 100$	T	T_{max}	$Gap \times 100$	T	T_{max}
B Ran 0.1	2.11	34.61	480.88	2.32	38.82	1106.0
B Ran 0.5	1.82	44.21	637.26	2.33	212.14	1270.8
B Cor 0.1	1.54	7.35	103.40	1.76	45.94	319.90
B Cor 0.5	0.72	3.22	44.03	0.74	47.96	359.88
C Ran 0.1	4.94	0.76	6NS	4.97	0.90	6NS
C Ran 0.5	3.74	1.75	6NS	1.75	6.31	6NS
C Cor 0.1	1.84	0.50	6NS	1.92	0.72	6NS
C Cor 0.5	0.00	0.87	6NS	0.00	0.53	6NS

5.9.3 Assessment of the different components

In this section, we highlight certain of the contributions of the different components that influence the solution of the instances. In particular, we

report the percentage of reduction of the degree-delay preprocessing (see section 5.6.1), the gap and the computational time of all the algorithms in which the LP preprocessing has not been executed and the computational comparison of the usage of the lifted constraints (5.31) and of the unlifted constraints (5.30) for formulation F_1 and F_2.

Table 5.3: Gap and computational times for the algorithm without the LP preprocessing

	F_1			F_2		
Problem	$Gap \times 100$	T	T_{max}	$Gap \times 100$	T	T_{max}
B Ran 0.1	9.44	0.13	1.30	7.54	5.94	99.31
B Ran 0.5	6.37	0.77	2.34	3.83	4.46	43.22
B Cor 0.1	5.27	0.21	1.27	2.79	1.97	15.23
B Cor 0.5	3.52	0.45	2.69	0.97	0.83	17.52
C Ran 0.1	7.60	189.90	2NS	5.44	68.02	5NS
C Ran 0.5	5.84	82.52	3NS	2.48	127.14	4NS
C Cor 0.1	6.28	210.50	2NS	2.39	22.25	5NS
C Cor 0.5	2.46	94.73	1NS	0.04	88.80	6NS

	F_3			F_4		
Problem	$Gap \times 100$	T	T_{max}	$Gap \times 100$	T	T_{max}
B Ran 0.1	4.02	27.47	24.46	4.66	163.9	1210.2
B Ran 0.5	2.28	43.55	549.3	2.33	212.1	1270.8
B Cor 0.1	1.45	4.62	36.14	1.76	45.94	319.90
B Cor 0.5	0.64	2.72	21.18	0.74	47.96	359.88
C Ran 0.1	4.93	1.14	6NS	4.97	4.37	6NS
C Ran 0.5	1.39	315.30	5NS	2.04	7.57	6NS
C Cor 0.1	1.87	173.00	5NS	2.09	1.68	6NS
C Cor 0.5	0.47	0.54	6NS	0.00	0.55	6NS

In Table 5.4, we present the mean percentage of reduction of the num-

Table 5.4: Degree-delay preprocessing reduction

Problem	$\%n$	$\%m$	$\%arc$
B Ran 0.1	45.85	15.50	49.97
B Ran 0.5	38.32	13.87	31.99
B Cor 0.1	46.98	15.48	47.50
B Cor 0.5	34.06	12.25	28.78
C Ran 0.1	61.94	14.09	59.51
C Ran 0.5	51.32	12.65	45.06
C Cor 0.1	60.08	12.81	56.00
C Cor 0.5	51.94	12.65	45.49

ber of nodes, destinations and arcs performed only using the degree-delay preprocessing (the LP preprocessing is not performed in this case) for the different instances we have generated. If n is the original number of nodes and n' the number of nodes in the reduced problem in column $\%n$ we report the mean of the percentage of the values $(n-n')/n$ over all the instances belonging to the same class of problems (similarly for column $\%m$ and $\%arcs$). The number of nodes is almost halved and there is a consistent reduction on the number of arcs, the reduction is more effective on the class C than on the class B and the effect of the preprocessing based on the delay can be noticed in the higher percentage of reduction of the size of the problem when Δ is only ten percent more than the value that make the problem feasible ($\Delta = 1.1 * MP$). When only the delay-degree preprocessing is performed to reduce the size of the problem, the relations among the formulations in terms of gap and computational time do not change as it is easy to see in Table 5.3. In most of the problems the gap is slightly reduced. We have not reported here another table to show the solution time of the different algorithms on the original graph (that is on the graph where no preprocess-

