ALMA MATER STUDIORUM
UNIVERSITÀ DEGLI STUDI DI BOLOGNA

Dottorato di Ricerca in
Automatica e Ricerca Operativa

MAT/09 - XXII Ciclo

Orientation and layout problems on graphs, with applications

Emiliano Traversi

Il Coordinatore
Prof. Claudio Melchiorri

Tutor
Prof. Alberto Caprara

A.A. 2006–2009

Keywords

Integer Linear Programming

Graph Orientation Problems

Personal Rapid Transit

Minimum Linear Arrangement

Betweenness Variables

Contents

List of Figures

List of Tables

Preface

This Ph.D. thesis includes the research activity developed together with my Advisor Professor Alberto Caprara during my Ph.D. course in "Aut. and O.R." ("Automatica e Ricerca Operativa") at the University of Bologna.

The topic of my thesis concerns optimization over graphs, and arises from the collaboration with Professor Joerg Schweizer from the transportation institute of the department of Civil Engineering of the University of Bologna and with Professors Gerhard Reine and Marcus Oswald from the department of Combinatorial Optimization of the University of Heidelberg.

Two problems are presented in this work: a Graph Orientation Problem and a Graph Lay-Out problem.

The Orientation Problem arises from real-world application studied together with the transportation institute and with several private companies.

The Layout Problem is a classical problem known from the 1960s and in this joint work with the Heidelberg University we provide interesting advancement on this subject.

In both problems I used a wide range of Operation Research subjects: Mixed Integer Linear and Non-Linear Programming Theory, Graph Theory and Polyhedral Theory are the main tool needed to solve the problem faced.

Solving problem arising from real-world situation means working together with people coming from academic fields different from mine and with people coming from private companies, this has given me the opportunity to learn how to communicate with them and to see how much Operation Research can be flexible and effective in practice.

On the other hand the Layout Problem has been useful for work with people from the same field of mine, allowing me to improve my knowledge of the subject and to see different ways to work.

finally, it is also surprising how the knowledge obtained from working in solving a problem can be reused in solving another one with a complete diverse subject, this is due to the flexibility of the instrument used and is something I learned to appreciate during this years.

Chapter 1

Introduction

Graph theory is a really wide area that has been studied deeply in the last decades.

Graphs are a flexible and useful mathematical tool, capable to model and solve a wide range of really different problem provided by real-world situation (vehicle routing, network design, timetabling are only few example of applications that makes an intense use of graphs).
On the other hand graph theory provides a lot of challenging questions on itself really hard to solve.

In this work we will talk about Orientation Problems, Flow Problems and Lay-Out problems, three big classes of problems on graphs.

A general Orientation Problem can be stated as follows:

Definition 1 (Orientation Problem) ***Input:***
an undirected or mixed graph $G = (V, E \cup A)$ with costs and capacities associated to its edges and its nodes, plus eventually other additional condition (like connectivity condition).
Output:
an Orientation of G, i.e a directed graph $G' = (V, A')$ with $A' = A \cup A''$ and $A'' = \{a \equiv (i, j) \mid \exists\{i, j\} \in E \vee \exists\{j, i\} \in E\}$.
Subject to:
a certain optimum criteria on the final orientation.

A lot of problems fall into this class, unfortunately it does not exist an exhaustive and unambiguous classification that really helps in identifying them. Roughly speaking, some of the requirements that can be imposed on the final orientation are:

- Connectivity
  given a set of pairs of nodes, it is required to ensure the existence of one ore more (edge or node disjoint) paths between each pair in this set. Finding a strongly connected orientation for a given mixed graph falls into this category and has been proved. in 1985 by to be polynomial algorithm for solving this problem [8].
  A lot of reliability issues are part of this section: by imposing the existence of more than one edge disjoint path for each pair of nodes we are sure the the graph will remain connected even if one arcs is deleted.

- forbid certain class of subgraphs in the final orientation
  the most frequent issue is to demand for a directed graph with no cycles (notice how this property is somehow complementar of asking for a strongly connection).

- upper or lower bounds on the nodes degree
  notice thatimposing the each nodes in the final orientation must have a certain value k for his in-degree or out-degree value is different from imposing connectivity requirements.

- adding more general constraints like $\sum_{a\in\delta^+(S)} x_a \geq f(S) \forall S \subset V$
  In [13] Khanna et al. proved that this problem is polynomial if we impose require f to be submodular and if we impose a linear objective function on the use of one arc .

For a partial survey on network orientation problems see [12] or TODO.

Flows problem are one of the earlier applcation of graph theory and linear programming. Shortest Path Problem, Min Cost Flow Problem, are member of this important class of problems. Among all these there is the well known Multicommodity Flow Problem.

Let $G = (V, A)$ be a directed graph with arcs lengths ℓ_a and arcs capacities c_a, let d_i be the demand (if positive) or the offer (if negative) of flow for a node i, moreover let f_a be a set of continuous flow variable, one for each arc. the min xost multicommodity flow Problem is stated as follows:

$$\min_{a\in A} f_a l_a \tag{1.1}$$

$$\sum_{a\in\delta^+(i)} f_a - \sum_{a\in\delta^-(i)} f_a = -d_i, \quad \forall i \in V \tag{1.2}$$

$$f_a \leq c_a, \qquad \forall a \in A \tag{1.3}$$

$$f_a \geq 0, \qquad \forall a \in A \tag{1.4}$$

Constraints (1.2 are the flow constraints and constraints (1.2) are the cpacity constraints.

In the first part of this work we deal with a problem that combines aspect of the Multicommodity Flow Problem with other related to the Orientation Problem. The basic idea is, given an undirected graph, to provide the orientation with the lowest multicommodity flow. Other constraints are added in order to meet the practical requirements of the application studied but the idea remains basically the same.

Even if the problem is not completely new in the literature(see for exaple [7] or [12]) no extensive computational experiments has been done so far for practically solve instancese of decent size.

The practicall input of our work comes from the demand for algorithms able to design large scale Personal Rapid Transit (PRT) networks.

PRT is an innovative pubblic transport system, based on automatic vehicles running on a dedicated guideways, studied for providing an hight quality and capillary service.
The most simliar system to PRT are the Automatic Guided Vehices (AGV), from our application point of view the main difference between AGV and PRT instances is that with PRT the dimension of PRT instances is of several order of magnitude bigger that an typical instance of an AGV instance.

In order to solve the problem we make extensive use of mathematical models. Beside the issues related to the solution of these models, also the definition of a model itself is a non trivial task. For this reason a wide range of formulation are proposed, in order to be able to select the one who better fits the application faced.

In addition we use a PRT network simulator to validate the models provided.
A real wrold application is tested and we show how the results obtained in the simulation are similar to the one obtained by the ILP model used.

Once we have ensured that the models used are valid, solving instances of decent size for these models is not a trivial task. Exact and heuristic algorithm will be provided, comparing their performances on different set of instances.

The main issue in modelling the problem is how to deal with the vehicle flows behaviour and, more specifically, how to handle capacity limit and consequently congestions.

As first approximation we can suppose the problem as uncapacitated and feel free to send

any aammount of flow in an arc.
Techical specification of PRT systems says that the vehicle speed can be assumed constant as long as the total arc flow remains under a certain limit. If the number of vehicles per section exceed this limit for security reason the speed must decrease and this means that the path of some vehicle can no longer be the shortes (in terms of time spent).

In real PRT instances congestions occurs quite likely, hence how to deal with them is not a theroetical curiosity but one of the main issues in the desing of a PRT network . Models are presented with an increasing level of complexity, directly related to the hipothesis used to represents congestions.

Lay-out problem are another family of classical graph theory problem, they can be stated as follows:

Definition 2 (Graph Lay-out Problem) ***Input:***
an undirected graph $G = (V, E)$ with costs associated to its edges.
Output:
a Lay-out of G, i.e a permutation $p : |V| \rightarrow |V|$.
Subject to:
a certain optimum criteria on the final lay-out.

The definition "lay-out" comes from the fact that the problem can be visualized as a linear placement of the nodes on one with the distance between two nodes equal to the difference between their position, i.e. $|p(i) - p(j)|$.

Some example of lay-out problems are :

- Minimum Bandwidth Problem
  the objective function is $\min \max_{e \equiv \{i,j\} \in E} |p(i) - p(j)|$, i.e. the minimizing the maximum distance between two nodes joined by and edge.
  In [14], Caprara et al. provided an efficient enumerative algorithm for solving the problem.

- Minimum Linear Arrangement Problem
  the objective function is $\min \sum_{e \equiv \{i,j\} \in E} |p(i) - p(j)|$, i.e. minimizing the

For a more detailed survey on Graph layout problems see [15].

In the second part of this work we deal with a new Integer Programming formulation for the Linear Arrangement Problem. With this new approach we have been able to solve

benchmark instances never solved before and to drastically decrese the computational time for the instances solved so far.

In Chapter 2 talks about the Graph Orientation Problem.
In Section 2.1 and Section 2.2 we presents the problem and its application.
In Section 2.3 we introduce the models used and in Section 2.4 complexity aspect are treated. Section 2.5 is dedicated to the comparison between models used and the PRT simulator in order to validate the models used.
Section 2.6-2.8 describes the method used to solve the models introduced. In Section 2.9 we introduce the instances used and in Section 2.10 we discuss about the computational results.

In Chapter 3 the Linear arrangement problem is faced.
In Section 3.1 the problem is formally introduced and in Section 3.2 the previous solution approaches are presented.
In Section 3.3 we present our approach based on a new IP formulation and in Section 3.4 a Polyhedral analysis for this new formulation is described.
In Section 3.5 we provide preliminary computational results.

Chapter 2

Graph Orientation Problem

2.1 Motivation and previous work

This work is motivated by the requirements to design optimal, large scale Personal Rapid Transit (PRT) networks for entire urban areas. PRT is an innovative type of public transport [6], with the first system planned to operate in public by the end of 2010 at the new terminal 5 of London-Heathrow airport. PRT is composed of a fleet of fully-automated and electrically-driven vehicles for up to 6 passengers, running on a dedicated network of one-way guide-ways with small dimensions. Similar to Taxis, PRT vehicles are available on-demand and 24 hours a day. The access stations are off-line, ensuring that all vehicles can reach their pre-programmed destinations without transfers or intermediate stops. PRT is seen as a truly sustainable urban mobility alternative that offers high-quality, emission-free and low energy-usage transportation which is accessible to and affordable for all social groups. This is why PRT has been chosen as the exclusive transport system within the sustainable, completely energy self-sufficient city of the Masdar ("The Masdar-Initiative", Dubai, United Emirates), covering a 5x5 kilometers area with approximately 100 stations and 40 kilometers of guide-ways. Capacity limits of PRT systems are a crucial issue and are vital to the feasibility of large-scale networks. In principle, exceeding these capacity limits can be avoided in three ways: (i) by reducing headways between vehicles, (ii) by an intelligent, congestion-avoiding vehicle routing, (iii) by network that is optimized for a-priory known trip demand patterns. The present work is concerned with the last option, defining models in order to find an optimized layout for a non trivial PRT network: starting from a network of initially undirected links, travel costs and a demand matrix between origin- and destination-nodes, the proposed method will orient all the links so as to obtain the best orientation, according to the traffic hypothesis.

2.2 Introduction

As mentioned before, this work deals with models aimed at defining a "good" lay-out for a PRT network. For a better understanding of the problem we must specify what are the data and the decision variables involved in its definition. Suppose that we want to implement a PRT system in a certain area. As input of the problem we know in advance: (i) where the stations are positioned

(ii) all the available connections between the stations where the guide-ways for the PRT network can lie

(iii) one (or a set of) origin-destination (OD) matrix, with the demand of vehicles between each pair of stations.

Al the models examined take the above data as starting point; roughly speaking thsi corresponds to an undirected graph with a demand associated to some pairs of nodes. More specifically we do not take into consideration the possibility of paying a cost for using a connection between nodes. Hence all the connections can be used for free. Moreover as starting analysis we consider only one single OD-matrix, usually the one corresponding to the peak-hour demand in order to "tune" the system to the worst possible scenario.

Under this assumption all the models described share (almost) the same feasible region, i.e. they provide an orientation of an undirected graph, representing how the guide-ways must be directed in the final lay-out. Note that once the orientation is fixed the problem reduces to finding an optimal multicommodity flow in the resulting oriented graph of vehicles that satisfies the demand between stations.

Among all the possible orientations we must choose the one that provides the best level of service for the customers. In our work we assume that this corresponds to minimizing the sum of the total time spent by each of them in the system. A crucial assumption related to this aspect is how to deal with capacities and congestions. In this work we deal with several different ways to model congestions, with an increasing level of complexity. For each model we propose exact and heuristic methods. In order to validate the effectiveness of the models we compare their results with a software simulating a PRT systems.

2.3 Problem formulations

2.3.1 Common definitions

Let $G = (V, E)$ be an undirected graph, with a length ℓ_e and a capacity $c_e = c_{i,j}$ associated with each edge $e = \{i, j\} \in E$. Moreover, let $R \subset V \times V$ be a set of origin-destination pairs, with a demand d_r associated with each origin-destination pair $r = (s_r, t_r) \in R$.

Let $S \subset V$ be the set of all the stations defined as $\{v \in V : \exists r = (s_r, t_r) \in R, s_r \equiv v \vee t_r \equiv v\}$.

We let an *orientation* of G be a directed graph $D = (V, A)$ such that each arc $(i, j) \in A$ corresponds to an edge $\{i, j\} \in E$ and, for each edge $\{i, j\} \in E$, at most one of the arcs (i, j) and (j, i) is in A.
Finally let $n = |V|$, $m = |E|$ and $s = |S| \leq n$.

2.3.2 Network Orientation - basic formulation

We address the problem of finding an orientation D of G, along with a path in D joining each source-destination pair, so as to minimize the weighted sum of the lengths of these paths, where the weight of the path joining each origin-destination pair r is equal to its demand d_r, and the length of each arc $a = (i, j)$ is equal to $\ell_a = \ell_{i,j}$.

The natural MILP formulation of the problem is the following. For convenience, let $\overline{A}$ denote the set of the possible arcs arising from orientations of the edges in E. Moreover, for a vertex $i \in V$, let $\delta^+(i)$ and $\delta^-(i)$ denote, respectively, the set of arcs in $\overline{A}$ exiting from and entering in V. The MILP formulation contains binary variables $x_a \equiv x_{i,j}$, equal to one if the arc $a \equiv (i, j) \in \overline{A}$ is present in D, i.e. if edge $\{i, j\}$ is oriented from node i to node j, and binary variables $y^r_a \equiv y^r_{i,j}$, equal to one if the path joining origin s_r to destination t_r uses arc $a \equiv (i, j) \in \overline{A}$. The corresponding Network Orientation Problem (NOP) reads:

Network Orientation Problem - Basic Formulation (NOP-BF)

$$\min \sum_{r\in R} \sum_{(i,j)\in\overline{A}} d^r \ell_{i,j} y^r_{i,j} \tag{2.1}$$

$$x_{ij} + x_{ji} \leq 1, \qquad \forall \{i, j\} \in E \tag{2.2}$$

$$\sum_{(i,j)\in\delta^+(i)} y^r_{ij} - \sum_{(j,i)\in\delta^-(i)} y^r_{ji} = \begin{cases} 1, & \text{if } i = s_r \\ -1, & \text{if } i = t_r \\ 0, & \text{otherwise} \end{cases}, \quad \forall i \in V, \forall r = (s_r, t_r) \in R \tag{2.3}$$

$$y^r_{ij} \leq x_{ij}, \qquad \forall (i, j) \in \overline{A}, \forall r \in R \tag{2.4}$$

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

$$y^r_{ij} \geq 0, \qquad \forall (i, j) \in A, \forall r \in R \tag{2.6}$$

Constraints (2.2) impose that D is an orientation of G. Equations (2.3) guarantee that the arcs a with $y^r_a = 1$ define a path from s_r to t_r in D, whereas inequalities (2.4) link the y and the x variables. These paths represents flows of vehicles carrying users from their original station to their destination (called full vehicle flows).

Note that the problem simply calls for an orientation of the edges of G so that the weighted sum of the shortest-path distances between origin-destination pairs in the resulting directed

graph D is minimized.

From now on we call (2.1)-(2.5) the Network Orientation Problem in the Basic Formulation (NOP).
Two extensions can be easily defined to take into account empty vehicles and capacities.

2.3.3 Network Orientation - empty vehicles

In order to guarantee that each vehicle goes back to its origin station, the full vehicle flow described in section 2.3.2 must be counterbalanced by empty vehicle flows.
For this reason we add in each station empty vehicle flow conservation constraints by defining a fictitious demand $D_{i,res}$ of empty vehicles for each station:

$$D_{i,res} = \sum_{r \in R: s_r = i} d_r - \sum_{r \in R: t_r = i} d_r \quad \forall i \in V \tag{2.7}$$

This fictitious demand represents the algebraic difference between the number of exiting and entering full vehicles at each demand-node, i.e. $D_{i,res} > (<)\ 0$ means that there is a demand (offer) of empty vehicles in node i.
In addition we add a set of continuous variables $w_a \equiv w_{i,j}$, representing the empty vehicle flow on link a and we obtain the Network Orientation Problem with Empty Vehicles (NOP-EV):

Network Orientation Problem with Empty Vehicles (NOP-EV)

$$\min \textstyle\sum_{r \in R} \sum_{(i,j) \in \overline{A}} d^r \ell_{ij} y^r_{ij} + \sum_{(i,j) \in \overline{A}} \ell_{ij} w_{ij} \tag{2.8}$$

$$x_{ij} + x_{ji} \leq 1, \qquad \forall \{i,j\} \in E \tag{2.9}$$

$$\sum_{(i,j) \in \delta^+(i)} y^r_{ij} - \sum_{(j,i) \in \delta^-(i)} y^r_{ji} = \begin{cases} 1, & \text{if } i = s_r \\ -1, & \text{if } i = t_r \\ 0, & \text{otherwise} \end{cases}, \quad \forall i \in V, \forall r = (s_r, t_r) \in R \tag{2.10}$$

$$y^r_{ij} \leq x_{ij}, \qquad \forall (i,j) \in \overline{A}, \forall r \in R \tag{2.11}$$

$$\sum_{(i,j) \in \delta^+(i)} w_{ij} - \sum_{(j,i) \in \delta^-(i)} w_{ji} = -D_{i,res} \qquad \forall i \in V, \tag{2.12}$$

$$w_{ij} \leq M_w x_{ij}, \qquad \forall (i,j) \in \overline{A} \tag{2.13}$$

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

$$w_{ij} \geq 0 \qquad \forall (i,j) \in \overline{A} \tag{2.15}$$

$$y^r_{ij} \geq 0, \qquad \forall (i,j) \in A, \forall r \in R \tag{2.16}$$

Constraints (2.12) impose a multi-origin multi-destination flow in the oriented graph and

inequalities (2.13) link the w and the x variables, in this case we need a "big M" because the $w_{i,j}$ are not binary; in fact we can set M to the sum of the demand of empty vehicles

$$M_w = D_{tot} = \sum_{i \in V : D_{i,res} \geq 0} D_{i,res} \tag{2.17}$$

The new objective function (2.8) takes into consideration also the total distance traveled by the empty vehicles.
From a practical point we want to minimize the empty vehicle flow because we want to make vehicles available as soon as possible for the incoming customers, according with the idea that the customers must wait as little as possible.
From now on, we always consider models with also empty vehicles, noting that the equivalent with no empty vehicle flows can be obtained simply by fixing to zero all the w_{ij} variables and setting $D_{i,res} = 0 \forall i \in V$.

2.3.4 Network Orientation - capacitated version

NOP-EV can be modified in order to take into consideration capacity constraints by adding the following constraints:

$$w_{i,j} + \textstyle\sum_{r \in R} d^r y^r_{i,j} \leq c_{i,j} x_{i,j}, \quad \forall (i,j) \in \overline{A} \tag{2.18}$$

As explained in more detail in Section (2.4), without the capacity constraints, having a 2-connected graph guarantees that we always have a feasible solution; on the other hand with the capacity condition (2.18) finding a feasible solution becomes hard.
In this cases we can add a new set of continuous variables z_{ij} representing the overflow along arc $\{i,j\}$, redefining the following relaxed version of the capacity constraints:

$$w_{ij} + \textstyle\sum_{r \in R} d^r y^r_{ij} \leq c_{ij} x_{ij} + z_{ij}, \quad \forall (i,j) \in \overline{A} \tag{2.19}$$

and substituting the objective function (2.8) with

$$\min \textstyle\sum_{r \in R} \sum_{(i,j) \in \overline{A}} d^r \ell_{i,j} y^r_{i,j} + \sum_{(i,j) \in \overline{A}} \ell_{i,j} w_{i,j} + M_z \sum_{(i,j) \in \overline{A}} z_{ij} \tag{2.20}$$

obtaining the the Network Orientation Problem with Hard Capacity constraints (NOP-

HCC).
Here, M_z is fixed to a sufficiently large value in order to give priority to the minimization of the total overflow.

