Conjoint design of railway lines and frequency setting under semi-congested scenarios

Author:

Francisco López Ramos (*)

Supervisor:

Esteve Codina Sancho

Department of Statistics and Operations Research
Universitat Politècnica de Catalunya - Barcelona Tech

Monday 16th December, 2013

Thesis appointed to research projects TRA2008-06782-C02-02 and TRA2011-27791-C03-01/02

(*) In receipt of the Spanish government grants: BES-2009-026743, EEBB-2011-43487 and EEBB-I-2012-03711

Abstract

This thesis develops mathematical programming models which integrate network design (ND) and line frequency setting (LFS) phases. These appear in transport planning studies that extend an existing urban public transportation system ($UPTS$) and are suitable for underground and rapid transit systems. The ND phase extends the working $UPTS$, taking as inputs the locations of candidate stretches and stations on the new lines, as well as construction costs which cannot exceed the available infrastructure budget. Regarding the LFS phase, frequencies and vehicles are assigned to functioning and newly built lines, providing that they do not exceed link frequency capacities, vehicle acquisition budgets, and time horizons. The developed models take into account the type of service patterns that may operate on the lines of the transport system. They include local services, where vehicles halt at every node in the line, and express services, in which vehicles halt at only a subset of nodes in the line. A passenger assignment model solves the ND and LFS phases at the same time a global optimum under inelastic demand.

The combined model has two variants: one which deals with inelastic demand and another which faces elasticities in demand. The latter originates from changes in the modal choice proportions of travelers and may result from modifications in the public transport system. The former does not take into account competition among several modes of transportation and it is formulated as a mixed-integer linear programming problem. In contrast, the latter allows passengers to travel via two modes of transportation: public transport and private car. It is formulated as a mixed-integer linear bilevel programming problem ($MILBP$) with discrete variables only in the upper level. In both models, a complementary network is used to model transfers among lines and to reach the passenger's origin and/or destination nodes when the constructed $UPTS$ does not cover them.

The model with inelastic demand is initially solved by means of the commercial solver *CPLEX* under three different mathematical formulations for the ND phase. The first two are exact approaches based on extensions of Traveling Salesman Problem formulations for dynamic and static treatment of the line's subtours, whereas the last one is an approximation inspired by constrained k-shortest path algorithms. In order to deal with large-sized networks, a quasi-exact solution framework is employed. It consists of three solving blocks: the corridor generation algorithm (CGA), the line splitting algorithm (LSA), and the specialized Benders decomposition (SBD). The LSA and CGA methods are heuristic techniques that allow skipping some of the non-polynomial properties, which are the more difficult properties of the mathematical programming problem. They are related to the number of lines under construction and the number of feasible corridors (line segments) that can be generated. As for the SBD, it is an exact method that splits the original mathematical programming problem into a series of resolutions, composed of two mathematical problems which are easier to solve. Regarding the elastic demand variant, it is solved under the same framework as the specialized Benders decomposition adaptation for solving $MILBP$, which results from this variant formulation.

The inelastic demand variant is applied to two test cases based on underground network models for the cities of Seville and Santiago de Chile. The test cases were built for academic purposes. Origin destination trip matrices, construction costs and other parameters required by the models have been set to likely values

using maps and published studies. They do not come from any data collection study or survey. The purpose of these networks is to test the models and algorithms on realistic scenarios, as well as to show their potentialities. Reported results show that the quasi-exact approach is comparable to approximate techniques in terms of performance. Regarding the elastic demand variant, the model is more complex and can be applied only to smaller networks.

Finally, some further lines of research for both modeling and algorithmic issues are discussed. Furthermore, it is stressed that the current mathematical structure of both models, as well as techniques for solving them, can be preserved when applying any of these lines of research.

Keywords: Public transportation, network design, line planning, integrating, express service design, elastic demand, constrained k-shortest paths, Benders' Decomposition, Pareto-optimality, mixed-integer linear programming, mixed-integer linear bilevel programming.

Acknowledgements

The author expresses gratitude to the support of projects TRA2008-06782-C02-01/02 and TRA2011-27791-C03-01/02 as well as grants BES-2009-026743, EEBB-2011-43487 and EEBB-I-2012-03711 from the Spanish Government. Beyond economic support, the author would like to thank: his PhD supervisor, Dr. Esteve Codina Sancho of the Department of Statistics and Operational Research at *Universitat Politècnica de Catalunya - Barcelona Tech*, for his continuing help; Dr. Angel Marín Gracia, of the Department of Applied Mathematics and Statistics at the *Universidad Politécnica de Madrid*, for his expertise in applying Benders Decomposition; and Dr. Roberto Cominetti, of the Department of Industrial Engineering at the *Universidad de Chile*, for his contribution to improving the mathematical formulation of the model. Last but not least, the author would also like to thank his parents for encouraging him to pursue this thesis. Without their help, this work could not have been finished.

Contents

List of Tables

List of Figures

Mathematical Notation

Definition of parameters

a_r^w Origin coordinate of the line which approximates the entropy function (7.32) - (7.33) for the o-d pair w in the interval r.

b_r^w Slope of the line which approximates the entropy function (7.32)-(7.33) for the o-d pair w in the interval r.

B Vehicle fleet capacity expressed as the number of available vehicles with no acquisition cost.

c_i^c Cost of building a stop at node $i \in N_{TP}^N$.

c_i^m Cost of stop maintenance at node $i \in N_{TP}^N$.

c_{ij}^c Cost of building a stretch at link $(i,j) \in A_{TP}^N$.

c_{ij}^m Cost of stretch maintenance at link $(i,j) \in A_{TP}^N$.

c_i^f Cost of operation at stop $i \in N_{TP}$.

c_{ij}^f Cost of operation on stretch $(i,j) \in A_{TP}$.

c_b^s Cost of assigning a vehicle to any line.

c_b^a Cost of acquiring a new vehicle.

$\bar{c}_{net}$ Available budget for infrastructure.

$\bar{c}_{veh}$ Available budget for acquiring new vehicles.

D^c A 0-1 column vector holding the active stretches for corridor c.

d_{ij}^{TP} In-vehicle travel distance from node i to node j in the public transit network.

d_w^{PRI} Average traveled distance for a user belonging to the o-d pair w that uses the private mode in the elastic demand model.

E^c A 0-1 column vector holding the active nodes for corridor c.

$\overline{f}$ Maximum number of vehicles per unit of time that can be reached.

$\bar{f}_i$ Capacity of the node $i \in N_{TP}^N$, defined as the maximum number of vehicles that can go through per time unit.

$\bar{f}_{ij}$ Capacity of the link $(i,j) \in A_{TP}$, defined as the maximum number of vehicles that can go through per time unit.

g_w Number of passengers per unit of time belonging to o-d demand pair w.

g_p Number of passengers per unit of time belonging to all o-d demand pairs with origin at p.

q Vehicle capacity expressed as the maximum number of users that a train can hold.

$\hat{h}$ Planning time horizon.

K^r Maximum number of k-shortest rectilinear corridors which can be found.

K^c Maximum number of k-shortest circular corridors which can be found.

T In-vehicle travel time matrix for every arc linking two pair of nodes $i, j \in N_{TP}^N$.

$\tilde{T}$ A copy of T which may be modified in order to compute a new feasible detour.

t_{ij}^{TP} In-vehicle travel time from node i to node j in the public transit network.

t_{ij}^{COM} Walking travel time from node i to node j in the complementary transit network.

t_l Layover time of the line.

t_s Passenger service time at any working or constructed station in the public transit network.

t_a Time per passenger spent on boarding a vehicle halting at a service node.

t_x Time per passenger spent on waiting in a vehicle halting at a service node.

t_y Time per passenger spent on alighting from a vehicle halting at a service node.

t_w^m Average time of a user belonging to the o-d pair w that wants to use the mode m in the elastic demand model.

$\overline{x}$ Maximum allowable number of stretches in each constructed line.

α Objective weight factor whose value must be within the interval $(0, 1)$.

θ Economic cost of a unit of time for any OD pair when using the inelastic demand model.

θ^w Economic cost of a unit of time for the OD pair w when using the elastic demand model.

β_w^{PRI} Parking cost at origin $p(w)$ and destination $q(w)$ of a user belonging to the o-d pair w that uses the private mode in the elastic demand model.

β_w^{TP} Fare cost of a user belonging to the o-d pair w that uses the public transportation mode.

γ_w A factor that weighs the average traveled distance for a user belonging to the o-d pair w that uses the private mode in the elastic demand model.

Φ_{max}^r A parameter characterizing the maximum length of a detour $P_{i-j}^k \subset P^k$ with $k > 1$ and P^k rectilinear relative to the shortest path from i to j (P_{ij}^1).

Φ_{max}^c A parameter characterizing the maximum length of a detour $P_{i-j}^k \subset P^k$ with $k > 1$ and P^k circular relative to the shortest path from i to j (P_{ij}^1).

Δ_{min}^r A parameter characterizing the minimum length of a detour $P_{i-j}^k \subset P^k$ with $k > 1$ and P^k rectilinear relative to the time length of path P^k from i to j ($t(P_{ij}^1)$).

Δ_{min}^c A parameter characterizing the minimum length of a detour $P_{i-j}^k \subset P^k$ with $k > 1$ and P^k circular relative to the time length of path P^k from i to j ($t(P_{ij}^1)$).

Definition of indexes

c A subindex denoting the identifier of a corridor.

i Subindex denoting the identifier of a node.

k A supra index denoting the iteration of the algorithm.

m Supra index denoting the identifier of a transportation mode.

l A supra index denoting the identifier of a line.

p A supra index denoting the identifier of a demand origin.

q A supra index denoting the identifier of a demand destination.

r Subindex denoting the identifier of an interval of the entropy discretization function in the elastic demand model.

s Subindex denoting the identifier of the source node of a path.

t Subindex denoting the identifier of the destination/terminal node of a path.

w A supra index denoting the identifier of an origin-destination pair of demand.

(i, j) A double index denoting the identifier of a link.

$a(i)$ A subindex denoting the identifier of a boarding link related to identifier node i.

$y(i)$ A subindex denoting the identifier of an alighting link related to identifier node i.

$x(i)$ A subindex denoting the identifier of a remaining link related to identifier node i.

$pred(i)$ The node which immediately precedes node i on a certain shortest path.

$pred(i, j)$ The node which immediately precedes node j and is on the shortest path starting at i.

Definition of functions

$C(P^k)$ A function which maps the ordered node set P^k with its corresponding construction cost.

h_k^n A function which refers to an equation which made up the right hand side of the Benders cuts (7.85) for n = 1 or (7.86) for n = 2.

$NS(P^k, P_\cdot)$ A function which sets the role of the node $P_\cdot$ in path P^k as service node.

$t(P^k)$ A function which maps the ordered node set P^k with its total time.

U_w^m Value of the utility function of a user belonging to the o-d pair w that wants to use the mode m in the elastic demand model.

Definition of sets

A Set of links contained in the global network.

A_i The arc adjacency list of node i used in Dijkstra's shortest path algorithm.

A_{TP} Set of links belonging to the public transport network.

A_{TP}^{IN} Set of in-vehicle links belonging to the public transport network.

A_{TP}^{E} Set of stretches contained in the working lines.

$A_{TP}^{E}(l)$ Set of stretches contained in the working line l.

A_{TP}^{N} Set of stretches which can be used by the new lines.

$A_{TP}(i)$ Set of stretches in the public transport network containing stop i.

$A_{xya}^{N}(i)$ Set of links related to in-station passenger flows which are adjacent to stop i.

$A_{x}^{N}(i)$ Set of links related to remaining in-station passenger flows which are adjacent to stop i.

$A_{x}^{+}(i)$ Set of links representing the remaining passenger flows in-vehicle at station that goes out of node i.

$A_{x}^{-}(i)$ Set of links representing the remaining passenger flows in-vehicle at station that goes into node i.

A_{COM} Set of links belonging to the complementary transit mode network.

$A_{COM}(i)$ Set of links belonging to the complementary transit mode network containing node i.

B A set containing the current computed feasible candidate line.

D_p Set of o-d demand pairs which originate at node p.

$E_{TP}(i)$ Set of stops emerging from stop i belonging to the public transport network.

$I_{TP}(i)$ Set of stops incident to stop i belonging to the public transport network.

G A graph containing the global network, i.e., all subgraphs representing different transport networks.

G_{TP} A subgraph representing the public transportation network.

G_{COM} A subgraph representing the complementary network which allows passengers to transfer from line to line or from one mode to another on foot.

G_{PRI} A subgraph representing the private network where passengers travel in their private vehicles.

L Set of lines belonging to the public transport network.

L_{ij} Set of lines that contain stretch (i, j).

L_i Set of lines that contain node i.

L^E Set of working lines already in operation contained in the public transport network.

L_R^E Set of working rectilinear lines already in operation contained in the public transport network.

L_C^E Set of working circular lines already in operation contained in the public transport network.

L^N Set of new lines which can be constructed in the public transport network.

$l_{a(i)}$ Set of adjacent links to public transportation node $i \in N_{TP}$ which are related to passenger flows boarding a vehicle.

$l_{y(i)}$ Set of adjacent links to public transportation node $i \in N_{TP}$ which are related to passenger flows alighting from a vehicle.

M Set of transportation modes under consideration (i.e., $M = \{TP, PRI\}$).

N Set of nodes contained in the global network.

N_{TP} Set of nodes which can be used as stops by the public transport network.

N_{TP}^{S+} Set of incoming station nodes belonging to the public transport network.

N_{TP}^{S-} Set of outgoing station nodes belonging to the public transport network.

$N_{TP}^{P}(l)$ Set of passing points contained in the working line l.

N_{TP}^{N} Set of stops which can be used by the new lines.

$N_{TP}(i)$ Set of stretches adjacent to stop i belonging to the public transport network.

O Set of origins of the demand.

P A double set containing all the feasible K-shortest paths linking a pair of nodes.

P^1 An ordered set containing the nodes corresponding to the shortest path linking a pair of nodes.

P^k An ordered set containing the nodes corresponding to the k-feasible shortest path linking a pair of nodes.

P_1^k The node contained in the first position of the feasible k-shortest path.

P_t^{k-1} The node contained in the last position of the feasible k-shortest path.

P_i^k Node contained in the i^{th} position of the feasible k-shortest path.

P_{i-j}^k A subpath of P^k containing the subpath of the k-shortest path linking node P_i^k to node P_j^k.

Q A set containing the current computed feasible candidate lines.

R_w Set of intervals in which the entropy function for the o-d pair w has been discretized.

S Set of nodes forming a subline.

S_c Complementary of set S (i.e., $S \backslash N_{TP}^N$), containing the nodes not involved in the subline S.

W Set containing all the o-d demand pairs.

δ_S Set of edges that are incident to a group S of interconnected nodes which are disconnected from the remaining elected nodes S_c.

Λ Set of candidate corridors to be assigned to any of the lines under construction.

Ω Set of optimality benders cuts of the subproblem added to the master problem.

δ_n A set containing the list of nodes adjacent to n.

$pred$ The node precedent list used in Dijkstra's shortest path algorithm.

Definition of decision variables

b^l Discrete non-negative variable denoting the number of vehicles working on line l.

Δb Discrete non-negative variable denoting the number of new vehicles obtained with the current budget working on some lines.

f^l Continuous non-negative variable denoting the number of vehicles per time unit working on line l.

f^l_{lay} Continuous non-negative variable denoting the number of vehicles per time unit changing the direction they go throughout line l.

f^l_i Continuous non-negative variable denoting the number of vehicles per time unit working at a service stop i in the new line l.

f^l_{ij} Continuous non-negative variable denoting the number of vehicles per time unit working at stretch (i, j) and its reverse on the new line l.

g^m_w Number of trips of the OD-pair w assigned to the mode m in the elastic demand model.

n^l_i Continuous non-negative variable denoting the number of units of flow injected at node i of line l.

u^p_{ij} Continuous variable in $[0, 1]$ denoting the portion of passenger flow for the demand originating at p traversing the walking link (i, j).

$v^{p,l}_{ij}$ Continuous variable in $[0, 1]$ denoting the portion of passenger flow for the demand originating at p traversing the public transportation link (i, j) on line l.

$\tilde{v}^{p,l}_{ij}$ Continuous variable in $[0, 1]$ denoting the portion of passenger flow for the demand originating at p traversing in-vehicle stop i of line l without halting.

x^l Binary variable denoting if a new line l is constructed.

x_{ij} Binary variable denoting if stretch (i, j) and its reverse is constructed.

x^l_{ij} Binary variable denoting if stretch (i, j) and its reverse is contained in the new line l.

y_i Binary variable denoting if stop i is constructed.

y^l_i Binary variable denoting if node i is in the corridor used by the new line l ($l \in L^N$).

$\tilde{y}^l_i$ Binary variable denoting if there is a passenger flow exchange at node i in the corridor used by the new line l ($l \in L^N$) or, equivalently, that vehicles assigned to line l halt at node i.

z^l_{ij} Continuous non-negative variable denoting the amount of fictitious flow traversing a strecth $(i, j) \in A^N_{TP}$.

δ^l_c Binary variable denoting if corridor c is assigned to line l.

Δ^l_{yx} Binary variable denoting if the corridor assigned to line l is rectilinear.

δ_k Binary variable denoting if Benders cut (7.86) is active.

ϕ^l_i Binary variable denoting if node i works as source on line l.

Γ^l_i Binary variable denoting if node i works as source and sink at the same time on line l.

ϕ_w^m Value of the entropy function of a user belonging to the o-d pair w that wants to use the mode m in the elastic demand model.

ω Continuous variable associated with the active optimality Benders cuts (6.2).

ω_1 Continuous variable associated with the active optimality Benders cuts (7.85).

ω_2 Continuous variable associated with the active optimality Benders cuts (7.86).

$\beta_{i,p}^k$ Continuous non-negative dual variable related to flow balance constraint (2.50).

$\rho_{i,p}^{l+,k}$ Continuous non-negative dual variable related to flow balance constraint (2.51).

$\rho_{i,p}^{l-,k}$ Continuous non-negative dual variable related to flow balance constraint (2.52).

$\chi_{i,p}^{l,k}$ Continuous non-negative dual variable related to flow balance constraint (2.53).

$\gamma_{s,p}^{l,k}$ Continuous non-negative dual variable related to flow balance constraint (2.54).

$\pi_{s,p}^{l,k}$ Continuous non-negative dual variable related to flow balance constraint (2.55).

$\tau_{ij}^{l,k}$ Continuous non-negative dual variable related to flow balance constraint (2.56).

η Continuous dual variable related to linking constraint (7.74).

Introduction

This research work studies public transportation networks in urban scenarios and develops models to help operators make strategy and planning decisions. In this introductory chapter, we provide a global overview of the research to be done as well as the structure of the thesis report.

Study Objectives

This thesis focuses on developing a model which can help operators to make strategy and planning decisions for public transportation networks in urban scenarios. With this model, they should be able to consider high demand levels, and thus take into account some effects related to the phenomenon of congestion. To this end, the following objectives must be fulfilled:

- 1. Finding the main elements involved in the following:
  - 1.1. The costs and limitations of the network resources which are controlled by the transport operators. For instance, the number of available vehicles and their maintenance and acquisition costs.
  - 1.2. Some planning policies which help operators save money and users save time.
  - 1.3. Passenger behavior as they travel throughout the public transportation network.
  - 1.4. The consideration of some effects from congestion in the network.
- 2. Adapting the elements in points 1.1 - 1.4 to a mathematical programming problem.
- 3. Experimenting with case studies to comprehensively evaluate the final version of the model and to objectively analyze the quality of its solution.
- 4. Developing solving techniques that can provide near-optimal solutions in a period of time that is considered reasonable by operators.

Points 1.1 - 1.4 have been finally incorporated into a mixed-integer linear programming model which has been solved in different ways. Initially, it was solved directly by means of a CPLEX solver, version 12.4.0. However, the time needed to solve even small-sized network was too much, and thus some decomposition techniques and enumerative methods were used to overcome that limitation. The results for real-sized networks are not bad, although they can still be improved. Points 1.2 and 1.3 need further analysis, and some more related features should be incorporated into the current model.

Contributions

The contributions of the present research are summarized in the following subsections, each of which attempts to explain the motivations behind specific issues.

Integration of the network design and frequency setting phases

State-of-the-art methods have focused mainly on sequentially solving the network design (ND) and line frequency setting (LFS) phases. Firstly, the layout of the network is determined (line segments and stops served), and then frequencies and passengers are assigned to it.

However, the sequential solving approach is known to have many drawbacks in other areas such as the airline industry (Barnhardt [9]); but improvements can be made by building and solving models which integrate some of the planning problems. The airline industry has been a leader in the development of integrated approaches for schedule planning and recovering from disruptions. There has been research on integrating problems such as flight scheduling and fleet assignment (Lohatepanont & Barnhart [121], Cadarso & Marín [19]); fleet assignment and aircraft routing (Papadakos [148]); aircraft routing and crew scheduling (Mercier *et al*. [138]); and scheduling and competitive effects (Pita *et al* [152], Cadarso *et al*. [21]). All these problems were first developed and solved in a sequential fashion. However, their integration has proven to outperform sequential approaches, as demonstrated in every cited paper. Therefore, integrated planning may improve traditional sequential planning approaches. Cadarso & Marín [18] and Cadarso *et al*. [20] demonstrate that this fact also applies in the railway industry, which must consider additional factors such as integrated timetable planning and rolling stock assignment. Marín *et al*. [132] and Walker *et al*. [179] also developed integrated approaches for the railway industry.

Developing a quasi-exact solving methodology

In the last decade, some works integrating ND and LFS have been developed. The proposed solving methodologies rely entirely on heuristics and metaheuristics (Quak [153], Mauttone [135], Fernández *et al* [72], Fan & Machemehl [68]). The authors justify their use because of the NP-hard nature of the problem (Ignizio [94], Magnanti & Wong [126]). Thus, it seems unrealistic that real-sized instances could be solved to near-optimality within a reasonable amount of time. However, none of these authors have tried to formulate the problem with an appropriate mathematical structure, nor have any exact decomposition techniques been used to solve it. In airline applications, Mercier *et al*. [138] and Papadakos [148] have shown that mathematical programming decomposition techniques are worth using to solve related mathematical programming models with real-sized instances to near-optimality.

Modeling issues

Although many models have been presented to tackle ND and LFS in different ways, no literature has been found that addresses the problem of developing a model for expanding a transit network while also considering the effects of the new facilities on those already in operation, and vice versa. The design of express services or lines has also been addressed in this thesis. Unlike previous works, this aspect is integrated into the global model and is not considered locally, as in the works of Sun *et al* [170], Chen *et al* [47], Leiva *et al* [117], Chiraphadhanakul & Barnhart [49], and Larraín *et al* [111]. In other words, those works consider that the vehicles of a transit line may or may not halt at a bus stop while at the same time they take into account the demand and operations of the global network. Moreover, none of the previous works have taken into account the bus stop construction and maintenance costs as well as some or all passenger perception times, which are key elements for correctly modeling express services. Finally, the assignment demand models usually assume fixed modal demand, or the modal choice is carried out sequentially. First a set of lines is determined, and then the portion of each o-d demand is assigned to them (Hasselström [90], Soehodho & Koshi [165], among others). Only the work of Marín & García-Rodenas [130] determines modal splitting, but without considering line frequencies.

Extended flexibility for coping with more modeling issues

The resulting model and its solving techniques are developed in such a way that further considerations of modeling issues and algorithmic improvements do not have to be discarded. Therefore, the present research aims to introduce a preliminary model and demonstrate its viability in terms of the computational effort required for solving it, as well as its desirability for real-world applications.

Published and submitted articles

During the development of the present research work, some related papers have either already been published or they are in the revision phase. They include articles in ISI journals, fully extended papers in referred conferences, and conference presentations.

A preliminary study of a related problem has recently been published online in TOP journal [39]. This problem is based on the regional railway service disruption in an urban area of Spain's capital city of Madrid. This disruption severely affected a huge amount of demand for those who traveled throughout this railway network sector, and thus an auxiliary bus network was designed to alleviate this problem. Many of the modeling issues regarding this frequency setting phase have been adopted in this model, although some simplifications have been made due to time horizon restrictions.

Complementary to this journal publication, we had two publications in the referred conferences: Modelos de Optimización para la Planificación Robusta y la Gestión de Servicios de Transporte Público en caso de Emergencia (MORE) [34] and Automatic Control, Modeling & Simulation (ACMOS) [38]. These conferences were held in Madrid in November 2010 and in Saint Malo & Mont Saint Michel in April 2012, respectively. Moreover, we made four presentations in: the Conference on Numerical Optimization and Applications in Engineering ($CNumOpen$) [36], the Congreso de Ingeniería del Transporte (CIT) [33], the European Conference on Operational Research ($EURO$) [35] and the International Federation of Operational Research Societies ($IFORS$) [37]. They took place in Barcelona in October 2010, in Madrid in July 2010, in Lisbon in July 2010 and in Melbourne in July 2011, respectively.

The first publication strictly related to this research was made in the referred conference: 12th Conference on Advanced Systems for Public Transport (CASPT12) [122], which was held in Santiago de Chile in July 2012. This paper comprises part of the work described in chapters 2 and 5. An extension of it was submitted in December 2012 to the Public Transport journal [123], and in August 2013 we received the first revision, which asked for some corrections in order for it to be published.

Finally, a recent presentation was made in the new edition of the European Conference on Operational Research ($EURO$) [124], which took place in Rome. It included the entire work described in these chapters, plus the methodologies contained in chapters 4 and 6, as well as the experiments carried out with the Santiago de Chile underground network, which are reported in chapter 8.

Apart from these publications, we are preparing three additional papers, which will be submitted to ISI journals. The first one is an extension of the presentation given at Rome $EURO$ [124], which will include further experiments on the Santiago de Chile underground network as well as an additional study case based on the Seville railway network. The second and third papers will be related to subsection 3.5.6 and chapter 7. Firstly, we will introduce an extension of an innovative and general methodology, explained in subsection 3.5.6.2, which is capable of solving a subclass of mixed-integer linear bilevel problems where discrete variables only appear on the upper level. Then, we will introduce a practical application, based on an improved version of the elastic demand model presented in chapter 7.

Thesis Outline

The present thesis report is organized as follows. Chapter 1 gives a summary of the state of the art, but focusing on the works related to the expected contributions (see subsection). The following chapter introduces the mathematical programming model which integrates the ND and LFS phases that address the objectives outlined in points 1.1 - 1.4. Chapter 3 shows an overview of the developed solving techniques, which we adapt to our needs in the following chapters. Chapter 4 presents the corridor generator algorithm, which constructs a set of feasible corridors (incorporating constraints for topology, infrastructure, budget and some planning). Chapter 5 describes the line splitting algorithm which is used to efficiently solve instances where more than one line can be constructed. Chapter 6 explains all the techniques associated with the Benders Decomposition, which has been applied to the inelastic demand version. Chapter 7 extends these techniques in order to cope with the elastic demand version of the model. Chapter 8 shows the experimentation work carried out with two study cases: the Seville and Santiago de Chile underground networks, respectively. The model testing results have not been included in this chapter. Instead, they have been appended to the last subsection of chapters 4 - 7. Finally, Chapter 9 gives the final conclusions and provides some lines for further research.