Table 5.5: Improvement of the lifted constraints (5.31) with respect to the unlifted constraints (5.30)

		(5.31)		*(5.30)*	
Problem		$Gap \times 100$	T	$Gap \times 100$	T
B Ran 0.1	F_1	9.44	0.12	14.78	0.12
B Ran 0.5	F_1	6.37	0.77	13.60	1.06
B Cor 0.1	F_1	5.28	0.21	12.84	0.14
B Cor 0.5	F_1	3.52	0.45	12.12	0.47
B Ran 0.1	F_2	7.54	5.94	7.85	10.31
B Ran 0.5	F_2	3.83	4.46	3.93	14.73
B Cor 0.1	F_2	2.79	1.98	2.92	2.76
B Cor 0.5	F_2	0.97	1.77	0.98	2.57

ing is performed), because even some of the instances of the class B are not solved within the time limit.

In Table (5.5), we compare the impact of the lifted constraints (5.31) with respect to the unlifted constraints (5.30). For formulation F_1, the usage of constraints (5.31) strongly reduces the value of *Gap*, but they do not improve the solution time, whereas for formulation F_2 constraints (5.31) reduce the computational time, but do not decrease significantly the *Gap*.

Table 5.6: Computational time of the class B and C for the Steiner Tree problem

Problem	T	T_{max}
B	0.47	2.68
C	104.14	827.53

Finally, taking a sufficiently big value of Δ the delay constraints become redundant for the optimal solution and in this case we have solved the Steiner Tree problem on the graph reduced by using only the degree preprocessing; the mean and the maximal computational time for the class B and C with formulation F_1 are reported in Table 5.6. All the instances of the Steiner Tree problem are solved within the time limit.

5.10 Conclusions

In this chapter, in order to guarantee a Quality of Service in the communications, we have considered the Delay-constrained Steiner Tree problem. We have proposed four different formulations for modelling the problem together with a preprocessing based on the degree-delay characteristics and on the reduced costs properties, for reducing the size of the problems. The computational results, we have provided, suggest the usage of different techniques for solving those problems that are not solved so far. Another interesting problem to deal with is to apply the delay constraints to the wireless Ad-Hoc networks.

List of symbols

- $\mathbb{R}$: the set of real numbers
- $\mathbb{R}_+$: the set of nonnegative real numbers
- $\mathbb{Z}$: the set of integer numbers
- $\{0,1\}^n$: is the cartesian product of n copies of the set $\{0,1\}$
- Let $a \in \mathbb{R}$, $\lfloor a \rfloor := \max\{c \in \mathbb{Z} : c \leq a\}$
- Let $a \in \mathbb{R}$, $\lceil a \rceil := \min\{c \in \mathbb{Z} : c \geq a\}$
- $\mathbf{1}$ is a vector with all the components equal to 1
- $|U|$ is the number of elements beloging to the set U
- Let V be a set and $S \subseteq V$, define $S^c := V \setminus S$.

References

[1] E. Althaus, G. Călinescu, I.I. Măndoiu, S. Prasad, N. Tchervenski, and A. Zelikovsky. Power efficient range assignment in ad-hoc wireless networks. In *Proceedings of the IEEE WCNC*, pages 1889–1894, 2003.

[2] E. Althaus, T. Polzin, and S. Vahdati Daneshmand. Improving linear programming approaches for the Steiner tree problem. Technical report, 2003.