If we want to deal with overflows in arcs in a more accurate way we need to introduce a direct dependency between flows in the arcs and the time necessary to traverse them via the introduction of non linearities as described in the next section.

2.3.5 Network Orientation - nonlinear formulation

If we keep the overflow equal to zero all the vehicles can drive at their maximum speed. At the same time a certain amount of overflow can be admitted without collapsing the system, the drawback of a congested connection being that the speed of the vehicles must decrease, hence the total traveling time increases. This (partial) dependency of the arc costs on the flow leads to a more precise description of the real behavior of the system.
We define the following parameters:
i) a first capacity limit c_a^0 , the maximum amount of flow that can be assigned to an arc if we do not want to have congestion
ii) a second capacity limit c_a^1 , the maximum amount of flow that can be assigned to an arc in any case
iii) the original length of an arc ℓ_{ij}^0, equal to the constant ℓ_{ij} in the previous formulations
iv) the final length of an arc $\ell'_{i,j}$ when the total flow in the arc is equal to c_a^1
We add continuous variables $f_{i,j}$ representing the overflow in arc (i,j) and $\ell_{i,j}$ representing the cost of an arc, being now continuous variables and not constant.
$\ell_{i,j}(f_{i,j})$ is defined as the following piecewise linear function :

$$\ell_{i,j}(f_{i,j}) = \begin{cases} \ell_{i,j}^0 + \frac{\ell'_{i,j}-\ell_{i,j}^0}{c_{i,j}^1-c_{i,j}^0} f_{i,j}, & \text{if } \ 0 \leq f_{i,j} \leq (c_{i,j}^1 - c_{i,j}^0) \\ +\infty, & \text{if } \ f_{i,j} > (c_{i,j}^1 - c_{i,j}^0) \end{cases} \tag{2.21}$$

Under this assumption the Network Orientation Problem with Flow-Dependent Costs (NOP-FDC) is the following :

Network Orientation Problem with Flow-Dependent Costs (NOP-FDC)

$$\min \sum_{r\in R}\sum_{(i,j)\in A} d^r \ell_{i,j} y^r_{i,j} + \sum_{(i,j)\in A} \ell_{i,j} w_{i,j} \tag{2.22}$$

$$x_{ij} + x_{ji} \leq 1, \qquad \forall \{i,j\} \in E \tag{2.23}$$

$$\sum_{(i,j)\in\delta^+(i)} y^r_{ij} - \sum_{(j,i)\in\delta^-(i)} y^r_{ji} = \begin{cases} 1, & \text{if } i = s_r \\ -1, & \text{if } i = t_r \\ 0, & \text{otherwise} \end{cases}, \quad \forall i \in V, \forall r = (s_r, t_r) \in R \tag{2.24}$$

$$y^r_{ij} \leq x_{ij}, \qquad \forall (i,j) \in \overline{A}, \forall r \in R \tag{2.25}$$

$$\sum_{(i,j)\in\delta^+(i)} w_{ij} - \sum_{(j,i)\in\delta^-(i)} w_{ji} = -D_{i,res} \qquad \forall i \in V, \tag{2.26}$$

$$w_{ij} \leq M_w x_{ij}, \qquad \forall (i,j) \in \overline{A} \tag{2.27}$$

$$\sum_{r\in R} d^r y^r_{i,j} + w_a \leq c^0 x_{i,j} + f_{i,j}, \qquad \forall (i,j) \in A \tag{2.28}$$

$$\ell_{i,j} \geq \ell^0_{i,j} + \frac{l'_{i,j} - l^0_{i,j}}{c^1_{i,j} - c^0_{i,j}} f_{i,j}, \qquad \forall a \in A \tag{2.29}$$

$$f_{i,j} \leq c^1_{i,j} - c^0_{i,j}, \qquad \forall (i,j) \in A \tag{2.30}$$

$$x_{ij}, \in \{0,1\}, \qquad \forall (i,j) \in A \tag{2.31}$$

$$w_{ij} \geq 0 \qquad \forall (i,j) \in \overline{A} \tag{2.32}$$

$$y^r_{ij} \geq 0, \qquad \forall (i,j) \in A, \forall r \in R \tag{2.33}$$

$$f_{i,j} \geq 0, \qquad \forall (i,j) \in A \tag{2.34}$$

$$l_{i,j} \geq 0, \qquad \forall (i,j) \in A \tag{2.35}$$

Inequalities (2.29) are used to describe the function (2.21) in the linear part and (2.30) fix the upper bound for $f_{i,j}$ and consequently also for $\ell_{i,j}$.
Note that the inequalities (2.28) do not ensure that y_{ij} or w_{ij} is greater than zero if and only if the corresponding x_{ij} is equal to one; this is why constrains (2.13) and (2.4) are necessary also in this model.
The source of nonlinearity is the objective function with the two products $\ell_{i,j} y^r_{i,j}$ and $\ell_{i,j} w_{i,j}$.
Notice that NOP-FDC is a generalization of the uncapacitated and capacitated versions:
i) if $c^0_a = c^1_a$, then it becomes equivalent to the capacitated version
ii) if $c^0_a = c^1_a = +\infty$, then it is equivalent to the uncapacitated version

2.4 Problem complexity

First of all, note that, in case the capacity constraint is imposed, even finding a feasible solution to the problem is easily seen to be difficult.

Proposition 1 *Testing if the NOP-HCC has a feasible solution is NP-complete.*

Proof. In case $c_e = 1$ for $e \in E$, the problem has a solution if and only if G contains $|R|$ edge-disjoint paths, one from s_r to t_r for $r \in R$. This is well known to be NP-complete. □

Of course, this implies that finding a feasible solution to NOP-FDC is strongly NP-complete. On the other hand, in case the capacity constraint is not imposed, finding a feasible solution is easy, though not entirely trivial.

Proposition 2 *In case the capacity constraint is not imposed, testing if NOP has a feasible solution can be done in linear time.*

Proof. Without capacity constraint, the problem has a solution if and only if there exists an orientation of the edges of G such that, for $r \in R$, there exists a directed path from s_r to t_r. This can be tested in linear time by the algorithm in [8]. □

(The algorithm in [8] can also be applied to a mixed graph, in which some of the edges are already oriented.) The above results, based on well-known facts, leave open the complexity of the problem without capacity constraints. This is easily settled by using an old (and not-so-well-known) result by [11].
In [12] Burkard et al. showed that the problem considered is NP-hard even in case the capacity constraint is not imposed and no empty vehicle flows are required. Here we provide an alternative proof of the same result

Proposition 3 *The NOP is strongly NP-hard*

Proof. In [11], it is shown that the following problem is NP-complete: given $G = (V, E)$, find an orientation D of G of diameter 2, i.e. such that each node can be reached from each other node by a path with at most two arcs. Given an instance of this problem, we define the instance of our problem (without the capacity constraint) on the same G in which $R := \{(i, j) : i, j \in V, i \neq j\}$, i.e. every ordered pair of nodes is an origin-destination pair, all demands $d_r = 1$ and all edge lengths $\ell_e = 1$. Note that, for each of the $|V|(|V| - 1)/2$ node pairs (i, j), considering the two origin-destination pairs (i, j) and (j, i), in every orientation D of G one of the paths will have weight 1, whereas the other one will have weight at least 2. This proves that the optimal value of our problem is at least $3/2|V|(|V| - 1)$. Moreover, the optimal value is exactly $3/2|V|(|V| - 1)$ if and only if there exists an orientation of diameter 2, which shows that by finding an optimal solution to our problem we can solve the problem of [11]. □

This implies that, besides NOP-HCC and NOP-FDC, also NOP-EV is strongly NP-hard.

2.5 Comparison of static vehicle flow assignment methods and micro-simulations

Even though the concept of PRT has been known for almost 50 years, it is surprising how little work has been done on the development of static assignment models for PRT as a transport offer. Planning studies have either been made by conventional assignment methods or by micro-simulators, mimicking the dynamics of individual vehicles [4]. Optimization models have been developed for the routing in automated guides vehicle systems (AGVs), used for freight movement [7].

However, static assignment models are essential to PRT planning for several reasons:

- They allow one to get a rough estimate of link-flows and determine the required capacities. This is useful to define PRT system requirements and to select the system.
- Static assignment models can be part of an algorithm that can optimize the topology of the PRT network.
- Static models can verify micro-simulations. This is very important since micro-simulators are very complex software tools and prone to errors and unexpected inaccuracies.
- Static models can be integrated with modal split or trip generation models.

Static models of PRT systems need to take into account at least two significant characteristics of PRT systems:

- The role of centralized control which has the knowledge of the position of all vehicles and the origin and destination of all users. The centralized control decides over the path of each vehicle and may adopt different strategies. These strategies must be taken into account when assigning the demand. Because of the centralized control system, system optimum traffic assignment methods can be applied. This is in contrast with road transport where path-choice is subject to the decision of the individual participant, resulting in sub-optimal solution known as the user equilibrium.
- The empty vehicle flows which are superimposed to the full vehicle flows with passengers. One needs to create a demand for the empty vehicles. In particular, if the travel demand is asymmetric, empty vehicle flows can become significant and cannot be neglected.

In the previous sections we have proposed several mathematical programming model to compute an optimal orientation of a PRT network. Every method proposed takes into consideration a different method for evaluating empty and full vehicle flows in a PRT network.

If we fix an orientation we can use the method in Section (2.3) to compute the static assignment of flows. The objective of this section is to verify whether the link flows computed by the static assignment methods and the dynamic link flows determined by the microsimulator give consistent results.
To better investigate this aspect a real case is analyzed. If static and averaged dynamic link flows matched we would be able to verify that the static models and the complex microsimulator do actually produce reliable results.

2.5.1 The PRT micro-simulator iTS

As mentioned above the main goal is to estimate the flows of full and empty vehicles on all network links. Even if in the static flow assignments we assume a constant trip demand throughout the period of observation, the micro-simulator produces random arrival times of users at stations with a unit distribution over the observation interval.

As micro-simulations software, we used the in-house developed PRT simulator called innovative Transport Simulator (iTS). This micro-simulator mimics the movement of each vehicle and individual passengers on an arbitrary PRT network. PRT networks can be edited with a graphical editor.

Regarding the PRT vehicle dynamics, the simulator is adopting an asynchronous vehicle follower control approach, where each vehicle adapts its speed to the speed of the preceding vehicle (or possibly a vehicle on a parallel track in merge and diverge situations). The vehicles keep always a safe distance to the vehicle in front. This safe distance depends on the speed and on the dynamic characteristics of the PRT system. For the present simulations, we assume a PRT system guarantees $1.5m/s^2$ acceleration and deceleration during normal operation and an emergency brake deceleration of $2.5m/s^2$. We further assume that the brick-wall stopping criteria must be satisfied (vehicle in front can stop instantly) and the time from detecting an emergency case to the actuation of the emergency brake is below $0.5s$. The maximum line speed is $12m/s$ ($43km/h$) and the headway is approximately $3s$ at maximum speed. These parameters are similar to the technical characteristics of Urban Light Transport, ULTra (2009), 2getthere (2009) and Vectus PRT (2008).

Passenger origin and destinations are determined by passing an Origin-to-Destination Matrix (ODM) to the simulator. Boarding times are simulated as unit distributions in the intervals 8s-12s for boarding and 5s-10s for alightning, assuming passengers with light luggage only.

Concerning the logistics we have implemented an innovative empty vehicle management that allows to cope with a particular problem of this PRT network: there are remote stations with a high demand (such as the station at the P&R). This is a challenge for conventional PRT

logistics because empty (or full) vehicles must be sent to the remote car park before the users arrive, otherwise prolonged waiting times are the consequence. If, on the contrary, the central control is sending more vehicles to a remote station than needed for users (who may or may not arrive at some time in the future), then there is the risk that vehicles which are already under way are no longer needed at the station at the time they arrive. The consequence would be that these empty vehicles would be needed elsewhere in the network. Moreover, if there is not enough vehicle buffer capacity at the remote station, the empty vehicles in excess must be sent back and added to the vehicle flow in other parts of the network. We have tried to limit these negative effects by some substantial enhancements to the control strategies:

- Special "buffer" stations which are placed at a strategic point where the empty vehicles enter a (remote) car-park area. These buffer stations have two roles: (1) to absorb and buffer empty vehicles that arrive at the car park and that are no longer needed; (2) to dispatch the empty vehicles in the buffer to local car-park stations that are most in need for empty vehicles. The idea behind is that if vehicles are located in a buffer-station closer to a potential destination station, then they can be assigned to high demand stations with a high probability of success (empty vehicles are still needed when they arrive).

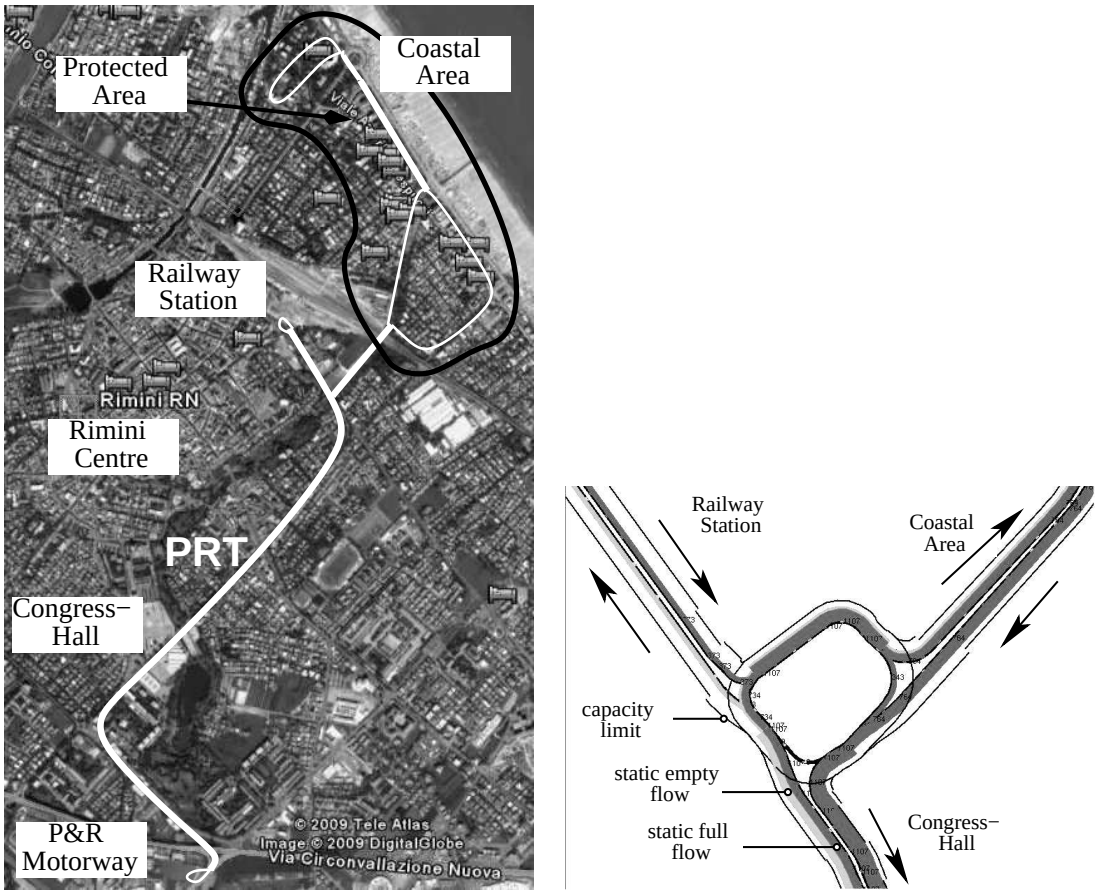
- Simple demand predictor: the predictor estimates the future demand for each station, assuming constant arrival rates. It is clear that arrival rates are not constant. For this reason the predictor updates its arrival rate estimation with the arrival time of every newly arriving passenger. In addition the aforementioned vehicle buffers can average out irregularities of user arrival times. Finally the estimated arrival rate is used to determine the rate at which the vehicle management must send vehicles (either empty of full) to a specific station. This strategy is particularly effective for long-range empty vehicle missions.

The NOP formulation introduced in Section 2.3.2 is based on the assumption the we are dealing with an assignment model named "all or nothing" (AON). From the basic AON the other mathematical programming formulations developed correspond to other assignment models, like system optimum assignments or flow constraints. Anyway, AON is the model that describes the state-of-the-art PRT control system behavior: the minimization of the total distance traveled by the vehicles, independent of the link flows.

2.5.2 The example network

The example network is taken from a feasibility study that we conducted for the province of Rimini: the objective was to establish a high-quality public transport service for tourists

and visitors between the highway exit and conference center (located south west of the city center) with the beach area, hotels and restaurants (located east of the center), see Fig. 2.1.



(a) (b) s

Figure 2.1: (a) Example network of Rimini, (b) Central network node which shows statically assigned empty and full vehicle flows together with capacity limits (that depend on local line speed).

As travel demand we have used the morning peak-hour demand during a major event at the congress hall. The zone-to-zone ODM in table 2.1 has been estimated based on data about the conference visitors and their origin, the number of overnight visits, the number of beds in the PRT covered area, number of parking spaces along the coast etc. The modal split for PRT is at 100% as there is the intention to close the coastal zone to non residential traffic. The zone-to-zone ODM has been equally distributed over the several PRT stations in case there were more than one station per zone.

	P&R	Station	Coastal Area	Congress hall	Center
P&R	0	0	398	0	20
Station	0	0	125	250	6
Coastal Area	20	13	0	750	20
Congress hall	0	25	75	0	38
Center	1	1	20	38	0

Table 2.1: One hour Zone-to-zone ODM of example network.

2.5.3 Results

The example network has been simulated by the micro-simulator with the peak-hour demand of Table 2.1. We denote by $\hat{f}_a$ the dynamic vehicle link flow on link a, averaged over one hour simulation time and by $\hat{w}_a$ the corresponding average empty vehicle flow. With the same example network and ODM we have determined the static vehicle link flows f_a and static empty vehicle link flows w_a using the method described in in Section 2.3.3. The static and dynamic vehicle flows are compared in Figure 2.2. In order to give a figure of merit we determined the weighted normalized standard deviation between static and dynamic flows. In Equation 2.36 we have defined e_w and e_f to validate the differences of empty and total vehicle flows, respectively:

$$e_w = \frac{\sqrt{\sum_{a\in A}(w_a - \hat{w}_a)^2}}{\sum_{a\in A} w_a} \, , e_f = \frac{\sqrt{\sum_{a\in A}(f_a - \hat{f}_a)^2}}{\sum_{a\in A} f_a} \tag{2.36}$$

For the present simulation we obtained $e_w = 0.19$ or 19% error between static and dynamic empty vehicle flows and $e_w = 0.04$ or 4% error between static and dynamic total vehicle link flows.

The larger error of the empty-vehicle flows is due to several effects:

- The static empty vehicle flows can be considered a lower bound for the average dynamic vehicle flows produced by the simulator. The reason is that the static assignment knows in advance the entire demand, while the micro-simulator does only know the demand for the users at the time they arrive at the station. This lack of information degrades the performance of the real-time empty vehicle management. In the present case we obtain from the static assignment a 26% empty vehicle share while the micro-simulator calculated a 48% empty vehicle share.

- There are transitions at the beginning and end of the simulation where more empty vehicles circulate.

- In general there may be more vehicles in circulation than necessary to satisfy flow

conditions in order to reduce waiting times — a quantity not considered in this paper.

- There are some vehicle "buffer" stations which are not used by the static assignment method, but needed in practice for the empty vehicle management.

2.6 LP-based approaches for solving NOP-EV

2.6.1 NOP surrogate formulation

Even without the capacity constraints (2.18), the direct solution of MILP (2.1)-(2.5) by a general-purpose MILP solver quickly becomes impractical as the size of G grows and the solver with a real instance usually runs out of memory.
For large scale problems, in order do have a compact formulation to deal with we restate the problem in a "surrogated" formulation with fewer variables involved.
we substitute the set of variable y^s_{ij} with a new set of continuous variable y^s_{ij} that represents the total flow of full vehicles originating from station s.