Chapter 1

Literature review

This chapter presents a summary of some of the literature works which are related to the Public Transit Network Design Problem ($PTNDP$). We have selected all these works which have some relationship to the contributions of the present research (see the introductory section for further details). The structure of the chapter is as follows. Firstly, we introduce the notion of the $PTNDP$ problem. Secondly, we present works related to the first two phases into which this problem is split. These phases are associated with the network design (ND) and the line frequency setting (LFS) problems. The latter is also divided into two subsections which describe two important features: the express services design and the consideration of elastic demand. Fourthly, we review the passenger assignment models (PAM) under un/congested scenarios. The PAM are used to cope with passenger behavior, and thus they are part of the optimization process when dealing with ND, LFS or an integration of both problems. The fifth section of the chapter is dedicated to discussing integrated approaches. Finally, we give a brief summary of the chapter while highlighting in which designing phase our contributions are set out.

1.1 Public Transit network design problem

The Public Transit network design problem ($PTNDP$) represents a very complex planning process. For that reason, the whole problem is not studied; instead, it is split into four phases [25] (see figure 1.1), where the solution of each phase depends on the resolution of its preceding one. The order and role of each phase is as follows:

1. Global line network planning.
2. Timetable setting.
3. Vehicle scheduling.
4. Crew scheduling.

The first phase involves the allocation of the public transportation stops and its interconnections by means of a network layout design. This design takes into account neither the frequencies nor the departure times of the lines. The state of the art is not very extensive and the vast majority of works are approximated methods (see Fernández *et al* [71], Baaj & Mahmassani [7], Mauttone & Urquhart [134]). Only the seminar works of Laporte [106], [107], [108] and the extensions of Marín [128] and Escudero & Muñoz [64] have dealt with exact methods.

The second phase, however, has been largely studied. It is split at the same time into two subphases:

1. 2.1. Frequency settings.

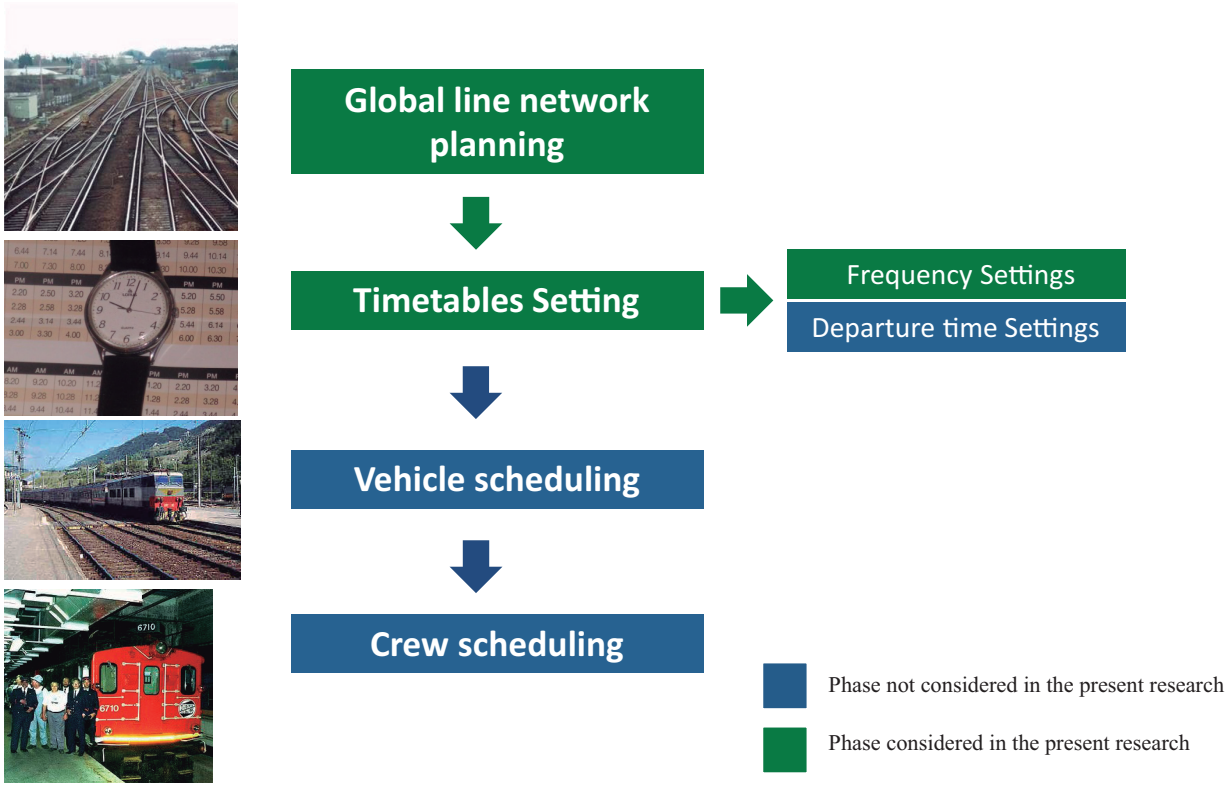


Figure 1.1: The four-steps framework for the Public Transit network design problem inspired by Ceder & Wilson 1986.

2. 2.2. Departure time settings.

Subphase 2.1. takes as inputs the set of lines established in the first phase and its goal is to determine the number of services that each line will carry. It also includes the assignment of vehicles which perform the services and the satisfaction of some requirements related to the quality of service offered to the passengers. To date, the most important work is that of Ceder [23], in which four different methods are proposed to determine the frequencies. They are categorized as point-based methods or route check-based methods.

Having determined the frequencies in subphase 2.2, a line synchronization process is carried out by minimizing the passenger waiting times at stops. Ceder & Tal [27] proposed a synchronization method, which was later improved on by C.B. Quak [153]. Moreover, Quak integrated this method into a set of methodologies described in Ceder [23], resulting in its successful application to the city of Amersfoort.

In certain cases, after subphase 2.1., there is an additional subphase in which a line's short turning is carried out. It consists of a reduction in some vehicle itineraries on a working line and starts at a subsequent stop of the first stop and finishes at a predecessor stop of the last stop. The reason for such a reduction is the change in the demand distribution along the line at different periods of the day. This change leaves some line stretches without passenger activity. For instance, from midnight to dawn, it is possible that the periphery sectors of a city have no demand. Scant literature related to this topic has been published and, among these works, we can find the seminar papers of Ceder [24] and [26], which describe different methods for short-turn trip categories.

The third phase aims to assign the minimum number of vehicles which carry out the services established by a line in the second phase. If the public transportation operator works with more than one depot, this problem is called the Multi-Depot Vehicle Scheduling Problem ($MDVSP$). Otherwise, if the operator has

only one depot, it is known as the Single-depot vehicle scheduling Problem ($SDVSP$). For a comprehensive survey of related formulations and solving techniques, the reader is referred to the works of Desrochiers *et al* [56], Daduna & Paixao [51], Löbel [120] and Mesquita & Paixao [139].

Finally, the fourth phase comprises two subphases: the assignment of duties to vehicles (which is known as the Duty Scheduling Problem (DSP)) and the assignment of these duties to drivers (which is called the rostering problem or crew rostering). The first subphase has been largely studied (see section 4.2. of Desaulniers & Hickman [55] for a comprehensive state of the art), whereas the second subphase has been paid scarce attention. A good review can be found in Odoni *et al* [144].

In the following subsections 1.2 and 1.3, we will explain in detail phase 1 and subphase 2.1. of this four-step framework, on which the present research is focused.

1.2 Global line network planning models

As discussed in the previous subsection, the global line network planning phase has received relatively little attention by researchers. Most of them assume that the line segments and the allocation of stops are given as inputs. This may be a valid simplification in a working network where the goal is to reassign planning resources in a way that increases demand and that minimizes operator costs. However, even these situations require the possibility of extending the network layout by adding some new lines. This need arises, for instance, in railway modes like underground or train. Furthermore, for non-railway-based systems like buses, rerouting does not entail operators investing a significant amount of money. Thus, the quality of setting line frequency and departure times will depend also on how good this routing phase performs.

To sum up, the $PTNDP$ problem requires that the network layout and the working line frequencies be determined in such a way that a given objective function is minimized, subject to a set of constraints. Different methodologies have been proposed to deal with this problem. They can be classified into the following main categories:

1. Practical rules and ad hoc procedures.
2. Optimization analytical models for idealized situations.
3. Heuristics and metaheuristic approaches applied to more real cases.

As shown in the chronogram of figure 1.1, Mandl [127] is the pioneer and the most cited work which tackles the $PTNDP$. Its solving approach consists of two phases. During the first one, a feasible initial route network is generated, with emphasis placed on service coverage and directness; whereas the second phase tries to minimize total travel time, including in-vehicle and waiting times. Both phases are solved by means of heuristic procedures. The limitation of this method is that it fails to consider the demand pattern during the first phase.

Dubois *et al* [60] also solved the $PTNDP$ in two steps. Firstly, they choose a set of links (representing streets), which will be used later to build the lines. The first step is solved by means of a heuristic procedure which minimizes in-vehicle travel times, subject to some construction costs. Then, they apply an optimization model for solving the second phase with the subset of streets provided by the previous phase. The limitation of this method is that its simplicity makes it incapable of solving large-sized networks.

Ceder & Wilson [25] solved the $PTNDP$ more efficiently and realistically by considering user behavior. The model aims to minimize excess travel time upon boarding, expressed as the sum of extra travel time (with respect to the shortest travel time), plus the transfer time (if any). This is subject to maximum o-d travel

time, bounds on the route length, and the maximum number of constructed routes. The model is solved by means of a heuristic procedure which performs a topological bread-first search with emphasis on constraint satisfaction.

Van Nes *et al* [175] carried out a survey on existing models and optimization techniques. They also proposed a version of the $PTNDP$ which maximizes the number of passengers reaching their destinations without transfers, using the maximum number of available vehicles as inputs. Furthermore, they also studied the relationship between passenger demand and network resources. As a result, they devised an elastic demand design model capable of splitting the passenger flow into different transportation modes. This model was solved by means of a heuristic procedure which simultaneously chooses the lines, sets their frequencies, and sets the number of passengers traveling throughout.

Baaj *et al* [5], [6] and [7] devised an artificial intelligence method consisting of three main blocks. The first one involves a line construction algorithm, which trades off between the user and operator costs. The second one consists of a line analysis method called $TRUST$, which assigns frequencies to lines. The last block implements a procedure called RIA, which improves the line routing obtained in the first block, so that they look more tractable from the operator's point of view.

The RIA procedure developed by Baaj *et al* in [7] was later improved upon by Mauttone & Urquhart [134]. The main differences lie in the strategy for inserting stops into the lines and in the way some operator costs are considered in relation to line configurations and total distances. Baaj et al [7] insert individual stops, whereas Mauttone & Urquhart [134] add two pairs of stops related to an od-demand pair. Thus, the latter approach allows serving more demand directly without transfers.

Fernández *et al* [71] proposed an interesting methodology for generating different types of lines according to their role. They define two types of line: trunk and feeder lines. The former are used to satisfy directly all o-d higher demand pairs, whereas the latter account for giving service to o-d lower demand pairs which perform some transfers. To build these lines, the authors developed three methods: one for each of the two types of line, plus a third one in order to reduce some lines which carried low levels of demand. These methods were applied to the construction of the line layout for the Bus Rapid Transit Network of Santiago de Chile.

Another interesting line of research is related to the resolution of the path enumerative problem (PEP). This problem aims to determine the k-shortest paths which are candidates for line use. One of the first papers which applies this problem to the $PTNDP$ is that of Van der Zijpp & Catalano [174]. Moreover, they implement some user satisfaction constraints within the solving method. These consist of two behavioral rules introduced by Schnabel and Lohse [159], which state that users do not consider taking routes whose travel times are a ϕ_{max} factor greater than the shortest route, given the origin and destination stops; and that they overlap by more than a Δ_{min} factor, since they are considered quite similar.

Following this line of research, some authors have used this method within a genetic framework. Fan & Machemehl [68], [69] construct a pool of preliminary candidate corridors, which are then filtered by taking into account some operator's requirements. They include maximum and minimum distance satisfaction. This filter provides the final pool of corridors, which are used in the genetic algorithm to determine the optimal frequencies and the passenger assignment under static and elastic variants. A similar approach is taken by Cipriani et al [31] for planing Rome's public transport system. However, they extend the topology of the corridors by generating, apart from the k-shortest paths, a TSP-like corridors. This subset aims to provide service to important service stops which share different transportation modes and/or are situated at strategic points of the city.

As explained, all these works are based on obtaining a feasible line configuration whose solution quality is not evaluated, nor is it carried out by means of an operator heuristic function (Mauttone [135]), which does not give real insight into the optimal solution. The first analytical models were developed at the beginning of 2000. Laporte et al [106] resolved the location of a stop set by considering the trace of the line, its terminal stations, and the maximum number of lines to be allocated. The authors developed a maximum demand coverage criterion by means of a metric which obtains the traveling distances throughout the urban network. In that manner, the number of users who can reach the stops from the departure points is determined, taking into account the maximum travel time which they are willing to spend.

Laporte et al [107] extend the previous model so that the line trace or part of it can be constructed. To do that, they incorporate a demand origin-destination matrix in collaboration with the maximum demand coverage criterion for choosing eligible stops. However, they leave out the exact approaches and propose different constructive heuristics for solving them. They report which of them are the most efficient based on the experiments performed on a Seville's underground network model. Laporte et al [108] propose another extension to deal with multiple line design. To do that, they split the problem into two phases. The first one consists of identifying the candidate stops, whereas the second phase constructs their interconnections so that the final network layout is obtained. This approach has an important drawback: the second phase needs an explicit enumeration of the terminal stops of each line. Thus, some knowledge of the solution is required beforehand.

Marín [128] overcomes that limitation by formulating an optimization model, where neither the number of lines to be constructed is predefined nor their terminal stops. The resulting optimization model is a pure binary mathematical programming problem which is solved by means of $CPLEX$. However, optimal solutions are only reported for small-sized instances, and the model is unable to give circular line topologies. Further works of this author in collaboration with Jaramillo [129] and [131] tackle multiperiod network construction and the application of decomposition methods to solve the model more efficiently. The former takes into account different od-demand matrices (one for each time window), and thus different line configurations may be obtained; whereas the latter deals with the network design problem for a single time window and it is solved by means of the classical Benders decomposition (CBD) [14]. Reported results show that CPLEX is faster than the CBD for all instances.

Escudero & Muñoz [64] accommodate Marín's model in [128] to tackle circular lines. Moreover, they managed to reduce the computing time quite significantly by solving the model in two steps. In the first step, they solve an integer mathematical programming problem which determines the used stops as well as their interconnections. In the second step, lines are constructed by means of an assignment procedure which links the interconnections of stops. This procedure is based on the grade of the node.

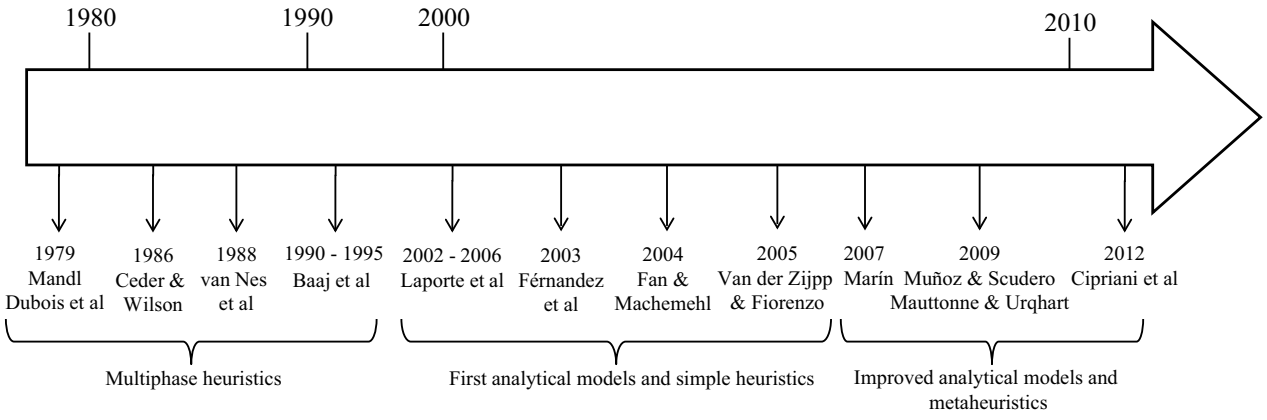


Figure 1.2: Chronogram of the works on the global line network planning phase.

The present research has focused on integrating the solving techniques for the PEP (shown in [174] and [68]) into an extension of the mathematical programming models presented by Marín [128] and Escudero & Muñoz [64]. This extended mathematical programming problem will take as inputs a pool of feasible line corridors, where feasibility is regarded as the verification of the line topology constraints as well as the infrastructure budget and planning horizon requirements. The optimization problem will determine which corridors are assigned to the lines and which nodes will carry out service based on passenger assignment. Furthermore, the existence of a network layout already in operation will be considered as well.

1.3 Line frequency setting models

As shown in figure 1.3, the research on line frequency setting models ($LFSM$) started in the early 80's. These models assume as inputs the network layout (the line segments and the location of stops), and their main goal is to determine the working frequencies or, in other words, the number of services to be carried out on the line.

The first solving techniques are very primitive and rely on experimental methods. The most popular is the passenger count data approach of Ceder [23], in which some measures are taken at some stops of a working transit network, followed by a statistic survey for creating estimates in the form of simple mathematical formulas that compute the correct frequencies without saturating the system. Despite its simplicity, the computational effort of this methodology is subject to the quality of the collected data, which means a significant financial investment. Thus, it is not viable for large networks.

This limitation motivated the emergence of analytical models. They have been formulated by means of a passenger assignment model (see next subsection) and in accordance with two different goals:

1. To know the required transit features (i.e., line frequencies and vehicles) or
2. To generate planning timetables.

The former have focused mainly on a planning horizon (see the works of Spiess & Florian [167], Constantin & Florian [43] and Noriega & Florian [142]), whereas the latter have been studied under rigid timetables (see the recent works of Fang & Xiaogang [193], Haupt *et al* [91], Han *et al* [89] o Papola *et al* [149]).

The present research has focused on the first approach. Moreover, some features are taken from our work published in [39], which presents an optimization model for overcoming service disruption in railway networks. Despite this focalization, there still exists a huge amount of research; and thus it is very difficult to analyze all of them. However, not many of them have considered some planning strategies such as the express service network design or competition among different modes of transportation. For that reason, in the following subsections we will bring some attention to the works which take these issues into account.

1.3.1 Express services design

In public transit systems with high demand levels, benefits are gained by both users and operators through limited-stop services (i.e., express services), which serve only a subset of stops along certain lines. As shown in the studies of Vuchic [178] and Ercolano [62], express services improve service for users because they reduce travel times by making fewer service stops and traveling at higher speeds between stops. As for operators, it fulfills demand with fewer vehicles, thanks to shorter bus cycles.

This practice has been implemented in many urban public networks. To mention some of them, Wilson et al [185] applied stop-skipping to a light rail line in Boston. The overall system performance was largely improved, although some portion of the demand was lost because of the increase in transfers between lines

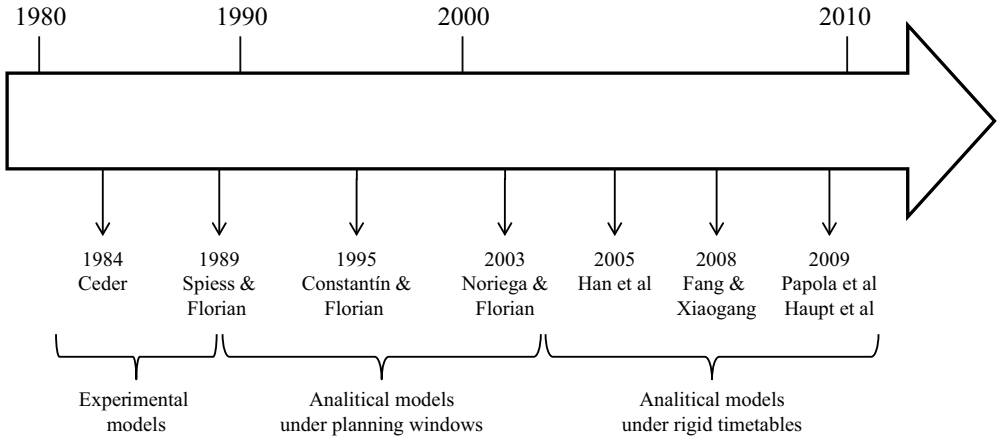


Figure 1.3: Chronogram of the outstanding works on frequency setting models.

that some passengers were not willing to do. Suh *et al* [168] introduced an express subway system in Seoul, in which a stop-skipping service was considered. The total time savings were estimated subject to a given origin-destination demand, distance between stations, headway and operating speed. It was found that the total passenger travel time decreased by up to 7.8 percent through this stop-skipping service.

Despite its importance, state-of-the-art works related to express service design (ESD) are scant. As shown in the chronogram of figure 1.4, Jordan & Turnquist [100] were the pioneers of this research. Their model assumes that all the demand goes to the same destination (called the central business district). Moreover, they do not take into account vehicle fleet size or vehicle capacity, and they do not allow passenger transfers. Under these assumptions, the model was stated as a working vehicle cost minimization problem and it was solved by means of forward dynamic programming. The underlying graph structure consists of corridor areas in which new inbound demand occurs. Thus, each area is processed from the nearest to the most outer, assigning a number of vehicles so that minimal line headway is assured and vehicle level occupancies are tolerable. Furth [80] generalized this approach for tackling bidirectional corridors (i.e., the demand has the same destination or emanates from the same origin stop). Moreover, he included light direction on deadheading (this concept will be explain in the following).

The works of Li *et al* [118], Eberlein *et al* [61], Fu *et al* [78], Liu *et al* [119], Sun & Hickman [169] and Corts et al [45] focused on real-time scheduling approaches, where some stops served in a certain line service may be skipped by a vehicle in the next service. In that manner, vehicles can arrive earlier at conflictive stops where demand levels are higher. This kind of scheduling operation is known as *deadheading* and has nothing to do with ESD. Thus, we do not provide a detailed description of these works here.

The following work on ESD is based on Ulusoy *et al* [173], who defined a more sophisticated optimization model. This model incorporates a fixed OD-demand matrix and is formulated as a non-linear mixed-integer programming problem (MIP), where the objective function seeks to minimize vehicle working costs as well as the user's travel costs, including waiting, transfer and in-vehicle times. They imposed bounds, in the form of constraints, on route headways, symmetrical route services and vehicle fleet size limitations. Moreover, transfers can only occur between express and local service routes; thus transfers between local service lines are not allowed. This MIP is solved by means of an exhaustive search algorithm, which was implemented in MATLAB and tested on a rail transit line in the northeastern region of the USA.

Goossens et al [85] also incorporated a fixed OD-demand matrix demand for a railway system. Firstly, the authors formulated a pure binary linear problem which dealt with a single line local service design prob-

lem. Its objective function sought to minimize the vehicle working costs subject to link frequency capacity. This problem was later extended to account for multiple lines which allowed transfers. At this point, its underlying graphs were transformed to cope with ESD based on the type of vehicles which could halt at a given stop. The vehicle type was defined according to the covering area: regional trains, interregional trains and intercity trains. The resulting graph associated each link with a type of vehicle and was used to formulate three equivalent optimization problems. The two latter problems had fewer variables and constraints, and thus they were used to solve three different instances related to some parts of the Dutch railway network.

In this line of research, categorizing the type of service to be performed depends on features of the planning resources. Freyss *et al* [77] developed a practical methodology to determine the distribution of two types of stops within an underground symmetric circular corridor. The type of stop determines the number of services to carry out. At stops of type AB, all vehicles halt; whereas at stops A or B, only half of them halt. The methodology is based on some important assumptions which do not hold in general. For instance, constant and identical dwell times at stations do not hold in congested scenarios.

ESD has been incorporated into Bus Rapid Transit Systems (BRT's) by Sun *et al* [170] and Chen *et al* [47]. Both authors first defined mathematical programming formulations whose objectives aim to minimize users waiting, in-vehicle travel times, and working costs, all of which are subject to headway bounds and vehicle fleet size limitations. Moreover, they are solved by means of genetic algorithms. The main differences rely on the scenarios to which the models were applied. Sun *et al* [170] reported results without considering high demand levels; whereas Chen *et al* [47] focused on very congested scenarios (during peak hour periods). The latter also comprehensively compared the results obtained from the ESD or the local service design.

These two later works incorporated many features. However, they also committed an important conceptual error: passenger waiting times were taken into account in the objective function while the model sought a system optimum solution. This controversy has been proven to violate the user equilibrium principle, which states that each user chooses their itinerary according to their own selfish behavior instead of cooperating with other users.

Leiva *et al* [117] overcome this error by integrating the common lines problem (CLP) of Chriqui & Robillard [50] into the optimization model. It was formulated as a non-linear mixed-integer problem (MIP) and their solving approach was a heuristic algorithm that consisted of an iterative call to a simplified version of this MIP, in which constraints related to the CLP dropped off. Between calls, some constraints were added to force a frequency increase in those lines where the CLP constraints were violated. Each simplified MIP was solved by means of the $MINOS$ commercial solver. This model, however, has some important drawbacks: the modeler has to provide all the possible service configurations explicitly, and no planning resource capacities are taken into account. Moreover, the authors do not provide computational times in the results, although they apply the model to a real case.

Chiraphadhanakul & Barnhart [49] developed another MIP, which includes fleet size and vehicle capacities. However, their model was limited to only one express service route and it also violated the user equilibrium principle stated above. The innovative aspect is that the authors reported good computational times for real-sized networks. This was accomplished by reducing the size of the MIP through bounded variables and some heuristic rules based on experimental results. Moreover, it was turned into a linear mixed-integer programming problem ($MILP$) by enumerating the values that the frequency variables could take. Then, the resulting $MILP$ was repeatedly solved for each enumerated frequency value, and the one that gave the best demand coverage was chosen as the optimal solution.

Quite recently Larraín *et al* [111] have presented a preliminary model, which is partially based on the

work of Leiva *et al* [117]. It determines all types of service routes, based on a pool of predefined corridors, and uses a framework in which the solving scheme presented in Leiva *et al* determines the frequencies for a given set of routes and their type of services. This route service set is determined by another module, which employs different heuristics developed in [110]. The framework has been tested on the underground network of Santiago de Chile and the authors plan to develop an SW aided tool. However, they do not consider vehicle fleet size limitations, although the cost of acquiring a new vehicle is taken into account in the objective function.