[3] K. Altinkemer, F.S. Salman, and P. Bellur. Solving the minimum energy broadcasting problem in ad hoc wireless networks by integer programming. In *Proceedings of the INOC*, pages B2.635–B2.642, 2005.

[4] E. Amaldi, P. Belotti, A. Capone, and F. Malucelli. Optimizing base station location and configuration in umts networks. *Annals of Operations Research*, to appear.

[5] K. Andersen and Y. Pochet. Coefficient strengthening: a tool for formulating mixed integer programs. *Submitted*.

[6] A. Balakrishnan and N.R. Patel. Problem reduction methods and a tree generation algorithm for the Steiner network problems. *Networks*, 17:65–85, 1987.

[7] E. Balas and S.M. NG. On the set covering polytope: I all the facet with coefficients in $\{0, 1, 2\}$. *Mathematical Programming*, 43:57–69, 1989.

[8] E. Balas and S.M. NG. On the set covering polytope: II lifting the facets in $\{0, 1, 2\}$. *Mathematical Programming*, 45:1–20, 1989.

[9] J.E. Beasley. An SST-Based algorithm for the Steiner tree problems in graph. *Networks*, 19:1–16, 1989.

[10] D. Bertsimas and J.N. Tsitsiklis. *Introduction to Linear Optimization*. Athena Scientific, 1997.

[11] D. Blough, M. Leoncini, G. Resta, and P. Santi. On the symmetric range assignment problem in wireless ad hoc networks. In *Proceedings of the 2nd IFIP International Conference on Theoretical Computer Science*, pages 71–82, 2002.

[12] S.A. Borbash and E.H. Jennings. Distributed topology control algorithm for multihoc wireless networks. In *Proceedings of IJCNN*, 2002.

[13] M. Cagalj, J.-P. Hubaux, and C. Enz. Minimum-Energy Broadcast in all-wireless networks: NP-completeness and distribution issues. In *Proceedings of the MOBICOM*, 2002.

[14] I. Caragiannis, C. Kaklamanis, and P. Kanellopoulos. Energy-efficient wireless network design. *Theory of computing system*, 39:593–617, 2006.

[15] J. Cargigny, D. Simplot, and I. Stojmenović. Localized minimum-energy broadcasting in ad-hoc networks. In *Proceedings of the IEEE INFOCOM*, 2003.

[16] S. Chopra, E. Gorres, and M.R. Rao. Solving the Steiner tree problem on a graph using branch and cut. *ORSA Journal on Computing*, 4:320–335, 1992.

[17] T. Christof and A. Löbel. Porta. Available on-line at the web address $http://www.zib.de/Optimization/Software/Porta/$.

[18] T. Chu and I. Nikolaidis. Energy efficient broadcast in modile ad hoc networks. In *Proceedings of AD-HOC Networks and Wireless*, 2002.

[19] V. Chvátal. Edmonds polytope and a hierarchy of combinatorial problems. *Discrete Mathematics*, 4:305–337, 1973.

[20] A. Clementi, P. Crescenzi, P. Penna, G. Rossi, and P. Vocca. On the complexity of computing minimum energy consumption broadcast subgraphs. In *Symposium on Theoretical Aspects of Computer Science*, pages 121–131, 2001.

[21] A. Clementi, P. Penna, and R. Silvestri. Hardness results for the power range assignment problem in packet radio networks. *Lectures Notes on Computer Science*, 1671:195–208, 1999.

[22] G. Cornuéjols and A. Sassano. On the $0, 1$ facets of the set covering polytope. *Mathematical Programming*, 43:45–55, 1989.

[23] A.K. Das, R.J. Marks II, M. El-Sharkawi, P. Arabshahi, and A. Gray. The minimum power broadcast problem in wireless networks: an ant colony system approach. In *Proceedings of the IEEE Workshop on Wireless Communications and Networking*, 2002.

[24] A.K. Das, R.J. Marks, M. El-Sharkawi, P. Arabshani, and A. Gray. Optimization methods for minimum power bidirectional topology construction in wireless networks with sectored antennas. *Submitted*.