The model reads:

Network Orientation Problem - surrogate formulation (NOP-EV-surr)

$$\min \sum_{s\in S}\sum_{(i,j)\in\overline{A}} \ell_{ij} y^s_{ij} + \sum_{(i,j)\in\overline{A}} \ell_{ij} w_{ij} \tag{2.37}$$

(2.2)

$$\sum_{(i,j)\in\delta^+(i)} y^s_{ij} - \sum_{(i,j)\in\delta^-(i)} y^s_{ij} = \begin{cases} \sum_{r\in R:s_r\equiv s} d^r, & \text{if } i = s \\ -d^{(s,i)}, & \text{if } (s,i)\in R \\ 0, & \text{otherwise} \end{cases}, \quad \forall i \in V, \forall s \in S, \tag{2.38}$$

$$y^s_{i,j} \le \sum_{r\in R:s_r\equiv s} d^r x_{i,j}, \qquad \forall (i,j)\in\overline{A}, \forall s\in S \tag{2.39}$$

$$\sum_{(i,j)\in\delta^+(i)} w_{ij} - \sum_{(j,i)\in\delta^-(i)} w_{ji} = -D_{i,res} \qquad \forall i \in V \tag{2.40}$$

$$w_{i,j} \le M_w x_{ij}, \qquad \forall (i,j)\in\overline{A} \tag{2.41}$$

$$w_{i,j} \ge 0 \qquad \forall (i,j)\in\overline{A} \tag{2.42}$$

$$x_{i,j} \in \{0,1\}, \qquad \forall (i,j)\in A \tag{2.43}$$

$$y^s_{i,ja} \ge 0 \qquad \forall (i,j)\in A, \forall s\in S \tag{2.44}$$

the new model contains $O(nm)$ variables and $O(n^3 + n^2 m)$ constraints, instead of $O(ms^2)$ variables and $O(n^2 + nm)$ constraints.
Note that this formulation is obtained from the NOP-EV one by first multiplying all the $y^r_{i.j}$ variables by d^r , by replacing them by the new "surrogate" variables $y^s_{i,j} = \sum_{r=(s,i)\in R} y^r_{i,j}$,

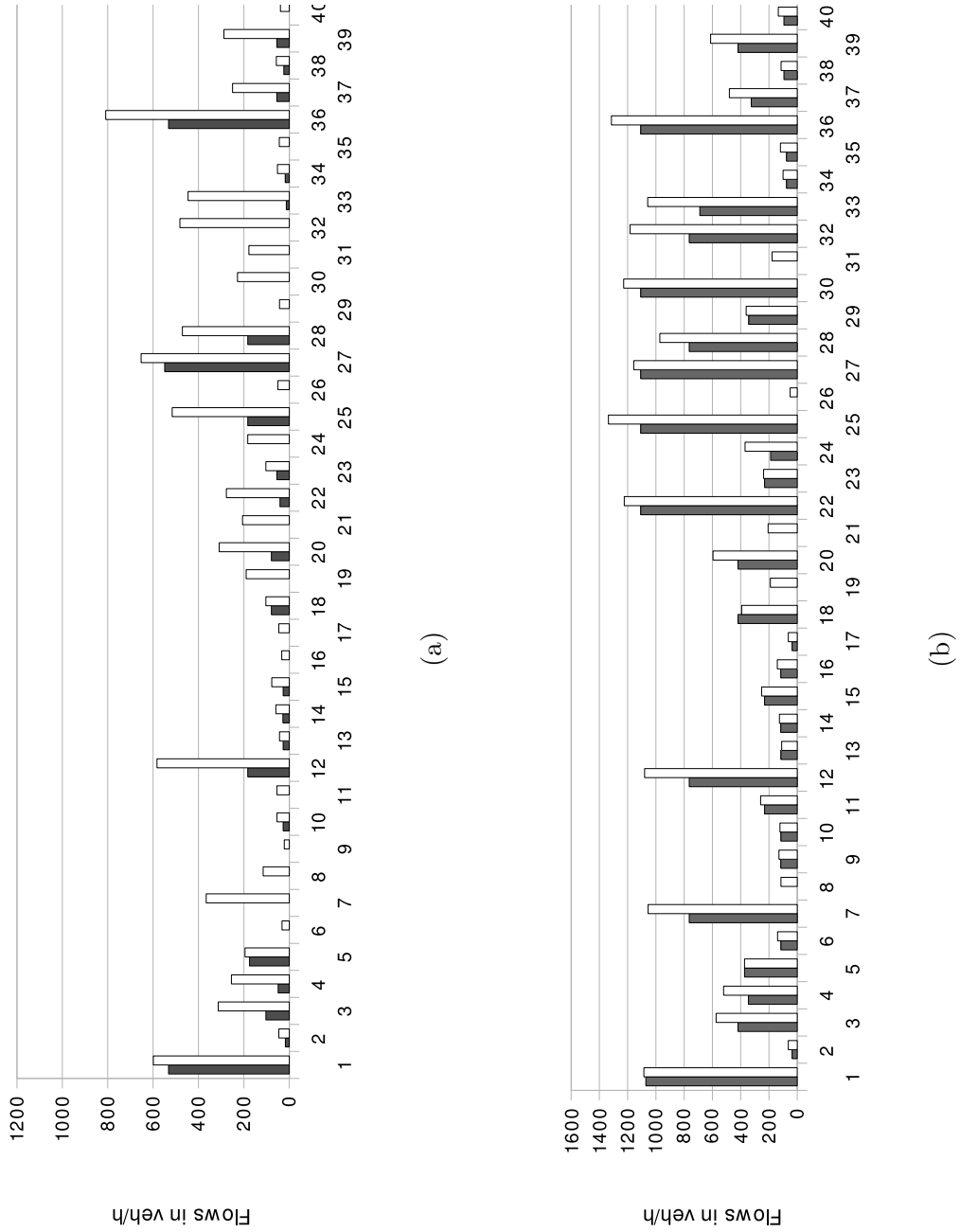


Figure 2.2: Static and averaged dynamic vehicle flows for all network links. Empty vehicles only in (a) and total vehicle flows in (b).

and by replacing constraints 2.11 by their surrogate version 2.39. This latter replacement makes the new model weaker than the NOP-EV one.

2.6.2 Benders Decomposition

We also solve the Linear Programming (LP) relaxation of NOP-EV by a Benders decomposition approach, with a Master Problem with the x and w_a variables along with auxiliary variables β^r expressing the weighted length of the path from s_r to t_r in D, the objective function being $\sum_{r \in R} \beta^r$.
The master problem is the following:

Benders Decomposition - Master Problem (MP)

$$\min \sum_{r \in R} \beta^r + \sum_{a \in \overline{A}} \ell_a w_a \tag{2.45}$$

$$(2.2), (2.12) - (2.15)$$

$$\sum_{a \in \overline{A}} u_a x_a + \beta^r \geq v_{t_r} - v_{s_r}, \quad \forall r \in R, \forall (u,v) \in E_P^r \tag{2.46}$$

$$\sum_{a \in \overline{A}} u_a x_a \geq v_{t_r} - v_{s_r}, \quad \forall r \in R, \forall (u,vs) \in E_R^r \tag{2.47}$$

$$x_a \in \{0,1\}, \quad \forall a \in A, \forall r \in R \tag{2.48}$$

$$\beta^r \geq 0, \quad \forall r \in R \tag{2.49}$$

Constraints (2.46) represent the optimality cuts, they provide lower bounds on the values of the weighted shortest path between each pair of stations.
Constraints (2.47) represent the feasibility cuts, they bound the extreme rays in the dual subproblem in order to have it bounded, to ensure primal feasibility.

Solving an LP relaxation via Benders decomposition requires a two level approach, alternating the solution of the Master Problem and the Subproblem until we have proved that the Master Problem solution is optimal.

The Primal Sub-Problem $PSP(\overline{x})$ associated with the Master Problem solution $\overline{x}$ is the following:

Benders Decomposition - Primal Sub-Problem ($PSP(\overline{x})$)

$$\min \sum_{r\in R}\sum_{a\in\overline{A}} d^r \ell_a y_a^r, \tag{2.50}$$

$$\sum_{a\in\delta^+(i)} y_a^r - \sum_{a\in\delta^-(i)} y_a^r = \begin{cases} 1, & \text{if } i = s_r \\ -1, & \text{if } i = t_r \\ 0, & \text{otherwise} \end{cases}, \quad \forall i \in V, \forall r = (s_r, t_r) \in R, \tag{2.51}$$

$$y_{i,j}^r \leq \overline{x_{i,j}}, \qquad \forall (i,j) \in \overline{A}, \forall r \in R, \tag{2.52}$$

$$y_a^r \in \{0,1\}, \qquad \forall a \in A, \forall r \in R. \tag{2.53}$$

The primal subproblem (PSP) decomposes into $|R|$ independent problems, for the sake of simplicity from now on we call each of them the subproblem $PSP_r(\overline{x})$, r being the corresponding route.

The Dual Sub-Problem associated with $PSP_r(\overline{x})$ is the following:

Benders Decomposition - Dual Sub-Pproblem ($DSP_r(\overline{x})$)

$$\max \sum_{a\in\overline{A}} \overline{x_a} u_a^r - v_{s_r} + v_{t_r} \tag{2.54}$$

$$-u_a^r - v_i^r + v_j^r \leq d^r l_a, \qquad \forall (i,j) \in \overline{A} \tag{2.55}$$

$$u_a^r \lesseqgtr 0, \qquad \forall a \in A \tag{2.56}$$

$$v_i^r \geq 0, \qquad \forall i \in V \tag{2.57}$$

E_P^r introduced in (2.46) is defined as the set of all the vertices (extreme points) of the polyhedron associated with the feasible region of DSP_r. E_R^r introduced in (2.47) is defined as the set of all the unbounded directions (extreme rays) of the polyhedron associated with the feasible reason of c. We notice that the feasible region of a $PSP_r(\overline{x})$ does not depend on $\overline{x}$.

Each primal subproblem corresponds to a min cost flow problem with capacity values $\overline{x}$; it is feasible if it admits an (s_r, t_r)-cut of value at least 1 for each route r. Therefore separating the inequalities (2.47) corresponds to solving $O(n^2)$ Min-cut problems.

Every possible $\overline{x}$ provides a cut, between two optimality cuts it is possible to establish a partial ordering and an optimality cuts can be strengthened by finding a cut that Pareto-dominates it with a procedure for the first time defined by Magnanti and Wong [10].

Let x^0 be a point in the interior of the set of all feasible points and let $z(\overline{x})$ be the optimal dual (or primal) solution associated to $\overline{x}$

The strengthened optimality cut can be found by solving an additional dual subproblem:

Benders Decomposition - Pareto Optimal Dual Subproblem ($DS_{Popt}(r)$)

$$\max \textstyle\sum_{a\in\overline{A}} x_a^0 u_a^r - v_{s_r} + v_{t_r} \tag{2.58}$$

$$-u_a^r - v_i^r + v_j^r \leq d^r l_a, \qquad \forall (i,j) \in \overline{A} \tag{2.59}$$

$$\textstyle\sum_{a\in\overline{A}} \overline{x}_a u_a^r - v_{s_r} + v_{t_r} = z(\overline{x}) \tag{2.60}$$

$$u_a^r \lesseqgtr 0, \qquad \forall a \in A \tag{2.61}$$

$$v_i^r \geq 0, \qquad \forall i \in V \tag{2.62}$$

The procedure to solve LP relaxation by Benders decomposition can be embedded in a Branch and Bound scheme in order to find an exact solution to the problem.

The procedure used for solving a node in the B&B tree is the following:

Benders Decomposition for a node in the BB tree

begin

set $BEST\_LB := 0$ and $STOP := 0$

repeat

solve MP, read β^r and the current solution $\overline{x}$

if no β^r is increased **then** $STOP := 1$

else

for $r = 1$ **to** $|R|$ **do**

solve $DSP(r)$

if $DSP(r)$ is feasible **then** store the optimality cut (2.46)

else add a feasibility cut (2.47)

if no feasibility cut has been found **then** add all the optimality cuts found and set $BEST\_LB := of$

until $STOP = 1$

return $BEST\_LB$

end

In the B&B we adopt a depth-first-search procedure. The search tree is partitioned by fixing at each node in the B&B tree one orientation in one edge (i, j) such that x_{ij} and x_{ji} are fractional.
Two degrees of freedom are involved in the choice of the edge to fix:
i) what edge to select
ii) what orientation to give to this edge.
Three basic policies have been tested in the selection of the edge:
i) the one with the value closest to an integer, namely the edge (i, j) with the highest fractional value $\max\{x_{i,j}, x_{j,i}\}$
ii) the one with the highest value π_e, where π_e is the dual variable associated with orientation constraints (2.2).
The reason for the first principle is due to the idea to change the integer solution as little as possible from the continuous relaxation. The second principle follows the idea that a constraint with a higher dual value is "strongly" tight. In our case it means that both directions are really important for the flows involved, hopefully this leads to a lower bound improvement on both branches.
A third and forth hybrid policies has been tested in order to mix the two contributions:
iii) picking the one with highest value $\pi_e(\max\{x_{i,j}, x_{j,i}s\})$; Once the edge is selected it is oriented in the direction of the higher fractional value.
iv) picking the one with the value closest to an integer, in case of two or more variables with the same fractional value selecting the one with the highest associated dual value .

The B&B proposed can be stopped once one feasible solution is found. Moreover the optimality cuts can be used only to provide an estimation of the bound in the tree, hence only some rounds of cuts can be added in each node in order to explore it more quickly.

Finally, a heuristic solution can be obtained by directly solving the Master Problem in the root node with binary x_{ij} variables binary, keeping the corresponding orientation and evaluating its value via a shortest path computation.

2.7 Non LP-based approaches for solving NOP-EV

Solving to optimality instances of realistic size is an hard task. The conjecture derived from the instances analysis is that the hardness is due to the poor information on the optimal integer solution that we can get from the LP relaxation and from the high amount of local optima. In particular, in the large majority of the optimal LP solutions the vast majority of the x variables takes values equal to $\frac{1}{2}$.

In this section we provide other Branch&Bound approaches with lower bounds computations that are combinatorial, and not LP-based, even if they are inspired by the LP behavior during the solution of the B&B tree.

The branching scheme is still based on orienting one edge of the graph at each node of the tree; the B&B schemes differ in the criteria used to select the edge to orient and to compute the LB.

In the LP-based procedure the connectivity is guaranteed by the flow conservation constraints. In the combinatorial approach we must explicitly take into consideration the basic observation that while we are orienting edges in the branching we need to ensure the connectivity of the resulting mixed graph.
Every time that a new edge is oriented we call the "chain orientation" procedure described below; this is done in order to:
i) avoid creating B&B nodes that are by definition infeasible
ii) obtain a partial orientation that is as strong as possible, i.e. with as many oriented edges as possible

The procedure introduced is the following:

Chain Orientation Procedure
input
A mixed graph $G = (V, E \cup A)$ with E set of edges and A set of arcs
begin
set $STOP = 0$ and $NO\_ORIENT = 0$
repeat

set $STOP = 1$

for $r = 1$ **to** $|R|$ **do**

let $r = (i, j)$, find a path p from i to j in G

if $\nexists p$ **then** set $STOP = 1$ and $NO\_ORIENT = 1$

else

let G' be equal to G with the edges of p oriented from j to i

find a path p' from i to j in G'

if $\nexists p'$ **then**

explore G' starting from i and mark all the visited nodes

if j is not reached **then**

find an edge $(k, l) \in G$ with only k marked

orient in G the edge (k, l) from l to k, update E and A

set $STOP = 0$

until $STOP = 1$
output
if $NO\_ORIENT = 1$ **then** the orientation is infeasible
else return the mixed graph $G = (V, E \cup A)$

The above procedure ensures that we deal with a partial orientation that admits a feasible solution.

In the following two sections two lower bound computation are described together with the branching rules. For a better understanding we use the same variables notation used in the NOP-EV mathematical formulation, i.e. $d^r y^r_{i,j}$ represents the quantity of full vehicle flow of route r along arc (i, j) and $w_{i,j}$ the quantity of empty vehicle flow of route along arc i, j.

2.7.1 Lower bound computation based on shortest path

This LB computation corresponds to relax the original NOP eliminating the orientation constraints.
At each node the LB can be computed as the sum of the full- and empty-vehicle flows in the mixed graph $G = (V, A \cup E)$: Let LB_{fv} (resp. LB_{ev}) be the LB contribution due to the full (resp. empty) vehicle flow:

$$LB^{sp}_{fv} = \sum_{r\equiv(i,j)\in R} d^r(\text{shortest path from } i \text{ to } j \text{ in } G) \quad (2.63)$$

$$LB^{sp}_{ev} = (\text{value of the min-cost-flow in } G \text{ with residual demands (2.7) }) \quad (2.64)$$

The Shortest Path computation can be done with the Dijkstra algorithm. At each node of the B&B tree we store the information about the edges involved in each shortest path, hence after orienting new edges we recompute the shortest paths only for the routes where some of these edges were used in the opposite direction.
The same applies for LB_{ev}: we recompute the min-cost-flow if and only if we reorient an edge in a direction opposite to the one used in the predecessor node.

2.7.2 Fake lower bound computation based on "$\frac{1}{2}$-graph"

This heuristic exploits the idea that the optimal solution of the LP relaxation usually contains the majority of the unfixed $x_{i,j}$ variables equal to $\frac{1}{2}$.
We comute an estimation of the LP value,which is not a valid lower bound in geneneral by assuming that the values of all unfixed $x_{i,j}$ variables are equal to $\frac{1}{2}$.

Let $G = (V, E \cup A)$ be the mixed graph in the current node. Let $G' = (V, A')$ be the capacitated directed graph defined as follows: for each arc $(i, j) \in A$ we add an arc from i to j in A' with capacity $c_{i,j} = 1$ and for each edge $\{i, j\} \in E$ we add an arc from i to j in A' and an arc from j to i in A' with capacity $c_{i,j} = c_{j,i} = \frac{1}{2}$. Finally, let Let G" $= (V, A$") be the capacitated directed graph defined as G' with the capacity redefined as follows: for each edge $(i, j) \in E$ the corresponding capacity is $c_{i,j} = c_{j,i} = \frac{M_w}{2}$ and for each arc $\{i, j\} \in A$ the corresponding capacity is $c_{i,j} = M_w$.

The LB contribution are the following :

$$LB^{\frac{1}{2}}_{fv} = \sum_{r\equiv(i,j)\in R} d^r(\text{ value of the min-cost-flow of one unit from } i \text{ to } j \text{ in } G') \quad (2.65)$$

$$LB^{\frac{1}{2}}_{ev} = (\text{ value of the min-cost-flow in } G' \text{ with residual demands (2.7) }) \quad (2.66)$$

Note that the computation of $LB_{ev}^{\frac{1}{2}}$ consists in a capacitated min cost flow, with arc capacity $\frac{M_w}{2}$ (almost the "big M"), this means that the computation of $LB_{ev}^{\frac{1}{2}}$ is based on an graph that in practice in uncapacitated.

As already mentioned, this LB computation corresponds to fixing all the free $x_{i,j} = \frac{1}{2}$. The chain orientation procedure ensures that we always deal with a feasible orientation, unfortunately the LB provided is not always valid in general, hence the corresponding Branch&Bound scheme is only an heuristic method for NOP-EV.

2.7.3 Edge selection

In both approaches described in Sections 2.7.1 and 2.7.2 the edge selection follows the same approach.

Let $Y_{i,j} = \sum_{r \in R} d^r y^r_{i,j} + w_{i,j}$ be the total flow of empty and full vehicles in the edge (i, j) in the direction from i to j.
The criteria used to orient one edge are the following:
i) the highest value $\max\{Y_{i,j}, Y_{j,i}\}$
ii) the highest value $\max\{\frac{Y_{j,i}}{1+Y_{i,j}}, \frac{Y_{i,j}}{1+Y_{j,i}}\}$
iii) the highest value $Y_{i,j} + Y_{j,i}$
th Criterion i) gives the priority to the arc with the highest amount of flow, criterion ii) has the same idea of i) but tries to avoid edges with high flow in both directions and on the other hand criterion iii) does the opposite, trying to branch on the most used edge.

Once the edge is selected we orient it according with the one with the higher quantity of flow between the two directions.

After the edge selection the "chain orientation" procedure is executed before processing the child nodes.