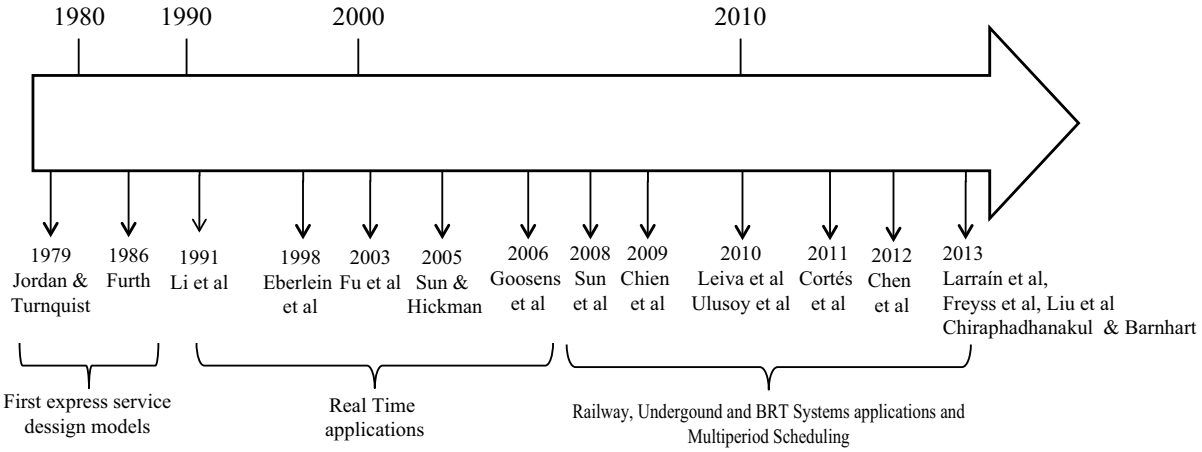


Figure 1.4: Chronogram of the works on express service design and related policies.

The present research is in the line of Leiva *et al* [117], Chiraphadhanakul & Barnhart [49] and Larraín *et al* [111], where static approaches with different combinations of express service patterns can be obtained. However, unlike these works, our model is capable of dealing with non-fixed line corridors as well as the budgetary limitations of infrastructure and planning, which are incorporated by means of constraints and costs in the objective function.

1.3.2 Elastic demand

In recent years several models for the design of public transport networks (line layout, frequency setting and many others) take into account other transportation modes. Accordingly, a fraction of the demand of the existing transportation modes may switch to the newly designed network. For evaluating the amount of captured trips, the most popular modal split model used in these design models has been multinomial logit. Because of this, the passenger trip assignment criterion of the network design model is elastic, with a modal split model as the source of demand elasticities in the public transportation mode. In this thesis, the term *design models with elastic demand* must be understood in this sense, and it is preferred over design models with modal split (or choice). When referring to elastic demand, the reader must bear in mind that a modal split model is in fact implicit.

Related state-of-the-art works are not very extensive and can be classified in several ways, according to selected criterea. In this review, we have reduced them to two possibilities for the sake of simplicity. The first criterion is related to the type of model used to evaluate modal splitting, whereas the second one is associated with the modal network decision variables.

Modal splitting models can be divided into probabilistic and zero-one assignment models. The former use a pre-fixed o-d demand matrix, which holds the total amount of demand for each o-d pair. These models evaluate the portions of each amount to be assigned to the different existing modes of transportation by eval-

uating a probabilistic function. The most used one is the multinomial logit function (MNL) of McFadden [137]. See the works of Soehodho & Koshi [165], Abdulaal & LeBlanc [1], LeBlanc [114], Ferrari [73], Lee & Vuchic [116], Laporte *et al* [107], Marín & García-Rodenas [130], Yoo *et al* [190], Gallo *et al* [82] and Cipriani *et al* [30]. In all these works, only two modes compete. The MNL function is not suitable for use where different classes of public transportation modes compete, since it assumes that changes in the network structure of a given mode will have proportionally equal effects on the rest (Bhat [12]). Despite this fact, only the works of Fan & Machemehl [68], [69] and [70] have adopted another approach, based on the nested logit model of Williams [183]. This model evaluates an MNL model at each non-leaf node of a tree. From this node, a bunch of emanating branches contains different mode alternatives, which are considered as belonging to a class of transportation mode, for instance, public transport. Thus, they are considered to compete in similar conditions.

Zero-one assignment models, like probabilistic models, use a pre-fixed o-d demand matrix. They can be regarded as simplified probabilistic models, where the probabilistic function assigns all the amount of an o-d demand to one mode, hence their name. These models are not realistic and tend to under/overestimate the demand. However, they are suitable for integration into an optimization model because of their easy implementation. In this line of research, we have found the works of van Oudheusden *et al* [176], Bruno *et al* [16], Marín [128], Escudero & Muñoz [64], Marín & Jaramillo [129] and [131].

Regarding the modal network decision variable criterion, we can divide the research into 3 categories:

1. Models where only the modal network layout variables are optimized.
2. Models where only modal network planning decision variables (frequencies, mainly) are computed or,
3. A combination of both.

Moreover, we can subdivide this criterion according to the solving scheme:

a) The layout and/or planning variables are determined first, and then the modal assignment is carried out (sequential scheme) or,

b) The layout and/or planning variables, as well as the modal assignment, are performed at the same time (simultaneous scheme).

Works related to category 1 include Laporte *et al* [107], Bruno *et al* [16], Marín & García-Rodenas [130], Marín [128], Escudero & Muñoz [64], Marín & Jaramillo [129] and [131]. The first two deal with the construction of a single alignment, but using different modal splitting models and algorithms. Laporte *et al* develop a set of constructive heuristics, where the ridership throughout a candidate stretch (link) is evaluated by means of an MNL model. Bruno *et al* use a K-shortest path algorithm to determine the candidate set of alignments that form the public transportation network. Then, a pedestrian subnet is constructed according to each computed alignment. Thirdly, a binomial logit model is used to evaluate the portion of demand assigned to each public transportation-pedestrian subnetwork structure and the private car network. This one is known beforehand, thus the shortest path cost are used in the probabilistic model. Moreover, in this step the evaluation of the operator and user costs are also computed. They are used in the following and last step to choose the best alignment. Marín & García-Rodenas formulate a non-linear bilevel problem in which the upper level operators construct the network layout that is limited by a budget and a set of maximum lines to be constructed. The lower level determines the modal assignment by means of a bilevel logit function. This function is approximated by a piecewise linear function and embedded into the operator level. This results in a single linear mixed-integer level problem, which is directly solved by *CPLEX solver*. The algorithmic details of the remaining works are skipped, since this information is already provided by the description in

the fourth paragraph of this section as well as in the third to last paragraph of section 1.2.

Under category 2, we have found the works of Abdulaal & LeBlanc [1], LeBlanc [114], Ferrari [73], Yoo *et al* [190] and Gallo *et al* [82]. The first two works are quite similar and the difference lies in the network structure of the public transportation mode. The latter adds two types of links to the network, which allows modeling transit time access and transfer times. In both works, the simultaneous scheme is carried out by means of an adaptation of the Hooke-Jeeves' heuristic, which finds good solutions in reasonable time. Regarding the other works, they are formulated as bilevel problems and thus they are solved by means of simultaneous schemes, but using heuristic methods. In those bilevel problems, the upper level determines the operator decision variables (mainly frequencies) and the lower level does the modal assignment flows. The work of Ferrari also includes transit fares on public transportation and some private car links as decision variables at the operator level.

The third category includes the works of Hasselstrm [90], van Oudheusden *et al* [176], Soehodho & Koshi [165], van Nes *et al* [175], Fan & Machemehl [68], [69], [70], Lee & Vuchic [116], Cipriani *et al* [30]. Excluding the work of Hasselstrm and Soehodho & Koshi, the remaining works use a full simultaneous scheme. Moreover, whatever the scheme is, the solving techniques rely on heuristics, metaheuristics or a combination of both methodologies, which are more or less sophisticated. We refer the reader to section 1.5 for further details on these works and some other related works which deal with inelastic demand.

We have skipped many other works which make use of some elastic demand since:

1. They were not applied to network design and/or line frequency setting problems; and/or,
2. The elastic demand was considered in the route choice of a single mode only.

The last point was adopted in the work of Ulusoy *et al* [173], mentioned in the previous subsection.

In this research, we have adopted a simultaneous scheme, which includes the network design and the line frequency setting problems. Moreover, it is solved using exact methods. As a start, we studied in detail the work of Marín & García-Rodenas and found out that their model did not reproduce exactly the bimodal logit function in the optimum. The reason for this drawback lies in the interpretation of the KKT optimality conditions. We refer the reader to study carefully the sections 7.3, 7.3.1 and 7.4 of chapter 7 for better understanding.

All the aforementioned works and their classified criteria are summarized in table 1.1. We would like to clarify that in the column labeled **Modal Split.**, we have written the word "prob. mod" for those cases in which we didn't find the type of probabilistic model used.

1.4 Transit passenger assignment models

A transit passenger assignment model ($TPAM$) is defined as the determination of the passenger flow going throughout a set of paths within the transit network (Desaulniers & Hickman [55]). The network layout is supposed to be known (i.e., the line segments, location of stops and its working frequencies), and the total demand amount per origin-destination is also pre-specified.

The $TPAM$ is formulated assuming that the passenger's goal is to minimize travel time or generalized cost. To do that, several elements are taken into account, such as:

1. The modelisation of stochastic elements which are related to time issues.

Year	Author	Modal Split.	Variables	Scheme	Methodology
1979	Abdulaal & LeBlanc	prob. mod.	freq.	simul.	heur.
1988	LeBlanc	prob. mod.	freq.	simul.	heur.
1998	Bruno *et al*	0-1 assignment	a single line	seq.	heur.
1999	Soehodho & Koshi	MNL	lines and freq.	seq. + simul.	heur.
1999	Ferrari	MNL	freq. and fares	paral.	heur.
2004	Fan & Machemehl	nested logit	lines and freq.	simul.	metaheur.
2005	Lee & Vuchic	MNL	lines and freq.	simul.	heur.
2006	Fan & Machemehl	nested logit	lines and freq.	simul.	metaheur.
2008	Fan & Machemehl	nested logit	lines and freq.	simul.	metaheur.
2007	Marín	0-1 assignment	lines	paral.	exact
2008	Marín & Jaramillo	0-1 assignment	lines	simul.	exact
2009	Marín & Jaramillo	0-1 assignment	lines	simul.	exact
2009	Escudero & Muñoz	MNL	lines	simul.	exact
2009	Marín & García-Rodenas	0-1 assignment	lines	simul.	exact
2010	Yoo *et al*	MNL	freq.	simul.	heur.
2011	Gallo *et al*	prob. mod.	freq.	simul.	heur. and meta.
2012	Cipriani *et al*	MNL	lines and freq.	simul.	metaheur.

Table 1.1: Summary of elastic demand model features.

2. The kind of strategy (behavior) that passengers follow. This strategy may or may not include transfers between lines and, if so, we need to take into account alighting or boarding movements as well as waiting times.

3. Congestion effects due to the capacity resources of the network, for instance, the maximum allowable density of users on a stop platform or in a transit vehicle.

The first $TPAM$ models assumed all elements were deterministic. Thus, the optimization goal focused on minimizing the travel times throughout the network links subject to the line's working frequencies and the passenger arrival rates at stops (Dial [57], Lampkin & Saalmans [105], le Clercq [115] (1972), Silman *et al* [164] and Last & Leak [112]). In that manner, for low and moderate congested scenarios, waiting times and boarding passengers at a stop can be estimated by means of the following formulas:

$$E[WT] = \frac{\alpha}{\sum_{l \in L_i} f_l} \tag{1.1}$$

$$P_{l'} = \frac{f_{l'}}{\sum_{l \in L_i} f_l} \tag{1.2}$$

where L_i stands for the set of lines going through stop i. Moreover, f_l is related to the frequency of line l, and α is a parameter which depends on the distribution of passenger arrivals at the stop. Normally, this parameter takes the value 0.5 for the Poisson distribution, whereas for uniform rates it is set to 1. Therefore, the estimated waiting time ($E[WT]$) and the probability of getting on a vehicle serving line l ($P_{r'}$) depends mainly on the line frequencies.

These models are rather simple since they rely on a large assumption: passengers take the first bus halting at the stop. Thus, no congestion effects due to finite vehicle capacity are taken into account. Later, Chriqui

& Robillard [50] extended these models to cope with the common line problem (CLP). The CLP deals with the choice of a line among a set of lines which share either all or part of the path of an o-d demand pair. Next, some models appeared which incorporated the concept of strategy. This idea was introduced by Spiess [166], but was characterized by Nguyen & Pallotino [141] as a subgraph of the expanded transit network. This subgraph was called hyperpath and consists of a subnetwork which holds a set of possible paths that can be chosen for a given o-d demand pair.

Spiess & Florian [167] were pioneers in integrating the models of Chriqui & Robillard and Nguyen & Pallotino into a single mathematical programming model whose dual form can be solved using a specific algorithm for combinatorial optimization. In the case of infinite frequencies, this becomes reduced to the classical Dijkstra algorithm.

The next generation of models have slightly introduced the phenomenon of congestion. We do not review them since congestion effects have not been considered in the present research, although the capacity of the link is taken into account. Regarding the concept of passenger strategies, it has been discarded as well and, instead, we have adopted a system optimum point of view. These two simplified hypotheses allow us to skip non-linearities and thus formulate the model as a linear mixed-integer mathematical programming problem.

Despite these simplifications, some passenger behavior features will be taken into account. For instance, situations where passengers remain in-vehicle at a service stop will be penalized in the objective function. Additionally, we model the time costs of boarding and alighting from a vehicle. All these costs will be considered as linear (i.e., proportional to the amount of demand related to these situations). To the best of our knowledge, works on the state of the art consider only boarding or alighting times (see de Cea & Fernández [53], Wu & Florian [186], Wu *et al* [187], Cominetti & Correa [42], Kurauchi *et al* [102], Cepeda *et al* [29] and Lam *et al* [103] and [104]). The last reference combines both times.

The first type of time gives innovation to our PAM and allows us to integrate correctly the express service design feature (see subsection 1.3.1) into the frequency setting part of the model. It is integrated by linking the demand flows of this time to the node's role. We have considered two possible roles: a node in which no vehicle halts (no flow exchange) and a node where all vehicles halt (some flow exchange). Demand flows related to the first role and that occur at nodes assigned to the second are penalized in the objective function.

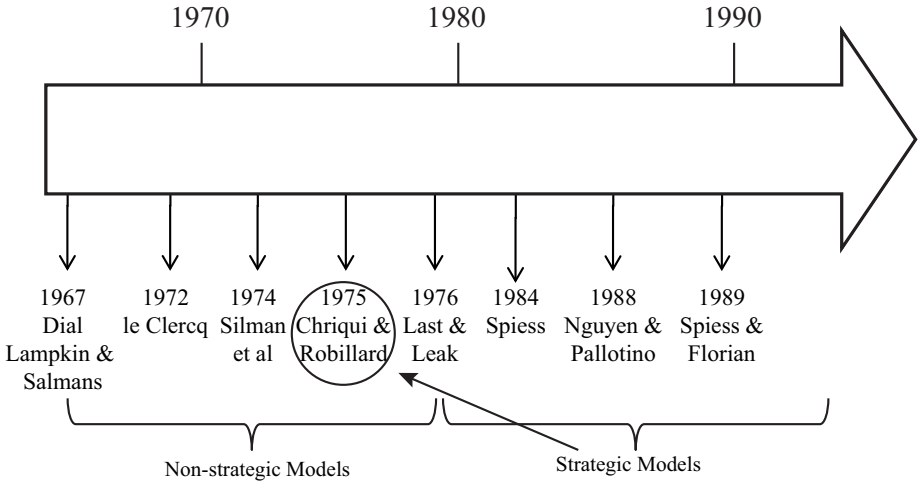


Figure 1.5: Chronogram of works related to the transit passenger assignment models with no congestion effects.

1.5 Integrated Global Line Planning Network and Line Frequency Setting models

This section is devoted to describing the main state-of-the-art works which integrate the Global Line Planning Network and the Line Frequency Setting phases presented in subsections 1.2 and 1.3. The vast majority of works solve these two phases by means of heuristics, metaheuristics or a combination of both. We have only found the works of Wan & Lo [180] and Borndörfel *et al* [15], which formulate them as mixed integer linear programming problems. However, they are too simplistic and their solving techniques are unable to solve medium- and large-sized networks.

Regarding modeling issues, the objective function criterion is not well-established. Many works do not consider operator costs together with passenger costs. In the works where both costs are taken into account, some important features of passengers and/or operators are left out. For instance, boarding or alighting times are seldom modeled and only vehicle working costs are considered as operator costs. Nonetheless, other operator costs are important, such as the construction and maintenance costs of infrastructure resources (stations and stretches) when designing a railway network. Apart from the objective criterion, the way the frequency setting is carried out is rather simplistic: no works incorporate express service design and very few works consider elastic demand. Thus, much work remains to be done in this area.

In the following subsections, we will provide a summary of each research work and/or a set of works which have similar features. Each summary contains the stated objective criterion, the constraints imposed, their solving approach and the case study, if any. Complementary to these summaries, we include table 1.2, which highlights the most important features of each research work. In that table, the column **Scheme** shows the way the network design and line frequency problems are solved (sequential or simultaneous). The next column **Method** reports the solving approach used (heuristic, metaheuristic or exact). The column **Op. Costs** indicates whether or not operator costs are included in the objective function. The column **Us. Costs** points out whether or not user costs are included in the objective function. The last column **E.D.** denotes whether or not elastic demand is taken into account.

1.5.1 Lampkin & Saalmans, Silman, Barzily and Passy's research work

Lampkin & Saalmans [105] and Silman *et al* [164] decomposed the problem into two stages. In the first one, the set of lines is constructed. In the second, their frequencies are determined. The first stage uses heuristic methods to construct "skeleton" lines, which are then expanded to cover the full set of nodes in the network.

The frequencies are determined by minimizing the total passenger travel time, which is calculated as the sum of the O-D demand multiplied by the travel time. This time the expected waiting time as a function of route frequencies serving each origin is included, along with any transfer node on the shortest path serving the O-D pair. Lampkin & Saalmans [105] used a random gradient-based search procedure to determine the final frequency values, whereas Silman *et al* [164] employed a gradient projection method to minimize the total travel time. Moreover, the latter adds a penalty to the objective for the estimated number of standees on the bus.

The solving methodologies in both works were capable of solving only small-sized networks.

1.5.2 Dubois, Bell and Llibre's research work

Dubois *et al* [60] decomposed the problem into three subproblems. The first one involves selecting the links in the street network where the service operates; the second one determines the lines themselves; and the

third one assigns them the optimal frequencies.

In the first step, a traditional network design problem is formulated, where the total passenger travel time is minimized. This time in-vehicle times subject to a budget constraint are included, along with binary decision variables indicating whether a street segment is in the final solution. A heuristic is used to solve it. In the second step, the maximum set of lines is generated by using three different heuristic rules. In the last step, the optimal line frequencies are found while considering waiting times. The solving approach consists of a gradient-based search heuristic, which is similar to that of Lampkin & Saalmans [105].

As in the previously mentioned work, the solving methodology was capable of solving only small-sized networks.

1.5.3 Hasselström's research work

Hasselström [90] proposed a two-stage process of network design in which lines and frequencies are determined simultaneously. In the first stage, an initial lines set is generated; in the second stage, these lines are refined and a detailed evaluation of them is performed along with passenger assignment.

In the first stage, the mathematical formulation includes a direct demand function, allowing the demand to be determined endogenously. The form of the direct demand model is based on a traditional gravity model with parameter β, where all terms that do not dependent on the route structure or on the frequencies are rolled into a constant term for each O-D demand pair. The remaining elements of the generalized cost are given by a function of the set of frequencies. The objective function maximizes consumer surplus and is equivalent to maximizing the number of passengers with this demand function. As constraints, the author considers a budget constraint that includes a cost per vehicle on each line and a required minimum frequency of service for a zone. The latter is implemented by making use of a binary variable which indicates whether or not a line serves the zone.

This solving approach was the first one to be applied to a real world case consisting of the city of Goteborg, Sweden, where a little more than 60 different modal lines were taken into account. Moreover, it was embedded into the Volvo package, a transit network planning software.

1.5.4 Marwah, Farokh, Umrigar and Patnaik's research work

Marwah *et al* [133] proposed a three stage process to simultaneously construct routes and assign to them frequencies in the context of bus transit systems. The first stage determines the links on the network where the passenger flow is concentrated. Its objective is to minimize vehicle working costs as well as passenger riding time under under no constraints. Links and nodes in which no passenger flow is involved are eliminated and the resulting reduced network is given as inputs to the second phase. This phase constructs a large set of candidate routes subject to bounds on route length and maximum detour constraints. The resulting route set is given to the last phase, which selects a subset of them and assigns frequencies. Its objective function aims to minimize only vehicle working costs and is subject to vehicle fleet size limitation, bounds on the line frequencies and maximum flow load restriction on each link.

The first and second phases are solved by means of constructive heuristics, whereas the third one tackles a continuous linear programming problem. Reported results are based on the city of Ahmedabad (India), whose network representation holds 134 nodes, 1028 links (reduced to 426 links in the first phase), and 8911 o-d demand pairs. The second phase generates 457 candidate routes, which are given to the continuous optimization problem.

1.5.5 Ceder & Wilson's research work

Ceder & Wilson [25] formulated two sequential mathematical models for solving the bus network design problem. The first one considers the passenger objective of minimizing excess travel time upon boarding, expressed as the sum of "excess" travel time (larger than the shortest travel time with a direct route), plus the transfer time (if any), and summed across all o-d pairs. This objective is minimized according to constraints on the maximum o-d travel time (as a percentage above the shortest path), lower and upper bounds on the route length (expressed in units of running time), and a constraint on the maximum number of routes. The second model adds the passenger waiting time and vehicle operating and capital costs to the objective function; it also includes constraints on the minimum frequency for each route and a constraint on the maximum fleet size.

Reported results are based only on a small-sized network with 5 nodes, 7 arcs and 20 o-d demand pairs; so it is unknown how the solving approach would perform on a larger network.

1.5.6 van Oudheusden, Ranjithan and Singh's research work

van Oudheusden *et al* [176] devised a framework for operators to iteratively solve a bus network design problem. Initially, operators must provide a set of candidate routes with feasible working frequencies, from which the framework will select a subset of them and reassign new frequencies.

For the route selection phase, the authors developed two mathematical programming formulations, which are inspired by the classical Set Covering Problem (SCP) and the Simple Plant Location Problem ($SPLP$), respectively. The former is used when the whole input od-demand matrix is going to be allocated to the bus network, whereas the latter is employed in the case where the demand competes with different modes of transportation. Moreover, each of these formulations have two variants with similar mathematical structures. They differ from the type of demand pattern considered: one is related to the many-to-one case, whereas the other is associated with the many-to-many case.

Both formulations and demand pattern cases are conceived as pure binary linear mathematical programming problems, where the objective functions aim to minimize the fixed route costs plus fare costs (in the case of elastic demand) and subject to full demand coverage requirements. Although, there are no planning resource limitations and transfers are not allowed, the problems are flexible and can be easily extended to cope with these restrictions (i.e., by simply adding extra constraints). Additionally, the authors stated that working lines can be considered by setting to zero their corresponding objective function costs, so that these lines will be selected in the optimal solution.

These mathematical problems are solved by means of the Erlenkotter algorithm [63] in a repeating fashion, wherethe line frequencies in each resolution are set out by operators using an external procedure. The formulation related to the SCP was applied to the network of Kanpur, which holds 60 centroids; whereas the one associated with the $SPLP$ was used in the network of Ranjitahn, which contains 41 centroids.

1.5.7 van Nes, Hamerslag and Immers' research work

van Nes *et al* [175] proposed a model in which routes and frequencies are determined simultaneously. The objective function maximizes the number of direct trips (i.e., trips without transfers) served in the network, for a given fleet size. A direct demand model is proposed to estimate the origin-destination trips by public transit; trips are proportional to the attraction of the origin and destinations zones, and are an exponential function of the cost, similar to that of Hasselström [90].

A distinct feature of Hasselström's work is the consideration of different vehicle types and limited resources. Moreover, each working type vehicle incurs a different cost factor. To implement this feature, a variable indicator is used to know the type of vehicle assigned to a route. Regarding constraints, they include a vehicle working budget constraint, the vehicle availability and lower and upper bounds on the set of feasible frequencies. These bounds are set by means of the round-trip time on the routes.

The solution technique adopted is a heuristic in which all frequencies on proposed routes are initially set to 0. Each route is then evaluated with respect to its potential to improve "efficiency", defined as the ratio of passengers added by the direct service to the additional cost of increasing the frequency, evaluated by the mentioned constraints. The route with the highest efficiency is selected and the frequency on that route is increased, until the budget and the available vehicles are consumed.

The reported results are based on Groningen, a Netherlands city whose network contains 182 nodes, from which 150 are centroids and 8 are candidate routes. The number of links and o-d demand pairs is omitted. Furthermore, the authors claim that the solving approach is capable of solving instances of up to 250 nodes, from which 150 are centroids and 750 are candidate routes.

1.5.8 Israeli & Ceder's research work

Israeli [97] and related works (Israeli & Ceder [96], [98]; Ceder & Israeli [28]) formulated the bus network design problem as a multiobjective programming problem with two objectives: the total passenger cost (Z1) and the operator fleet size (Z2). Z1 includes the in-vehicle passenger hours spent between the origin and destination, the waiting and transfer time spent traveling from the origin to the destination, and the empty seat-hours on a route. Constraints in the formulation include the passenger assignment from a fixed demand matrix and minimum frequencies on each route. This problem is solved with a 7-step heuristic, where different sets of routes and frequencies are determined in pairs of sequential steps.

Reported results are based only on a small-sized network with 8 nodes and 14 links, so it is unknown how the solving approach would perform on a larger network.

1.5.9 Baaj & Mahmassani's research work

Baaj & Mahmassani [5], [6] and [7] decomposed the network design problem into three elements: a route generation step, in which routes and frequencies are constructed; a network analysis procedure, defining measures of effectiveness at the network-, route-, and stop-level; and a route improvement algorithm to improve the route design. These procedures are applied to different sets of weights, which examine the total travel time (including walk time, wait time, in-vehicle travel time and transfer-related time), total demand satisfied, and the required fleet size to operate the system.

The case study consists of a reduced network representing Austin, a city in the state of Texas. Authors do not mention all the network elements and, so far, we found that they use 106 nodes and 952 o-d demand pairs.

1.5.10 Pattanaik, Tom and Mohan's research work

Pattanaik *et al* [150] is the first work which uses a genetic algorithm-based approach in the context of network design and frequency setting. Their model's goal is to minimize total user travel time, including in-vehicle, waiting and transfer times, as well as vehicle operation costs. As constraints, they impose bounds on line frequency and length as well as a maximum load factor.