[25] A.K. Das, R.J. Marks, M. El-Sharkawi, P. Arabshani, and A. Gray. Minimum power broadcast trees for wireless networks: integer programming formulations. In *Proceedings of the IEEE INFOCOM*, 2003.

[26] M. Desrochers and G. Laporte. Improvements and extensions to the Miller-Tuckler-Zemlin subtour elimination constraints. *Operation Research Letters*, 10:27–36, 1991.

[27] E.W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.

[28] J.J. Dongarra. Performance of various computers using standard linear algebra software in a fortran environment. Technical Report CS-89-85, University of Tennessee, 2003.

[29] D.Z. Du, B. Lu, H.Ngo, and P.M. Pardalos. *Steiner tree problems*, pages 227–290. Kluwer Academic Publishers, 2001.

[30] F. Eisenbrand. On the membership problem for the elementary closure of a polyhedron. *Combinatorica*, 19:297–300, 1999.

[31] M. Fischetti and A. Lodi. Optimizing over the first Chvátal closure. *Submitted*.

[32] M.L. Fisher. The lagrangian relaxation method for solving integer programming problems. *Management Science*, 27(1):1–18, 1981.

[33] L.R. Ford and D.R. Fulkerson. *Flows in Networks*. Princeton University Press, 1962.

[34] K. Fukuda. cdd. Available on-line at the web address *http* : $//www.ifor.math.ethz.ch/ \sim fukuda/cdd\_home/cdd.html$.

[35] R.E. Gomory. Outline of an algorithm for the integer solutions to linear programs. *Bulletin of the American Mathematical Society*, 64:275–278, 1958.

[36] S. Guo and O. Yang. Minimum-energy broadcast routing in wireless multi-hop networks. In *IEEE PCC*, 2003.

[37] S. Guo and O. Yang. QoS-aware minimum energy multicast tree construction in wireless ad hoc networks. *Ad Hoc Networks*, 2:217–229, 2004.

[38] G.Y. Handler and I. Zang. A dual algorithm for the constrained shortest path problem. *Networks*, 10:293–310, 1980.

[39] M. Haouari and P. Dejax. Plus court chemin avec dapplication aux problèmes de tournées. *RAIRO*, 31:117–131, 1997.

[40] Z. Huang, C.-C. Shen, C. Srisathapornphat, and C. Jaikaeo. Topology control for ad hoc networks with directional antennas. In *Proceedings of the Eleventh International Conference on Computer Communications and Networks*, 2002.

[41] F.K. Hwang, D.S. Richards, and P. Winter. The Steiner tree problem. *Annals of Discrete Mathematics*, 53, 1992.

[42] R.J. Marks II, A.K. Das, M. El-Sharkawi, P. Arabshani, and A. Gray. Minimum power broadcast trees for wireless networks: optimizing using the viability lemma. In *Proceedings of the IEEE ISCAS*, 2002.

[43] X. Jia, D. Li, and D. Du. QoS topology control in ad hoc wireless networks. In *Proceedings of the INFOCOM*, 2004.

[44] H.C. Joksch. The shortest route with constraints. *Journal of Mathematical Analysis and Applications*, 14:191–197, 1966.

[45] Richard M. Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, pages 85–103. Plenum, 1972.

[46] L. Kirousis, E. Kranakis, D. Krizanc, and A. Pelc. Power consumption in packet radio networks. *Theoretical Computer Science*, 243:289–305, 2000.

[47] T. Koch and A. Martin. Solving Steiner tree problems in graph to optimlity. *Networks*, 32:207–232, 1998.

[48] T. Koch, A. Martin, and S. Voβ. SteinLib. *http : //elib.zib.de/steinlib.*

[49] V.P. Kompella, J.C. Pasquale, and G.C. Polyzos. Multicast routing for multimedia communication. *IEEE ACM*, 1(3):289–292, 1993.