2.8 LP-based approaches for NOP-FDC

The nonlinearity in NOP-FDC is restricted to the objective function which is *bilinear* in the $\ell_{i,j}$, $y^r_{i,j}$, $w_{i,j}$ variables, i.e. either by fixing $\ell_{i,j}$ or by fixing $y^r_{i,j}$ and $w_{i,j}$ it becomes linear. This special case has been considered in [5] by McCormick, who provided a valid MILP relaxation for a general bilinear problem. By solving the MILP relaxation we obtain valid LB to the problem.
Let $z^r_{i,j}$ and $u_{i,j}$ be two sets of continuous variables representing respectively the linearization

of $\ell_{i,j} y^r_{i,j}$ and of $\ell_{i,j} w_{i,j}$.
The variables $\ell_{i,j}$, $y^r_{i,j}$ and $w_{i,j}$ have the same definition provided in Section 2.3.5, hence they are bounded as follows:

$$LO^{\ell}_{i,j} \leq \ell_{i,j} \leq UP^{\ell}_{i,j} \tag{2.67}$$
$$LO^{y}_{i,j,r} \leq y^r_{i,j} \leq UP^{y}_{i,j,r} \tag{2.68}$$
$$LO^{w}_{i,j} \leq w_{i,j} \leq UP^{w}_{i,j} \tag{2.69}$$
$$\tag{2.70}$$

where

$$LO^{\ell}_{i,j} = \ell^0_{i,j} \tag{2.71}$$
$$UP^{\ell}_{i,j} = \ell^1_{i,j} \tag{2.72}$$
$$LO^{y}_{i,j,r} = 0 \tag{2.73}$$
$$UP^{y}_{i,j} = 1 \tag{2.74}$$
$$LO^{w}_{i,j} = 0 \tag{2.75}$$
$$UP^{y}_{i,j} = M_w \tag{2.76}$$

Using the procedure defined in [5] we can rewrite the connection between the new variables z , u and $\ell_{i,j}$, $y^r_{i,j}$, $w_{i,j}$ as follows:

$$z^r_{i,j} \geq UP^{y}_{i,j,r} \ell_{i,j} + UP^{\ell}_{i,j} y^r_{i,j} - UP^{\ell}_{i,j} UP^{y}_{i,j,r} \tag{2.77}$$
$$z^r_{i,j} \geq LO^{\ell}_{i,j} y^r_{i,j} + LO^{y}_{i,j,r} \ell_{i,j} - LO^{y}_{i,j,r} LO^{\ell}_{i,j} \tag{2.78}$$
$$\tag{2.79}$$
$$u_{i,j} \geq UP^{w}_{i,j} \ell_{i,j} + UP^{\ell}_{i,j} w_{i,j} - UP^{\ell}_{i,j} UP^{w}_{i,j} \tag{2.80}$$
$$u_{i,j} \geq LO^{\ell}_{i,j} w_{i,j} + LO^{w}_{i,j} \ell_{i,j} - LO^{w}_{i,j} LO^{\ell}_{i,j} \tag{2.81}$$
$$\tag{2.82}$$

By substituting UP and LO and adding it into the model we obtain the following linearization of NOP-FDC:

$$\min \sum_{r\in R}\sum_{(i,j)\in A} d^r z^r_{i,j} + \sum_{(i,j)\in A} u_{i,j} \tag{2.83}$$

$$z^r_{i,j} \geq \ell^0_{i,j} y^r_{i,j}, \qquad \forall (i,j)\in A, \forall r \in R \tag{2.84}$$

$$z^r_{i,j} \geq \ell'_{i,j} y^r_{i,j} + \ell_{i,j} - \ell'_{i,j}, \qquad \forall (i,j)\in A, \forall r \in R \tag{2.85}$$

$$u_{i,j} \geq \ell^0_{i,j} w_{i,j}, \qquad \forall (i,j)\in A \tag{2.86}$$

$$u_{i,j} \geq \ell'_{i,j} w_{i,j} + D^{res}_{+}\ell_{i,j} - D^{res}_{+}\ell'_{i,j}, \qquad \forall (i,j)\in A \tag{2.87}$$

$$x_{ij} + x_{ji} \leq 1, \qquad \forall \{i,j\}\in E \tag{2.88}$$

$$\sum_{(i,j)\in\delta^+(i)} y^r_{ij} - \sum_{(j,i)\in\delta^-(i)} y^r_{ji} = \begin{cases} 1, & \text{if } i = s_r \\ -1, & \text{if } i = t_r \\ 0, & \text{otherwise} \end{cases}, \quad \forall i \in V, \forall r = (s_r, t_r) \in R \tag{2.89}$$

$$y^r_{ij} \leq x_{ij}, \qquad \forall (i,j)\in \overline{A}, \forall r \in R \tag{2.90}$$

$$\sum_{(i,j)\in\delta^+(i)} w_{ij} - \sum_{(j,i)\in\delta^-(i)} w_{ji} = -D_{i,res} \qquad \forall i \in V, \tag{2.91}$$

$$w_{ij} \leq M_w x_{ij}, \qquad \forall (i,j)\in \overline{A} \tag{2.92}$$

$$\sum_{r\in R} d^r y^r_{i,j} + w_a \leq c^0 x_{i,j} + f_{i,j}, \qquad \forall (i,j)\in A \tag{2.93}$$

$$\ell_{i,j} \geq \ell^0_{i,j} + \frac{l'_{i,j} - l^0_{i,j}}{c^1_{i,j} - c^0_{i,j}} f_{i,j}, \qquad \forall a \in A \tag{2.94}$$

$$f_{i,j} \leq c^1_{i,j} - c^0_{i,j}, \qquad \forall (i,j)\in A \tag{2.95}$$

$$x_{ij}, \in \{0,1\}, \qquad \forall (i,j)\in A \tag{2.96}$$

$$w_{ij} \geq 0 \qquad \forall (i,j)\in \overline{A} \tag{2.97}$$

$$y^r_{ij} \geq 0, \qquad \forall (i,j)\in A, \forall r \in R \tag{2.98}$$

$$f_{i,j} \geq 0, \qquad \forall (i,j)\in A \tag{2.99}$$

$$l_{i,j} \geq 0, \qquad \forall (i,j)\in A \tag{2.100}$$

An interesting remark is that if for certain $(i,j) \in A$ the corresponding $z_{i,j}$ or $u_{i,j}$ variables take the value of their lower or upper bound the linearization is no longer an approximation but provides the exact value of the corresponding products $\ell_{i,j} y^r_{i,j}$ and $\ell_{i,j} w_{i,j}$.

As usual the MILP relaxation can be solved with a B&B technique, branching on the integer variable. In addition we can solve the MINLP by branching on the continuous variables y , ℓ and w. If we branch on a continuous variable one of its corresponding bounds (UP or LO) changes, hence it is possible to rewrite a strengthened linearization constraints. Note that this is a binary branching in the sense that at each node we generate two new nodes, on the other hands we can branch several times on the same continuous variable, reducing its upper bound or increasing its lower bound. The following example illustrates this process:

Example Let us consider the constraints related to the arc $(1,2)$ with $LO^{\ell}_{1,2} = 2$, $UP^{\ell}_{1,2} = 6$ and $M_w = 100$:

$$z^r_{1,2} \geq 2y^r_{1,2}, \qquad \forall r \in R \tag{2.101}$$
$$z^r_{1,2} \geq 6y^r_{1,2} + \ell_{1,2} - 6, \qquad \forall r \in R \tag{2.102}$$
$$u_{1,2} \geq 2w_{1,2} \tag{2.103}$$
$$u_{1,2} \geq 6w_{1,2} + 100\ell_{1,2} - 600 \tag{2.104}$$
$$\tag{2.105}$$

if we branch on $\ell_{1,2}$ with the disjunction $\{\ell_{1,2} \leq 3\} \vee \{\ell_{1,2} \geq 3\}$ we obtain the following new set of inequalities for the first part of the disjunction (i.e. the one with $UP^{\ell}_{1,2} = 3$) :

$$z^r_{1,2} \geq 2y^r_{1,2}, \qquad \forall r \in R \tag{2.106}$$
$$z^r_{1,2} \geq 3y^r_{1,2} + \ell_{1,2} - 3, \qquad \forall r \in R \tag{2.107}$$
$$u_{1,2} \geq 2w_{1,2} \tag{2.108}$$
$$u_{1,2} \geq 3w_{1,2} + 100\ell_{1,2} - 300 \tag{2.109}$$
$$\tag{2.110}$$

and the following for the second part (i.e. the one with $LO^{\ell}_{1,2} = 3$) :

$$z^r_{1,2} \geq 3y^r_{1,2}, \qquad \forall r \in R \tag{2.111}$$
$$z^r_{1,2} \geq 6y^r_{1,2} + l_{1,2} - 6, \qquad \forall r \in R \tag{2.112}$$
$$u_{1,2} \geq 3w_{1,2} \tag{2.113}$$
$$u_{1,2} \geq 6w_{1,2} + 100l_{1,2} - 600 \tag{2.114}$$
$$\tag{2.115}$$

□

As showed in the example by fixing one bound a set of constraints is changed, and remains valid also for the child nodes.
Branching on continuous variables does not guarantee finiteness of the B&B process unless we introduce a tolerance on the gap between LO and UP in order to stop splitting a variable range of values. (It would also be possible to ensure finiteness in a methodologically more elegant way, which however turns out to be less efficient in practice.)

Branching on the continuous variables bound has also an intuitive interpretation: with regard to the ℓ variables, decreasing the UP corresponds to limit the total overflow of an

arc; conversely increasing the LO means requiring that a minimum amount of overflow must be assigned to an arc. Similarly, by fixing the upper or lower bound for a w or y variable we decrease the maximum quantity allowed of empty vehicle flow or increase the minimum quantity of flows in an arc.

In the next section we present several procedures based on the above linearization.
Note that even if the orientation is assigned (i.e. all the x variables are fixed) the problem and B&B framework reduces to a non trivial NLP. Also in this case the branching scheme described can be used.
Generally speaking the procedures tested can be divided into two sets:

- "two phase" approaches:
  we first orient the graph by fixing the x variables and then we assign the vehicle flows. This procedure is intrinsically heuristic and we can use all the approaches described in the previous sections in order to find the orientation.
- "integrated" approaches:
  we simultaneously solve the orientation and the flow assignment.

2.8.1 Two phase approaches

Splitting the optimization into two phases is not only a practical choice due to the necessity of limiting the computational effort. In several cases, once the orientation is fixed, forecasting the exact flow behavior is an interesting and useful task on itself.
As phase one we use one of the procedure used for solving the NOP-EV or the NOP-HCC, if the optimal value is not available we use the best solution found.

Once the orientation is completed we start the B&B approach for solving the flow assignment. At each node of the B&B we solve the LP relaxation in order to obtain a valid LB for the node. Note that the subproblem associated with the nodes has no integer variables because all the integer variables have been fixed.
Once the flows are assigned we can find a valid upper bound (UB) by simply computing the corresponding overflows associated with the current y and w and consequently updating the real arc costs ℓ.
If the current UB is bigger than the best known UB we can prune the node.
If all the ℓ are at the upper or lower bound the MILP relax value correspond to the optimal MINLP value hence we don't need to branch more because the current solution will remain optimal.

Among the three sets of continuous variables we branch on the ℓ variables; this choice is due to the fact that this affects the linearization of both the y and the w variables.

Let $\overline{\ell}$ be the current values of the ℓ variables. In all the B&B tested we fix one variable bound at each node. The criteria used to select the arc are the following:
i) the one with the lowest value $\dfrac{\overline{\ell}_{i,j} - \frac{UP^{\ell}_{i,j}+LO^{\ell}_{i,j}}{2}}{UP^{\ell}_{i,j} - LO^{\ell}_{i,j}}$
ii) the one with the highest value $Y_{i,j}(1 - \dfrac{\overline{\ell}_{i,j} - \frac{UP^{\ell}_{i,j}+LO^{\ell}_{i,j}}{2}}{UP^{\ell}_{i,j} - LO^{\ell}_{i,j}})$

Approach i) represents the normalized distance of $\ell_{i,j}$ from the middle value in the range that it can assume; this value goes from 0 (exactly in the middle) to $\frac{1}{2}$ (value corresponding to one bound).
Approach ii) is an attempt to weigh the the value i) with the total flow in the arc, in order to give the priority to the most "important" edges.

Additionally one heuristic option can be activated to speed up the B&B convergence to a heuristic solution, at the price of losing guaranteed optimality. This "early fixing procedure" aims at diminishing the number of free variables in order to focus only on the important one. More specifically, every time that a continuous variable takes its current lower or upper bound we fix this value, decreasing the solution space. Notice that by current value we mean the bound updated with the branching done so far.
The conjecture under this "early fixing procedure" is that if at some point an arc flow is at its upper bounds it will generally remain at the upper bound in the child nodes of the B&B tree. The more we branch the more this procedure is called, accelerating the process when we are close to the end.

2.8.2 Integrated aproach

This method integrates the branching rules explained in the second phase of the "two phase approach" with the ones described in section 2.6.
At each node we compute the continuous relaxation of the MILP relaxation of the original problem. The two branching approaches there are applied simultaneously by fixing one x variable and one ℓ variable bound at each branchig here.
also in this case, even if we operate in a B&B framework we do not aim at solving the instances to optimality. We restrict ourselves to a diving approach, using the information provided by the LP relaxation to guide the fixing order of the variables.

The branching scheme used for the x and ℓ variables are the same described in Section 2.6.2 and Section 2.8 respectively. Instead of a specific branching rule for the "one phase" approach we use a "priority" criterion to manage the two fixing previously introduced: after selecting the edge to fix we fix the bound for one $\ell_{i,j}$ variable only if the corresponding $x_{i,j}$ is equal to 1 in the last LP value. The idea of the "priority" criterion is that it worth to branch on a continuous variable only if it corresponds to an arc that will be likely used in the final orientation.

2.9 Description of the instances

We tested our algorithm on the following artificial and real-world instances.

- Grid instances.
  Artificial instances based on a grid graph with n_r rows and n_c columns. The number of nodes is equal to $n = n_r n_c$ and the number of edges is $m = 2n_r n_c - n_c - n_r$. Each edge has unitary length.
  Two ODMs are tested: i) one asymmetric OD-matrix, with demand d_r for each pair of nodes uniformly distributed in $[0, 99]$ and ii) a symmetric OD-matrix with demands d'_r defined as follows: for every $d_{i,j}$ and $d_{j,i}$ (demands in the asymmetric case) let $d'_{i,j} = d'_{j,i} = \frac{d_{i,j}+d_{j,i}}{2}$.

- Bologna
  Instance based on the city of Bologna public transport network. The graph corresponds the whole area covered by buses and trams. The ODM corresponds to the demand measured in the afternoon peak hour.

- Masdar
  Instance taken from the real world application cited in Section 2.1. The implementation of the system has been splitted into different steps. At each step several configurations and several OD-matrices are considered. Each step include the previous connections plus new connections and new stations. One of the particular aspects of the PRT system used in this application is that the guide-ways orientation can be easily changed from phase to phase. This flexibility is due to the way the guideways are implemented: they just need a flat surface to run, with corridors defined by barriers fixed on the ground. The vehicles use an electric engine and the control system is not installed in the corridors.
  Under the mentioned assumptions the network lay out can drastically change from step

to step according to the necessity.

The instances tested are the following:

- phase 3 instances
  mm_city_g05 consists of 533 nodes and 602 edges. for this phase the ODMs are known and two of them are tested, corresponding to the morning and afternoon peak hours.
  masdar_g02 consists of 467 nodes and 659 edges. It corresponds to an alternative lay out, with stations placed in different positions, the ODM is unknown and two ODMs are tested: i) an "all-one" ODM and ii) a random ODM with demands uniformly defined in $[0, 99]$.

2.10 Computational results

We have implemented all the the algorithms described in C language and we have executed the tests on a PC XXX, Y GB Ram, ZZ GHz. as MILP and LP solver we have used CPLEX 10.1 with its standard settings.
When it is not specified the time limit is always 36000 seconds.

Objective of this analysis is to investigate the performance of the algorithms described in the previous sections. As starting point we consider the grid instances, whom goal is to define a benchmark set composed of instances having the same structure.
Among the grid instances the performances of the algorithms change if we apply the asymmetric demand or not, this gives an interesting hint on which procedure to use if we know in advance how much the instance is asymmetric.

2.10.1 NOP-EV solution

In this section we discuss the algorithm used to solve NOP 2.3.3.

Grid Instances

In Table 2.2 we consider the following MILP models:

- NOP-EV
  this is the basic formulation with empty vehicle flows described in Sections 2.3.3 and 2.3.2.

objval ILP represents the best objective function found within the time limit, *objval LP* is the LP relaxation value.

- NOP-EV-surr
  this corresponds to the formulation introduced in Section 2.6.1. *objval ILP surr* is the best objective function found within the time limit and *objval LP* is the LP relaxation value.

In both cases the CPU time is reported.
The standard LP relaxation is available up to grid9x10, for bigger instances the solver runs out of memory. The LP relaxation for the surrogated version does not have this problems for instances of this size.
For instances bigger than grid5x6 both ILP formulations do not provide any feasible solution within the time limit.
The *Gap* columns report the difference between the optimum value of the two linear relaxation and the integer optimum, defined as :

$$Gap = \frac{objval\ ILP - objval\ LP}{objval\ ILP} \tag{2.116}$$

The gap is showed only when the MILP optimum is available. It is not surprising that the gap for the surrogated version is very weak: from 7% to 15% worse than for the standard LP relaxation, though fairly small in any case. On the other hand the computation time is always very small.
An interesting result is that the solution time for the ILP version in the surrogated version is competitive with the original formulation.

In Table 2.3 we report the results concerning the following models:

- NOP-EV solved by Benders Decomposition (*LP Bend* and *ILP Bend*)
  This model is described in Section (2.6.2). The time for computing the LP relaxation corresponds to the time for computing the root node in the B&B tree. As branching selection in the B&B tree we use the option iii), i.e. picking the one with highest value $\pi_e(\max x_{i,j}, x_{j,i})$.

- NOP-EV formulation with empty vehicles (*lp cplex* and *ILP cplex*)
  like in Table 2.2, as comparison with the Benders results.

graph	od matrix	t tot lp	objval lp	Gap	t tot lp surr	objval lp surr	Gap	t tot ilp	t tot ilp surr	objval ilp
grid3x4_gra	grid3x4_odm_asymm	0	291890	0.13	1	241180	0.28	4	0	333740
grid3x4_gra	grid3x4_odm_symm	1	277920	0.12	1	227380	0.28	3	1	314700
grid4x4_gra	grid4x4_odm_asymm	0	575700	0.1	1	502000	0.21	31	6	636780
grid4x4_gra	grid4x4_odm_symm	1	548020	0.1	1	474740	0.22	32	6	608280
grid4x5_gra	grid4x5_odm_asymm	3	968920	0.12	1	853960	0.22	220	71	1095900
grid4x5_gra	grid4x5_odm_symm	2	925200	0.11	1	810640	0.22	315	84	1039600
grid5x5_gra	grid5x5_odm_asymm	9	1687700	0.11	1	1536680	0.19	5041	2864	1902280
grid5x5_gra	grid5x5_odm_symm	8	1629120	0.11	1	1478920	0.19	2078	3992	1820900
grid5x6_gra	grid5x6_odm_asymm	35	2546050	0.11	1	2340840	0.18	18725	10003	2865780
grid5x6_gra	grid5x6_odm_symm	20	2480800	0.1	1	2276640	0.18	19887	36000	2770200*
grid6x6_gra	grid6x6_odm_asymm	26	4193540	-	1	3925160	-	-	-	-
grid6x6_gra	grid6x6_odm_symm	31	4085860	-	1	3818760	-	-	-	-
grid6x7_gra	grid6x7_odm_asymm	56	5939230	-	1	5572020	-	-	-	-
grid6x7_gra	grid6x7_odm_symm	61	5801580	-	1	5436300	-	-	-	-
grid7x7_gra	grid7x7_odm_asymm	104	8843020		1	8400840	-	-	-	-
grid7x7_gra	grid7x7_odm_symm	84	8640040	-	1	8200120	-	-	-	-
grid7x8_gra	grid7x8_odm_asymm	328	12027970	-	1	11463720	-	-	-	-
grid7x8_gra	grid7x8_odm_symm	221	11800160	-	1	11238780	-	-	-	-
grid8x8_gra	grid8x8_odm_asymm	558	17107680	-	1	16444120	-	-	-	-
grid8x8_gra	grid8x8_odm_symm	600	16757280	-	1	16097000	-	-	-	-
grid9x9_gra	grid9x9_odm_asymm	2711	30583490	-	1	29605700	-	-	-	-
grid9x9_gra	grid9x9_odm_symm	2397	29949120	-	1	28976480	-	-	-	-
grid9x10_gra	grid9x10_odm_asymm	4925	38655890	-	2	37467700	-	-	-	-
grid9x10_gra	grid9x10_odm_symm	6434	38095960	-	1	36913960	-	-	-	-
grid10x10_gra	grid9x10_odm_asymm	O.o.M	-	-	2	49936020	-	-	-	-

Table 2.2: Grid Instances - Cplex Results.

The *time ratio* columns report the ratio between the solution time with Benders and with the standard model:

$$time\ ratio\ MILP = \frac{t\ sol\ ILP\ cplex}{t\ sol\ ILP\ Bend} \tag{2.117}$$

$$time\ ratio\ LP = \frac{t\ sol\ LP\ cplex}{t\ sol\ LP\ Bend} \tag{2.118}$$

The time ratio is showed only when the ilp optimum is available.
In other words, a time ratio of 0.1 means that Benders decomposition is 10 times faster than the standard model. The LP solution with Benders decomposition is from 5 to 20 times faster than with the standard model. Also the MILP solution is faster: from 3 to 9 times faster.
Unfortunately even if the trend seems promising for instances bigger than grid5x6 not even Benders decomposition is able to solve the problem.