The solving scheme consists of two main blocks: a candidate route generation algorithm and a genetic algorithm (GA), which choose the optimal set of routes. The former takes into account the line length bounds and two additional constraints. One is related to the overlap factor and the other one is associated with the maximum detour factor. Both factors are measured with respect to the shortest path for a given o-d demand pair. In the second phase, the GA considers two different coding schemes: the typical fixed string length and a new variable string length. In the former, routes are considered as variables and the number of routes remains fixed; whereas in the latter, both elements are optimized simultaneously. The latter scheme is improved in the authors' later work [172].

Reported results are based on a case study of an urban area of Madras, a metropolitan city in the South of India. The network contains 25 nodes, 39 links and 600 o-d demand pairs.

1.5.11 Soehodho & Koshi's research work

Soehodho & Koshi [165] formulated a bilevel programming problem in which the upper level controls the passenger and user costs, whereas the lower level performs the user equilibrium. The user costs include the use of a car (for those who prefer the private mode) and in-vehicle travel time as well as transfers and crowding costs (for the public mode users). Crowding costs are evaluated as the number of standees that travel in-vehicle. Regarding operators, only bus operations are taken into account as costs. The lower level costs are comprised of the bimodal travel times and the entropy function, which allows establishing in the optimum a logit demand modal distribution. All these costs are subject to modal equilibrium constraints as well as minimal route frequency and maximal vehicle fleet size requirements.

The solving approach consists of a heuristic methodology made up of five modules. Three of them are related to the route construction and its improvements, whereas the two remaining ones perform the frequency and bimodal assignment and the fleet size determination.

Results are based on Sioux Falls City, whose network consists of 24 nodes, 76 links and 540 o-d demand pairs.

1.5.12 Chien, Yang and Hou's research work

Chien *et al* [48] proposed a model which determined an optimal feeder bus route and its headway. This route aims to feed a major intermodal transfer station or a central business district in the service area under consideration. The objective function minimizes user travel costs by including access, waiting and in-vehicle travel times. It also minimizes supplier costs in the form of vehicle working costs. As constraints, they imposed bounds on the route headways so that the planning budget is not exceeded (minimum headway) and so that all the demand is satisfied (maximum headway).

Their solving approach consisted of a two-phase genetic algorithm in which the first (initial) phase determines a feasible set of candidate routes, which are used in the second phase to generate new (nest) routes while keeping headways feasible. This scheme is applied to a service area of $4 \times 10km^2$, which holds 28 nodes, 160 links and 128 o-d demand pairs. The authors prove that their algorithm is capable of solving instances of that network to optimality by carrying out an exhaustive search procedure.

1.5.13 Carrese & Gori's research work

Carrese & Gori [22] developed a model in which constructed lines are categorized as main and feeder lines, in the same way as stated by Fernández *et al* [71], where main lines are called trunk lines.

The main lines are used for minimizing: 1) the increase in time spent by in-vehicle travelers along the constructed route, in terms of their shortest route; 2) the total waiting time on each route; and 3) the unused vehicle capacity, expressed as the number of free seats on each route. The constraints include the satisfaction of the whole demand, minimum level of service, maximum route length, winding path of the route, maximum number of transfers, door to door travel time, maximum link flow, and fleet size limitation. Complementary feeder lines are used for improving the level of service in order to cover the demand which is not directly served by the main lines.

The solving approach consists of three steps: 1) defining the skeleton of the main lines, 2) identifying the optimal main lines, and 3) defining the feeder lines. Reported results were based on a large zone of Rome, the capital city of Italy. Its network representation holds around 1.400 nodes, 7.700 links and 560 centroids, with 550.000 trips/h during the peak hour.

1.5.14 Wan & Lo's research work

Wan & Lo [180] are one of the few authors who formulated and solved a mixed integer lineal programming problem ($MILP$) that integrated the network design and line frequency setting problems. However, the modeling approach has many drawbacks. To cite just a few of them: a) routes can be only acyclic, b) each o-d demand pair must be assigned to a single route (in other words, transfers are not allowed and all the demand must be covered by the transit network), c) no budgetary limitations are imposed, and d) only operator frequencies are taken as costs in the objective function.

This $MILP$ is delivered directly to the CPLEX commercial solver for computing only a small-sized network of 10 nodes, 19 links and 9 o-d demand pairs, with three lines under construction. Furthermore, the authors do not report any performance measures.

1.5.15 Ngamchai & Lovell's research work

Ngamchai & Lovell [140] proposed a model focused mainly on determining the transfer locations in the network and the coordination of headways between the chosen routes. Its objective function seeks the minimization of vehicle operation costs and user travel times, including in-vehicle and waiting times.

The solving approach consists of three major components: 1) the route generation algorithm, which builds an initial set of TSP-like-routes; 2) the route evaluation algorithm, which designs the service frequency and applies headway coordination; and 3) the route improvement algorithm, which modifies the existing route configuration.

The latter phase is carried out by a genetic algorithm (GA), which implements a variety of genetic operators based on different problem elements. Each of them is applied according to some discrete distribution related to the efficacy of the operators. This way of handling the genetic operators, together with the variety of them, gives their work more innovation than previous works associated with GA.

Reported results were based on the same study case of Pattanaik *et al* [150]. Moreover, some performance features of the author's approach were compared with those of Pattanaik *et al* [150] algorithm.

1.5.16 Lee & Vuchic's research work

Lee and Vuchic [116] stated that transit demand should depend on the network configuration and associated route frequencies. Therefore, they presented an iterative approach to tackle the dynamic characteristics of the transit route network design problem. The objective of this approach was to minimize only the total user travel time (including in-vehicle, transfer and waiting times), subject to frequency constraints on each

route. To simultaneously estimate the transit demand and generate the optimal transit network, a modal split modeling procedure was proposed. Furthermore, an AI-based heuristic method was adapted from the solution approach used by Rea [154] to solve the transit network design problem as well as to consider variable transit demand under a fixed total travel demand. Sensitivity analysis examined the relationship between the optimal transit network and the design inputs.

Reported results are based on a small-sized network taken from Rea [154], so it is unknown how the solving approach would perform on a larger network.

1.5.17 Hu, Shi, Song and Xu's research work

Hu *et al* [93] proposed two optimization models: one to tackle the transit network problem (TNP), and the other to determine the optimal headways (THP). The former seeks the maximization of direct demand coverage, subject to bounds on: route length, link capacities, maximum route detour factor (with respect to the shortest route), and maximum allowable transfer time constraints. The latter is formulated as a minimization problem, where the objective costs include passenger costs (waiting, in-vehicle and transfer times) as well as vehicle working costs. As constraints, they imposed bounds on the service time at stops, transfers coordination, maximum transfer time and maximum allowable headway.

These models are solved sequentially and use different metaheuristics approaches. The TNP faces an ant colony algorithm, whereas the THP is optimized by means of an improved genetic algorithm. Two case studies are proposed, one for each optimization model. The one related to the TNP is based on Changchun, the capital city of Jilin Province of China, whose working network holds 273 nodes, 504 links, 115 transit routes, 904 buses and 1180 mini-buses. The other case study, which is associated with the THP, consists of the underground network of Hong Kong. It has six working lines, which contain 63 nodes and 114 links.

1.5.18 Zhao's research work

Zhao presented in [191] and [192] a model for finding network structures and headways, where the objective function minimizes user travel times, including in-vehicle, waiting and transfer times. As for constraints, he imposed bounds on route length and headway, maximum load route factor, limited vehicle fleet size, maximum number of transfers, and minimum route directness (expressed as the ratio between the current user travel cost from a path linking a pair of nodes and its shortest path cost).

The solving approach is based on an iterative methodology which combines two algorithms: a simulated annealing search, which looks for a set of feasible routes; and a fast descent search, which finds their corresponding headways.

Reported results were based on two case studies. One is based on the Mandl network [127] and the other is related to the service area of Miami-Dade, a southern city in the USA. Its network holds 2804 nodes, 4300 links and 120000 o-d demand pairs, with a total of 161944 trips/day.

1.5.19 Chakroborty's research work

Chakroborty [46] showed the effectiveness of the genetic algorithms (GA) in solving the urban transit network design problem ($UTNDP$). Firstly, he pointed out the features of the $UTNDP$ that make it a difficult problem for traditional techniques, and then he suggested directions through his presentation of GA-based methodologies for the $UTNDP$.

Its modeling approach consisted of defining two sequential problems: the transit routing problem ($TrRP$) and the transit scheduling problem ($TrSP$). Both approaches are solved by different GA's. The $TrRP$ was

not formally described, whereas the $TrSP$ was formulated as a non-linear mixed-integer problem (MIP). This MIP minimizes transfer and user waiting times and is subject to bounds on the service time at stops as well as the route headways. Furthermore, it coordinates transfer and arrival times, allows only one transfer per person, and limits the maximum waiting time per person. Reported results are based on the Mandl small-sized network [127].

1.5.20 Agrawai & Mathew's research work

Agrawai & Mathew [2] exploited the parallel features of the genetic algorithms (GA) to apply them more efficiently to a model which minimizes global user travel times, including walking, waiting, in-vehicle and transfer times, as well as vehicle working costs. As constraints, they impose bounds on the line frequencies, a maximum allowable load factor, and demand coverage satisfaction.

The solving scheme is similar to the vast majority of GA-based approaches. Initially, a generation of a large set of potential routes is carried out, and then a subset of them is chosen. In this second phase, the authors applied two different parallel GA's: one consists of a global parallel processing environment, which uses parallel virtual machine (PVM) libraries; the other one is a global message passing interface, which substitutes the PVM libraries. Both GA algorithms are driven by means of a route evaluation module, which evaluates the computed route set. Reported results are based on Delhi, the capital city of India. Its network representation contains 1332 nodes, 4076 links and 6000 o-d demand pairs.

1.5.21 Fan & Machemehl, Barra, Carvalho, Teypaz, Cung and Balassiano's research work

Fan & Machemehl [68] solve the network design problem by examining some metaheuristic techniques such as: simulated annealing, tabu search, genetic algorithms, local search, and random search. As with other previous methods, all these metaheuristics begin with a set of skeleton routes and they generate additional routes. The output is run through a network evaluation tool, which is used to improve the quality of the solution at subsequent iterations.

Their objective is to minimize the sum costs of user, operator and unsatisfied demand. User costs include walking, waiting, transfer and in-vehicle times, whereas operator costs refer to the costs of operating the required vehicles. The last cost is computed by means of the difference between the total satisfied demand and the demand going through the transfer paths. Regarding constraints, they formulate five sets related to headway feasibility, maximum load factor, maximum fleet size, bounds on trip length, and a maximum number of constructed routes.

Reported results include three experimental networks labeled as small, medium and large. The small one has 35 nodes, 82 links, 42 o-d demand pairs and 286 skeleton routes, whereas the medium one consists of 72 nodes, 180 links and 182 o-d demand pairs. Finally, the large network has 160 nodes, 418 links and 756 o-d demand pairs. For the medium and large networks, the size of the skeleton routes are not specified.

The later works of Fan & Machemehl [69] and [70] are part of the mentioned work with no apparent differences. In contrast, the work of Barra *et al* [10] try to solve the Fan & Machemehl's model by means of constraint programming techniques. However, they do not report results on medium- or large-sized networks. Their analysis focused on a modified Mandl's network [127] consisting of 15 nodes, 42 links and 210 o-d demand pairs.

1.5.22 Borndörfer, Grotschel and Pfetsch's research work

Borndörfer *et al* [15] proposed a multicommodity flow model for line planning, in which line segments are predefined a priori. The model is formulated as a mixed-integer linear programming problem ($MILP$),

where decision variables include the selection of lines (discrete variables), the determination of their frequencies and the passengers assigned to them (both are continuous non-negative variables).

The objective function minimizes passenger travel times (only in-vehicle times) and some operator costs, both of which are related to line frequency setting and fixed operating costs, respectively. It has constraints on flow conservation, link capacity, line capacity, line frequencies, and edges. It also has some linking constraints, which associate frequencies with lines.

The solving approach consists of a column-generation-based algorithm with length-restricted lines. The algorithm solves first an LP relaxation of the $MILP$, and then a heuristic procedure constructs a good integer solution from a line pool consisting of lines with nonzero frequencies in the optimal LP solution.

The authors report some experiments carried out in the 1998 line system of Potsdam, which then had 951 nodes, 1321 edges and 110 o-d demand pairs.

1.5.23 Fernández, de Cea and Malbran's research work

Fernández *et al* [72] developed a bilevel problem in which the upper level performs the physical design of the transport system, whereas the lower level carries out the operational design and determines the user behavior.

The objective function of the upper level is not specified, whereas that of the lower level minimizes user travel times, including access, in-vehicle, waiting at stops and transfer times, and the production costs of the services. As constraints, they impose topology on the network, flow balance, some user behavior rules and link capacities.

The upper level is solved by means of the heuristic procedures proposed by Fernández *et al* 2003, whereas the lower level is formulated as a continuous non-linear programming problem, which tackles three different algorithms: the Hooke and Jeeves' algorithm; simulated annealing; and augmented Lagrangian. The first one was reported as the best.

Reported results were based on the Bus and underground networks of Santiago de Chile, the capital city of Chile. The whole system has 2042 nodes, 8267 links and 409 centroids with 584.833 trips/h for the peak hour (from 7.30 a.m. to 8.30 a.m.).

1.5.24 Pacheco, Alvarez, Casado and González-Velarde's research work

Pacheco *et al* [146] studied an urban transport problem applied to Burgos, a northern Spanish city. The problem consists of designing n routes and assigning the available buses to them. The objective is to minimize the waiting times of users of stops and their overall in-vehicle travel time. Routes are subject to a pair of pre-defined nodes, where they must begin, end, and pass through twice in the same service. Moreover, each vehicle assigned to the same line is assumed to carry out the same number of services.

The solving approach consists of a specific procedure that constructs an initial solution, plus two alternative methods: a local search and a tabu search strategy. Both methods work in two alternating phases: modifying first the route designs, and then assigning buses to them. The initial procedure is based on a heuristic insertion which incorporates techniques related to TSP, such as Or-exchanges and cross-exchanges.

Reported results are based on Burgos, a northern Spanish city. The largest tested instance has 382 stops, of which 48 are pairs of terminal nodes and an overall of 24 routes are constructed. No data was available on the number of links and o-d demand pairs.

1.5.25 Mauttone's research work

Mauttone [135] reviews some state-of-the-art mathematical programming problems and presents some new ones, including a bilevel programming problem. However, only one problem is used to solve two small-sized networks. It consists of an extension of the Spiess model, where frequencies are integer decision variables subject to a given set of possible values. Additionally, a constraint on vehicle fleet size is imposed. The set of eligible routes are given as inputs. They have been previously computed by means of a construction heuristic procedure, called PIA, which takes into account additional objective costs and constraints. The line cycle is considered as the operator costs, and some bounds related to the route duration are imposed: maximum time duration and maximum circuit factor. The latter is measured by means of a ratio between the line cycle and the shortest path of a given o-d demand pair.

The PIA procedure is later incorporated into a metaheuristic technique called Greedy Randomized Adaptive Search procedure (GRASP). It consists of a repeated execution of a greedy randomized variant of the PIA, such that different sets of routes are constructed, followed by a local search in which new frequencies are determined.

The GRASP metaheuristic is used for solving a medium-sized network representing the bus system of Rivera, a city of Uruguay. This network consists of 84 nodes, 143 links and 370 o-d demand pairs.

1.5.26 Szeto & Wu's research work

Szeto & Wu [171] formulated a non-linear mixed integer programming problem (MIP), where the objective function minimizes the number of transfers as well as the total passenger travel time, including access, in-vehicle and waiting times. Regarding constraints, the routes are imposed to start and end at a pre-defined set of nodes and the number of intermediate nodes visited is limited. Moreover, the number of assigned vehicles is also limited to a given fleet size; the line frequencies must have a minimum value; and the access time is bounded by a maximum allowable time.

The MIP is solved by means of an integrated method in which a genetic algorithm tackles the route design problem and a neighborhood search heuristic faces the frequency setting problem. Reported results are based on Tin Shui Wai, a suburban residential area in Hong Kong. The network has 28 nodes, 45 links and 115 o-d demand pairs.

1.5.27 Cipriani, Gori and Petrelli's research work

The most recent works on GA are those of Cipriani *et al* [30], [31] and [32]. The authors proposed an objective function which considers total distance and travel times as operator costs. It also takes into account user disutility, which is measured by in-vehicle travel times, waiting times and transfer penalties.

The solving approach gives as outputs the bus lines as well as the working frequencies and suitable vehicle sizes. It consists of two phases. In the first one, three different sets of complementary feasible routes are generated. They are called A-type routes, B-type routes and C-type routes. The first type is composed of the shortest paths between node pairs with demand trips longer than a given minimum value. The second type is related to general routes, which aim to generate trunk and feeder lines. The last type accounts for the working transit network's routes. The second phase uses the GA for choosing a sub-set of feasible routes from among the three mentioned sets, as well as for planning their configuration. The GA uses a fitness function, which is made up of three terms: the objective function, one term associated with the unsatisfied demand, and another one related to the line-level coordination of conflicting transfer points. The second term explicitly considers the minimum percentage of demand to be satisfied, whereas the third term coordinates the lines by assigning additional penalties related to the number of transfer points and the number of lines

going through these transfer points. During this phase a modal splitting is also performed by means of the evaluation of a binomial logit function.

Reported results are based on the central area of Rome, the capital city of Italy. Its network representation contains 1300 nodes, 7000 undirected links, 450 centroids with an amount of 565000 trips/h corresponding to the peak hour.

Year	Author	Scheme	Method	Op. Costs	Us. Costs	E.D.
1967	Lampkin & Saalmans	Sequential	Heuristic	Yes	No	No
1974	Silman *et al*	Sequential	Heuristic	Yes	No	No
1979	Dubois *et al*	Sequential	Heuristic	No	Yes	No
1981	Hasselström	Simultaneous	Heuristic	No	Yes	No
1884	Marwah *et al*	Simultaneous	Heur. + Exact	Yes	Yes	No
1986	Ceder & Wilson	Sequential	Heuristic	Yes	Yes	No
1987	Van Oudheusden *et al*	Simultaneuous	Heur. + Exact	Yes	No	Yes
1988	van Nes *et al*	Simultaneous	Heuristic	No	Yes	No
1989	Israeli & Ceder	Sequential	Heuristic	Yes	Yes	No
1990	Baaj & Mahmassani	Simultaneous	Heuristic	Yes	Yes	No
1992	Baaj & Mahmassani	Simultaneous	Heuristic	Yes	Yes	No
1992	Israeli	Sequential	Heuristic	Yes	Yes	No
1995	Baaj & Mahmassani	Simultaneous	Heuristic	Yes	Yes	No
1995	Israeli & Ceder	Sequential	Heuristic	Yes	Yes	No
1998	Ceder & Israeli	Sequential	Heuristic	Yes	Yes	No
1998	Pattnaik *et al*	Simultaneous	Metaheuristic	Yes	Yes	No
1999	Soehodho & Koshi	Sequential	Heuristic	Yes	Yes	Yes
2001	Chien *et al*	Simultaneous	Metaheuristic	Yes	Yes	No
2003	Tom & Mohan	Simultaneous	Metaheuristic	Yes	Yes	No
2003	Wan & Lo	Simultaneous	Exact	Yes	No	No
2003	Ngamchai & Lovell	Simultaneous	Metaheuristic	Yes	Yes	No
2003	Zhao	Simultaneous	Metaheuristic	No	Yes	No
2003	Chakroborty	Sequential	Metaheuristic	No	Yes	No
2004	Carrese & Gori	Sequential	Heuristic	No	Yes	No
2004	Fan & Machemehl	Simultaneous	Metaheuristic	Yes	Yes	Yes
2004	Agrawal & Mathew	Simultaneous	Metaheuristic	Yes	Yes	No
2005	Lee & Vuchic	Simultaneous	Metaheuristic	No	Yes	Yes
2005	Cipriani *et al*	Simultaneous	Metaheuristic	Yes	Yes	Yes
2005	Hu *et al*	Sequential	Metaheuristic	Yes	Yes	No
2006	Fan & Machemehl	Simultaneous	Metaheuristic	Yes	Yes	Yes
2006	Zhao	Simultaneous	Metaheuristic	No	Yes	No
2007	Borndörfer *et al*	Simultaneous	Exact	Yes	Yes	No
2008	Fan & Machemehl	Simultaneous	Metaheuristic	Yes	Yes	Yes
2008	Fernández *et al*	Simultaneous	Heuristic	Yes	Yes	No
2009	Pacheco *et al*	Simultaneous	Metaheuristic	No	Yes	No
2011	Mauttone	Simultaneous	Metaheuristic	Yes	Yes	No
2011	Szeto & Wub	Simultaneous	Metaheuristic	No	Yes	No
2012	Cipriani *et al*	Simultaneous	Metaheuristic	Yes	Yes	Yes

Table 1.2: Summary of the main features for Integrated Global Planning and Line Frequency Setting models.

1.6 Summary

This chapter has shown several state-of-the-art works which are related either to the network design problem (NDP), line frequency setting problem ($LFSP$), or both. We have also studied some emerging works on $LFSP$, where express service design and elastic demand features are taken into account. Almost all of them are solved by means of heuristics, metaheuristics or a combination of both techniques. Only the works of Marín [128], Marín & Jaramillo [129] and [131], Escudero & Marín [64], Wan & Lo [180] and Borndörfel *et al* [15] solve them by means of exact approaches. However, their models are rather simplistic.

Regarding modeling issues, there is no work on the state of the art which integrates NDP and $LSFP$ problems by simultaneously taking into account express service design and elastic demand features. Moreover, the passenger assignment models do not consider or only partially consider passenger time costs at stations related to boarding, alighting, and waiting in-vehicle. These costs are very important when demand levels are not so low. Finally, the inclusion of a working network has been only slightly considered by van Oudheusden *et al* [176]. Thus, more modeling effort is needed.

Finally, it is worth-mentioning that the size of the networks which these methods are capable of solving is rather limited, even when using approximated methods. Not many authors report any results on real-sized networks (see the works of Marwah *et al* [133], Carrese & Gori [22], Zhao [192], Agrawai & Mathew [2], Fan & Machemehl [68], Borndörfer *et al* [15], Fernández *et al* [72] and Cipriani *et al* [32]). Moreover, computational times are seldom encountered in these works.

In the following chapters, we will show that a mathematical programming approach can integrate the mentioned modeling aspects, and that it is capable of solving medium- and large-sized networks to (near-) optimality. It solves these networks through the following simplifications:

- Line corridors are given as inputs to the optimization model.
- Lines are constructed sequentially.

The line corridors will be obtained by means of a constrained k-shortest path algorithm, which is external to the optimization model. These simplifications are rather reasonable, as shown in the preliminary results subsections 4.2 and 5. The optimum or a near-optimum can be reached as long as the generated set of feasible line corridors is rich enough and the networks do not exhibit a high connectivity.

Complementary to these simplifications, we will apply some decomposition techniques, which allow splitting the original model into small problems so that they are easier to solve without losing optimality.

Chapter 2

Modeling Approach

This chapter presents the mathematical formulation for the inelastic version of the rapid transit network design and planning model ($RTNPD$), which integrates the network layout design (NLD) and the line frequency setting (LFS) for urban public networks (UPN) in underground and railway systems. The NLD phase extends the working UPN, taking as inputs the locations of the candidate stretches and stations of the new lines, as well as the construction costs, which cannot exceed the infrastructure budget. Regarding the LFS phase, frequencies and vehicles are assigned to the working and newly built lines, providing that the link frequency capacities, the vehicle acquisition budget, and the planning horizon are not exceeded. NLD and LFS phases are solved at the same time by means of a passenger assignment model under a system global optimum. The structure of the chapter is as follows. Firstly, we introduce the problem statement. Then, we show the data structure in which the mathematical formulation of the model lies. It includes two graphs: 1) an undirected graph for routing and frequency assignment purposes, and 2) a network flow for the passenger assignment phase. The next section is devoted to the mathematical formulations. It comprises the objective function as well as the set of constraints sorted by functionality. Finally, we present some preliminary results in order to justify which formulation is the best.

2.1 Problem statement

The problem under consideration concerns the design and planning of rapid transit network extensions, where a set of lines may be already in operation. The resulting system must satisfy a known demand of trips for this public transportation mode (PT). The extension of the network can be considered subject to budgetary constraints, which apply both to the line construction costs and the possible enlargement of the fleet of vehicles, which may be required for it to operate. The maximum number of new lines to be added to the rapid transit system is fixed a priori, and the layout of these new lines must comprise a predetermined set of segments or line stretches, some of them already built and being part of the itinerary of an already existing line. Ending points of these segments may be chosen optionally as stops for some or all of the new lines, or simply as points where vehicles pass through but do not halt.

The approach integrates into a single model both the determination of the layout of the new lines (under construction) and the frequency setting of the whole system. This allows us to take into account the effect that the new lines have on the rest of the transportation system. The assignment of vehicles is performed globally, taking into account the existing vehicles already in operation on the lines plus the (possibly) newly purchased vehicles on the whole set of lines of the system (existing + added). In the following, we will refer to the existing vehicles operating on the lines as simply the working vehicles, and the currently operating lines as simply the working lines. Also, we will refer to the segments as line stretches. By stop, we refer to a rapid transit platform where passengers can board or alight from a public transportation vehicle. Physical characteristics or parameters of a stretch may differ, according to the direction of movement.

We refer to line frequency as the number of vehicles per time unit that can perform at least one service on a line. A service is a complete line cycle carried out by a vehicle assigned to that line. According to the line's layout, the vehicle stops at or just visits the line stops. Lines under consideration in the model can be of the express service type, i.e., point-to-point lines or also lines whose vehicles halt only at some of the stops along their route. It is assumed that a fleet of existing vehicles can be assigned to the set of all lines and that there exists the possibility of renting/purchasing a specified number of additional ones if so required.

A complementary transit mode (COM) will be considered in order to allow transfers among lines so that passengers can reach their destination or nearest boarding stop.

The problem described above will be modeled and solved by means of mathematical programming techniques under the following additional hypothesis:

- The line topology could be rectilinear or circular. Additionally, whatever the topology is, lines must be also symmetrical, i.e., the line trace starts and ends at the same stop node and all links and nodes are visited twice, one in each direction.
- The fleets of vehicles are homogenous, i.e., all vehicles have the same capacity. Also, the average speed on a link is a characteristic common to all vehicles.
- Passengers travel following a system optimum assignment model so that the global user times will be minimized. These times will include not only the in-vehicle travel times but also boarding and alighting from vehicle times and waiting in-vehicle times during service at service nodes. To obtain these times, an extended flow network will be modeled (see section 2.2 for further information).
- The number of vehicles allocated on a line will be treated as discrete variables.
- The maximum line frequency depends on the stretch capacity, measured as the maximum number of vehicles going through per time unit.
- The passenger demand for the urban area under study is known in advance for every origin and destination stop. However, when we refer to elastic demand, the whole OD demand pair should not be necessarily assigned to the rapid transit system. Some of them can go through the private transportation mode.
- The average passenger service time at stops as well as the public vehicles average speed per link are also known in advance.