[50] Z. Kun, W. Heng, and L. Feng-Yu. Distributed multicast routing for delay and delay variation-bounded Steiner tree using simulated annealing. *Computer Communications*, 28:1356–1370, 2005.

[51] V. Leggieri, M. Haouari, S. Layeb, and C. Triki. Delay-constrained Steiner tree problem. *Working paper*, 2007.

[52] V. Leggieri, P. Nobili, and C. Triki. Minimum power multicasting problem in wireless networks. *Submitted.*

[53] J. Leino. Optimal multicast routing in ad hoc networks. Technical report, 2002.

[54] F. Li, S. Mannor, and A. Lippman. Probabilistic optimization for energy-efficient broadcast in all-wireless networks. In *Proceedings of the 2005 Conference on Information Sciences and Systems, The Johns Hopkins University*, 2005.

[55] W. Liang. Constructing minimum-energy broadcast trees in wireless ad hoc networks. In *Proceedings of MOBIHOC*, pages 112–122, 2002.

[56] E. Lloyd, R. Liu, M. Marathe, R. Ramanathan, and S. Ravi. Algorithmic aspects of topology control problems for ad hoc networks. In *Proceedings of the ACS MobiHoc*, pages 123–134, 2002.

[57] T.L. Magnanti and L. Wolsey. *Optimal trees* , volume 7, pages 503–615. North-Holland, 1995.

[58] C.E. Miller, A.W. Tucker, and R.A.Zemlin. Integer programming formulation of the Traveling Salesman problems. *J. ACM*, 7:326–329, 1960.

[59] M. Minoux. Plus court chemin avec contraintes: algorithmes et applications. *Annales des télécommunications*, 30:383–394, 1975.

[60] R. Montemanni and L.M. Gambardella. Exact algorithms for the minimum power symmetric connectivity problem in wireless networks. *Computers and Operations Research*, 31(10):1667–1680, 2004.

[61] R. Montemanni, L.M. Gambardella, and A.K. Das. The minimum power broadcast tree problem in wireless networks: a simulated annealing approach. In *Proceedings of the IEEE WCNC*, 2005.

[62] R. Montemanni, L.M. Gambardella, and A.K. Das. *Models and algorithms for the MPSCP: an overview*, pages 133–146. Auerbach Publications, 2006.

[63] R. Montemanni, V. Leggieri, and C. Triki. Mixed integer formulations for the probabilitic minimum energy broadcast in wireless networks. *Submitted*.

[64] G.L. Nemhauser and L.A. Wolsey. *Integer and Combinatorial Optimization*. Wiley, 1988.

[65] C. Noronha and F. Tobagi. Optimum routing of multicast streams. In *INFOCOM*, pages 865–873, 1994.

[66] C.A.S. Oliveira and P.M. Pardalos. A survey of combinatorial optimization problems in multicast routing. *Computers & Operations Research*, 32:1953–1981, 2005.

[67] M. Padberg and L. Wolsey. Trees and cuts. *Annals of Discrete Mathematics*, 17:511–517, 2003.

[68] C.H. Papadimitriou and K. Steiglitz. *Combinatorial optimization, Algorithms and Complexity*. Dover, 1998.

[69] T. Polzin and S. Vahdati Daneshmand. A comparison of Steiner tree relaxations. *Discrete Applied Mathematics*, 112:241–261, 2001.

[70] R.C. Prim. Shortest connection networks and some generalizations. *Be14.32 13/04/2007ll System Technical Journal*, 36:1389–1401, 1957.

[71] R. Ramanathan and R. Rosales-Hain. Topology control of multihop wireless networks using transmit power adjustment. In *Proceedings of the IEEE Infocom*, pages 404–413, 2000.

[72] T. Rappaport. *Wireless Communications: Principles and Practices*. Prentice Hall, 1996.

[73] F. Rossi, A. Sassano, and S. Smriglio. Models and algorithms for terrestrial digital broadcasting. *Annals of Operations Research*, 3(107):267–283, 2001.