In Table 2.4 we compare different Benders-based heuristics. All of them are based on a depth fist search in a B&B tree, stopping at the first integer value found.
The different columns represents the different branching strategies:

- *x most int*
  at each node the most integer x_{ij} is selected

graph	od matrix	t sol ilp cplex	t sol ilp Bend	time ratio ilp	t sol lp cplex	t sol lp Bend	time ratio lp	lp val	objval ilp
grid3x4_gra	grid3x4_odm_asymm	4	1	0.25	1	1	1	291890	333740
grid3x4_gra	grid3x4_odm_symm	3	1	0.33	1	1	1	277920	314700
grid4x4_gra	grid4x4_odm_asymm	31	13	0.42	1	1	1	575700	636780
grid4x4_gra	grid4x4_odm_symm	32	7	0.22	1	1	1	548020	608280
grid4x5_gra	grid4x5_odm_asymm	220	99	0.45	3	1	0.33	968920	1095900
grid4x5_gra	grid4x5_odm_symm	315	41	0.13	2	1	0.50	925200	1039600
grid5x5_gra	grid5x5_odm_asymm	5041	1286	0.26	9	3	0.33	1687700	1902280
grid5x5_gra	grid5x5_odm_symm	2078	568	0.27	8	1	0.13	1629120	1820900
grid5x6_gra	grid5x6_odm_asymm	18725	5998	0.32	35	4	0.11	2546050	2865780
grid5x6_gra	grid5x6_odm_symm	19887	2965	0.15	20	3	0.15	2480800	2770200
grid6x6_gra	grid6x6_odm_asymm	-	-	-	26	7	0.27	4193540	-
grid6x6_gra	grid6x6_odm_symm	-	-	-	31	8	0.26	4085860	-
grid6x7_gra	grid6x7_odm_asymm	-	-	-	56	12	0.21	5939230	-
grid6x7_gra	grid6x7_odm_symm	-	-	-	61	12	0.20	5801580	-
grid7x7_gra	grid7x7_odm_asymm	-	-	-	104	21	0.20	8843020	-
grid7x7_gra	grid7x7_odm_symm	-	-	-	84	15	0.18	8640040	-
grid7x8_gra	grid7x8_odm_asymm	-	-	-	328	32	0.10	12027970	-
grid7x8_gra	grid7x8_odm_symm	-	-	-	221	30	0.14	11800160	-
grid8x8_gra	grid8x8_odm_asymm	-	-	-	558	55	0.10	17107680	-
grid8x8_gra	grid8x8_odm_symm	-	-	-	600	49	0.082	16757280	-
grid9x9_gra	grid9x9_odm_asymm	-	-	-	2711	153	0.06	30583490	-
grid9x9_gra	grid9x9_odm_symm	-	-	-	2397	99	0.04	29949120	-
grid9x10_gra	grid9x10_odm_asymm	-	-	-	4925	214	0.04	38655890	-
grid9x10_gra	grid9x10_odm_symm	-	-	-	6434	185	0.03	38095960	-
grid10x10_gra	grid10x10_odm_asymm	-	-	-	O.o.M.	386	-	51271780	-

Table 2.3: Grid Instances - Benders Decomposition

- *fist x then* π
  at each node the most integer x_{ij} is selected, if this occurs for more than one edge, the one with the corresponding highest dual value $\pi_{i,j}$ is chosen

- *fist x then* π
  at each node the x_{ij} with the highest product between x_{ij} and the corresponding dual variable $\pi_{i,j}$ is selected.

For grid instances it seems that guiding the edge orientation with the help of the dual variables not helpful.

Table 2.4 provides more details about the comparison between the LP and MILP optimal solution and the diving using the first branching rule. Columns *delta LP* (resp. *delta MILP*) represents the gap between the best Upper Bound and the optimal LP (resp. MILP) value:

$$gap\ MILP = \frac{\text{best UB} - \text{opt val MILP}}{\text{opt val MILP}} \tag{2.119}$$

$$gap\ LP = \frac{\text{best UB} - \text{opt val LP}}{\text{opt val LP}} \tag{2.120}$$

The LP delta value is slightly decreasing with the increasing of the instances size. Also the ILP delta value has the same behavior. An advantage of this diving heuristic is that it provides solutions about 7% worse then the optimal value in a small amount of time.

graph	od matrix	lp val	objval ILP	t tot Bend Dive x most int	best UB x most int	t tot fist x then π	best UB first x then π	t tot π x	best UB π x
grid3x4_gra	grid3x4_odm_asymm	291890	333740	0	369600	1	346060	1	335400
grid3x4_gra	grid3x4_odm_symm	277920	314700	1	342380	0	332100	0	332100
grid4x4_gra	grid4x4_odm_asymm	575700	636780	1	690760	2	648100	2	699100
grid4x4_gra	grid4x4_odm_symm	548020	608280	2	658920	1	684800	1	677580
grid4x5_gra	grid4x5_odm_asymm	968920	1095900	4	1179560	5	1203320	3	1221100
grid4x5_gra	grid4x5_odm_symm	925200	1039600	4	1106000	5	1194700	3	1152000
grid5x5_gra	grid5x5_odm_asymm	1687700	1902280	10	2023860	9	1992400	11	1953900
grid5x5_gra	grid5x5_odm_symm	1629120	1820900	10	1961000	9	1933880	11	2013920
grid5x6_gra	grid5x6_odm_asymm	2546050	2865780	24	2991560	23	3095500	21	3146560
grid5x6_gra	grid5x6_odm_symm	2480800	2770200	17	2932100	24	3126940	21	3125760
grid6x6_gra	grid6x6_odm_asymm	4193540	-	50	5094640	58	5089880	54	5151680
grid6x6_gra	grid6x6_odm_symm	4085860	-	46	4915800	60	4888880	64	5194340
grid6x7_gra	grid6x7_odm_asymm	5939230	-	115	6842460	118	7362360	117	7515840
grid6x7_gra	grid6x7_odm_symm	5801580	-	88	6680120	102	6891900	100	7083280
grid7x7_gra	grid7x7_odm_asymm	8843020	-	224	10449760	241	10796780	278	10850080
grid7x7_gra	grid7x7_odm_symm	8640040	-	188	10121340	241	10378220	294	10629180
grid7x8_gra	grid7x8_odm_asymm	12027970	-	431	14120640	466	14838820	480	14525120
grid7x8_gra	grid7x8_odm_symm	11800160	-	336	13706000	522	14324340	521	14232640
grid8x8_gra	grid8x8_odm_asymm	17107680	-	785	19931660	884	20478060	1037	20706400
grid8x8_gra	grid8x8_odm_symm	16757280	-	779	19379520	950	19891420	1194	20199700
grid9x9_gra	grid9x9_odm_asymm	30583490	-	2741	34531720	3636	37349500	4246	37192260
grid9x9_gra	grid9x9_odm_symm	29949120	-	2350	34141860	3885	38454520	3324	35431520
grid9x10_gra	grid9x10_odm_asymm	38655890	-	4927	44843540	7154	50093320	8192	47669760
grid9x10_gra	grid9x10_odm_symm	38095960	-	3740	42846760	8468	48428200	8045	46253600
grid10x10_gra	grid10x10_odm_asymm	51271780	-	7113	57827440	12742	64094120	10760	60911380
grid10x10_gra	grid10x10_odm_symm	-	-	7904	56613360	13308	62221640	12369	60819420

Table 2.4: Grid Instances - Benders Diving for NOP

graph	od matrix	lp val	objval ILP	t tot Bend Dive x most int	best UB x most int	gap LP	gap ILP
grid3x4_gra	grid3x4_odm_asymm	291890	333740	0	369600	0.27	0.11
grid3x4_gra	grid3x4_odm_symm	277920	314700	1	342380	0.23	0.09
grid4x4_gra	grid4x4_odm_asymm	575700	636780	1	690760	0.20	0.08
grid4x4_gra	grid4x4_odm_symm	548020	608280	2	658920	0.20	0.08
grid4x5_gra	grid4x5_odm_asymm	968920	1095900	4	1179560	0.22	0.08
grid4x5_gra	grid4x5_odm_symm	925200	1039600	4	1106000	0.20	0.06
grid5x5_gra	grid5x5_odm_asymm	1687700	1902280	10	2023860	0.20	0.06
grid5x5_gra	grid5x5_odm_symm	1629120	1820900	10	1961000	0.20	0.08
grid5x6_gra	grid5x6_odm_asymm	2546050	2865780	24	2991560	0.17	0.04
grid5x6_gra	grid5x6_odm_symm	2480800	2770200	17	2932100	0.18	0.06
grid6x6_gra	grid6x6_odm_asymm	4193540	-	50	5094640	0.21	-
grid6x6_gra	grid6x6_odm_symm	4085860	-	46	4915800	0.20	-
grid6x7_gra	grid6x7_odm_asymm	5939230	-	115	6842460	0.15	-
grid6x7_gra	grid6x7_odm_symm	5801580	-	88	6680120	0.15	-
grid7x7_gra	grid7x7_odm_asymm	8843020	-	224	10449760	0.18	-
grid7x7_gra	grid7x7_odm_symm	8640040	-	188	10121340	0.17	-
grid7x8_gra	grid7x8_odm_asymm	12027970	-	431	14120640	0.17	-
grid7x8_gra	grid7x8_odm_symm	11800160	-	336	13706000	0.16	-
grid8x8_gra	grid8x8_odm_asymm	17107680	-	785	19931660	0.17	-
grid8x8_gra	grid8x8_odm_symm	16757280	-	779	19379520	0.16	-
grid9x9_gra	grid9x9_odm_asymm	30583490	-	2741	34531720	0.13	-
grid9x9_gra	grid9x9_odm_symm	29949120	-	2350	34141860	0.14	-
grid9x10_gra	grid9x10_odm_asymm	38655890	-	4927	44843540	0.16	-
grid9x10_gra	grid9x10_odm_symm	38095960	-	3740	42846760	0.12	-
grid10x10_gra	grid10x10_odm_asymm	-	-	7113	57827440	-	-
grid10x10_gra	grid10x10_odm_symm	-	-	7904	56613360	-	-

Table 2.5: Grid Instances - Benders Diving for NOP - Deltas

graph	od matrix	t tot	best UB non-LP	t tot Bend Dive x most int	best UB x most int
grid3x4_gra	grid3x4_odm_asymm	0	378480	0	369600
grid3x4_gra	grid3x4_odm_symm	0	322020	1	342380
grid4x4_gra	grid4x4_odm_asymm	0	729900	1	690760
grid4x4_gra	grid4x4_odm_symm	0	680660	2	658920
grid4x5_gra	grid4x5_odm_asymm	0	1296640	4	1179560
grid4x5_gra	grid4x5_odm_symm	1	1129200	4	1106000
grid5x5_gra	grid5x5_odm_asymm	0	2116220	10	2023860
grid5x5_gra	grid5x5_odm_symm	1	1842580	10	1961000
grid5x6_gra	grid5x6_odm_asymm	0	3049920	24	2991560
grid5x6_gra	grid5x6_odm_symm	1	2823280	17	2932100
grid6x6_gra	grid6x6_odm_asymm	2	5230300	50	5094640
grid6x6_gra	grid6x6_odm_symm	2	4704820	46	4915800
grid6x7_gra	grid6x7_odm_asymm	3	7063980	115	6842460
grid6x7_gra	grid6x7_odm_symm	2	6435220	88	6680120
grid7x7_gra	grid7x7_odm_asymm	4	10469340	224	10449760
grid7x7_gra	grid7x7_odm_symm	6	9904920	188	10121340
grid7x8_gra	grid7x8_odm_asymm	8	14873460	431	14120640
grid7x8_gra	grid7x8_odm_symm	8	13447420	336	13706000
grid8x8_gra	grid8x8_odm_asymm	13	21385720	785	19931660
grid8x8_gra	grid8x8_odm_symm	14	18546680	779	19379520
grid9x9_gra	grid9x9_odm_asymm	34	45286040	2741	34531720
grid9x9_gra	grid9x9_odm_symm	32	34036880	2350	34141860
grid9x10_gra	grid9x10_odm_asymm	52	57661000	4927	44843540
grid9x10_gra	grid9x10_odm_symm	56	54042560	3740	42846760
grid10x10_gra	grid10x10_odm_asymm	82	73159410	7113	57827440
grid10x10_gra	grid10x10_odm_symm	86	74400320	7904	56613360

Table 2.6: Grid Instances - Benders Diving vs combinatorial bounds

Finally, in Table 2.6 we present the comparison between the UB founds by the best Benders diving and the diving based on the $\frac{1}{2}$-lower bound described in Section 2.7.1 (*best UB non-LP*), showing that the latter is much slower but the solution found by the former are much better.

Real-World Instances

In this Section we take into consideration the Bologna and Masdar instances.
In Table 2.7 we report the following information about the Masdar instances:

- LP relaxation and best MILP value obtained from the surrogated version (*LP surr* and *MILP surr*)
  like in Table 2.2, the surrogated method is not able to find the optimal solution within the time limit. We do not report the results concerning the standard formulation because the problem size is too big and the solver goes out of memory.

- LP relaxation value provided by the solution of Benders decomposition (*best LB Bend*)

graph	od matrix	t tot LP surr	objval LP surr	best LB Bend T_LIM 36000	best UB surr T_LIM 36000	gap LP surr	gap LP Bend
mm_city_g05	mm_city_ddummy_03	19	1558982		2404591	0.54	
mm_city_g05	mm_city_dam_01	21	4866754	5607194	7150847	0.47	0.28
mm_city_g05	mm_city_dpm_01	20	3230226		4749240	0.47	
et_city_g02	mm_city_ddummy_01	18	1480853	1878239	2379618	0.61	0.27
et_city_g02	mm_city_ddummy_01_rand01	18	73803766	92320069	116704013	0.58	0.26

Table 2.7: Masdar - ILP

graph	od matrix	t tot LP cplex	t tot LP Surr	t tot LP Bend	objval LP	objval LP Surr	objval cplex ILP surr T_LIM	objval cplex ILP T_LIM
MODBO_GRAPH	MODBO_ODM	40	1	9	593820635	508019263	680280050	682294090

Table 2.8: Bologna - ILP

Benders decomposition is not able to solve the LP relaxation to optimality, this is the best available lower bound.

The two columns *gap LP* describe the gap between the best UB found and the two LP provided:

$$delta = \frac{\text{best UB} - \text{objval LP surr (or best LP Bend)}}{\text{objval LP surr (or best LP Bend)}} \tag{2.121}$$

Benders decomposition turns out to be useful in order to increase the best LB found. It is not clear if the high gaps between th beste LB and UB is due to the weakness of the first or second value or on the weakness of the model itself. It is clear in any case the usefulness of Benders decomposition in order to improve more than 30% the gap with the upper bound. Finally it is interesting to note how the gap remains the same in both the symmetric and asymmetric cases.

In Table 2.8 we report the same information described in Table 2.7 but concerning Bologna instance. In addition is provided also the LP relaxation and the best upper bound found in the solution of the standard formulation. Also in this case Benders decomposition is faster than the standard formulation in computing the lower bound. An interesting result is that the surrogated formulation, even relying on a weaker LP relaxation, provides an upper bound better than the one provided by the standard formulation.

In Table (2.9) Benders heuristics are tested:

- Best solution found within T_LIMIT (*Benders*)
- Benders diving (*Dive*)

For each modality the branching method described for Table 2.4 are reported.
The column *delta* describes the difference of the heuristics proposed with the best upper

graph	od matrix	mode	t tot	t sol lp	lp val	best UB	delta
MODBO_GRAPH	MODBO_ODM	ILP surr	36000	1	508019263	680280050	0
MODBO_GRAPH	MODBO_ODM	Dive πx	45	9	593820635	694651520	0.021
MODBO_GRAPH	MODBO_ODM	Dive x most int	54	9	593820635	741077810	0.089
MODBO_GRAPH	MODBO_ODM	Dive first x most int then π	57	9	593820635	727781100	0.070
MODBO_GRAPH	MODBO_ODM	Benders x most int	36000	8	593820635	735229800	0.081
MODBO_GRAPH	MODBO_ODM	Benders πx	36000	9	593820635	681784380	0.002
MODBO_GRAPH	MODBO_ODM	Benders first x most int then π	36000	9	593820635	707993160	0.041

Table 2.9: Bologna - Benders

bound known so far, provided by the surrogated formulation:

$$delta(mode) = \frac{\text{UB mode} - \text{best UB surr}}{\text{best UB surr}} \tag{2.122}$$

$$\tag{2.123}$$

Among the three branching policies, πx is the most effective. It is interesting how in just 45 seconds it provides a solution 2% worst then the best solution found. In addition if we let the program run until the time limit the two algorithms are almost comparable.

2.10.2 NOP-FDC

In this section we provide the results concerning the algorithms proposed for solving the MINLP model described in Section (2.3.5).

Grid instances

In Table 2.10 we report the following information concerning the MILP relaxation of the problem:

- LP relaxation (*lpcplex*)
  This is the value of the LP relaxation of the MILP model. For instances of size greater than grid3x4 it is the only available lower bound for the problem (only for grid3x4 we knows the optimal value)

- best UB found for MILP relaxation (*objval MILP cplex*)
  as mentioned above only for grid3x4 it is possible to solve to optimality the problem. For instances bigger than grid5x5 no feasible solution has been found within the time limit.

graph	od matrix	t tot lp	objval lp cplex	t tot ilp	objval ilp cplex	real objval ilp	gap	delta ilp - real objval
grid3x4_gra	grid3x4_odm_asymm	2	291890	512	8492142.92	13391262.32	44.88	0.58
grid3x4_gra	grid3x4_odm_symm	1	277920	398	7409627.36	11646666.64	40.91	0.57
grid4x4_gra	grid4x4_odm_asymm	11	575700	36000	19343782.25	66119100.10	113.85	2.42
grid4x4_gra	grid4x4_odm_symm	8	548020	36000	17189029.65	55792359.75	100.81	2.25
grid4x5_gra	grid4x5_odm_asymm	49	968920	36000	63001134.23	256652634.34	263.89	3.07
grid4x5_gra	grid4x5_odm_symm	36	925200	36000	51488472.30	164419003.63	176.71	2.19
grid5x5_gra	grid5x5_odm_asymm	308	1687700	36000	222873171.77	674768100.49	398.82	2.03
grid5x5_gra	grid5x5_odm_symm	209	1629120	36000	195711056.44	580045911.15	355.05	1.96
grid5x6_gra	grid5x6_odm_asymm	1272	2546050	T_LIM	-	-	-	-
grid5x6_gra	grid5x6_odm_symm	1107	2480800	T_LIM	-	-	-	-
grid6x6_gra	grid6x6_odm_asymm	4360	4193540	T_LIM	-	-	-	-
grid6x6_gra	grid6x6_odm_symm	12336	4085860	T_LIM	-	-	-	-
grid6x7_gra	grid6x7_odm_asymm	15645	5939230	T_LIM	-	-	-	-
grid6x7_gra	grid6x7_odm_symm	17366	5801580	T_LIM	-	-	-	-
grid7x7_gra	grid7x7_odm_asymm	36000	8843020	T_LIM	-	-	-	-
grid7x7_gra	grid7x7_odm_symm	T_LIM	-	T_LIM	-	-	-	-

Table 2.10: Grid Instances - NOP Flow-Dependent Costs - ILP relaxation

- real objective function value (*real objval ilp*)
  This is the value of the nonlinear objective function with the empty- and full-vehicle flows in the best solution of the ILP model.

Two columns are added in order to compare the data described above:

- $gap = \dfrac{\text{real objval ILP} - \text{objval LP cplex}}{\text{objval LP cplex}}$
  Tthis value indicates how much the UB found is bigger than the current LB

- $delta\ ILP - real\ objval = \dfrac{\text{real objval ILP} - \text{objval ILP}}{\text{objval ILP}}$
  This indicator gives an idea about how much the MILP relaxation is far from the real value.

For instances of realistic size the lower bounds provided are really poor: from 40 to almost 400 times lower than the best upper bound found. In addition the real objective function is from 0.5 to 3 times higher than the best MILP relaxation found.
This high differences between the real objective function value and its estimators is due to the linearization, which turned out to be fairly weak.