2.2 Network structure

The passenger flow goes through a directed graph $G = \{N, A\}$, which is split into three subgraphs: the public transport network $G_{TP} = \{N_{TP}, A_{TP}\}$, the complementary network $G_{COM} = \{N_{COM}, A_{COM}\}$, and the private network $G_{PRI} = \{N_{PRI}, N_{PRI}\}$. The G_{TP} topology represents a simplification of a rapid transit system under the following hypothesis:

- The nodes $i \in N_{TP}$ work as origin or destination nodes of the demand as well as stop nodes. Consequently, there can be a passenger flow exchange at any node. Moreover, these nodes turn into network flow extensions in order to represent the passenger flow exchanges (see Figures 2.2 and 2.3).
- The links $(i, j) \in A_{TP}$ are directed. However, for routing purposes, we work with a non-directed subgraph $G^N_{TP} = \{N^N_{TP}, A^N_{TP}\}$, where $N^N_{TP} \subset N_{TP}$ and $A^N_{TP} \subset A_{TP}$ in which $\forall (i, j) \in A^N_{TP}$:

$i < j$ (see Figure 2.1). In that manner, we build symmetrical lines, where each node i is visited twice, once in each direction. Despite that simplification, the traveling costs $t^{TP} : \{t_{ij}^{TP}\}$ are not necessarily symmetrical, so that, $t_{ij}^{TP} \neq t_{ji}^{TP}$.

As a consequence of the second hypothesis, the degree of a node $i \in N_{TP}$ satisfies that $|I_{TP}(i)| = |E_{TP}(i)| \leq 2$, where $I_{TP}(i)$ and $E_{TP}(i)$ stand for the sets of incoming and outgoing nodes. As a result, a line l is defined as a set of K 2-connected links, which form subtours in G_{TP} so that link $(i, j) \in A_{TP}$ has its opposite $(j, i) \in A_{TP}$.

Regarding G_{COM} topology, the links $(i, j) \in A_{COM}$ represent the facilities where passengers may transfer, whereas the nodes $i \in N_{COM}$ can work as nodes $i \in N_{TP}$ or as street intersection nodes. The traveling costs $t^{COM} : \{t_{ij}^{COM}\}$ may not be symmetrical, so that: $t_{ij}^{COM} \neq t_{ji}^{COM}$.

Finally, the G_{PRI} network represents a simplification of the road network used by the users who are willing to travel by car. The nodes $i \in N_{PRI}$ stand for all the origin and destination nodes where the PT stops are located. Regarding the links $(i, j) \in A_{PRI}$, they play the role of the shortest paths linking the origin demand node i to the destination demand node j.

Figure 2.1 depicts the type of network used for routing purposes. It contains 9 nodes which can work as stations, excluding node 2, which is a street intersection. Moreover, it has two types of link traces: the thicker ones, which represent the links that may share all the subgraphs; and the thinner ones, which symbolize the links that belong only to the G_{COM} and G_{PRI} subgraphs.

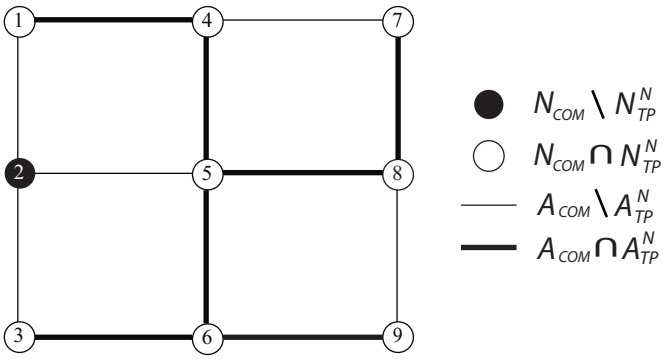


Figure 2.1: A routing network example.

Notice that it is possible to set a rectilinear line linking, for example, nodes 1 to 9. A possible line containing these nodes may be $1 - 4 - 5 - 6 - 9$. However, this network representation is not enough to deal with the passenger flow assignment because of the following reasons:

1. It has no way of circulation and thus the itinerary of a pair of o-d demand (i, j) cannot be directly distinguished from its reverse (j, i).

2. It has no flow exchange representation at a stop node. When a vehicle halts at a node, some passengers remain in-vehicle, whereas others alight from or board this vehicle, coming from or going out through a walking link.

3. As an extended consequence of the two aforementioned reasons, passenger transfers among lines cannot be well represented. Times for alighting from a vehicle at line l and boarding another line l' will be missed.

To correctly represent the demand assignment in the public transportation network, we have extended the routing graph, such that the aforementioned requirements can be satisfied. Taking as example the one depicted in Figure 2.1, the network flow representation will be like the one shown in Figure 2.2.

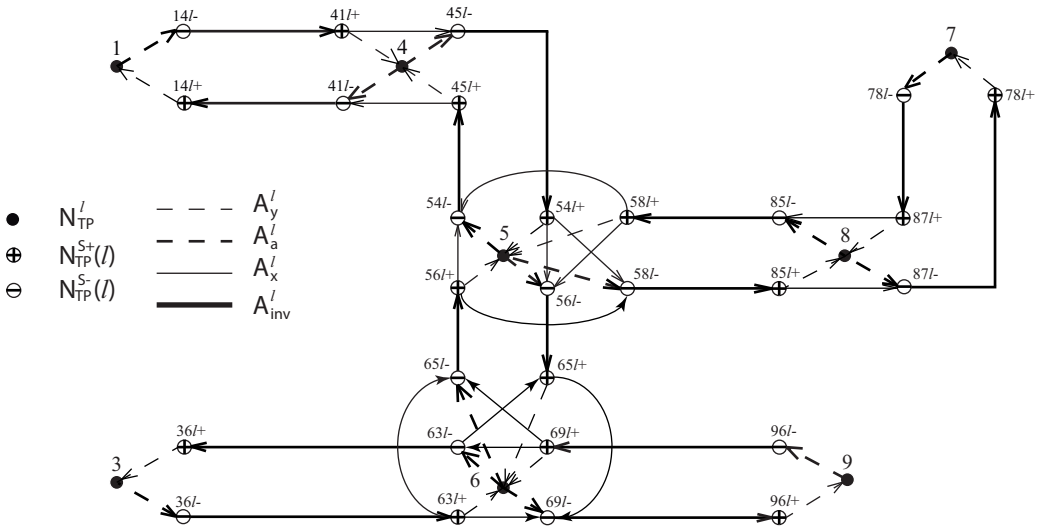


Figure 2.2: Extension of the Routing graph example to carry out the passenger flow assignment.

For the sake of clarification, we have deleted the A_{COM} and A_{PRI} links. Additionally, we have only drawn the network extension for a single candidate line l. The reader can see that two new types of nodes and three types of directed links have come into play. The links come in the "positive" nodes and in the black ones; and go out through the negative nodes and through the black ones. The latter are shared by all the network extension nodes and represent the passenger flow exchange between the G_{PT} and G_{COM} subgraphs. So, these black nodes represent the origin and destination of the passenger flow . The "positive" and "negative" nodes are fictitious and represent the in-vehicle passenger flows coming in and going out of the station. They are tagged as follows. In the first place, the "positive" nodes take the black node identifier, where the alighting link $(i, j) \in A^l_y$ and some waiting-in-vehicle links $(i, j) \in A^l_x$ are connected. In the second place, we add the black node identifier related to the "negative" node, which is linked by means of an in-vehicle traveling link $(i, j) \in A^l_{inv}$. Finally, we append to them the line identifier to which the node is related and the positive sign "+". For instance, if we want to represent the "positive" node of station 5 (black node 5), which is linked to the negative node of station 4, its tag will be $54l+$. To tag the "negative" node, the procedure is quite similar, the only difference is the replacement of the positive sign with the minus one. Moreover, the first black node to be identified is linked by means of a boarding link $(i, j) \in A^l_a$ and some waiting-in-vehicle links.

To see the details of the passenger flow exchange in these extended station nodes, the reader is directed to the following Figure 2.3. It shows the network extension for a given (black) node $i \in N^l_{TP}$, which has six adjacent black nodes: three of these are incoming nodes (h, s, $j \in N^l_{TP}$) and the other three are outgoing ones (g, r, $k \in N^l_{TP}$). To simplify, we have limited node i to be located on a new single line l. Moreover, we have only taken into account flows which originate at a unique node p. Observe that we have used an abstract notation, where the letters stand for the black node identifiers. The nodes ending with a "+" sign represent the positive nodes, whereas the ones ending with a "-" sign are related to the "negative" nodes. Regarding flow notation, the in-vehicle passenger flows coming in to and going out of the station are represented by $v^{p,l}_{v^+(i)}$ and $v^{p,l}_{v^-(i)}$, respectively. The former are split into three types of flows: the passenger flow alighting from the vehicle ($v^{p,l}_{y(i)}$), the passengers remaining in-vehicle ($v^{p,l}_{x(i)}$) and the ones remaining in a vehicle which does

not halt at a stop ($\tilde{v}^{p,l}_{x(i)}$). The latter are also split into three types of flows: the passenger flow boarding the vehicle, $v^{p,l}_{a(i)}$ and the already mentioned $v^{p,l}_{x(i)}$ and $\tilde{v}^{p,l}_{x(i)}$.

Flows $v^{p,l}_{y(i)}$ and $v^{p,l}_{a(i)}$ are also used, in collaboration with walking passenger flows coming in to and going out of the station $u^{p}_{i^-}$ and $u^{p}_{i^+}$, respectively. This makes the passenger flow exchange possible between the G_{PT} and G_{COM} subgraphs at node i.

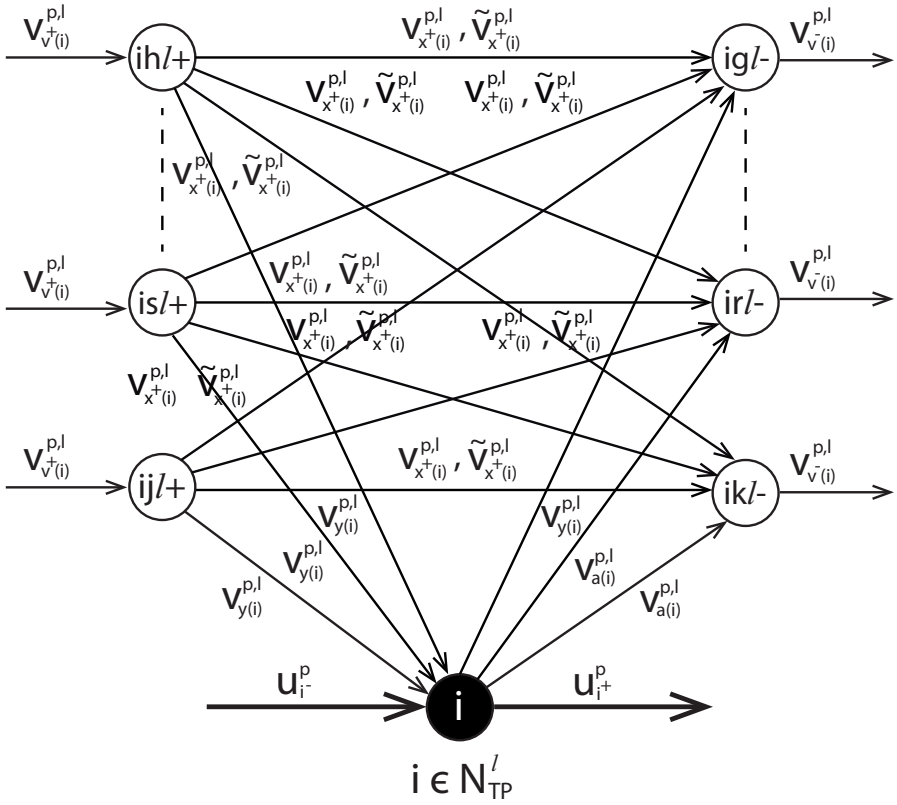


Figure 2.3: Network flow representation for a candidate node i on a new line l.

Passenger flows $v^{p,l}_{y(i)}$, $v^{p,l}_{a(i)}$ and $v^{p,l}_{x(i)}$ have time unit costs t_y, t_a and t_x, respectively, whereas flows $\tilde{v}^{p,l}_{x(i)}$ have no extra cost. Thus, this representation allows us to distinguish a twofold role of a node. Notice that if a node is treated as a passing point for some line l, some $\tilde{v}^{p,l}_{x(i)} > 0$ and all $v^{p,l}_{x(i)} = v^{p,l}_{y(i)} = v^{p,l}_{a(i)} = 0$ so $v^{p,l}_{inv^+(i)} = v^{p,l}_{inv^-(i)}$. In contrast, if the node works as a service stop, every $\tilde{v}^{p,l}_{x(i)} = 0$ and all $v^{p,l}_{x(i)} + v^{p,l}_{y(i)} = v^{p,l}_{x(i)} + v^{p,l}_{a(i)} > 0$. It is worth-noticing that $t_x < t_a + t_y$, so that it prevents pax flows coming from the same $p \in O$ from alighting and boarding a vehicle at the same stop.

The reader is referred to the illustrative example 1 of the appendix for a complete understanding of the need for including these in-station movements.

2.3 Model formulation

This section presents the mathematical formulation for the $RTNPD$ model under inelastic demand, where the passenger flows are expressed for each origin of demand. They can be easily accommodated to work with destinations; however, this step will be omitted for the sake of simplification. The objective of the model is twofold: it designs the infrastructure of the new lines to be constructed as well as the services to be

performed. The following subsections introduce the mathematical notation as well as the objective function and constraints of the model.

2.3.1 Objective Function

$$z = \min \alpha \cdot z_{pax} + (1 - \alpha) \cdot z_{op} \tag{2.1}$$

The objective function (2.1) represents a tradeoff between the passenger costs (z_{pax}) and operator costs (z_{op}). The former includes the traveling costs throughout the G_{COM} and G_{TP} subgraphs (2.2), whereas the latter comprises the planning costs related to the line frequencies and vehicles, as well as the construction and maintenance costs of the network infrastructure (2.3).

$$z_{pax} = \theta \sum_{p \in O} g_p \sum_{(i,j) \in A} \left(\sum_{l \in L_{ij}} t_{ij}^{TP} \cdot v_{ij}^{p,l} + t_{ij}^{COM} \cdot u_{ij}^{p} \right) \tag{2.2}$$

$$\begin{aligned} z_{op} = & \sum_{i \in N_{TP}^N} \left(c_i^c \cdot y_i + c_i^m \sum_{l \in L^N} \tilde{y}_i^l \right) + \sum_{(i,j) \in A_{TP}^N} \left(c_{ij}^c \cdot x_{ij} + c_{ij}^m \sum_{l \in L^N} x_{ij}^l \right) + \\ & + \sum_{(i,j) \in A_{TP}} \sum_{l \in L_{ij}} c_{ij}^f \cdot \tilde{f}_{ij}^l + c_b^s \cdot \sum_{l \in L} b^l + c_b^a \cdot \Delta b \end{aligned} \tag{2.3}$$

The t_{ij}^{TP} costs include the times associated with the in-vehicle links as well as the times for boarding, remaining in, and alighting from a vehicle. On the other hand, the t_{ij}^{COM} costs represent the shortest walking times between nodes.

The weighting of each cost is carried out as follows. On the one hand, the costs related to passenger times (2.2) are weighted by parameter θ, defined as the value of time for any OD-demand pair. On the other hand, the costs related to operators (2.3) are comprised of the following terms. Firstly, the line construction costs take into account the amount of money required to build the stops and stretches (c_i^c and c_{ij}^c). Secondly, the cost for resource maintenance includes the amount of money spent to maintain the stops and stretches in good condition (c_i^m, c_{ij}^m). Thirdly, the costs related to the planning phase, which include the frequency weights c_{ij}^f, defined as the amount of money spent per unit of time utilization through stretch (i, j) and the cost of allocating and obtaining additional vehicles (c_b^s, c_b^a), respectively. Finally, the global costs z_{pax} and z_{op} are weighted by means of non-negative constant α.

2.3.2 Infrastructure Budgetary constraint

$$\sum_{i \in N_{TP}^N} c_i^c \cdot y_i + \sum_{(i,j) \in A_{TP}^N} c_{ij}^c \cdot x_{ij} \leq \bar{c}_{net} \tag{2.4}$$

Constraint (2.4) sets the maximum number of infrastructure resources to be constructed according to the available infrastructure budget ($\bar{c}_{net}$) and the number of constructed resources. The latter is captured by decisional variables y_i and x_{ij}, which stand for the number of constructed stations and stretches, respectively.

2.3.3 Network design constraints

This subsection presents three different formulations for the network design submodel. The first two describe exact formulations whereas the latter is an approximation, but it is the only one that seems capable of solving

medium-sized networks according to the preliminary results shown in section 2.4. All these formulations are subject to the following common constraints:

$$\tilde{y}_i^l \leq y_i, \quad \forall i \in N_{TP}^N,\ l \in L^N \tag{2.5}$$
$$x_{ij}^l \leq x_{ij}, \quad \forall (i,j) \in A_{TP}^N,\ l \in L^N \tag{2.6}$$
$$\tilde{y}_i^l \leq y_i^l, \quad \forall i \in N_{TP}^N,\ l \in L^N \tag{2.7}$$

Constraints (2.5)-(2.6) capture which stretches and stops have been built in the public transport network. Additionally, (2.7) sets the correct role of the stop according to their allocation to the lines.

2.3.3.1 Routing Model M1

This first formulation stands for the routing of lines without fixing in advance their layouts. Consequently, we take into account the possible line topologies (rectilinear or circular) as well as the symmetric property (i.e., all stretches and stops will be visited twice, one in each direction). Given these requirements, the routing model ($M1$) can be formulated by means of the following constraints:

$$\sum_{j \in N_{TP}^N(i), i<j} x_{ij}^l + \sum_{j \in N_{TP}^N(i), i>j} x_{ji}^l \geq y_i^l, \quad \forall i \in N_{TP}^N,\ \forall l \in L^N \tag{2.8}$$
$$\sum_{j \in N_{TP}^N(i), i<j} x_{ij}^l + \sum_{j \in N_{TP}^N(i), i>j} x_{ji}^l \leq 2 \cdot y_i^l, \quad \forall i \in N_{TP}^N,\ \forall l \in L^N \tag{2.9}$$
$$\sum_{(i,j) \in A_{TP}^N, i<j} x_{ij}^l \leq M_l \cdot x^l, \quad \forall l \in L^N \tag{2.10}$$
$$\sum_{(i,j) \in A_{TP}^N, i<j} x_{ij}^l \geq \sum_{i \in N_{TP}^N} y_i^l - x^l, \quad \forall l \in L^N \tag{2.11}$$
$$\sum_{(i,j) \in A_{TP}^N, i<j} x_{ij}^l \leq \sum_{i \in N_{TP}^N} y_i^l, \quad \forall l \in L^N \tag{2.12}$$
$$\sum_{(r,s) \in \delta_S} x_{rs}^l \geq y_i^l + y_j^l - 1, \quad \forall i \in S,\ \forall j \in S_c,\ S \subset N_{TP}^N,\ \forall l \in L^N \tag{2.13}$$

where constraints (2.8) - (2.9) set the grade of the node and link the line stretches to their stops. In both equations, the set $N_{TP}^N(i)$ denotes the number of nodes which are adjacent to node i, and expressions $i < j$ and $i > j$ indicate that the cardinality of the adjacent nodes under consideration must be less than and greater than node i, respectively. The next constraint (2.10) limits the maximum number of stretches that can be assigned to a line by means of the binary variable x^l, which denotes if line l is constructed, and the weight M_l, whose value can be set as $0.5 \cdot |A_{TP}^N| + 1$.

On the other hand, constraints (2.11) - (2.12) establish the line topology. Notice that when constraint (2.11) has a strict equality and constraint (2.12) has a strict inequality, the line topology is rectilinear (i.e., the number of stretches is one unit greater than the number of stops). On the other hand, when the line topology is circular (i.e., the number of nodes and links are equal), constraint (2.12) has a strict equality and constraint (2.11) has a strict inequality.

Constraints (2.8) - (2.12) are not enough to carry out the routing correctly since they do not prevent disconnections among selected stretches. To overcome these situations, we add constraints (2.13) dynamically as long as sublines emerge. In equation (2.13), δ_S stands for the set of edges that are incident to a group of

interconnected nodes (captured by set S), which are disconnected from the remaining selected nodes (held in set S_c). Formally, $\delta_S = \left\{ \forall (i,j) \in A_{TP}^N \mid \ i \in S,\ j \in S_c \right\}$

To illustrate the workings of constraint (2.13), let us consider a solution consisting of two groups of interconnected edges called Subline A and Subline B for a complete 6-node railway network with only one line under construction (see Figure 2.4). Observe that we must consider the possibility of allowing subline A, which alone forms a circular layout, to be a valid line topology solution. To eliminate this disconnection, it suffices to choose a single pair of disconnected nodes and set its corresponding constraint (2.13). If we take, for instance, the pair $(1,3)$, we yield the following constraint:

$$x_{12} + x_{13} + x_{16} + x_{24} + x_{34} + x_{46} + x_{25} + x_{35} + x_{56} \geq y_1 + y_3 - 1 \tag{2.14}$$

which states that if we include nodes 1 and 3 on the line ($y_1 = 1$ and $y_3 = 1$), at least one edge with an extreme node in the group of interconnected nodes $S = \{1,4,5\}$ must have the other extreme node in the set of selected and disconnected edges $S_c = \{2,3,6\}$.

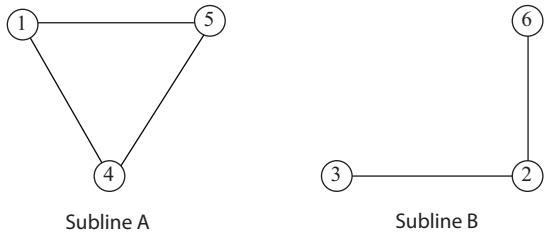


Figure 2.4: Example of a non-valid line trace generated from the 6-node network.

2.3.3.2 Routing Model M2

This second formulation stands for the routing of lines without fixing the layout in advance; but unlike the routing model M1, it explicitly includes the elimination of sublines within a line under construction by means of a one-commodity network flow formulation. It is inspired by Gavish & Graves 1979 [84], who discovered a valid compact formulation for the symmetrical traveling salesman problem ($STSP$). Its mathematical representation is as follows:

$$\min_{x_{ij}} \quad \sum_{(i,j)\in A} c_{ij} \cdot x_{ij} \tag{2.15}$$

$s.t.$

$$\sum_{j \in E_{TP}^N(i)} x_{ij} + \sum_{j \in I_{TP}^N(i)} x_{ji} = 2, \quad \forall i \in N \tag{2.16}$$

$$\sum_{j \in E_{TP}^N(i)} y_{ij} - \sum_{j \in I_{TP}^N(i)} y_{ji} = b_i, \quad \forall i \in N \tag{2.17}$$

$$y_{ij} \leq (|N| - 1) \cdot \tilde{x}_{ij}, \quad \forall (i,j) \in A^{'} \tag{2.18}$$

where b_i and $\tilde{x}_{ij}$ stand for the following expressions:

$$b_i = \begin{cases} |N| - 1 & \text{if} \quad i = 1 \\ -1 & \text{if} \quad i \neq 1 \end{cases}$$

$$\tilde{x}_{ij} = \begin{cases} x_{ij} & \text{if} \quad i < j \\ x_{ji} & \text{if} \quad i > j \end{cases}$$

Unlike the classical traveling salesman problem (TSP) formulation, two types of variables are taken into account: the original discrete variables x_{ij} representing whether or not a link is selected $(i,j) \in A$ and the new non-negative continuous variables y_{ij} representing the flow through the link $(i,j) \in A^{'}$. Set A refers to the undirected graph links, whereas set $A^{'}$ is related to the directed version. Regarding the constraints set, equations (2.16) state that the grade of a node must be equal to one, whereas equations (2.17) perform the flow balance at each node. This balance consists of injecting $|N-1|$ units of flow into the source node, which is set arbitrarily to any node (for instance, node 1). In the remaining nodes, one unit of that flow must be consumed. Finally, equations (2.18) link the flows to the network configuration. They are called the "forcing" constraints because they prevent the emergence of subtours.

We have developed an adaptation of the constraints set (2.16) - (2.18) for routing model M2 as follows. Firstly, we have replaced the node grade constraints (2.16) with the constraints (2.8) and (2.9) of model M1, since we want to construct both circular and rectilinear lines. Secondly, we have directly incorporated the forcing constraints (2.18). Finally, we have constructed a valid flow balance left term b_i to adapt the flow balance constraints (2.17). It includes the definition of two additional variables ϕ_i^l and Γ_i^l. The former defines which node will be the source, whereas the latter determines which node will be considered as source and sink at the same time, providing that the constructed line is circular. Otherwise, that variable is meaningless. So, by taking into account the y_i^l variables, which define which nodes are allocated on the line, we formulate the remaining constraints of routing model M2 as follows:

$$\sum_{j \in E_{TP}^N(i)} z_{ij}^l - \sum_{j \in I_{TP}^N(i)} z_{ji}^l = n_i^l + \phi_i^l - \Gamma_i^l - y_i^l, \quad \forall i \in N_{TP}^N,\ l \in L^N \tag{2.19}$$

$$\sum_{i \in N_{TP}^N} \phi_i^l = x^l, \quad \forall l \in L^N \tag{2.20}$$

$$\phi_i^l \leq y_i^l, \quad \forall i \in N_{TP}^N,\ l \in L^N \tag{2.21}$$

$$\Gamma_i^l \leq \phi_i^l, \quad \forall i \in N_{TP}^N,\ l \in L^N \tag{2.22}$$

$$n_i^l = \sum_{k \in N_{TP}^N} y_k^l \cdot \phi_i^l, \quad \forall i \in N_{TP}^N,\ l \in L^N \tag{2.23}$$

$$z_{ij}^l \leq (|N_{TP}^N| - 1) \cdot \tilde{x}_{ij}^l, \quad \forall (i,j) \in A_{TP}^N,\ l \in L^N \tag{2.24}$$

where we have labeled the flow variables y_{ij}^l as z_{ij}^l for the sake of clarification. Observe that constraints (2.19) are analogous to the flow balance constraints (2.17), except that the right term b_i of the equality is replaced with an expression in which several decision variables appear. The n_i^l variable is associated with the linearization of constraint (2.23). It simply states that the node i will have an injection of $\sum_{k \in N_{TP}^N} y_k^l$ units of flow, provided that it is the source. Otherwise it will have no injection. That linearization is carried out by means of the simplified Grover's equations (9)-(10) (shown in the appendix section 9.3), since there is one binary variable ϕ_i^l alone in the product. Having applied them, we yield the following constraints:

$$0 \leq n_i^l \leq \phi_i^l \cdot |N_{TP}^N|, \quad \forall i \in N_{TP}^N,\ l \in L^N \tag{2.25}$$

$$\sum_{k \in N_{TP}^N} y_k^l - |N_{TP}^N| \cdot (1 - \phi_i^l) \leq n_i^l \leq \sum_{k \in N_{TP}^N} y_k^l, \quad \forall i \in N_{TP}^N,\ l \in L^N \tag{2.26}$$

To understand the workings of constraints (2.19) - (2.24), we refer the reader to the following scenarios:

1. A single line l is constructed as rectilinear and not circular.

2. A single line l is constructed as rectilinear and circular.

The first scenario implies that no node can work as source and sink at the same time. Consequently, all $\Gamma_i^l = 0$. However, there must be one node working as the source, i.e., one $\phi_i^l = 1$. This is carried out by means of constraints (2.20) and (2.22). Notice that constraints (2.22) do not force any ϕ_i^l to be non-zero. They just simply force the node that is considered a sink to work also as a source. However, adding constraints (2.21) assures that the node considered as the source will be assigned to the line. Consequently, if node i is the source, the right side expression of balance flow constraint (2.19) will take the value $\sum_{k \in N_{TP}^N} y_k^l - 1$. In contrast, if the node i is not the source but is part of the line, the expression will take the -1 value. Finally, if the node i is neither the source nor a node of the line, the expression will take the 0 value. Thus, it allows creation of a fictitious flow throughout the line trace.