[74] A. Ruszczyński and A. Shapiro. *Stochastic Programming*. Elsevier, 2003.

[75] H.F. Salama, D.S. Reeves, and Y. Viniotis. The delay-constrained minimum spanning tree problem. *ISCC*, pages 699–704, 1997.

[76] A. Sassano. On the facial structure of the set covering polytope. *Mathematical Programming*, 44:181–202, 1989.

[77] A. Sassano. *Modelli e algoritmi della ricerca operativa*. FrancoAngeli, 1999.

[78] S. Singh, C. Raghavendra, and J. Stepanek. Power-aware broadcasting in mobile ad hoc networks. In *Proceedings of the IEEE PIMRC*, 1999.

[79] R. Sriram, G. Manimaran, and C. Siva Ram Murthy. Algorithms for delay-constrained low-cost multicast tree construction. *Computer Communications*, 21(18):1693–1706, 1998.

[80] S.Y. Tseng, Y.M. Huang, and C.C. Lin. Genetic algorithm for delay- and degree-constrained multimedia broadcasting on overlay networks. *Computer Communications*, 29:3625–3632, 2006.

[81] E. Uchoa, M. Poggi de Aragao, and C.C. Ribeiro. Preprocessing Steiner problems from VLSI layout. Technical report, 1999.

[82] P.-J. Wan, G. Călinescu, X.-Y. Li, and O. Frieder. Minimum energy broadcast routing in static ad hoc wireless networks. In *Proceedings of the IEEE Infocom*, pages 1162–1171, 2001.

[83] R. Wattenhofer, L. Li, P. Bahl, and Y.M. Wang. Distributed topology control for power efficient operation in multihop wireless ad hoc networks. In *Proceedings of the Infocom*, pages 1388–1397, 2001.

[84] J. Wieselthier, G. Nguyen, and A. Ephremides. On the construction of energy-efficient broadcast and multicast trees in wireless networks. In *Proceedings of the IEEE INFOCOM*, pages 585–594, 2000.

[85] J.E. Wieselthier, G. Nguyen, and A. Ephremides. Algorithms for energy-efficient multicasting in static ad hoc networks. *Mobile Networks and Application*, 6:251–263, 2001.

[86] L.A. Wolsey. *Integer Programming*. Wiley-Interscience, 1998.

[87] R.T. Wong. A dual ascent approach for Steiner tree problems on a directed graph. *Mathematical programming*, 28:271–287, 1984.

[88] D. Yuan. An integer programming approach for the minimum-energy broadcast problem in wireless networks. In *Proceedings of the INOC*, pages B2.643–B2.650, 2005.

[89] Q. Zhu, M. Parsa, and J.J. Garcia-Luna-Aceves. A source-based algorithm for the delay-constrained minimum cost multicasting. In *INFOCOM*, pages 377–385, 1995.

Acknowledgments

At the end of this dissertation, I would like to express my gratitude to all the people who have helped me during my Ph.D studies.

First of all, I would like to thank my supervisor Prof. Paolo Nobili, for his assistance and encouragement during these years.

Thanks to Dr. Chefi Triki and to all the other people in the research group of Operations Research at the University of Lecce for many fruitful discussions and suggestions.

My gratitude goes to Prof. Yves Pochet and Prof. Laurence Wolsey for their kind hospitality during my stay in Louvain-la-Neuve and for their assistance in my research activities at CORE.

Thanks to Prof. Mohamed Haouari for giving me the opportunity of collaborating with him and for his friendly hospitality at the Ecole Polytechnique de Tunis.

Thanks to Dr. Roberto Montemanni for many fruitful discussions.

Thanks to all my colleagues at the Department of Mathematics in Lecce and to all people that I met in the laboratory of Tunis and at CORE.

And, finally, I would like to thanks for their patience and support Fabrizio and my family.

Valeria Leggieri

April 2007