In Table 2.11 we report the following results concerning the "two-phase" approach (see Section 2.8.1). As first phase we solve NOP-HCC, described in Section (2.3.4), with capacity equal to the maximum capacity available c^1 and edge length equal to the maximum length l^1. The idea of this approach is to solve the orientation problem as "fully congested", i.e. with all all the overflows at their maximum values. Once the orientation is fixed we branch on the ℓ variables. At each node in the B&B tree we compute the real objective function value for

graph	od matrix	t tot	first objval	objval	improvement
grid3x3_11_gra	grid3x3_11_odm_asymm	2	1141139	1123693	0.02
grid3x3_11_gra	grid3x3_11_odm_symm	1	381600	377389	0.01
grid3x4_11_gra	grid3x4_11_odm_asymm	15	13965250	13965250	0
grid3x4_11_gra	grid3x4_11_odm_symm	9	11576078	11576078	0
grid4x4_11_gra	grid4x4_11_odm_asymm	231	54119486	42602919	0.21
grid4x4_11_gra	grid4x4_11_odm_symm	145	55792359	55019912	0.014
grid4x5_11_gra	grid4x5_11_odm_asymm	1544	232420918	156466041	0.33
grid4x5_11_gra	grid4x5_11_odm_symm	1524	210694999	145387561	0.31
grid5x5_11_gra	grid5x5_11_odm_asymm	22845	648673630	343999750	0.47
grid5x5_11_gra	grid5x5_11_odm_symm	14839	762228750	344547967	0.55

Table 2.11: Grid - NOP Flow-Dependent Costs - first x then ℓ

the current empty and full vehicle flows. The search is finished when no more branching is needed and before the backtracking.
The information provided by Table 2.11 is the following:

- best solution found (*objval*)
- first solution found (*first objval*)
  this is the real objective function corresponding to the empty- and full-vehicle flows assigned after phase one, it is used to verify how much is important the second phase in the improvement of the upper bound.
  Column *improvement* measures this aspect:

$$improvement = \frac{\text{first objval} - \text{objval}}{\text{first objval}} \tag{2.124}$$

The second phase is fundamental in the improving of the solution: for big instances the branching on ℓ decreases the phase one solution by about 30% to 50%.

Table 2.12 reports the results for a "one-phase" approach. As explained in Section (2.8.2) we branch on the x and ℓ variables together. The usual behavior is that first we start to branch only on the x variables (because the ℓ are at their lower bounds). After few branching some ℓ values start to move and we start the double branching. At each node in the B&B tree the branching scheme is done with the following disjunction:

$$\{x_{i,j} = 1 \wedge \ell_{k,h} \leq \bar{\ell}_{k,h}\} \vee \{x_{i,j} = 0\} \vee \{x_{i,j} = 1 \wedge \ell_{k,h} \geq \bar{\ell}_{k,h}\} \tag{2.125}$$

The data reported are relative to the same B&B method with two stopping criteria:

graph	od matrix	tot nodes x&l	t tot dive x&l	objval dive x&l	objval T_LIM 3600
grid3x4_gra	grid3x4_odm_asymm	13	12	35682144.27	13391262.32
grid3x4_gra	grid3x4_odm_symm	11	19	21848298.76	11646666.64
grid4x4_gra	grid4x4_odm_asymm	17	222	104673326.39	66119100.11
grid4x4_gra	grid4x4_odm_symm	23	439	63363401.29	55792359.75
grid4x5_gra	grid4x5_odm_asymm	29	2309	264418032.71	315068140.96
grid4x5_gra	grid4x5_odm_symm	22	1791	244234455.19	261516625.79
grid5x5_gra	grid5x5_odm_asymm	43	11949	730874518.43	1318384973.65
grid5x5_gra	grid5x5_odm_symm	28	14591	777763155.27	no sol found
grid5x6_gra	grid5x6_odm_asymm	-	T_LIM	-	-

Table 2.12: Grid Instances - NOP Flow-Dependent Costs - x & ℓ Heur

graph	od matrix	objval first x then ℓ	objval x and ℓ
grid3x4_gra	grid3x4_odm_asymm	13965250	13391262.32
grid3x4_gra	grid3x4_odm_symm	11576078	11646666.64
grid4x4_gra	grid4x4_odm_asymm	42602919	66119100.10
grid4x4_gra	grid4x4_odm_symm	55019912	55792359.75
grid4x5_gra	grid4x5_odm_asymm	156466041	256652634.34
grid4x5_gra	grid4x5_odm_symm	145387561	164419003.63
grid5x5_gra	grid5x5_odm_asymm	343999750	674768100.49
grid5x5_gra	grid5x5_odm_symm	344547967	580045911.15

Table 2.13: Grid Instances - NOP Flow-Dependent Costs - *first* x *then* ℓ vs x *&* ℓ

- diving (*dive*)
  The exploration ends all x variable are either fixed or integer and all ℓ variables are equals to one of their bounds.
  Notice that the tree depth is not fixed because the number of branching on the ℓ variables is not unique.

- time limit (*T_LIM 3600*)
  Notice that, even if the time necessary to dive the whole tree is greater than 3600 seconds, if after this time limit all the x variables has been fixed a valid UB for the problem is available in any case.

In Table 2.13 a comparison of the two heuristics is provided. It is clear how the two phases optimization provides better results, on the other hand the one-phase approach is faster and is able to provide a feasible solution in only one hour of computational time.

graph	od matrix	t tot ilp	objval ilp cplex	real objval ilp	t tot lp	objval lp cplex
MODBO_GRAPH	MODBO_ODM	36000*	747317670.96	2006306172.93	218	593820635

Table 2.14: Bologna- NOP Flow-Dependent Costs - ILP

graph	od matrix	t tot dive x&l	objval dive x&l	Real objval ILP 3600
MODBO_GRAPH	MODBO_ODM	10212	1640246128.24	

Table 2.15: Bologna - NOP Flow-Dependent Costs - Heur compare

Real-World Instances

In this section we mainly focus on the Bologna instance because the Masdar instance is too large to be solved with the method proposed.
In Table 2.14 we provide the results related to the MILP relaxation. In Table 2.15 the $x\&l$ diving results are reported.

The $x\&l$ diving provides a better solution than the one provided by the computation of the MILP relaxation.

Chapter 3

Linear Arrangement Problem

3.1 Problem Description

MinLAP can be stated as follows: given an undirected graph $G(V, E)$ with $|V| = n$, $|E| = m$ and a weight w_e associated with each edge e , find a labeling $\pi : V \rightarrow \{1, \ldots, n\}$ that minimizes

$$\sum_{(i,j)\in E} w_{i,j}|\pi(i) - \pi(j)|.$$

For a given graph G we call $lap(G)$ the linear arrangement optimum value.
From now on we define $S(n)$ as the set of all possible labeling of n elements. Moreover when we deal with a graph G we always refers to V and E as the sets denoting respectively his vertices and edges, and with $n = |V|$, $m = |E|$ their cardinality.
Notice that in the rest of this section we will consider the unweighted case, i.e. when $w_e = 1$ $\forall e \in E$, all the results and th e procedure described hold also for the weighted case.

3.2 Previous Integer Linear Programming Formulations

MinLAP can be formulated as an Integer Linear Programming in a descriptive way.
We introduce a set of binary variables x_{ij}, one for every pair of nodes $i, j, \in V$, x_{ij} is equal to one if the node i is placed in position j (i.e. if $\pi(i) = j$), and a set of variables $d_e \equiv d_{\{i,j\}}$, one for every edge $e \equiv \{i, j\} \in E$, representing the distance between nodes i and j, i.e. $d_{\{i,j\}} = |\pi(i) - \pi(j)|$.

$$min \sum_{e \in E} d_e \tag{3.1}$$

$$\sum_{j \in V} x_{i,j} = 1 \qquad i \in V, \tag{3.2}$$

$$\sum_{i \in V} x_{i,j} = 1 \qquad j \in V, \tag{3.3}$$

$$d_{\{i,j\}} \geq |p - q|(x_{ip} + x_{jq} - 1), \quad \{i,j\} \in E, p, q \in V, \tag{3.4}$$

$$x_{ij} \in \{0,1\}, \qquad i,j \in V \tag{3.5}$$

Constraints (3.2) and (3.3) force every node to be placed in exactly one position ad every position to be used by exactly one node. (3.4) are the linking constraints between d_e and x_{ij}. Defining an effective solution method based on this formulation is hard because of the weakness of the LP relaxation (it is easy to see that the LP solution $x_{ij} = 1/n, \forall i,j \in V$ and $d_e = 0 \ \forall e \in E$ is feasible and has value zero) and of the huge number of constraints.

Liu and Vannelli [1] considered a formulation based only on the d_e variables. This choice is due to the consideration that the variables x_{ij} do not appear in the objective function, hence if we do not pretend to have a compact formulation for the problem they can be eliminated. A general condition on the d_e variables are the so-called *rank-inequalities*, one for every subgraph G' of G. With this set of inequalities we obtain the following *sparse formulation with* d_e:

$$min \sum_{e \in E} d_e \tag{3.6}$$

$$\sum_{e \in E(G')} d_e \geq lap(G'), \forall G' \in \mathcal{G} \tag{3.7}$$

Where $\mathcal{G}$ is the set of all subgraphs of G .
The rank inequalities are based on the simple but really useful observation that the sum of the distances associated with the edges of a subgraph G' must be at least equal to the optimal value $lap(G')$ associated with it.
If we enumerate all the (exponentially many) rank inequalities by definition we obtain an exact formulation of the problem (because also G is a subgraph of itself) but this is not helpful because it is as hard as solving the problem itself.
If we restrict ourselves to add only a subset of the rank-inequalities we can compute valid lower bounds for the problem, and the strength and the difficulty of the LP depends on how

we chose this subset, for example we can focus only on special class of subgraphs of G, like stars, cliques, paths. Among all the possible subgraphs the separation of the rank-inequalities concerning stars turns out to be very easy.
The key aspect in the procedure suggested by Liu and Vannelli is the simple and general principle that we can use information concerning a subgraph of G to obtain information on G, this concept has been used also in the next approach described by Caprara et al. [2] and in our work.

The sparse formulation uses one distance variable d_e for every edge existing in the graph G, this minimizes the number of variables but limts its "flexibility".

In [2] Caprara et al. proposed an approach based on a *dense formulation* with the distance variables, let $K_n = (V, F)$ be the complete graph on n vetices and let $\mathcal{K}$ be the set of all edge induced subgraphs K' of K_n and let $F(K')$ be the set of edges in subgraph K'. They introduce a set of variables $d_{\{i,j\}}$, for each edge $\{i, j\}$ of F (i.e. one for each pair of vertices in V) that represent the distance between vertices i and j in the arrangement.

$$\min \sum_{e \in E} d_e \tag{3.8}$$

$$\sum_{\{i,j\} \in F(H)} d_{\{i,j\}} \geq lap(K), \qquad k \in \mathcal{K} \tag{3.9}$$

$$d_{\{i,j\}} \leq d_{\{i,k\}} + d_{\{k,j\}}, \quad \forall i, j, k \subseteq V \tag{3.10}$$

As we see the objective function takes into account the edges in the original graph G. Working on the complete graph allows one to add the set of *triangle inequalities* (3.10 that allows one to obtain stronger LP relaxations than in the sparse case.
Like for the sparse formulation also in this case one must use a subset of $\mathcal{K}$ to obtain practically computable lower bounds.

The link between sparse and dense formulation provided by Caprara et al. is the so called *projected LP relaxation*, the idea is to remove the rank inequalities and substitute the $d_{\{i,j\}}$ variables in the dense formulation with the variables d_e representing the shortest path between i and j in the original graph G. Practically speaking it is possible to precompute all the shortest paths between each pair of nodes in the original graph G and then do the separation in the complete graph.
It is easy to prove that the feasible region of the projected relaxation is the projection over $\Re^E$ of the feasible region of the dense formulation.

As we will see in the next sections it is interesting how the parallelism between sparse and dense formulation exists also in our work, even if we deal with a different set of variables.

3.3 New Integer Linear Programming Formulation

As mentioned in the previous section for MinLAP we have: i) the existence of a dense and a sparse formulation, and ii) the possibility to use information about subgraphs of the original graph to obtain valid inequalities for it.

The new ILP formulation uses the so called betweenness variabless, already known in the literature [CIT].
Given a graph G, for each triple i, j, k , $(i, j) \in E$, $k \in V \backslash \{i, j\}$ and a labeling $\pi \in S(n)$ we define a binary variable x_{ikj} s.t.

$$x_{ikj} = \begin{cases} 1 & \text{if } \begin{array}{l} \pi^{-1}(i) < \pi^{-1}(k) < \pi^{-1}(j) \text{ or} \\ \pi^{-1}(i) > \pi^{-1}(k) > \pi^{-1}(j) \end{array} \\ 0 & \text{otherwise} \end{cases} \tag{3.11}$$

In other words, x_{ikj} is equal to one if k is between i and j in the labeling π, note that no information is directly provided about the distance between two of the three elements of the triple i, j, k, the only information required being if k lies between i and j.
Even if there is no difference between the notation x_{132} and x_{231}, for sake of uniformity we always use an ordered notation with the first index lower than the third one, hence among the two notation we alway use the first one. For a given graph G the number of betweenness variable is $m(n-2)$.

We introduce the betweenness polytope for a given graph $G = (V, E)$

$$P^G_{BTW} = conv\{x \in \Re^{|E||V-2|} | x \text{ is a valid set of betweenness variable for a given } \pi \in S(n)\} \tag{3.12}$$

Even if it is not explicit the information about distances in the permutation can be easily obtained by the following relation:

$$|\pi(i) - \pi(j)| = 1 + \sum_{k \in V \backslash \{i,j\}} x_{ikj} \tag{3.13}$$

Thus we can redefine the MinLAP objective function as follows :

$$\min_{x \in P^G_{BTW}} \sum_{(i,j) \in E} (1 + \sum_{k \in V \setminus \{i,j\}} x_{ikj}) = m + \min_{x \in P^G_{BTW}} \sum_{(i,j) \in E} \sum_{k \in V \setminus \{i,j\}} x_{ikj} \tag{3.14}$$

the new x_{ikj} variables can be viewed as an "expansion" of the previous d_e variable, something like splitting the general integer variable into several binary variable, obtaining an higher quantity of information.

As we will see better in the next section some of the constraints valid for P^G_{BTW} have a counterpart in the formulation by Caprara et al. but some other are specific for the betweenness approach and exploit the addition of the "middle index", that gives the opportunity of introducing other inequalities that turned out to be really useful in practice.

3.4 Polyhedral Study

In this section we will describe some results related to P^G_{BTW}, comparing them with the results obtained for the d_e formulation by Caprara et al.

Optimizing over the dominant of a polytope is equivalent to optimizing over the polytope if the objective function is non-negative, thus sometimes we will focus on the dominant for P^G_{DIST} and for P^G_{BTW}.

Before starting we define the polytope P^G_{DIST} , his dominant D^G_{DIST} and D^G_{BTW}, the dominant of P^G_{BTW}

$$P^G_{DIST} = conv\{d_e \in \Re^{|E|} | \exists \pi : d_e = |\pi(i) - \pi(j)| \ \forall e \equiv (i,j) \in E\} \tag{3.15}$$

$$D^G_{DIST} = \{d_e \in \Re^{|E|} | \exists d'_e \in P^G_{DIST} : d_e \geq d'_e\} \tag{3.16}$$

$$D^G_{BTW} = \{x_{ikj} \in \Re^{|E||V-2|} | \exists x'_{ikj} \in P^G_{BTW} : x_{ikj} \geq x'_{ikj}\} \tag{3.17}$$

Like for the formulations based on distance variables also for the betweenness variables it is possible to introduce a dense formulation where instead of optimizing over G we optimize over the corresponding complete graph K_n:

$$\min_{x \in P_{BTW}^{K_n}} \sum_{(i,j) \in E} \sum_{k \in V \setminus \{i,j\}} 1 + \ x_{ikj} = m + \min_{x \in P_{BTW}^{K_n}} \sum_{(i,j) \in E} \sum_{k \in V \setminus \{i,j\}} x_{ikj} \tag{3.18}$$

In this new formulation we give a unitary cost only to the edges that belong to G.
Note that this approach can be generalize optimizing over every graph obtained from G and adding to it any number of missing edges.
Let $G' = (V, E')$ with $E' \subseteq E$, the set of variable defining $P_{BTW}^{G'}$ is a subset of the one defining P_{BTW}^{G}.

The valid inequalities for P_{BTW}^{G} include:
i) triangle equations
ii) d_e inequalities
iii) "subgraph" inequalities
iv) cut polytope inequalities

the last category is even more than a simple set of inequalities, it is the tool used to reduce the dense formulation to the sparse formulation in the betweenness variable polytope.

3.4.1 Triangle Equation

The so called triangle equations are the core of the betweenness approach.

Proposition 4 *For any triple of vertices $i, j, k \in V$ the following* triangle equation *is valid for $P_{BTW}^{K^n}$:*

$$x_{ikj} + x_{ijk} + x_{jik} = 1 \tag{3.19}$$

Proof. It is trivial to see that in any feasible lay-out exactly one index lays between the two others. □

In addition when we talk about *triangle inequalities* we are referring to (3.19) with $\geq$ instead of $=$.
For a complete graph K_n the number of triangle inequalities is $\binom{n}{3}$

The introduction of the triangle equation gives us the opportunity to establish a bound on the dimension of $P_{BTW}^{K_n}$:

Proposition 5 *The dimension of $P_{BTW}^{K_n}$ is bounded from above by*

$$2\binom{n}{3} \tag{3.20}$$

Proof. As mentioned before the number of betweenness variable is $m(n-2)$. For a complete graph the number of edges is equal to $\binom{n}{2}$.

In a complete graph the number of triangle equation is $\binom{n}{3}$. Hence the dimension of $P_{BTW}^{K_n}$ is bounded by:

$$\binom{n}{2}(n-2) - \binom{n}{3} = \frac{n!\ (n-2)}{(n-2)!\ 2!} - \binom{n}{3} \tag{3.21}$$

$$= 3\frac{n!}{(n-3)!\ 3!} - \binom{n}{3} = 3\binom{n}{3} - \binom{n}{3} \tag{3.22}$$

$$= 2\binom{n}{3} \tag{3.23}$$

□

3.4.2 d_e Inequalities

Starting from 3.13 it is possible to establish the correlation between distance variables d_e and betweenness variables x_{ikj}:

$$d_{\{i,j\}} = 1 + \sum_{k \in V \setminus \{i,j\}} x_{ikj} \tag{3.24}$$

By relation (3.24) we can use all the inequalities defined by Caprara et al. [2] and reuse them in our model. For the sake of brevity we report only the main results related to stars and clique:

Proposition 6 *For any $n \geq 2$, and for any $S \subseteq N(i)$, the star inequality*

$$\sum_{j \in S} d_{\{i,j\}} \geq \lfloor (|S|+1)^2/4 \rfloor \tag{3.25}$$

is valid for D_{DIST}^G and faced inducing if $|S| \neq 2$

if we apply equation 3.24 to 3.25 we obtain the star inequality for P_{BTW}^G :

$$\sum_{j \in N(i)} \sum_{k \in N(i), k \neq j} x_{ikj} \geq \lfloor (|S|-1)^2/4 \rfloor \tag{3.26}$$

For a complete graph K_n the number of p-star inequalities is $\binom{n}{p+1}$, with $p \leq n-1$.

Proposition 7 *For any $n \geq 2$, and for any $S \subseteq V$ inducing a clique in G, the clique inequality*

$$\sum_{\{i,j\} \subseteq S} d_{\{i,j\}} \geq \binom{|S|+1}{3} \tag{3.27}$$

is valid and facet inducing for D^G_{DIST}

For a complete graph K_n the number of p-clique inequalities is $\binom{n}{p}$ with $p \leq n$.

Equation (3.24) can be used to translate one inequality valid for P^G_{DIST} into one valid for P^G_{BTW}.

Example A valid inequality for $P^{K_n}_{DIST}$ related to a 3-star centered in $i = 1$ is the following:

$$d_{1,2} + d_{1,3} + d_{1,4} \geq 4 \tag{3.28}$$

hence the corresponding valid inequality for P^G_{BTW} is:

$$\sum_{k \in V \ \{i,j\}} x_{1k2} + \sum_{k \in V \ \{i,j\}} x_{1k3} + \sum_{k \in V \ \{i,j\}} x_{1k4} \geq 1 \tag{3.29}$$

□

Unfortunately if we apply the substitution (3.24) the inequalities (3.25) and (3.27) are no longer facet defining for D^G_{BTW} or P^G_{BTW}, as shown in the example in the following section.

3.4.3 Cut Polytope Inequalities

Together with the triangle equations (3.19) the following proposition is the main advantage of working with P^G_{BTW} instead of P^G_{DIST}: From a computation point of view they turned out to be really effective in the computation of good lower bound for the problem. In addition to the practical aspect also from a theoretical point of view the cut polytope inequalities are useful in the classification of facet defining inequalities of simple graphs.