Regarding the second scenario, the reasoning is quite similar. Now when a node i is considered to be the source, it works also as the sink. So $\phi_i^l = \Gamma_i^l = 1$. In that case, the right side expression of flow balance constraint (2.19) will take the value $\sum_{k \in N_{TP}^N} y_k^l - 2$, which implies that only $\sum_{k \in N_{TP}^N} y_k^l - 1$ links will carry flow and are thus allocated to the line. The remaining possible situations are equivalent to the rectilinear case.

Notice that the adapted flow balance constraints (2.19) also assure that every node will be connected; and since constraints (2.24) prevent the emergence of sublines, the connectivity constraints (2.11) - (2.12) as well as the subline elimination constraints (2.13) of routing model M1 are not needed.

2.3.3.3 Model with fixed corridors (M3)

This third formulation stands for the routing of lines with fixed line segments (corridors), which are computed off-line by means of the algorithm shown in chapter 4. As a result, we yield the following constraints set:

$$x_{ij}^l = \sum_{c \in \Lambda} D_{ij}^c \cdot \delta_c^l, \quad \forall i \in A_{TP}^N,\ l \in L^N \tag{2.27}$$

$$y_i^l = \sum_{c \in \Lambda} E_i^c \cdot \delta_c^l, \quad \forall i \in N_{TP}^N,\ l \in L^N \tag{2.28}$$

$$\sum_{c \in \Lambda} \delta_c^l \leq 1, \quad \forall l \in L^N \tag{2.29}$$

where (2.27)-(2.28) map the chosen corridor setup to the stretches and nodes to be included in a given line l, whereas constraint (2.29) establishes that only one corridor can be selected per line. The matrices D and E hold all the 0-1 combinations of x_{ij}^l and y_i^l variables, respectively. So, the matrix column vector D^c stands for the x_{ij}^l values for corridor c, whereas E^c relates to the y_i^l values for corridor c. The binary variable δ_c^l denotes whether or not corridor c takes part of the line l.

2.3.4 Line Frequency setting constraints

The line frequency setting is based on the following law:

$$b^l \cdot \hat{h} \geq z^l \cdot c^l \tag{2.30}$$

where variables b^l, z^l and c^l represent the number of vehicles, services and cycles of line l, respectively. Additionally, parameter $\hat{h}$ stands for the planning horizon. From this law, one can see that the optimal planning solution is to strictly set the number of vehicles needed to carry out the z^l services on the line while not exceeding $\hat{h}$. Despite its simplicity, this rule has a cumbersome implementation. Notice that the c^l associated to lines under construction is not known beforehand, so a non-linearity given by the product of z^l and c^l emerges. Moreover, variables z^l are integer, and thus this product cannot be easily linearized.

To work with a tractable planning model, we have reformulated the z^l variables as the number of vehicles per time unit (f^l). This variable is continuous and its relationship with z^l is given by the following equation:

$$z^l = \hat{h} \cdot f^l \tag{2.31}$$

Consequently, the aforementioned law (2.30) turns into the following:

$$b^l \geq f^l \cdot c^l \tag{2.32}$$

where the non linearity $f^l \cdot c^l$ can now be linearized in a straightforward manner by means of the Grover approach stated in the Appendix section 9.3.

The following constraints sets (2.33) - (2.38) implement the linearized version of (2.32) as well as some additional planning requirements related to the fleet size (B) and the planning budget ($\overline{c}_{veh}$).

$$\sum_{l \in L} b^l - B \leq \Delta b \leq \left\lfloor \frac{\overline{c}_{veh}}{c_b} \right\rfloor \tag{2.33}$$

$$b^l \geq \sigma^l, \quad \forall l \in L \tag{2.34}$$

$$\sum_{l \in L_{ij}} \tilde{f}^l_{ij} \leq \overline{f}_{ij}, \quad \forall (i,j) \in A_{TP} \tag{2.35}$$

$$c^l \leq \hat{h}, \quad \forall l \in L^N \tag{2.36}$$

$$f^l_{lay} = \Delta^l_{yx} \cdot f^l, \quad \forall l \in L^N \tag{2.37}$$

$$f^l_i = \tilde{y}^l_i \cdot f^l, \quad \forall i \in N^N_{TP},\ l \in L^N \tag{2.38}$$

$$f^l_{ij} = x^l_{ij} \cdot f^l, \quad \forall (i,j) \in A^N_{TP},\ l \in L^N \tag{2.39}$$

Firstly, (2.33) sets the lower and upper bounds on the number of new vehicles to be acquired (Δb), given the available planning budget, the current size of the fleet of vehicles and the number of vehicles assigned to the lines. Secondly, (2.34) establishes a lower bound on b^l by means of σ^l, which is equivalent to the following expression:

$$\sigma^l = \begin{cases} \displaystyle\sum_{(i,j) \in A^N_{TP}} t^{TP}_{ij} \cdot f^l_{ij} + 2 \cdot \left(t_s \sum_{i \in N^N_{TP}} f^l_i + t_l \cdot f^l_{lay} \right) & \text{if} \quad l \in L^N \\ f^l \cdot \displaystyle\sum_{(i,j) \in A^E_{TP}(l)} t^{TP}_{ij} + 2 \cdot f^l \cdot \left(t_s \cdot |N^S_{TP}(l)| + t_l \cdot \Delta^l_{yx} \right) & \text{if} \quad l \in L^E \end{cases} \tag{2.40}$$

Equation (2.40) consists of the product $f^l \cdot c^l$, which is directly implemented for the working lines because the travel times of the stretches (t^{TP}_{ij}), the average service stop times (t_s) and the layover time (t_l) are known in advance. The latter is active if the topology of the line is rectilinear. This effect is carried out by means of Δ^l_{yx}, whose mathematical expression is as follows:

$$\Delta^l_{yx} = \begin{cases} \displaystyle\sum_{i \in N^N_{TP}} y^l_i - \sum_{(i,j) \in A^N_{TP}, i<j} x^l_{ij} & \text{if} \quad l \in L^N \\ |N_{TP}(l)| - A'_{TP}(l) & \text{if} \quad l \in L^E \end{cases} \tag{2.41}$$

According to this formula, $\Delta^l_{yx} \longrightarrow 1$ when the line is rectilinear (i.e., $|N_{TP}(l)| = A'_{TP}(l)+1$), whereas $\Delta^l_{yx} \longrightarrow 0$ when the line is circular (i.e., $|N_{TP}(l)| = A'_{TP}(l)$). Set $A'_{TP}(l) = \{(i,j) \in A_{TP}(l) \mid i < j\}$.

Other values for Δ^l_{yx} are not allowed since they would not satisfy some of the line's topology requirements (see sections 2.1 and 2.2 for further details).

Regarding the lines under construction, the product $f^l \cdot c^l$ is reformulated as the weighted sum of selected stretches and service stops, plus the frequencies of vehicles changing ways (f^l_{ij}, f^l_i and f^l_{lay} respectively). These frequencies are linked to the those of the line (f^l) by means of constraints (2.37) - (2.39), which are non-linear. However, since in both right side expressions one of the variables or expressions are made of binary variables ($\tilde{y}^l_i$, y^l_i, x^l_{ij}) and line frequencies f^l are continuous non-negative variables, they can be linearized in a straightforward manner by means of the simplified Grover equations (9)-(10), shown in the appendix section 9.3, as follows:

$$\overline{f}_i \cdot \tilde{y}^l_i \geq f^l_i \geq 0, \quad \forall i \in N^N_{TP},\ l \in L^N \tag{2.42}$$

$$f^l \geq f^l_i \geq f^l - \overline{f}^l \cdot (1 - \tilde{y}^l_i), \quad \forall i \in N^N_{TP},\ l \in L^N \tag{2.43}$$

$$\overline{f}_{ij} \cdot x^l_{ij} \geq f^l_{ij} \geq 0, \quad \forall (i,j) \in A^N_{TP},\ l \in L^N \tag{2.44}$$

$$f^l \geq f^l_{ij} \geq f^l - \overline{f}^l \cdot (1 - x^l_{ij}),\ \forall (i,j) \in A^N_{TP},\ l \in L^N \tag{2.45}$$

$$\overline{f}^l \cdot \Delta^l_{yx} \geq f^l_{lay} \geq 0, \quad \forall l \in L^N \tag{2.46}$$

$$f^l \geq f^l_{lay} \geq f^l - \overline{f}^l \cdot (1 - \Delta^l_{yx}), \quad \forall l \in L^N \tag{2.47}$$

where upper bounds $\overline{f}_i$, $\overline{f}_{ij}$ and $\overline{f}^l$ establish the maximum number of vehicles per time unit at the node station i, the stretch (i,j) and the line l, respectively. Δ^l_{yx} denotes the line topology and its expression is shown above in (2.41).
Thirdly, (2.35) limits the maximum number of vehicles which can go through the stretches according to their capacities ($\overline{f}_{ij}$) and the whole frequency assigned to the link (i,j). The latter is obtained by means of function $\tilde{f}^l_{ij}$ whose mathematical expression is as follows:

$$\tilde{f}^l_{ij} = \begin{cases} f^l & \text{if} \quad l \in L^E \\ f^l_{ij} & \text{if} \quad l \in L^N \quad \text{and} \quad i < j \\ f^l_{ji} & \text{if} \quad l \in L^N \quad \text{and} \quad i > j \end{cases} \tag{2.48}$$

Finally, (2.36) prevents the line cycle of the lines under construction to exceed the planning horizon. The expression of this line cycle is as follows:

$$c^l = \sum_{(i,j) \in A^N_{TP}} x^l_{ij} \cdot \left(t^{TP}_{ij} + t^{TP}_{ji}\right) + 2 \cdot \left(t_s \sum_{i \in N^N_{TP}} \tilde{y}^l_i + t_l \cdot \Delta^l_{yx} \right) \tag{2.49}$$

2.3.5 Passenger flow balance constraints

$$\sum_{l\in L_i}\left(\sum_{j\in l_{a(i)}} v_{ij}^{p,l} - \sum_{j\in l_{y(i)}} v_{ji}^{p,l}\right) + \sum_{j\in A_{COM}(i)} u_{ij}^{p} - \sum_{j\in A_{COM}(i)} u_{ji}^{p} = t_i^p,\ \forall i\in N,\ p\in O \tag{2.50}$$

$$v_{inv^+(i)}^{p,l} = v_{y(i)}^{p,l} + \sum_{j\in A^l_{x^+(i)}} \left(v_{ij}^{p,l} + \Psi_{ij}^{p,l}\right),\quad \forall l\in L,\ i\in N_{TP}^{S+}(l),\ p\in O \tag{2.51}$$

$$v_{inv^-(i)}^{p,l} = v_{a(i)}^{p,l} + \sum_{j\in A^l_{x^-(i)}} \left(v_{ji}^{p,l} + \Psi_{ji}^{p,l}\right),\quad \forall l\in L,\ i\in N_{TP}^{S-}(l),\ p\in O \tag{2.52}$$

$$v_{inv^+(i)}^{p,l} = v_{inv^-(i)}^{p,l},\quad \forall\, l\in L^E,\ i\in N_{TP}^{P}(l),\ p\in O \tag{2.53}$$

$$\sum_{(r,s)\in A_{xya}^N(i)} v_{rs}^{p,l} \le \tilde{y}_i^l,\quad \forall l\in L^N,\ i\in N_{TP}^N,\ p\in O \tag{2.54}$$

$$\sum_{(r,s)\in A_{x}^N(i)} \tilde{v}_{rs}^{p,l} \le (y_i^l - \tilde{y}_i^l),\ \forall l\in L^N,\ i\in N_{TP}^N,\ p\in O \tag{2.55}$$

$$\sum_{p\in O} g_p\cdot v_{inv(i,j)}^{p,l} \le q\cdot \tilde{f}_{ij}^l,\quad \forall (i,j)\in A_{TP},\ l\in L_{ij} \tag{2.56}$$

Firstly, (2.50) represents the passenger flow exchange at stops between the G_{PT} and G_{COM} graphs. The constraint's right side is set by means of parameter t_i^p, whose mathematical expression is as follows:

$$t_i^p = \begin{cases} 1 & \text{if}\quad i = p(w) \\ -\frac{g_{pi}}{g_p} & \text{if}\quad i\neq p(w),\ i\in D_p \\ 0 & \text{if}\quad i\notin D_p \end{cases} \tag{2.57}$$

where parameter D_p stands for the set of demand OD-pairs which originate from p, whereas g_p and $g_{p,i}$, respectively, are related to the demands of all OD-pairs with origin in p and the demand of OD-pair (p,i). Secondly, (2.51)-(2.55) perform the passenger flow balance at stops. Constraint (2.51) balances the in-vehicle passenger flow coming in the station by means of stop extension node $i\in N_{TP}^{s,l+}(l)$, whereas (2.52) balances the in-vehicle passenger flow going out of the station by means of stop extension node $i\in N_{TP}^{s,l-}(l)$. As we don't know in advance whether or not the stop will carry out service to passengers, we introduce the function $\Psi_{ij}^{p,l}$, whose mathematical form is as follows:

$$\Psi_{ij}^{p,l} = \begin{cases} \tilde{v}_{ij}^{p,l} & \text{if}\quad l\in L^N \\ 0 & \text{if}\quad l\in L^E \end{cases} \tag{2.58}$$

Constraints (2.54)-(2.55) enable the proper type of balance by means of the binary variable $\tilde{y}_i^l$, which represents the role of the stop. In that manner, if the stop carries out service ($\tilde{y}_i^l = 1$), passenger flows will exchange (all $\tilde{v}_{rs}^{p,l} = 0,\ (r,s)\in A_x^N(i)$). In contrast, if the stops work as a passing point ($\tilde{y}_i^l = 0$), no passenger flows will exchange ($v_{rs}^{p,l} = 0,\ \forall (r,s)\in A_{xya}^N(i)$). For further details see Figure 2.3. Constraint (2.53) represents the balance continuity between those already allocated stops which do not perform any service. Finally, (2.56) establishes the maximum passenger flow traveling through an in-vehicle link, which is obtained from the directed and not extended link $(i,j)\in A_{TP}$ with the help of the mapping function $v(i,j)$. That maximum is reached when it equals the so-called line link capacity ($q\cdot\tilde{f}_{ij}^l$), defined as the vehicle's capacity q times the link line frequency $\tilde{f}_{ij}^l$.

2.3.6 Improving the problem's performance

In this section, we present some changes we have made to the model in order to speed-up its performance, especially when applying a Branch and Bound scheme ($B\&B$). It includes changing the definition of some decision variables as well as the inclusion of new constraints. In the following, we are going to explain each of them in separate subsections according to their original idea.

2.3.6.1 Breaking symmetries

When applying a Branch and Bound scheme ($B\&B$) such as the one used by the *CPLEX* solver, the integrality of some discrete variables is relaxed at each node of the $B\&B$ tree; whereas others are fixed to an integer value. This process does not detect symmetries, i.e., combinations of decisional variables that lead to the same optimal solution. Thus, it entails exploring many combinations which do not improve the optimality gap but increase the computational time. In this model, the process can be sped up by adding the following set of constraints:

$$c^l \leq c^{l-1}, \quad \forall l \in L^N \backslash \{1\} \tag{2.59}$$

which give an enumeration of the lines, such that line 1 must have a line cycle, c^1, less than or equal to the cycle of line 2, c^2, and so on. The line cycle expression c^l is given by (2.49).

2.3.6.2 Relaxing the integrality of some variables

By studying in more detail the mathematical structure of the model constraints, we have found that the integrality of some discrete variables can be either directly relaxed or relaxed with some extra constraints. For instance, the binary variables x_{ij} and y_i can be stated as continuous in the $[0, 1]$ interval if we add the following constraints:

$$y_i \leq \sum_{l \in L^N} \tilde{y}_i^l, \quad \forall i \in N_{TP}^N \tag{2.60}$$

$$x_{ij} \leq \sum_{l \in L^N} x_{ij}^l, \quad \forall (i,j) \in A_{TP}^N \tag{2.61}$$

These additional constraints prevent fractional values from being obtained when the solution is not optimal. In contrast, when we reach optimality, (2.5)-(2.6) suffice to force them to be a 0-1 integer.

Variables y_i^l can be relaxed as well. If we work with the routing model M3, which is characterized by constraints set (2.27)-(2.29), we do not need to add extra constraints. Observe that all the matrix cells E_i^c have 0-1 integer values, the corridor selection variables δ_c^l are binary, and only one of them can be active per line according to constraint (2.29). Thus, the result of the linear convex transformation (2.28) will be a 0-1 integer as well. In contrast, if we work with routing models M1 or M2, we must include the following constraints:

$$x_{ij}^l \leq y_i^l, \quad \forall (i,j) \in A_{TP}^N, \; l \in L^N \tag{2.62}$$

$$x_{ij}^l \leq y_j^l, \quad \forall (i,j) \in A_{TP}^N, \; l \in L^N \tag{2.63}$$

which, in collaboration with the node grade constraints (2.8) and (2.9), force the y_i^l variables to be a $0-1$ integer value even if the solution is feasible non-optimal.

Finally, we also notice that variables x_{ij}^l can also be relaxed providing that we work with the routing model M3. The reasoning is exactly the same as the case with the relaxation of variables y_i^l. In that case, we refer to the matrix cells D_i^c, which are used to carry out the linear convex transformation (2.27)that results in 0-1 integer values for variables x_{ij}^l.

2.3.7 Summary of the model formulation

To sum up, we finally have four groups of discrete variables: b^l, $\tilde{y}_i^l$, x_{ij}^l and x^l, if we work with routing models $M1$ and $M2$; or only three groups of discrete variables: b^l, $\tilde{y}_i^l$ and δ_c^l, if we work with routing model M3. Additionally, variable Δb should be discrete as well, although we could relax its integrality, given constraint (2.33), if we solve the model to optimality.

The $RTNPD$ model with inelastic demand is given by the objective function (2.1) - (2.3) and the following groups of constraints:

- The infrastructure budgetary constraint (2.4)
- The network design constraints (2.5)-(2.7).
- The routing constraints:
  - (2.8)-(2.12) if the routing models M1 is chosen.
  - (2.8)-(2.9), (2.19)-(2.22) and (2.24)-(2.26) if the routing model M2 is chosen.
  - (2.27)-(2.29) if the routing model M3 is chosen.
- The line frequency setting constraints (2.33)-(2.36) and (2.42)-(2.45).
- The passenger flow balance constraints (2.50)-(2.56).
- The breaking symmetry constraints (2.59).
- The relaxing integrality constraints:
  - (2.60)-(2.63) if the routing models M1 or M2 are chosen.
  - (2.60)-(2.61) if the routing model M3 is chosen.

2.4 Preliminary Results

In order to show the efficiency of each mathematical programming formulation, i.e., the type of routing model selected from sections 2.3.3.1 - 2.3.3.3 and applied to the network layout design, two test networks have been used. The first one is a complete 6-node railway network shown in Figure 8.1, whereas the other one is a 9-node and 26-link railway network depicted in Figure 8.2. We refer the reader to section 8.1.1 for further details regarding the input parameters associated with both networks.

The following table 2.1 shows the main results. All mathematical programming formulations have been coded in AMPL, using CPLEX v.12.4.0 solver. The tests have been carried out on a working station R5500 with processor Intel(R) Xeon(R) CPU E5645 2.40 GHz and 48 Gbytes of RAM.

In the table, the column Network denotes the test network used. The label $N1$ refers to the 6-node network, whereas the label $N2$ is related to the 9-node network. Column $Exp.$ stands for the experiment

Network	Exp.	$\|L\|$	R.M.	$\|\Lambda\|$	F.Obj	F. Users	T_{CPU}	N_{SEC}
N1	1	2	M1	-	184034	183225	1	0
			M2	-			2	-
			M3	1967			5	-
	2	3	M1	-	157036	155962	42	0
			M2	-			48	-
			M3	1967			82	-
	3	4	M1	-	147612	146471	1586	0
			M2	-			2543	-
			M3	1967			4392	-
N2	1	2	M1	-	782332	781481	8	3
			M2	-			3	-
			M3	295			5	-
	2	3	M1	-	758634	757608	7386	19
			M2	-			1125	-
			M3	295			42	-
	3	4	M1	-	742175	741038	> 86400	≥ 11
			M2	-			5001	-
			M3	295			1963	-

Table 2.1: General results for the experiments performed in the test networks.

identifier. The next column, $|L|$, is the maximum number of lines to be constructed, whereas column, $R.M.$ shows which routing model has been used by denoting one of the following tags: $M1$, $M2$ and $M3$. The $M1$ and $M2$ tags stand for the models with non-fixed corridors. The difference lies in the way they treat the sublines. The former does not include them in the formulation from the very beginning. Instead, they are added dynamically as needed, whereas the latter treat them in a static fashion. For details of their formulations, see sections 2.3.3.1 and 2.3.3.2, respectively. The $M3$ tag refers to the model with fixed corridors (see section 2.3.3.3 for further details). The next column $|\Lambda|$ reports the number of corridors fixed in the routing model $M3$. Columns $F.Obj$ and $F.Users$ show the total objective function (2.1) and the user costs (2.2) weighted by β, respectively. Column $T.CPU$ reports the total number of elapsed seconds for the method used in the experiment. Finally, column $N.SECs$ shows the number of subline elimination constraints (2.13) added to the model $M1$ in order to obtain a valid line trace.

To compute the number of fixed corridors in set Λ, we used the Corridor Generation Algorithm (CGA). We adapted Yen's k-shortest path algorithm [189] for the possible line topologies and added some constraint satisfaction (see chapter 4 for further information). The inputs are set according to the values shown in row three of Table 4.8 in section 4.2. The number of computed corridors allows us to obtain optimal line routing configurations that are the same as those obtained by $M1$ and $M2$ models, so that we could correctly compare the performance of model $M3$ with models $M1$ and $M2$.

Results show that as the number of lines increases, the computational times have an exponential growth. Observe that each increase of one line under construction entails consuming one extra order of magnitude in CPU time. Regarding model comparison, Model $M1$ outperforms models $M2$ and $M3$ when sublines do not appear (see the first 9 rows of columns $T.CPU$ and $N.SECs$ in Table 2.1). However, when sublines emerge, models $M2$ and $M3$ have lower computational times (see the following 9 rows of columns $T.CPU$ and $N.SECs$ in Table 2.1). Moreover, model $M3$ clearly outperforms model $M2$ for the instances where $|L^N| > 2$. It then seems more suitable to use this model rather than models $M1$ and $M2$ to cope with bigger networks.

Now, let's move on to the analysis of the solution. Figures 2.5 and 2.6 show the network layout for the 6- and 9-node networks, respectively. Notice that we have used different colors to identify each link and node belonging to the same line. Furthermore, when links and nodes are used by more than one line, different non-overlapping traces with the same width are drawn.

Let us begin by explaining the layout solution of the 6-node network (shown in Figure 2.5). In that picture, we can see that circular lines L1 and L2 coincide in experiments 1 and 2 (i.e., L1: $2-3-6-4-5-2$ and L2: $1-3-5-2-6-4-1$); whereas in experiment 3, all lines are different (i.e., three rectilinear lines, L1: $1-2-5-6-3$, L2: $1-3-2-5-4-6$, L3: $1-4-6-2-3-5$ and one circular line, L4: $1-5-2-4-3-6-1$). Moreover, whatever the experiment and topology of the line is, all nodes perform services (i.e., their corresponding $\tilde{y}_i^l = 1$).

Regarding the 9-node network layout solution (shown in Figure 2.6), we observe that rectilinear line L1 coincides in all experiments (i.e., L1: $4-6-9$). Furthermore, rectilinear line L2 is also identical in experiments 1 and 2 (i.e., L2: $1-2-3-5-4-7-6-8$). The remaining lines are different: rectilinear line L3 in experiment 2 is made up of $6-5-3-2-4-7$; whereas rectilinear lines L2, L3 and L4 in experiment 4 consist of $2-3-5-4$, $1-3-2-4-7$ and $2-1-3-4-7-6-8$, respectively. Like in the 6-node network, all nodes carry out service no matter the experiment and the topology of the line is.

Finally, Table 2.2 includes some details of the experiments regarding the behavioral aspects of the model. In the table, columns NL and NLC report the number of built lines and the ones that have a circular topology. The next two columns, NV and NZ, stand for the number of assigned vehicles and their corresponding services per horizon time carried out. The following four columns report some measures regarding the level of line utilization. The first two columns ($\overline{U}_a$ and $\hat{U}_a$) show the average and maximum used capacity of the links. This capacity corresponds to input parameter q_{ij}, which stands for the maximum number of vehicles per time unit allowed through the link (i, j). The other two columns ($\overline{U}_l$ and $\hat{U}_l$) refer to the average and maximum capacity used on the line, computed as the total number of vehicles per time unit carrying out f^l times the maximum vehicle capacity, q_l. The two final columns ($T.TP$ and $T.COM$) report the time spent by passengers that travel through the public transportation network and its complementary, respectively.

Network	Exp.	NL	NLC	NV	NZ	$\overline{U}_{ij}$	$\hat{U}_{ij}$	$\overline{U}_l$	$\hat{U}_l$	T_{TP}	T_{COM}
N1	1	2	2	5	36	77 %	100 %	100 %	100 %	5025	198558
	2	3	2	6	45	81 %	100 %	100 %	100 %	6096	167195
	3	4	1	6	54	70 %	100 %	100 %	100 %	6459	156287
N2	1	2	0	4	45	69 %	100 %	100 %	100 %	3232	865080
	2	3	0	5	54	66 %	100 %	100 %	100 %	4150	837637
	3	4	0	6	72	75 %	100 %	100 %	100 %	4956	818419

Table 2.2: Detailed results for the experiments performed in the test networks.