For our purpose we need first to define the cut polytope P^G_{CUT} for a generic graph G as the convex hull of the incidence vectors of all edge sets of cuts of G.
We also need the definition of the semimetric polytope $P^G_{SEMIMETRIC}$

Definition 3 *Let $G = (V, E)$ be an undirected graph, $\mathcal{C}$ be the set of all chordless cycles of G and $\overline{E}$ be the set of the edges of G that do not belong to a 3-edge cycle of G. For an edge*

function $x \in R^E$ *and an edge subset* $F \subseteq E$, *let* $x(F) = \sum_{e \in F} x_e$. *The semimetric polytope* $P^G_{SEMIMETRIC}$ *associated with* G *is defined by the followiong linear system:*

$$x(C \diagdown F) - x(F) \leq |F| - 1, \forall F \subseteq C \text{ with } |F| \text{ odd and } C \in \mathcal{C} \tag{3.30}$$

$$0 \leq x_e \leq 1, e \in \overline{E} \tag{3.31}$$

Proposition 8 *Let* $P^{G|k}_{BTW} \subseteq \Re^{|E|}$ *be the projection of* P^G_{BTW} *onto the variables* x_{ikj} *for a fixed middle node* k.
$P^{G|k}_{BTW}$ *is isomorphic to the cut polytope* $P^{G'}_{CUT}$ *with* $G' = G \backslash k$

Proof. If we take a generic labeling π and we consider all the variables x_{iaj} for a fixed vertex a, the corresponding vector χ^{π}_{iaj} is equal to one if and only if a is between i and j, hence a can be viewed as the cut that divides the rest of the vertices into two shores, regardless of the position of i and j, this is exactly the definition of a cut in $G \backslash \{a\}$. □

Proposition 8 is extremely useful as it gives us the opportunity to use all the knowledge about the cut polytope P^G_{CUT} in the search of valid inequality for P^G_{BTW}.

Proposition 9 (trivial lifting - $G|k$**)** *A valid inequalities for* $P^{G|k}_{BTW}$ *remains valid even if lifted to* P^G_{BTW} .

Proof. it is a direct consequence of Proposition 8. □

More precisely a really useful property is the relation with the semimetric polytope of a graph $P^G_{SEMIMETRIC}$, that is $P^G_{SEMIMETRIC} \subseteq P^G_{CUT}$.

Among all the possible inequalities we made extensive use of the following theorem by Jünger, Reinelt and Rinaldi:

Proposition 10 *Consider an LP-solution over the semimetric polytope of a connected graph* $G(V, E)$. *Let a* $P(u, v) - path$ *be a path in* G *from* u *to* v *and let* $x(S) = \sum_{e \in S} x_e$. *For each missing edge* $e = (u, v) \notin E$ *lower and upper bounds of the (artificial) LP value* $\overline{x}_e$ *are given by:*

$$\xi_l = \max\{\overline{x}(F) - \overline{x}(P \backslash F) - |F| + 1 : P(u, v) - path, F \subseteq P, |F| odd\} \tag{3.32}$$

$$\xi_u = \min\{-\overline{x}(F) + \overline{x}(P \backslash F) + |F| : P(u, v) - path, F \subseteq P, |F| even\} \tag{3.33}$$

Condition (3.33) can be stated as inequality in the following form:

$$x_{ikj} \leq -x(F) + x(P \backslash F) + |F|, \quad P(i,j)\text{-path in } G, F \subseteq P, |F| \text{ even} \tag{3.34}$$

Among all the possible inequalities (3.34) the ones obtained by fixing $|F| = 0$ play a special role in our separation procedure :

$$x_{ikj} \leq x(P), \quad P(i,j)\text{-path in } G \tag{3.35}$$

Note that any path from i to j provides a valid inequality but if we restrict to the path with no cycles it is enough to be sure to exclude dominated inequalities.
Inequalities (3.34) are valid for $P^G_{SEMIMETRIC}$ and consequently for P^G_{CUT}. Hence (3.35) apply to $P^{G|k}_{BTW}$, providing the so-called (*simple odd-cycle inequality*) the inequalities (3.35):

$$x_{ikj} \leq x(P), \quad P(i,j)\text{-path in } G|k \tag{3.36}$$

Example Let's consider K_4, all the valid *simple odd-cycle inequality* for $K_4|3$ are the following

$$x_{132} \geq x_{134} + x_{234} \tag{3.37}$$
$$x_{134} \geq x_{132} + x_{234} \tag{3.38}$$
$$x_{234} \geq x_{132} + x_{134} \tag{3.39}$$

Analogously it is possible to write the *simple odd-cycle inequality* for $K_4|1$ and $K_4|2$. □

Beside being a valid inequality for P^G_{BTW}, *simple odd-cycle inequalities* become useful in obtaining valid inequalities for P^G_{BTW} from valid inequalities for $P^{K_n}_{BTW}$.

Note that the right hand side of the inequality (3.34) consists of variables belonging only to P^G_{BTW}, hence if $(i,j) \in E(G)$ the inequality is directly valid for P^G_{BTW}. Additionally (3.34) can be used to substitute in an inequality of the form $\alpha x \geq \beta$, valid for $P^{K^n}_{BTW}$, all the variables that belong only to $P^{K^n}_{BTW}$ and not to P^G_{BTW} with positive coefficient α.

Example Suppose to work with the following graph $G = (V, E)$ with $V = \{1,2,3,4,5\}$ and $E = \{\{1,2\},\{1,4\},\{1,5\},\{2,3\},\{3,4\},\{4,5\}\}$. The following triangle inequality

$$x_{123} + x_{213} + x_{132} = 1 \tag{3.40}$$

is valid only for for $P_{BTW}^{K_5}$ but not for P_{BTW}^{G} because $\{1,3\} \notin E$ hence variable x_{123} can not be used.
It is possible to move around this limitation via adding the following odd cycle inequality:

$$x_{123} \leq x_{124} + x_{324} \tag{3.41}$$

obtaining the inequality:

$$x_{124} + x_{324} + x_{213} + x_{132} \geq 1 \tag{3.42}$$

that is valid also for P_{BTW}^{G}. □

With the last example it is more clear how simple odd cycle inequalities are used in order to work only implicitly on the complete graph, keeping the formulation smaller from the number of variable point of view.

If we substitute a missing variable x_{ikj} with a suitable path from i to j in $G|k$ we can work with a complete graph .

For a given fractional solution x^* we call $G^*|k$ the graph isomorphic to G but with every edge $\{i,j\} \in E$ with a weight associated equal to x^*_{ikj}.

Because we need to find a path that is violated as much as possible we need do find a path that provides an upper bound as low as possible , hence a good choice is to search for the shortest path in $G^*|k$ between i and j.

Notice that all the procedure described for simple odd cycle inequalities can be extended to the case with odd cycle inequalities 3.34 with $|F| \geq 1$. In Section 3.4.7 this approach is described in details.

Finally, for a better understanding of the relation between d_e-inequalities on one side and triangle and cut-polytope inequalities on the other we give the following example:

Example

Let's consider the following simple concerning valid inequalities for $P_{DIST}^{K_4}$ and $D_{BTW}^{K_4}$. In the $d_{\{i,j\}}$ space this is a valid inequality:

$$d_{ij} + d_{ik} + d_{il} \geq 4 \tag{3.43}$$

it is a 3-star inequality (3.25), centered in i. As shown in Proposition 6 it is faced defining fo $D_{DIST}^{K^4}$.

The equation (3.24) applied to the distance variable is the following:

$$d_{ij} = 1 + x_{ikj} + x_{ilj} \quad d_{ik} = 1 + x_{ijk} + x_{ilk} \quad d_{il} = 1 + x_{ikl} + x_{ijl} \tag{3.44}$$

and we obtain

$$x_{ikj} + x_{ilj} + x_{ijk} + x_{ilk} + x_{ikl} + x_{ijl} \geq 1 \tag{3.45}$$

which can be rewritten as sum of the triangle equation (3.19)

$$x_{jlk} + x_{kjl} + x_{jkl} = 1 \tag{3.46}$$

and 3 odd-cycle inequalities

$$x_{ilk} + x_{ilj} - x_{jlk} \geq 0 \quad x_{ijk} + x_{ijl} - x_{kjl} \geq 0 \quad x_{ikl} + x_{ikj} - x_{jkl} \geq 0 \tag{3.47}$$

hence (3.25) is no longer facet defining (since it is the sum of others valid inequalities). □

3.4.4 Subgraph Inequalities

Under this definition we identify all the constraints that exploit the following property:

This leads to the property that :

Proposition 11 (trivial lifting - $G \subseteq G'$) *Let G and G' be two graphs with $G \subseteq G'$. every valid inequality $\alpha x \geq \beta$ for P^G_{BTW} is still valid for $P^{G'}_{BTW}$,*

For small graphs it is possible to compute the complete description of P_{BTW} using PORTA [3], a tool for computing the outer description of a polyhedron via Fourier-Motoring elimination.

Valid inequalities for K_n

Proposition 12 *For $P^{K_3}_{BTW}$ the complete description is given by the following equation:*

$$x_{123} + x_{132} + x_{213} = 1 \tag{3.48}$$

$$x \geq 0 \tag{3.49}$$

note that triangle are inequality are valid for all complete graphs

Proposition 13 *For all permutation of* $\{1,2,3,4\}$ *these are valid facets for* $P_{BTW}^{K_4}$ *:*

$$x_{123} + x_{132} + x_{214} + x_{314} \geq 1 \tag{3.50}$$

Inequalities (3.50), together with triangle equation are the complete description of $P_{BTW}^{K_4}$

Besides the computational proof provided by PORTA, the validity of inequality (3.50) can also be proved with logic considerations. With the indexing it has the following explanation: in any possible arrangement one of the following sentence must be true:
i) 1 is either between 2 and 4 or between 3 and 4
ii) 1 is either not between 2 and 4 or not between 3 and 4

The presence of equation makes the complete description not uniquely defined.
Inequality (3.50) is the sum of the triangular equation

$$x_{123} + x_{132} + x_{213} = 1 \tag{3.51}$$

and the simple odd cycle inequality

$$x_{213} \leq x_{214} + x_{314} \tag{3.52}$$

Hence from $P_{BTW}^{K_4}$ we don't get really useful information about new inequality, qualitatively different from the cut-polytope inequalities.

Proposition 14 *For all permutations of* $\{1,2,3,4,5\}$ *these are facets for* $P_{BTW}^{K_5}$*:*

$$x_{132} + x_{154} + x_{245} + x_{315} + x_{324} \geq 1 \quad (3.53)$$

$$x_{125} + x_{132} + x_{134} + x_{135} + x_{142} + x_{154}$$
$$+x_{213} + x_{214} + x_{215} + x_{243} + x_{325} + x_{354} \geq 3 \quad (3.54)$$

$$x_{125} + x_{132} + x_{143} + x_{153} + x_{213} + x_{214}$$
$$+x_{215} + x_{234} + x_{243} + x_{345} + x_{354} + x_{425} \geq 3 \quad (3.55)$$

$$x_{132} + x_{134} + x_{135} + x_{142} + x_{143} + x_{145}$$
$$+x_{153} + x_{154} + x_{213} + x_{214} + x_{325} + x_{425} \geq 3 \quad (3.56)$$

$$x_{125} + x_{132} + x_{134} + x_{135} + x_{143} + x_{154}$$
$$+x_{213} + x_{214} + x_{215} + x_{243} + x_{254} + x_{325} + x_{345} + x_{354} + x_{425} \geq 3 \quad (3.57)$$

$$x_{124} + x_{125} + x_{132} + x_{134} + x_{143} + x_{152}$$
$$+x_{213} + x_{214} + x_{243} + x_{245} + x_{315} + x_{324} + x_{354} + x_{435} \geq 4 \quad (3.58)$$

$$x_{124} + x_{125} + x_{132} + x_{134} + x_{142} + x_{153}$$
$$+x_{213} + x_{215} + x_{243} + x_{254} + x_{435} + x_{324} + x_{345} + x_{435} \geq 4 \quad (3.59)$$

$$x_{125} + x_{132} + x_{134} + x_{135} + x_{142} + x_{215}$$
$$+x_{253} + x_{143} + x_{145} + x_{153} + x_{213} + x_{214} + x_{325} + x_{354} + x_{425} \geq 4 \quad (3.60)$$

$$x_{123} + x_{125} + x_{132} + x_{143} + x_{145} + x_{154}$$
$$+x_{214} + x_{215} + x_{253} + x_{435} + x_{435} \geq 5 \quad (3.61)$$

$$x_{123} + x_{125} + x_{132} + x_{134} + x_{142} + x_{145}$$
$$+x_{153} + x_{154} + x_{214} + x_{215} + x_{235} + x_{243} + x_{253} + x_{254} + x_{435} + x_{315} + x_{324} + x_{345}$$
$$+x_{425} + x_{435} \geq 6 \quad (3.62)$$

Inequalities (3.53)-(3.62), together with triangle equation and inequalities (3.50) are the complete description of $P_{BTW}^{K_5}$

The presence of inequalities with more than three middle indices ensure that those inequalities are not sum of triangle and odd cycle inequalities, providing new inequalities.

For $n \geq 6$ the complete description is too big for being computed explicitly by PORTA.

Valid inequalities for Stars

Proposition 15 *Let $K_{1,3}$ be a 3-star with center node 1. The complete description of $P_{BTW}^{K_{1,3}}$ is*

$$x_{123} + x_{124} + x_{132} + x_{134} + x_{142} + x_{143} \geq 1 \tag{3.63}$$
$$-x_{123} + x_{124} - x_{132} - x_{134} + x_{142} - x_{143} \geq -1 \tag{3.64}$$
$$+x_{123} - x_{124} + x_{132} - x_{134} - x_{142} - x_{143} \geq -1 \tag{3.65}$$
$$-x_{123} - x_{124} - x_{132} + x_{134} - x_{142} + x_{143} \geq -1 \tag{3.66}$$
$$-x_{123} + x_{124} + x_{132} - x_{134} - x_{142} + x_{143} \geq -1 \tag{3.67}$$
$$+x_{123} - x_{124} - x_{132} + x_{134} + x_{142} - x_{143} \geq -1 \tag{3.68}$$
$$x \geq 0 \tag{3.69}$$

From now on we will refer to inequality 3.63 as cycle inequality.

Proposition 16 *Let $K_{1,4}$ be a 4-star with center node 1. The complete description of $P_{BTW}^{K_{1,4}}$ is*

$$-x_{123} - x_{125} - x_{132} + x_{135} - x_{152} + x_{153} \geq -1 \tag{3.70}$$
$$-x_{123} + x_{125} - x_{132} - x_{135} + x_{152} - x_{153} \geq -1 \tag{3.71}$$
$$-x_{123} + x_{124} - x_{132} - x_{134} + x_{142} - x_{143} \geq -1 \tag{3.72}$$
$$-x_{124} - x_{125} - x_{142} + x_{145} - x_{152} + x_{154} \geq -1 \tag{3.73}$$
$$-x_{134} - x_{135} - x_{143} + x_{145} - x_{153} + x_{154} \geq -1 \tag{3.74}$$
$$x_{123} - x_{124} + x_{132} - x_{134} - x_{142} - x_{143} \geq -1 \tag{3.75}$$
$$-x_{134} + x_{135} - x_{143} - x_{145} + x_{153} - x_{154} \geq -1 \tag{3.76}$$
$$-x_{124} + x_{125} - x_{142} - x_{145} + x_{152} - x_{154} \geq -1 \tag{3.77}$$
$$x_{124} - x_{125} + x_{142} - x_{145} - x_{152} - x_{154} \geq -1 \tag{3.78}$$
$$x_{123} - x_{125} + x_{132} - x_{135} - x_{152} - x_{153} \geq -1 \tag{3.79}$$
$$x_{134} - x_{135} + x_{143} - x_{145} - x_{153} - x_{154} \geq -1 \tag{3.80}$$
$$-x_{123} - x_{124} - x_{132} + x_{134} - x_{142} + x_{143} \geq -1 \tag{3.81}$$

$$-x_{123} + x_{125} + x_{132} - x_{135} - x_{152} + x_{153} \geq -1 \tag{3.82}$$
$$-x_{124} + x_{125} + x_{142} - x_{145} - x_{152} + x_{154} \geq -1 \tag{3.83}$$
$$-x_{134} + x_{135} + x_{143} - x_{145} - x_{153} + x_{154} \geq -1 \tag{3.84}$$
$$x_{134} - x_{135} - x_{143} + x_{145} + x_{153} - x_{154} \geq -1 \tag{3.85}$$
$$x_{124} - x_{125} - x_{142} + x_{145} + x_{152} - x_{154} \geq -1 \tag{3.86}$$
$$x_{123} - x_{125} - x_{132} + x_{135} + x_{152} - x_{153} \geq -1 \tag{3.87}$$
$$x_{123} - x_{124} - x_{132} + x_{134} + x_{142} - x_{143} \geq -1 \tag{3.88}$$
$$-x_{123} + x_{124} + x_{132} - x_{134} - x_{142} + x_{143} \geq -1 \tag{3.89}$$

$$-x_{123} + 2x_{124} + x_{125} + x_{132} - 2x_{134} + x_{135} + x_{152} - x_{153} + 2x_{154} \geq -1 \tag{3.90}$$
$$2x_{132} - x_{134} + x_{135} - 2x_{142} + x_{143} + x_{145} + 2x_{152} + x_{153} - x_{154} \geq -1 \tag{3.91}$$
$$x_{123} - 2x_{124} + x_{125} - x_{132} + 2x_{134} + x_{135} - x_{152} + x_{153} + 2x_{154} \geq -1 \tag{3.92}$$
$$-2x_{132} + x_{134} + x_{135} + 2x_{142} - x_{143} + x_{145} + 2x_{152} - x_{153} + x_{154} \geq -1 \tag{3.93}$$
$$x_{123} + 2x_{124} - x_{125} + x_{132} + 2x_{134} - x_{135} + x_{152} + x_{153} - 2x_{154} \geq -1 \tag{3.94}$$
$$2x_{123} + x_{124} - x_{125} + x_{142} + 2x_{143} - x_{145} + x_{152} - 2x_{153} + x_{154} \geq -1 \tag{3.95}$$
$$x_{123} - x_{124} + 2x_{125} + x_{132} - x_{134} + 2x_{135} + x_{142} + x_{143} - 2x_{145} \geq -1 \tag{3.96}$$
$$x_{123} + x_{124} - 2x_{125} - x_{132} + x_{134} + 2x_{135} - x_{142} + x_{143} + 2x_{145} \geq -1 \tag{3.97}$$
$$-2x_{123} + x_{124} + x_{125} - x_{142} + 2x_{143} + x_{145} - x_{152} + 2x_{153} + x_{154} \geq -1 \tag{3.98}$$
$$-x_{123} + x_{124} + 2x_{125} + x_{132} + x_{134} - 2x_{135} + x_{142} - x_{143} + 2x_{145} \geq -1 \tag{3.99}$$
$$2x_{123} - x_{124} + x_{125} + x_{142} - 2x_{143} + x_{145} + x_{152} + 2x_{153} - x_{154} \geq -1 \tag{3.100}$$
$$2x_{132} + x_{134} - x_{135} + 2x_{142} + x_{143} - x_{145} - 2x_{152} + x_{153} + x_{154} \geq -1 \tag{3.101}$$

$$-x_{124} - x_{125} - 2x_{132} + 2x_{134} + 2x_{135} + x_{142} + x_{145} + x_{152} + x_{154} \geq -1 \quad (3.102)$$

$$x_{123} + x_{125} - x_{132} - x_{135} + 2x_{142} - 2x_{143} + 2x_{145} + x_{152} + x_{153} \geq -1 \quad (3.103)$$

$$-2x_{123} + 2x_{124} + 2x_{125} - x_{134} - x_{135} + x_{143} + x_{145} + x_{153} + x_{154} \geq -1 \quad (3.104)$$

$$-x_{123} - x_{125} + x_{132} + x_{135} - 2x_{142} + 2x_{143} + 2x_{145} + x_{152} + x_{153} \geq -1 \quad (3.105)$$

$$2x_{123} - 2x_{124} + 2x_{125} + x_{134} + x_{135} - x_{143} - x_{145} + x_{153} + x_{154} \geq -1 \quad (3.106)$$

$$x_{123} + x_{125} + x_{132} + x_{135} + 2x_{142} + 2x_{143} - 2x_{145} - x_{152} - x_{153} \geq -1 \quad (3.107)$$

$$x_{123} + x_{124} + x_{132} + x_{134} - x_{142} - x_{143} + 2x_{152} + 2x_{153} - 2x_{154} \geq -1 \quad (3.108)$$

$$x_{124} + x_{125} + 2x_{132} - 2x_{134} + 2x_{135} - x_{142} - x_{145} + x_{152} + x_{154} \geq -1 \quad (3.109)$$

$$2x_{123} + 2x_{124} - 2x_{125} + x_{134} + x_{135} + x_{143} + x_{145} - x_{153} - x_{154} \geq -1 \quad (3.110)$$

$$x_{123} + x_{124} - x_{132} - x_{134} + x_{142} + x_{143} + 2x_{152} - 2x_{153} + 2x_{154} \geq -1 \quad (3.111)$$

$$x_{124} + x_{125} + 2x_{132} + 2x_{134} - 2x_{135} + x_{142} + x_{145} - x_{152} - x_{154} \geq -1 \quad (3.112)$$

$$-x_{123} - x_{124} + x_{132} + x_{134} + x_{142} + x_{143} - 2x_{152} + 2x_{153} + 2x_{154} \geq -1 \quad (3.113)$$

$$-x_{123} + 2x_{124} - x_{125} - x_{132} - x_{134} + 2x_{135} + x_{143} - x_{145} + x_{152} - x_{154} \geq -2 \quad (3.114)$$

$$-x_{123} + x_{125} - x_{132} + x_{134} + 2x_{142} - x_{143} - x_{145} - x_{152} + 2x_{153} - x_{154} \geq -2 \quad (3.115)$$

$$2x_{123} - x_{124} - x_{125} - x_{134} + x_{135} + x_{142} - x_{143} - x_{152} - x_{153} + 2x_{154} \geq -2 \quad (3.116)$$

$$x_{123} - x_{125} - x_{132} - x_{134} + 2x_{135} + 2x_{142} - x_{143} - x_{145} - x_{152} + x_{154} \geq -2 \quad (3.117)$$

$$x_{123} - x_{124} - x_{132} + 2x_{134} - x_{135} - x_{142} + x_{145} + 2x_{152} - x_{153} - x_{154} \geq -2 \quad (3.118)$$

$$x_{124} - x_{125} + 2x_{132} - x_{134} - x_{135} - x_{142} - x_{143} + 2x_{145} - x_{152} + x_{153} \geq -2 \quad (3.119)$$

$$-x_{123} + x_{124} - x_{132} + x_{135} - x_{142} + 2x_{143} - x_{145} + 2x_{152} - x_{153} - x_{154} \geq -2 \quad (3.120)$$

$$2x_{123} - x_{124} - x_{125} + x_{134} - x_{135} - x_{142} - x_{143} + 2x_{145} + x_{152} - x_{153} \geq -2 \quad (3.121)$$

$$-x_{123} - x_{124} + 2x_{125} - x_{132} + 2x_{134} - x_{135} + x_{142} - x_{145} + x_{153} - x_{154} \geq -2 \quad (3.122)$$

$$-x_{123} - x_{124} + 2x_{125} + x_{132} - x_{135} - x_{142} + 2x_{143} - x_{145} - x_{153} + x_{154} \geq -2 \quad (3.123)$$

$$-x_{124} + x_{125} + 2x_{132} - x_{134} - x_{135} - x_{142} + x_{143} - x_{152} - x_{153} + 2x_{154} \geq -2 \quad (3.124)$$

$$-x_{123} + 2x_{124} - x_{125} + x_{132} - x_{134} - x_{143} + x_{145} - x_{152} + 2x_{153} - x_{154} \geq -2 \quad (3.125)$$

$$+x_{134} + x_{135} + x_{143} + x_{145} + x_{153} + x_{154} \geq 1 \quad (3.126)$$

$$x_{123} + x_{124} + x_{132} + x_{134} + x_{142} + x_{143} \geq 1 \quad (3.127)$$

$$x_{123} + x_{125} + x_{132} + x_{135} + x_{152} + x_{153} \geq 1 \quad (3.128)$$

$$x_{124} + x_{125} + x_{142} + x_{145} + x_{152} + x_{154} \geq 1 \quad (3.129)$$

$$x \geq 0 \quad (3.130)$$

Inequalities (3.126)-(3.126) are analogous to the inequalities (3.63).
As mentioned in Section (3.4.2), inequality (3.63) can be viewed as sum of the following d_e constraints and odd cycle inequalities applied to $P_{BTW}^{K_3}$:

$$x_{234} + x_{243} + x_{324} \geq 1 \tag{3.131}$$

$$x_{132} + x_{134} - x_{234} \geq 0 \tag{3.132}$$

$$x_{142} + x_{143} - x_{243} \geq 0 \tag{3.133}$$

$$x_{123} + x_{124} - x_{324} \geq 0 \tag{3.134}$$

With the example provided we have directly proved the following statement:

Proposition 17 *trivial lifting (11) does not keep the facing inducing property.*

As we will show in Section 3.5 inequalities (3.63) are still useful even if not faced defining in general after trivial lifting. It is still an open question for what classes of graph inequalities (3.63) after trivial lifting are still faced defining.

3.4.5 Solution Scheme

All the polyhedral information has been embedded in a Branch and Cut scheme based on the betweeness variables in the sparse formulation.

Starting point of the process is the LP model with the objective function 3.14 and the non negativity constraints.
A each node of the B&B tree we apply three steps:
i) solving the LP model
ii) feasibility test (if integer solution)
ii)separation routine

Because we don't have a complete formulation for a generic P_{BTW}^{G} we need to test if an integer solution corresponds to a feasible arrangement. We use a techniques borrowed from the Consecutive Ones Problem to test if a set of integer variable is feasible (more details about this procedure are explained in Section 3.4.6). If the current integer solution is found to be unfeasible we add the inequality

$$\sum_{i \in J} x_i - \sum_{i \in V \setminus J} x_i \leq |J| - 1 \tag{3.135}$$

with $J = \{i \in V \mid x_i = 1$ in the current solution$\}$ If the integer solution is feasible we stop the procedure, otherwise we resolve the LP model.
If the solution is fractional we run the separation procedure, adding violated inequalities explained in section in Sections 3.4.1, 3.4.2, 3.4.3 and 3.4.4.
This procedure is repeatedly executed as long as the increase in the linear programming objective function value remains greater than a certain ϵ fixed a priori.
As branching policy we use a best first approach branching on the most fractional betweeness variable.

3.4.6 Feasibility Test

Notice that we don't know a complete ILP formulation for P^G_{BTW}. Hence if an integer solution is found it is not clear if it is feasible or not.
For doing so we use a generalization of the linear arrangemente problem, the Consecutive Ones Problem [19]. The idea is transforming the lAP in a Consecutive Ones problem. Here, a 0/1 matrix M is given, with the task to permute the columns, such that in each row, the ones appear consecutively.
In our case, M will be a matrix with rows of dimension n. For every edge ik â E we look at all the vertices in $V_{ik} = j|x_{ijk} = 1$. If $V_{ik} = \emptyset$, we know that i and k should be adjacent in any arrangement, so we insert a row r, with $r_i = r_k = 1$ and 0 otherwise. Else, all the vertices in V_{ik} are adjacent to each other as well as i and k, with the latter ones lying at the margins. So we add two rows, one with ones at index i and $j \in V_{ik}$ and another with ones at index k and $j \in V_{ik}$. Now, there is a permutation to put M in consecutive ones form, if and only if there is an arrangement that is described by x. To solve the Consecutive Ones Problem, we use the P Q-tree algorithm that has linear run time and outputs all feasible permutations. An implementation is pro- vided by [CITE].

3.4.7 Separation Routine

Let $\overline{x}$ be the current fractional point in the betweeness space (i.e. $\overline{x} \in \Re^{|E|} \cap [0,1]^{|E|}$).

The Separation routine consists of the following phases:
i) preprocessing and odd-cycle separation
ii) triangle inequalities separation
iii) stars inequalities separation
iv) other separation

Each separation step adds a set of violated inequalities to the LP model. An upper bound n_{cut} on the total number of cuts to be added is imposed, in order to limit the LP size.

Additionally only if in the odd-cycle separation no cuts is added the procedure continues to the further steps, this is done in order to speed up the first rounds of separation, mainly consisting on adding odd-cycle inequalities.

Preprocessing

For every vertex $a \in V$ we construct a weighted graph $G_a = G$ with edges weights $\overline{x}_{iaj}$. Let $\overline{y}$ be a vector of size $|V|^3$. For every generic vertex a let $\overline{y}_{iaj}$ be equal to the value of the shortest path between i and j in the graph G_a.
For every triple i, j, k we store the corresponding vale y_{ikj} and the current shortest path in G_k.
During the computation of the shortest path if we find a triple of nodes $\{i', k, j'\}$ for which the edge $(i, j) \in G$ and where the value of the shortest path in G_k is lower than $\overline{x}_{ikj}$ then we have found a violated odd cycle inequality 3.34 and we add it directly to the LP. Notice that in this way all the violated odd-cycle inequalities are added to the LP.
$\overline{y}$ can be viewed as betweeness variables of an auxiliary complete graph, over his corresponding polytope we run all the next separation procedures. no violated odd-cycle inequalities is the necessary condition for start separating other inequalities, this is due to the fact the we want to have a $\overline{y}$ space consistent in order to obtain effective inequalities.

Cuts Separation

As first step we separate triangle equation for $\overline{y}$. We randomly enumerate all the triplets of nodes of G with an probability uniformly distributed. for every triple i, j, k. If we find a triple with

$$\overline{y}_{ikj} + \overline{y}_{ijk} + \overline{y}_{jig} < 1 \tag{3.136}$$

we add to the LP the equation

$$\overline{y}_{ikj} + \overline{y}_{ijk} + \overline{y}_{jik} = 1 \tag{3.137}$$

in doing so we substitute every $\overline{y}_{ikj}$ with the corresponding shortest path in G_k. Note that with this substitution the constrains is no longer an equation but becomes an inequality.
Moreover, we substitute a $\overline{y}_{ikj}$ variable if and only if the corresponding x_{ikj} does not exist, this is ensured by the fact that we separate on the $\overline{y}$ only if no odd-cycle inequality is violated, hence if an edge $e \equiv \{i, j\}$ exists, in any G_k the shortest path from i to j will be the edge

$\{i, j\}$ if we want the odd cycle inequality 3.36 to be valid.

Now it becomes more clear why we decided to separate on the $\overline{y}$ variable only if no odd cycle inequality is found. Instead of implicitly including them in several constraints we prefer to add them just once and use the corresponding x variable in the other inequalities.

After the triangle equation we start the star separation.
In this procedure we separate the inequalities (3.26) in the $\overline{y}$ space:

$$\sum_{j \in N(i)} \sum_{k \in N(i), k \neq j} \overline{y}_{ikj} \geq \lfloor (|S| - 1)^2 / 4 \rfloor \tag{3.138}$$

for a given center node i and neighborhood $N(i)$.
As mentioned in previous section 3-star inequalities can be seen as sum of triangle equation and odd cycle inequalities. Moreover also the complete description of $P_{BTW}^{K_{1,4}}$ does not contains inequalities (3.26). Hence we decide to focus on n-stars with $n \geq 5$.

As first search we enumerate all the stars with size greater than 5 that are subsets of the original graph G and test if the corresponding inequality (3.138) is violated. If a violation is found we substitute the $\overline{y}$ with the corresponding x variable and we add the inequality to the LP model.

As second step we execute the following procedure for every node i in the graph:
i) search among all the nodes $j \in V \setminus i$ the four index j, k, l, m that provides the lowest value

$$s_1 = \overline{y}_{ijk} + \overline{y}_{ikj} + \overline{y}_{ijl} + \overline{y}_{ilj} + \overline{y}_{ijm} + \overline{y}_{imj} + \overline{y}_{ikl} + \overline{y}_{ilk} + \overline{y}_{ikm} + \overline{y}_{imk} + \overline{y}_{ilm} + \overline{y}_{iml} \tag{3.139}$$

ii) once the first 4 nodes are fixed we test all the remaining nodes and check if the corresponding 5-star inequality 3.138 is violated. iii) keeping the same 4 initial nodes we do the same for n-stars with n strictly greaterr than 5 by enumerating all the possible pair, triples that together with the original node may provide a violated inequality.

Notice that the three separation procedure explained are executed starting from the fastest and that in any moment that we find n_{cut} cuts we stop the procedure and we solve again the LP.
As we will show in the computation part this hierarchy is chosen also in order to give privilege to the most effective classes.

3.5 Computational Results

We have implemented all the the algorithms described in C++ language and we have executed the tests on a PC XXX, Y GB Ram, ZZ GHz. as LP solver we have used CPLEX 8 with its standard settings.
As Branch and Cut framework we used the ABACUS software. More details about this framework can be found in [16].

As test instances we use the well-known graph drawing instances (gd) from the Petit test set [15], available at [17].
We also considered the bandwidth instances addressed in [14], available at [18].

Tables 3.1, 3.2 and 3.3 are related to the gd instances. we fixed 6 hours as time limit and we ran the code with two different setting for the LB computation:
i) *LB I*
with this setting the separation procedure explained in Section 3.4.7 is executed entirely; at each iteration we add at most $n_{cut} = 2700$ violated inequalities.
ii) *LB II*
with this option the separation routine does not include the star separation explained in the second part of Section 3.4.7; in other word only odd cycles and stars are generated; also in this case we keep $n_{cut} = 2700$

In Tables 3.1 and 3.2 we report our results related to the graph drawing instances. *time tot* and *lp sol time* indicate the total computational time and the time used in computing the LP model. *tot BB nodes* indicates how many branch and bound nodes are computed, *tot lp* indicates the number of times that the LP model is solved. Finally the best lower bound obtained is indicated in *best LB*.

In Table 3.3 a comparison of *LB I* and *LB II* is reported, compared with the results obtained by Caprara et al. in [2].

Bandwidth instances results are showed in table 3.4, i this case all the results a related to *LB I* and with a time limit of 2 hours.

For the first time we are able to solve the instances gd95c and gd96c in less than one day of computation, additionally we have provided the best LB so far for gd96b.

In the large majority of the instances tested our algorithm spends all the time in the root

instance	time tot	lp sol time	tot BB nodes	tot lp	best LB
gd95c	0:01:01	0:00:54	1	138	506
gd96b	6:00:00	0:49:58	1	2176	1395
gd96c	0:06:25	0:05:54	1	281	519
gd96d	6:00:00	5:18:15	1	594	1953

Table 3.1: gd instances - all separation procedures - *LB I*

instance	time tot	lp sol time	tot BB nodes	tot lp	best LB
gd95c	0:01:15	0:01:07	1	151	506
gd96b	6:00:00	5:47:26	35	5466	1199
gd96c	0:03:49	0.03:31	1	303	519
gd96d	6:00:00	0:03:57	1	236	1314

Table 3.2: gd instances - triangle equation and odd cycle inequalities - *LB II*

nodes, even if the instance is solved to optimality the cuts provided are sufficient in order to solve it.

Moreover, the comparison between *LB I* and *LB II* shows that *LB II* is enough for solving the instances gd95c and gd96c. Computational results shows how triangle and odd cycle inequalities inequalities are the breakthrough aspect of our method. Star inequalities comes into play when we want to increase the bound for big instances.
This results show also the validity of the hierarchical choice between the separation method.

Finally, the most interesting instance to look if we want to improve our method are the following: i) instances not solved in the rood node: bcspwr01 , bcspwr01 and nos4.mtx.rnd
ii) instances where a consistent quantity of time is spent in the separation: bcspwr04, dwt__245

for this instances the cuts provided are not enough and we need to branch (class i)) or the separationg procedure is too slow (class ii)).

	time *LB I*	*LB I*	time *LB II*	*LB II*	time LB Cap et al.	LB Cap et al.
gd95c	0:01:15	506	0:01:01	506	0:01:53	443
gd96b	6:00:00	1199	6:00:00	1395	0:08:13	1281
gd96c	0:03:49	519	0:06:25	519	0:03:38	402
gd96d	6:00:00	1314	6:00:00	1953	0:27:49	2021

Table 3.3: gd instances - *LB I* vs. *LB II* vs: previous bound

	time tot	lp sol time	tot BB nodes	tot lp	best LB	opt	best LB Cap et al.
bcspwr01.graph	0:00:08	00:00:04	23	42	106	y	91
bcspwr02.graph	0:00:06	00:00:05	1	22	161	y	144
bcspwr03.graph	0:31:16	00:25:25	1	349	662	y	588
bcspwr04.graph	2:00:00	01:17:18	1	224	2581	n	3700
can___24.graph	0:00:02	0:00:02	1	4	210	y	203
can___61.graph	0:17:19	0:17:04	1	138	1137	y	1119
can___62.graph	0:00:30	0:00:25	1	57	210	y	187
can___73.graph	2:00:00	1:59:00	1	296	1000	n	971
can___96.graph	2:00:00	0:01:11	1	228	1779	n	2105
can__144.graph	2:00:00	01:58:17	1	261	2413	n	2304
can__161.graph	2:00:00	01:57:48	1	227	2805	n	5657
can__187.graph	2:00:00	01:50:00	1	323	2664	n	3827
can__229.graph	2:00:00	01:42:42	1	214	3179	n	7461
dwt___59.graph	0:00:42	0:00:36	1	77	289	y	258
dwt___66.graph	0:00:01	0:00:01	1	8	192	y	192
cdwt___72.graph	0:01:47	0:00:25	27	71	167	y	150
dwt___87.graph	2:00:00	1:45:00	1	189	2869	n	897
dwt__162.graph	2:00:00	1:54:53	1	320	2052	n	2032
dwt__198.graph	2:00:00	1:56:33	1	333	2526	n	x
dwt__209.graph	2:00:00	1:44:53	1	229	3355	n	5905
dwt__221.graph	2:00:00	1:41:20	1	202	2617	n	3603
dwt__245.graph	2:00:00	1:04:49	1	236	2393	n	3422
lund_a.mtx.rnd	2:00:00	1:57:00	1	150	6311	n	10772
lund_b.mtx.rnd	2:00:00	1:58:25	1	156	6327	n	10712
nos4.mtx.rnd	1:13:57	1:08:22	21	298	1031	y	976
steam3.mtx.rnd	0:34:07	0:31:59	1	548	1416	y	1406

Table 3.4: bandwidth instances - all separation procedures

Bibliography

[1] W. Liu & A. Vannelli (1995) Generating lower bounds for the linear arrangement problem. Discr. Appl. Math., 59, 137-151.

[2] A. Caprara, A.N. Letchford & J.J. Salazar Gonzales Decorous Lower Bounds for Minimum Linear Arrangement To be published

[3] T. Christof. Low-dimensional 0/1-polytopes and branch-and-cut in combina- torial optimization. PhD thesis, Dissertation, Uni. Heidelberg, Shaker Verlag, Aachen, 1997.

[4] Andréasson, I (1994). Vehicle Distribution in Large Personal Rapid Transit Systems. Transportation Research Record, No. 1451, pp 95-99, Transportation Research Board, Washington, D.C.

[5] McCormick, G. P. (1976) . Computability of global solutions to factorable nonconvex programs Part I Convex underestimating problems. Mathematical Programming, 10:147â175.

[6] Anderson, J.E. et al. (1998). Special issue: emergin systems for pubblic transportation. Journal of Advanced Transportation, 32, 1, 1-128.

[7] Kaspi, M. and Tanchoco, J.M.A. (1990). Optimal flow path design of unidirectional AGV systems. International Journal of Production Research, 28, 1023-1030.

[8] Chung, F.R.K., Garey, M.R., Tarjan, R.E. (1985). Strongly connected orientations of mixed multigraphs. Networks, 15, 477-484.

[9] Benders, J.F. (1962). Partitioning procedures for solving mixed variables programming problems. Num. Math, 4, 238-252.

[10] Magnanti, T.L., Mireault, P., Wong, R.T.(1986) Tailoring Benders decomposition for uncapacitated network design. Mathematical Programming Study, 26, 112-154

[11] Chvatal, V., Thomassen, C. (1978). Distances in orientations of graphs. Journal of Combinatorial Theory Ser. B, 24, 61-75.

[12] Burkard, R.E., Feldbacher K., Klinz, B., Woeginger, G.J.(1999) Minimum-cost strong network orientation problems: Classification, complexity, and algorithms NETWORKS Volume: 33 Issue: 1 Pages: 57-70 .

[13] Khanna, S., Naor, J.S. , Shepherd, F.,2000 Directed network design with orientation constraints PROCEEDINGS OF THE ELEVENTH ANNUAL ACM-SIAM SYMPOSIUM ON DISCRETE ALGORITHMS : 663 2000

[14] Caprara , A. , Salazar-González , J.J., 2005 Laying Out Sparse Graphs with Provably Minimum Bandwidth NFORMS JOURNAL ON COMPUTING Vol. 17, No. 3, pp. 356-373

[15] Díaz, J. Petit, J. , Serna, M. , 2002 A Survey of Graph Layout Problems ACM Computing Surveys Volume 34 , Issue 3 Pages: 313 - 356

[16] Jünger, M. , Thienel, S. ,2000 The ABACUS system for branch-and-cut-and-price algorithms in integer programming and combinatorial optimization. Software Practice and Experience, 30:1325â1349

[17] http://www.lsi.upc.edu/â$\frac{1}{4}$jpetit/MinLA/Experiments/.

[18] http://www.informs.org/site/IJOC/article.php?id=42.

[19] Oswald, M., 2003 Weighted Consecutive Ones Problems. PhD thesis, Ruprecht- Karls-Universität Heidelberg,