Results show that the model tends to reach maximum vehicle capacity since the objective function (2.1) does not take into account any term related to passenger comfort. Regarding the link capacity utilization, its level of congestion is very high because of the demand intensity (corresponding to the peak hour period), but it goes down as long as the number of constructed lines grows. On the other hand, the number of used vehicles and cycles carried out increases, while the whole time spent in the complementary network decreases (since the objective function weight α is set to 0.9, and thus much more importance is attached to passenger cost than operator costs). As a consequence, users are encouraged more and more to use the public transportation network. Observe also that $T.TP$ is always much less than $T.COM$ because the

average vehicle speed is 80 km/h, whereas the average walking passenger speed is 4.5 km/h. Thus, the complementary traveling times are 20 times greater than the public transportation times.

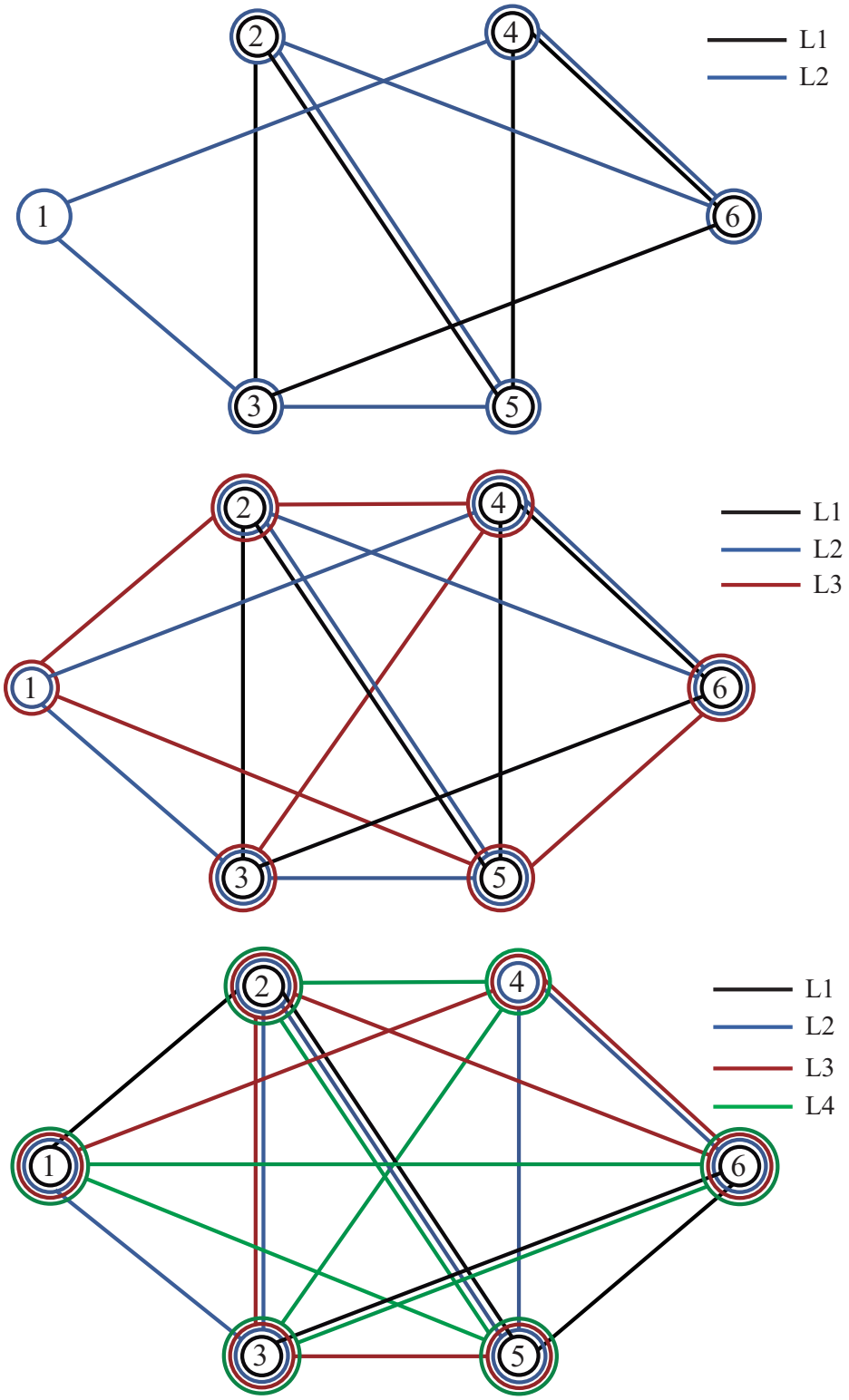


Figure 2.5: Layout configuration for the 6-node network. At the top, the layout for the experiment with two lines under construction. In the middle, the layout for the experiment with three lines under construction. At the bottom, the layout for the experiment with four lines under construction.

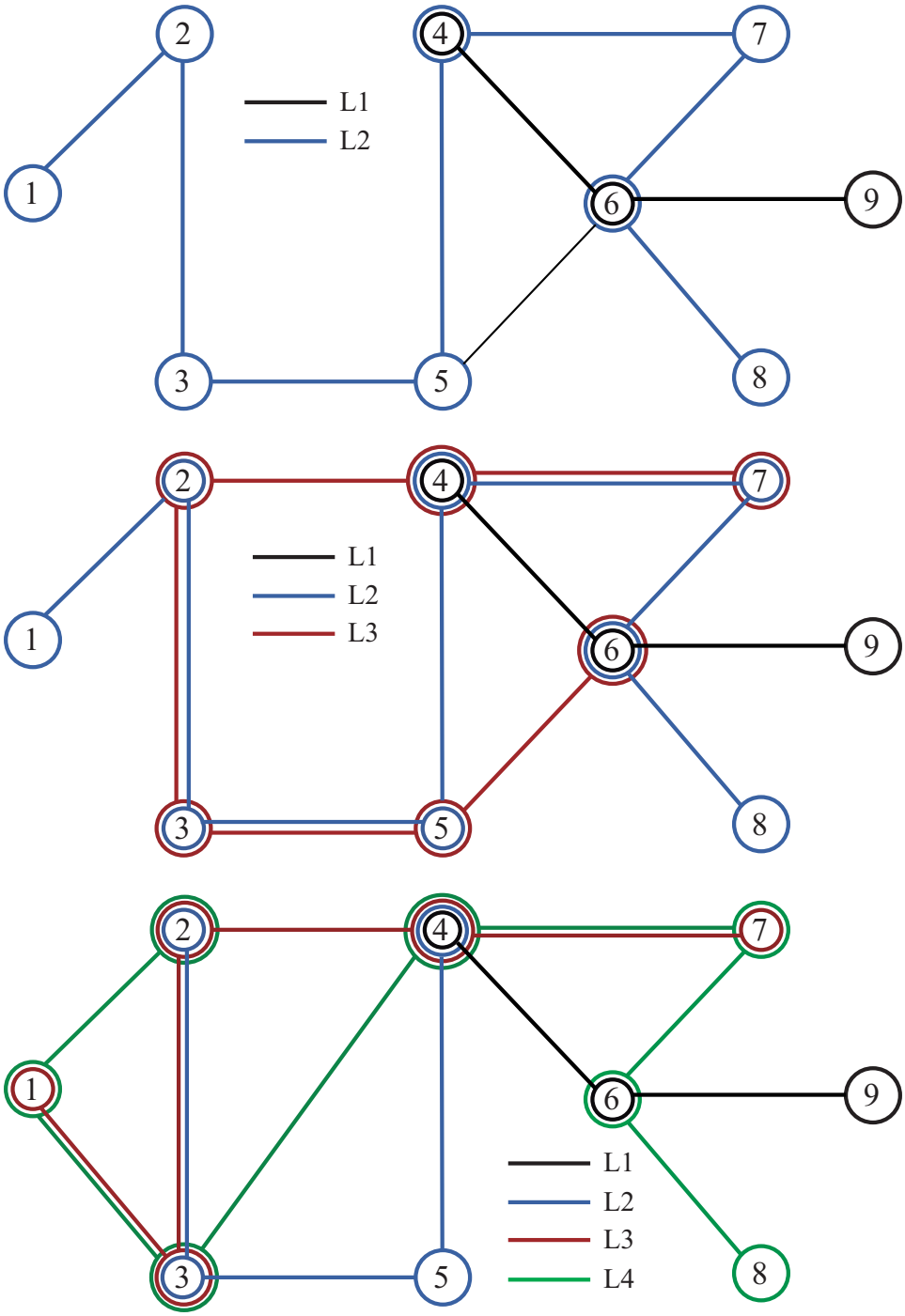


Figure 2.6: Layout configuration for the 9-node network. At the top, the layout for the experiment with two lines under construction. In the middle, the layout for the experiment with three lines under construction. At the bottom, the layout for the experiment with four lines under construction.

Chapter 3

Solution methodology

This chapter is devoted to providing an overview of the main solving blocks applied to the inelastic and elastic demand versions of the rapid transit network design and planning model ($RTNPD$), which was introduced in the previous chapter. They are a mixture of exact and heuristic techniques that allow us to obtain good near optimal solutions when coping with medium- and large-sized networks. The structure of the chapter is as follows. Firstly, we show the solution framework in which these modules are embedded, as well as their interconnections. They include the corridor generation algorithm (CGA), the line splitting algorithm (LSA) and the specialized Benders decomposition (SBD) for the $RTNPD$ model. Secondly, we lay out the skeleton of each module in the same order as they appear in the following chapters, which go into greater detail. Finally, we review the state-of-the-art techniques employed in these modules.

3.1 Solution Framework

The solution framework employed in this research is presented in Figure 3.1. It consists of three solving blocks: the corridor generation algorithm (CGA), the line splitting algorithm (LSA) and the specialized Benders decomposition (SBD) for the $RTNPD$ model. The LSA and CGA methods are heuristic techniques that allow us to skip the non-polynomial properties, which are hard properties of the mathematical programming problem. They are related to the number of lines under construction and the number of feasible corridors (line segments) that can be generated. Regarding the SBD decomposition, it is an exact method that splits the original mathematical programming problem into a series of resolutions for two mathematical problems which are easier to solve. Thus, the combination of these three techniques allows us to obtain good near optimal solutions when coping with medium- and large-sized networks.

The iteration and basic workings of these methods are as follows. The CGA is used as long as the routing model $M3$ is chosen (see section 2.3.3.3 for further details). It then determines the set of candidate corridors (line segments) from which the SBD will seek the L^N corridors that take part in the public transportation network layout. The SBD can be driven by the LSA, which splits the resolution of the $RTNPD$ model with or without a predefined set of corridors into a series of small $RTNPD$ models, where only a single new line can be constructed.

The use of CGA and LSA algorithms are optional and depend on user preferences. However for medium- and large-sized networks, we have found that obtaining good solutions within a reasonable time is only possible by using CGA and LSA algorithms (see Chapter 8 for further details). The following sections present an overview of these solving blocks.

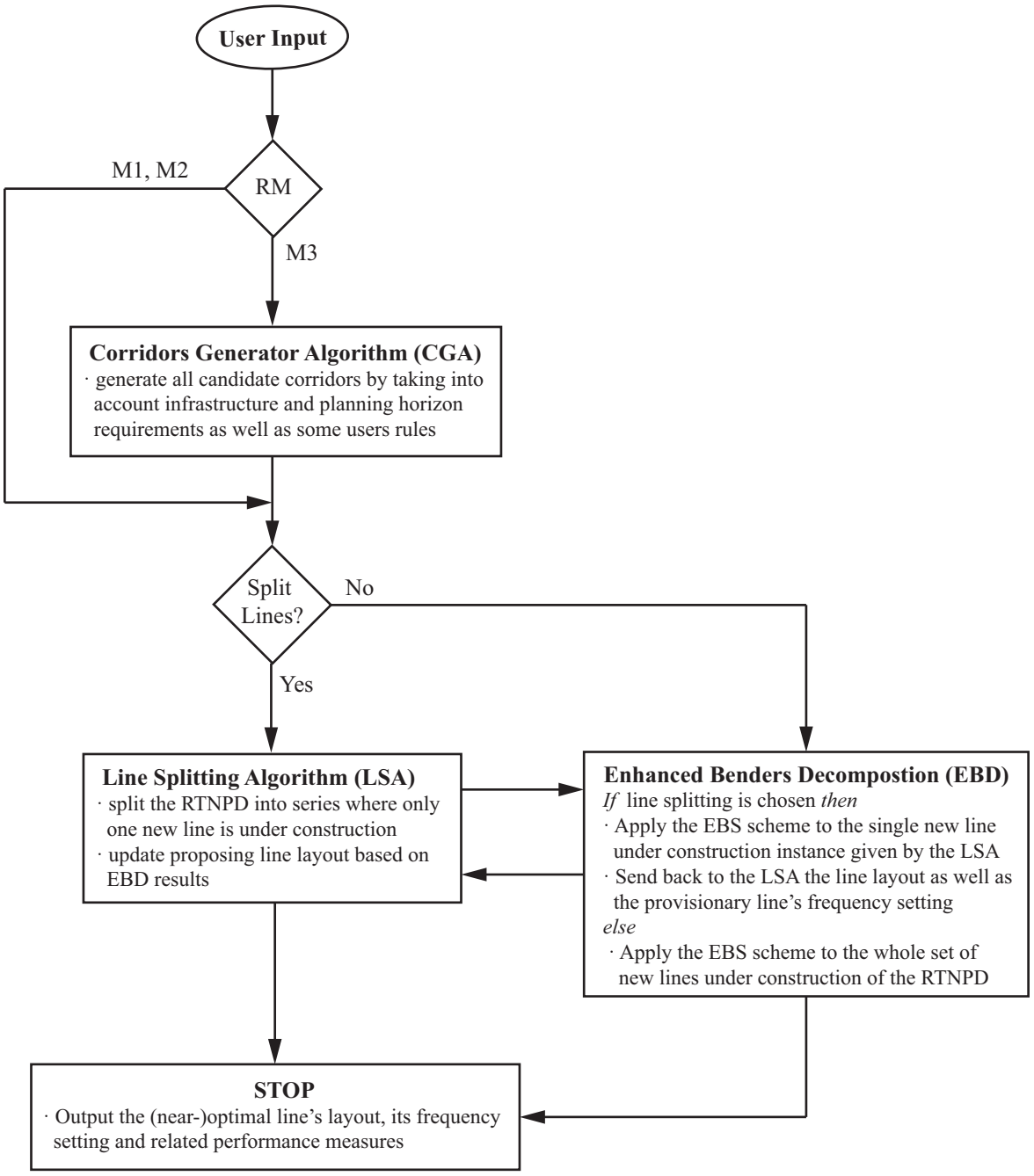


Figure 3.1: Framework of the main procedures developed.

3.2 The corridor generation algorithm

Figure 3.2 shows an overview of the CGA algorithm. It computes the number of corridors which can be assigned to the lines under construction. Thus, the routing part of the model is performed before the mathematical programming problem is solved. However, the decision of the node's role (whether it works as a service node or just as a passing point), is still computed in the optimization phase. The skeleton of the algorithm is as follows. It is split into two procedures: the rectilinear corridor generation algorithm ($RCGA$) and the circular corridor generation algorithm ($CCGA$). Both algorithms implement an ad hoc version of Yen's k-shortest path algorithm [189], which includes the verification of some requirements. They are also divided into two groups: the common corridor requirements and the specific corridor requirements. The

former constrains the corridors to infrastructure budgetary and planning horizon limitations, in a similar way as stated by equations (2.4) and (2.36); whereas the latter forces the corridors to meet some user behavior rules which establish lower and upper bounds on the length of the detours held in the k-shortest paths, for $k > 1$. These rules are set up by some parameters which depend on the type of corridor.

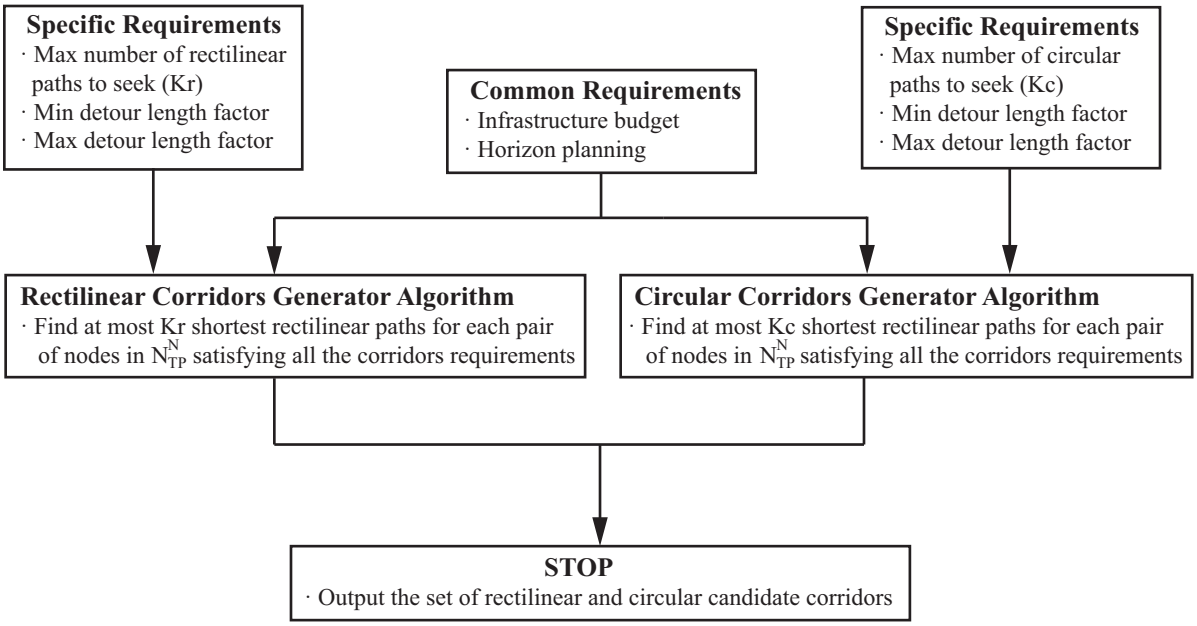


Figure 3.2: Overview of the Corridor Generation Algorithm.

In the state of the art, there are many authors that use constraint satisfaction when computing k-shortest paths. To cite just a few of them, Androutsopoulos & Zografos [4] and the references herein limit the time length of each path according to a given time window. Jim *et al* [99] do not allow building paths whose node departure times are not within a predefined list of times. More recently, and in the context of Public Transportation systems, Van der Zijpp & Catalano [174] and their earlier works referenced herein apply some user behavior rules to eliminate unwanted corridors.

The difference between the $RCGA$ and $CCGA$ algorithms lies in the way they find the shortest path for a given pair of nodes in each subphase. The $RCGA$ uses the Floyd & Warshall algorithm [75] in the first subphase and the Dijkstra algorithm [58] in the second subphase. The $CGCA$, however, employs a specialized algorithm which drives the Dijkstra algorithm [58] in the first subphase. It is also used in the second subphase in some cases. In other cases, the Dijkstra algorithm is directly employed.

The reader is referred to Chapter 4 to see the details of these algorithms and to subsections 3.5.1, 3.5.2 3.5.3 for a brief introduction of the solving techniques applied.

3.3 The line splitting algorithm

Figure 3.3 shows an overview of the LSA algorithm. It consists of the resolution of a series of small mathematical programming problems with the same mathematical structure as the original mathematical programming problem presented in Chapter 2, but with only one line under construction. Moreover, it includes two variants called incremental and non-incremental modes, which differ in the way they update the OD-pair demand data and the stopping criterion. The incremental mode assigns a portion of the OD-demand to each OD-pair each time a new line is added to the set of new lines, whereas the non-incremental mode assigns

the whole OD-demand to each OD-pair. If we use the non-incremental variant, the algorithm can be stopped early provided that the current solution of the reduced model does not build a new line and that we are not in the last iteration. To see this, let us consider that we are in a given iteration $k < |L^N|$ where we assigned the whole demand to n working lines (some of them are fictitious since the come from previous constructed lines) plus one extra possible line under construction. If the optimal solution indicates that all n working lines can satisfy the demand without constructing the new line, it then makes no sense to continue the LSA algorithm in the following iterations with the same demand level. Thus, the mathematical programming problem will give the same solution.

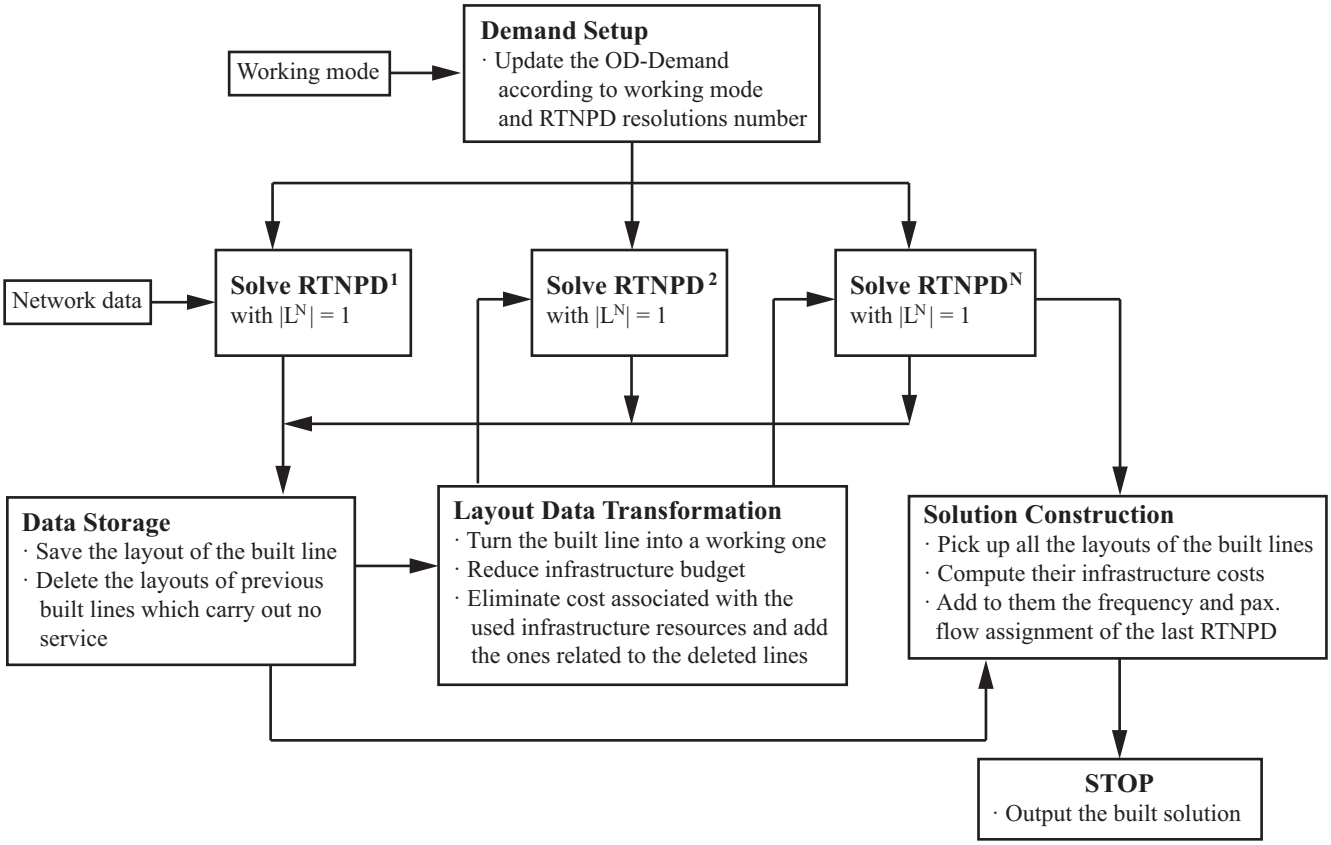


Figure 3.3: Overview of the line splitting algorithm.

The resolution of each reduced mathematical programming problem takes into account the working lines as well as the infrastructure and planning resources, which are updated according to the solution found in the last resolution. The main operations carried out in this updated are the transformation of the currently built line into a fictitious working line, the reduction of the infrastructure budget according to the new infrastructure resources used in this new line, and the elimination of the costs associated with these infrastructure resources.

The reader is refer to Chapter 5 to see the details of the algorithm and its variants. It also includes an illustrative example to understand the working of the algorithm.

3.4 The specialized Benders decomposition

The resolution of the $RTNPD$ model with a single line under construction has been done by means of a specialized version of the Benders decomposition. It includes the seminar work of J. Benders [14], the

enhancements of Wong & Magnanti [125], Papadakos N. [147], and McDaniel & Devine [136], as well as some ad hoc techniques. Figure 3.4 shows the big picture of the outer Benders scheme (OBS), whereas picture 3.5 presents the inner Benders scheme (IBS), which is embedded into the former.

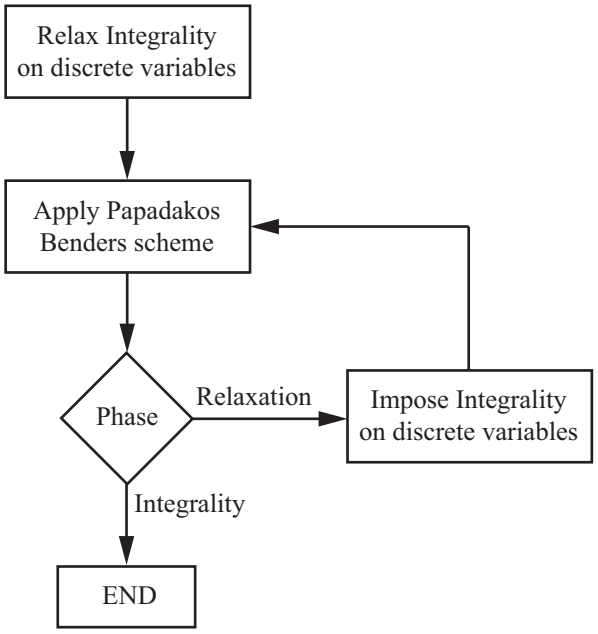


Figure 3.4: Overview of the final version of the Benders Decomposition used.

The OBS implements the McDaniel & Devine [136] scheme, which consists of two phases. They differ from the way they solve the master problem (MP). In the first phase, the integrality of the discrete variables of the MP are relaxed, whereas in the second phase, the integrality of these variables are imposed.

Regarding the IBS, it is an adaptation of the Papadakos N. [147] scheme, which enhances the Benders convergence when the subproblem is degenerated. At each iteration, three problems called the Independent Magnanti & Wong problem ($IMWP$), the master problem (MP), and the subproblem (SP) are solved. The MP is related to the network design and the line frequency setting, whereas the SP corresponds to the passenger flow assignment. The SP assumes that the multicommodity network topology and the line frequencies are known. Both are found by previously solving the MP. The $IMWP$ is quite similar to the SP except for the fact that the configuration of the network topology and the line frequencies come from computing a core point instead of the current MP solution. The notion of core point was first introduced by Wong & Magnanti [125], and it allows stronger or tighter Benders cuts to be obtained.

During the initialization phase of the IBS, some steps are different. If the relaxation phase is working (the one where the discrete variables are relaxed), the optimality Benders cut (OBC) is set to NULL and an initial core point must be computed. In contrast, if the integrality phase is active (the one where the integrality of the discrete variables is imposed), the $OBCs$ coming from the relaxation phase are kept and the core point is updated according to the last one obtained from the relaxation phase.

There is one additional step in the relaxation phase. Having solved the relaxed MP, its solution is rounded off in order to obtain a feasible multicommodity network topology and line frequency setting. This operation is straightforward as long as we apply routing model 3 (see section 2.3.3.3 for further details). It suffices to determine which corridor is best suited for the set of corridors assigned to each line. As a first attempt, we directly solve a subversion of the integral MP with a subset of Λ, keeping only the corridors assigned to some lines.

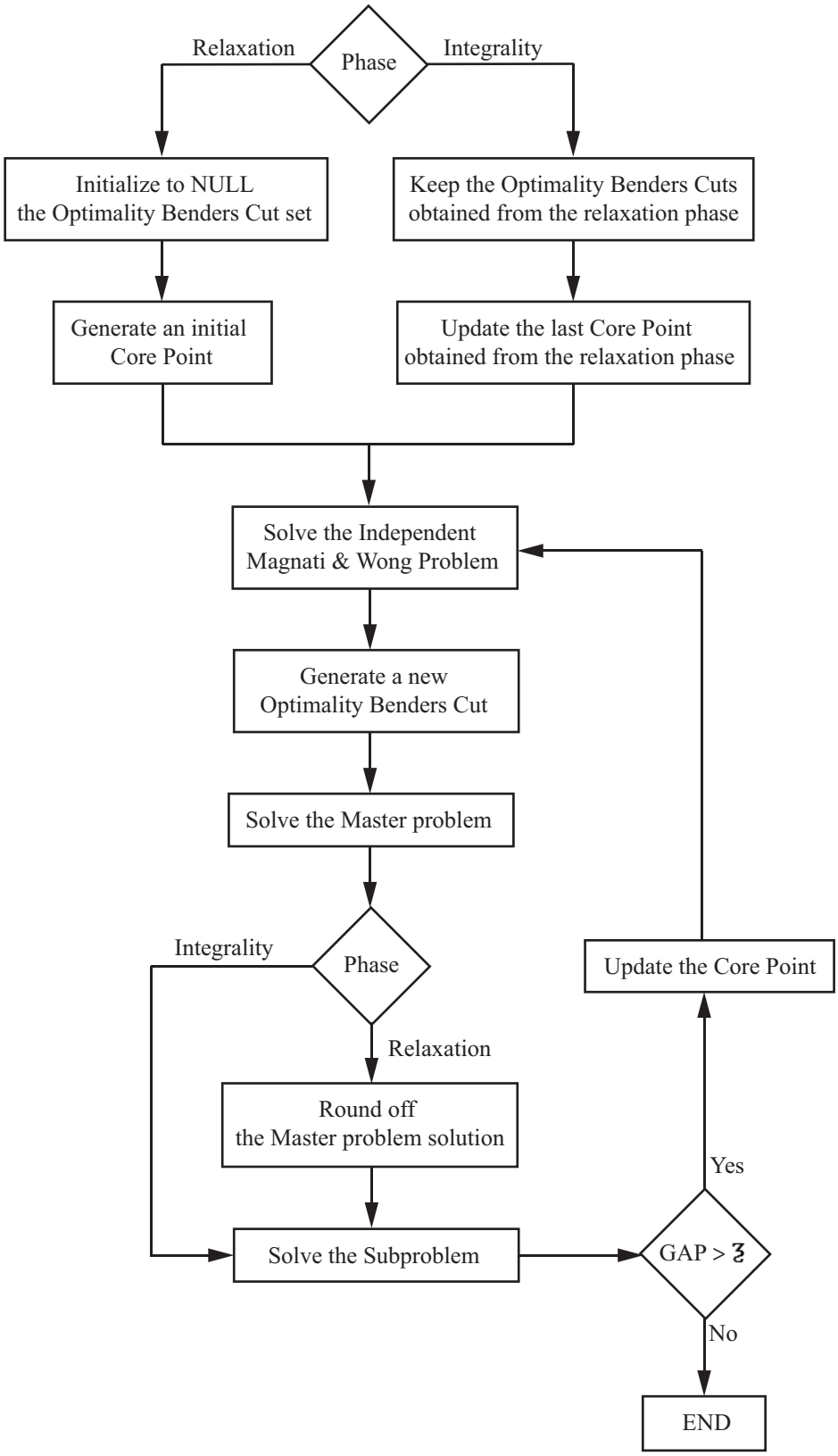


Figure 3.5: Overview of the adaptation of Papadakos Scheme used.

The reader is referred to Chapter 6 for the details of the algorithm, and to subsections 3.5.4 and 3.5.5 for a brief introduction to the solving techniques applied.

3.5 Solving techniques

This section holds all the state-of-the-art algorithms that have been used to develop the aforementioned building blocks, which conform to the framework in which the $RTNPD$ model is solved. They basically include three types of shortest path algorithms: the Floyd & Warshall all shortest path algorithm, the Dijkstra shortest path algorithm and Yen's k-shortest paths algorithm, all of which take part in the CGA algorithm. Three types of techniques are also included: the classical Benders decomposition, an enhanced Magnanti & Wong acceleration technique, and a type of bilevel programming, all of which are integrated into a specialized Benders decomposition. The latter is only used to solve the $RTNPD$ model under elastic demand.

3.5.1 Floyd & Warshall all shortest paths algorithm

The Floyd & Warshall algorithm (FWA) [75], [181] is a very efficient, simple program, and it is widely used method for finding the shortest paths between all pair of nodes in a directed graph, all at the same time. Furthermore, it has an important advantage over Dijkstra's algorithm (see section 3.5.2), in that it works when the arc weights are allowed to be negative, which means it will in fact allow us to detect negative-cost cycles.

The FWA is written down in pseudo-code in table 3.1. It basically works with an NxN matrix representing a directed graph $G = (N, A)$, where each cell element holds the shortest cost between its related pair of nodes, except for the diagonal elements, which are set to infinity to denote no connection (observe that the corresponding row-column identifiers represent the same nodes). To obtain the shortest paths in each cell, each triad of nodes $(k, i, j) \in N$ is verified whether $c_{ij} > c_{ik} + c_{kj}$. If so, the temporary cost of the shortest path from i to j is replaced with the sum of the costs of the two interconnected links. Otherwise it remains unchanged. Initially, the values of each cell $CT(i, j)$ are set to the cost of its corresponding edge $c_{i,j}$.

procedure FloydWarshall-AllSP (**in** C **out** CT)

$CT(i,j) \leftarrow C(i,j) \quad \forall (i,j) \in A,\ CT(i,j) \leftarrow \infty \quad \forall (i,j) \notin A$

for each $k, i, j \in [1..|N|]$ **such that** $i \neq j,\ k \neq i,\ k \neq j$ **do**

$c'_{ij} \leftarrow CT(i,k) + CT(k,j)$

if $c'_{ij} < CT(i,j)$ **then**

$CT(i,j) \leftarrow c'_{ij}$

end if

end for

return CT

end FloydWarshall-AllSP

Table 3.1: Pseudo-code of the Floyd & Warshall all shortest paths algorithm.

This is the general working of the algorithm. However, some modifications and/or additions are needed

for detecting negative cycles, for coping with some satisfaction path constraints, or for recovering the path itinerary. The former is not needed in our research since we work with time networks, whereas the two latter are implemented in the modified Floyd & Warshall algorithm, which is written down in table 4.2. So, for the sake of simplification, we do not give here further details.

3.5.2 Dijkstra's shortest path algorithm

According to the literature on network flow, the algorithmic approaches for solving the shortest path problem can be classified into two groups: label setting and label correcting. Both are iterative and both employ the labeling method in computing one-to-all (all-to-all) shortest paths. However, they differ in the way they update the estimate (i.e., the upper bound) of the shortest path associated with each node, in the step to step, and in how they converge to the optimal shortest path. In label setting algorithms, one label will be designated as permanent (optimal) at each iteration. However, in label-correcting algorithms, all labels will be considered as temporary until they all become permanent in the final step. Moreover, they require that the costs associated with the links be positive. According to Ahuja, Magnanti and Orlin [3], the label-correcting algorithms are more efficient than label setting algorithms. Also, since transport networks do not contain negative cycles, this group of shortest path algorithms has been chosen.

The most basic label-setting algorithm is Dijkstra's algorithm, which finds the shortest path from a given source node s to all other nodes in a directed network $G = (N, A)$ with non-negative arc costs. Dijkstra's algorithm creates labels associated with nodes representing the cost from the source node to each particular node. Within G, there exist two kinds of labels: temporary and permanent. Temporary labels are given to nodes that have not been reached. Their values can vary. The main idea of Dijkstra's algorithm is to change the temporary labels into permanent ones as the shortest path tree adds them to the model. A node's permanent label denotes identifies its nearest neighborhood node. Thus, for any given node, there must be a permanent label or a temporary label, but not both at the same time.

For a mathematical description of the algorithm, some notations are introduced. For node i, let A_i represent the arc adjacency list of node i. Let N denote the set containing all the nodes with permanent labels and $\bar{N}$ be the set containing all the nodes with temporary labels. As shown in table 3.2, every node initially has a temporary label and the cost from the source node s to all other nodes are initialized to ∞. At each step, the algorithm chooses the node $i \in \bar{N}$ with the least temporary label cost and makes it permanent. Then, it records its predecessor index and updates the temporary values of all nodes $j \in A_i$. This process is repeated until all nodes become permanent. Finally it outputs the list Pred, which holds the node labels starting at the destination node d and ending at the source node s.

3.5.3 Yen's k-shortest paths algorithm

The first step in analyzing a transport problem such as the $RTNPD$ is the explicit enumeration of the paths to be considered. In realistically sized networks the number of paths that can be constructed is virtually infinite. We will therefore define the path enumeration problem as a problem of generating a path-set of manageable size that contains the most relevant paths for a specific transport problem.

Different approaches for dealing with the path enumeration problem are known. Assuming that the issue of path relevance can be expressed by a number of constraints that refer to, for example, path overlap or path detour, the path enumeration problem can be objectively and efficiently solved by finding the K-shortest paths that satisfy these types of constraints.

Existing K-shortest path algorithms may be subdivided into algorithms that allow paths to have repeated links (cycles) (see Hoffman and Pavley [92], Bellman and Kalaba [13], Shier [162], [163], Eppstein [65],

procedure Dijkstra-SP (in $C,\ s,$ **out** Pred)

$P \leftarrow \emptyset,\ \bar{P} \leftarrow N$

$d_s \leftarrow 0,\ d_i \leftarrow \infty\ \ \forall i \in N \backslash s$

$\text{pred}(s) \leftarrow 0,\ \text{pred}(i) \leftarrow -1\ \ \forall i \in N \backslash s$

while $|P| < N$ **do**

Select $i \in \bar{P}$ **such that** $d_i = \min\left\{d_j \mid \forall j \in \bar{P}\right\}$

$P \leftarrow P \cup i,\ \bar{P} \leftarrow \bar{P} - i$

for each $(i,j) \in A_i$ **do**

if $d_j > d_i + C(i,j)$ **then**

$d_j \leftarrow d_i + C(i,j),\ \text{pred}(j) \leftarrow i$

end if

end for

end while

return Pred

end Dijkstra-SP

Table 3.2: Pseudo-code of the Dijsktra's shortest path algorithm.

[66]) and algorithms that only consider acyclic paths (see Yen [189], Lawler [113], Katoh *et al* [101], Hadjiconstantinou and Christofides [88]). The last two algorithms require the network to be undirected.

In the context of transport applications, cyclic paths are not of interest. Moreover, in our research the routing graph is undirected; thus the algorithm of Hadjiconstantinou and Christofides [88] (which is an enhancement of Katoh *et al* [101]) seems to be the most appropriate. However, the algorithm of Yen [189] has been chosen both because it is simple to implement and because it solves the network in a reasonable time.

Yen's k-shortest path algorithm is given by routine **Yen-KSP,** shown in table 4.3. To adapt this algorithm for our proposes, we have changed the original mathematical notation as follows. The link costs are denoted by means of matrix T; the computed feasible paths are stored in list B; and the subset of computed feasible paths not yet assigned to any k-shortest path (P_k) are saved in $Q \subset B$.

The algorithm begins by initializing list B to the first shortest path for the given pair $(i,j) \in N$ (P^1) and list Q to null. Then, the iterative part of the algorithm starts. Every k-iteration seeks all the existing new paths, such that they contain a common subpath (P_{1-i}^{k-1}) coming from the $k-1$ shortest path P^{k-1}, which starts at the source node P_1^{k-1} and ends at an intermediate node P_i^{k-1}. However, they differ from a detour subpath which starts at P_i^{k-1} and ends at the terminal node P_t^{k-1}.

To compute these new subpaths, the method works with a copy of the link cost matrix ($T^{'}$). This copy may be modified according to the position of the intermediate node P_i^{k-1} (under process) in the path P^{k-1} and the links held in the common subpath P_{1-i}^{k-1}. This modification is carried out as follows. It eliminates all the links adjacent to P_i^{k-1}, which are contained in the paths j of B and whose initial subpath from its first

procedure Yen-KSP (**in** $K,\ T,\ P^1,$ **out** P^{1-K})

$Q \leftarrow \emptyset,\ B \leftarrow P^1$

for each $k \in [2...K]$ **do**

for each $i \in [1...|P^{k-1}| - 1]$ **do**

$T' \leftarrow T$

for each $j \in [1...|B|]$ **such that** $|B^j| \geq |P^{k-1}_{1-i}|$ **and** $B^j_{1-i} = P^{k-1}_{1-i}$ **do**

$T'(P^{k-1}_i, B^j_{i+1}) \leftarrow \infty$

end for

$S \leftarrow$ Shortest rectilinear path from P^{k-1}_i to P^{k-1}_t by calling **DSP**($T', P^{k-1}_i, P^{k-1}_t, S$)

if $S \neq \emptyset$ **then**

$R \leftarrow P^{k-1}_{1-(i-1)} \cup S,\ Q \leftarrow Q \cup \{R\},\ B \leftarrow B \cup \{R\}$

end if

end for

$P^k \leftarrow$ **Select** R with minimum $t(R)$ from Q, $Q \leftarrow Q - \{R\}$

end for

return P^{1-K}

end Yen-KSP

Table 3.3: Pseudo-code of Yen's k-shortest path algorithm.

node B^1_j to the intermediate node B^i_j overlaps with the initial subpath of path $k-1$ from its first node P^{k-1}_1 to the intermediate P^{k-1}_i. This elimination is carried out virtually by setting to infinity the costs associated with the links, so it prevents them from being selected.

Having updated the matrix T' properly, it computes the shortest path from P^{k-1}_i to P^{k-1}_t by using a shortest path algorithm. Dijkstra's algorithm [58] is usually chosen, as it gives the lowest worst case time scenario. We do not provide here the description of the Dijkstra algorithm since section 3.5.2 is devoted to explaining it in detail. If Dijkstra's algorithm gives a new subpath (S) by linking the intermediate node P^{k-1}_i to the final node P^{k-1}_t, it is appended to the initial subpath $P^{k-1}_{1-(i-1)}$.

To complete the k^{th}-iteration, it chooses from the list of candidate paths Q the one with the shortest cost and sets it to the k-shortest feasible rectilinear path. Additionally, it deletes this chosen path from Q. The process is repeated at most $K-1$ times, providing that the list Q is not empty when it tries to set a k-shortest path with $k < K$. If so, Yen's algorithm finishes earlier.

3.5.4 Benders Decomposition

Benders decomposition [14] is an algorithm for mixed integer linear programming that has been applied successfully to a variety of applications. To cite just a few of them, Florian *et al* [74] have used the algorithm to schedule the movement of railway engines; Richardson [155] has applied the algorithm to airline routing;

and Geoffrion & Graves [84] have had great success in applying the algorithm to the design of industrial distribution systems. More recently, Cordeau *et al* [44] have employed the algorithm to simultaneously assign locomotives and cars in the context of passenger transportation. These contributions demonstrate the potential for using Benders decomposition to solve specially structured mixed integer programs.

A mixed integer linear programming problem can be stated as follows:

$$
\begin{aligned}
& z = \min_{x,y} \; c^t x + d^t y && (3.1)\\
& s.t. \\
& \; x \in X \\
& \; Ax + By \geq h \\
& \; y \geq 0
\end{aligned}
$$

where $x \in \Re^n$ and $y \in \Re^{r+}$ are decision variables of the problem, and supra-indexes n and r correspond to the dimension of their sets. The x variables are mainly discrete variables, although a subset of them can be continuous. In contrast, the set y must hold only continuous non-negative variables. The x variables are called the complicating or binding variables because fixing them to a specific value makes it much easier to solve the rest of the optimization problem.

J.F. Benders devised a clever approach for exploiting this type of structure. Firstly, he defined the following ordinary linear problem:

$$
\begin{aligned}
& \alpha(x) = \min_{y} \; d^t y && (3.2)\\
& s.t. \\
& \; By \geq h - Ax \\
& \; y \geq 0
\end{aligned}
$$

where the variable set x is fixed. Then, he proposed an algorithm for finding the optimal variable vectors x, y by employing a cutting-plane approach for building up proper representations of the extremal value of (3.2) as a function of the parameterized vector x and the set of values of x, for which (3.2) is feasible.

Linear Programming duality theory was employed to derive the natural families of cuts that characterize these representations, which were called Benders cuts due to the author's surname, and the dual of program (3.2) was used to generate them. Then, the optimal x variable vector was sought in the y-projection space instead of the x-y space by solving a series in the following reduced version of problem (3.1):

$$
\begin{aligned}
& \tilde{z} = \min_{x,\omega} \; c^t x + \omega && (3.3)\\
& s.t. \\
& \; x \in X && (3.4)\\
& \; \omega \geq (h - Ax)^t \, \hat{\pi}^j, \;\; \forall \hat{\pi}^j \in P && (3.5)\\
& \; (h - Ax)^t \, \overrightarrow{\pi}^j \leq 0, \;\; \forall \overrightarrow{\pi}^j \in Q && (3.6)
\end{aligned}
$$

where $\hat{\pi}^j$ and $\overrightarrow{\pi}^j$ stand for the extreme points and extreme directions, respectively, of the dual of (3.2) and are held in its respective sets P and Q. Problem (3.2) is characterized as follows:

$$\alpha(x) = \max_{\pi} \ (h - Ax)^t \pi \tag{3.7}$$
$$s.t.$$
$$B^t\pi = d$$
$$\pi \geq 0$$

Benders cuts are classified into two groups: the Benders optimality cuts (3.5) and the Benders feasibility cuts (3.6). The former is employed when, for a given parametrization of x, there exists at least one optimal solution to (3.7); whereas the latter is applied when (3.7) is unbounded for a given parameterization of x. In the latter case, to obtain the dual variables one can use the Theorem 2.6.6 of Bazaraa *et al* [11]. In our case, feasibility Benders cuts do not arise since whatever the parametrization of x is, the program (3.7) always renders a feasible solution.

Problems (3.3) and (3.7) are called the master problem (MP) and subproblem (SP), respectively. As shown in table 3.4, they are iteratively solved in that order until the following criterion is met:

$$GAP = \frac{UB^k - LB^k}{LB^k} \leq \epsilon \tag{3.8}$$

where UB^k and LB^k stand for the upper and lower bounds of the Benders algorithm at the current k-iteration. Their values are computed by means of the following expressions:

$$UB^k = \min\{UB^{k-1}, \ c^t x^{*,k} + \alpha(x^{*,k})\} \tag{3.9}$$
$$LB^k = \tilde{z}(x^{*,k}, \omega^{*,k}) \tag{3.10}$$

Equation (3.10) implies that, at each iteration, the solution of the MP gives a lower bound; whereas equation (3.9) states that the upper bound is computed as the minimum between the upper bound of the next to last iteration and the current optimal value of the independent objective term of the MP ($c^t x^{*,k}$), with the addition of the optimal value of the SP's objective function ($\alpha(x^{*,k})$).

1. Set $P \leftarrow Q \leftarrow \emptyset$, iteration $k \leftarrow 0$ and allowable error ϵ
2. Solve the MP^k.
3. Solve the dual of SP^k with x^k.
4. if dual of SP^k is unbounded then $Q \leftarrow Q \cup \{\overrightarrow{\pi}^k\}$ and go to Step 7, otherwise continue.
5. if $GAP < \epsilon$ then STOP, otherwise continue.
6. $P \leftarrow P \cup \{\hat{\pi}^k\}$.
7. $k \leftarrow k + 1$ and go to Step 2.

Table 3.4: Pseudo-code of Benders' Scheme.

3.5.5 Magnanti & Wong's acceleration technique and enhancements

The Magnanti & Wong acceleration technique [125] is used when applying a Benders decomposition [14] for a special type of mixed integer linear programming problem, where the resulting subproblem is degenerated.

This means that the dual of problem (3.2) has alternative optima and that the resulting Benders cuts may be weak, in the sense that they do not constrain the master problem (MP) as much as they can. Consequently, the number of Benders iterations increases dramatically and it becomes possible to obtain the (near-)optimal solution to problem (3.1). This difficulty arises in network optimization applications such us shortest route and transshipment problems.

Magnanti & Wong devise a clever idea for selecting good cuts from (3.7) based on pareto-optimality. They invented a new Benders scheme in which the optimality Benders cuts are obtained from the resolution of an extra problem, called the Magnanti & Wong problem (MWP), which is parameterized by the MP variable vector x, a core point, and the optimal objective function value of the SP.

Before going into the details, let's formalize some definitions. Let's say that the cut (3.11) dominates or is stronger than the cut (3.12) if (3.13) is met for all $x \in X$ and with a strict inequality for at least one point $x' \in X$. The set X is made up of points satisfying the MP constraints (3.4). Then, a cut is called pareto-optimal if no cut dominates it.

$$\omega \geq (h - Ax)^t \hat{\pi}^1 \tag{3.11}$$

$$\omega \geq (h - Ax)^t \hat{\pi}^2 \tag{3.12}$$

$$(h - Ax)^t \hat{\pi}^1 \geq (h - Ax)^t \hat{\pi}^2 \tag{3.13}$$

To generate a pareto-optimal cut, we need to find a core point. It is a point $x_0 \in ri(X^c)$, where $ri(X^c)$ stands for the relative interior of the convex hull set of X. Then, a core point can be formally expressed as follows:

$$x_0 = \sum_{j \in 1..s} \lambda_j \cdot x_j, \ \sum_{j \in 1..s} \lambda_j = 1, \ 1 > \lambda_j \geq 0 \tag{3.14}$$

where $x_1, \ x_2, \ ..., \ x_s \in X$. Observe that it is a convex combination of non-negative weights whose values are strictly less than 1, such that it ensures that the point x_0 is strictly in the interior of X. The finding of a core point is not trivial, so ad hoc procedures must be developed.

The core point is then used in collaboration with the MP variables and the optimal SP objective function value to solve the MWP, which generates the pareto-optimal cuts. Its dual form is as follows:

$$\alpha(x, x_0, \alpha(x^*)) = \max_{\pi} \ (h - Ax_0)^t \pi \tag{3.15}$$

$$s.t.$$

$$(h - Ax)^t \pi = \alpha(x^*) \tag{3.16}$$

$$B^t \pi = d$$

$$\pi \geq 0$$

where the linking constraint (3.16) ensures that an extreme point from (3.7) will be chosen. Moreover, the objective function (3.15) compares all possible cuts at a point $x_0 \in X$.

The Magnanti & Wong scheme shown in table 3.5 is then applied as follows. At every k-iteration, first the MP^k and SP^k problems are solved and then SP^k unboundedness is checked. If these parameters are fulfilled, a feasible Benders cut is generated and the MP^{k+1} is solved. This process is repeated as long as the SP^{k+1} does not give a feasible solution. In contrast, a valid core point is determined and convergence

is checked by means of equation (3.8). If the criterion is met, the algorithm stops. Otherwise, the core point as well as the MP^k solution x^k and SP^k optimal objective function are applied to the resolution of the MWP^k. Finally, the Benders optimality cut is generated and the algorithm loops to the next iteration.

1. Set $P \leftarrow Q \leftarrow \emptyset$, iteration $k \leftarrow 0$ and allowable error ϵ
2. Solve the MP^k.
3. Solve the dual of SP^k with x^k.
4. if dual of SP^k is unbounded then $Q \leftarrow Q \cup \{\overrightarrow{\pi}^k\}$ and go to Step 9, otherwise continue.
5. if $GAP < \epsilon$ then STOP, otherwise continue.
6. Find a core point x_0.
7. Solve the dual of MWP^k with $x^k,\ x_0^k,\ \alpha(x^{*,k})$.
8. $P \leftarrow P \cup \{\hat{\pi}^k\}$.
9. $k \leftarrow k+1$ and go to Step 2.

Table 3.5: Pseudo-code of Magnanti & Wong Scheme.

This scheme allows decreasing dramatically the number of Benders iterations. However, N. Papadakos [147] demonstrated that the primal form of (3.15) is numerically unbounded when it is provided with a suboptimal subproblem solution. The primal form of (3.15) is as follows:

$$\begin{aligned} &\beta(x, x_0, \alpha(x^*)) = \min_{y,\xi}\ d^t y + \alpha(x^*)\xi \\ &s.t. \\ &\quad By + (h - Ax)\,\xi \geq h - Ax_0 \\ &\quad \xi \text{ free} \end{aligned} \tag{3.17}$$

where ξ is the dual variable associated with linking constraint (3.16). This variable takes a large negative value when the optimal SP objective function $\alpha(x^*)$ is replaced with a suboptimal one. To see the complete proof, the reader is referred to section *2 - Limitations of the Magnanti & Wong's method* of Papadakos' work [147]. To overcome that limitation, the author proposed the following alternative problem to (3.15):

$$\begin{aligned} &\alpha(x_0) = \max_{\pi}\ (h - Ax_0)^t\,\pi \\ &s.t. \\ &\quad B^t\pi \leq d \\ &\quad \pi \geq 0 \end{aligned} \tag{3.18}$$

This problem was called the independent Magnanti & Wong problem ($IMWP$), since it is not dependent on the subproblem. The reader is referred to theorem 6 of [147] to see the proof that this problem also gives a valid pareto-optimal cut. N. Papadakos also devised a new Benders scheme which outperforms the Magnanti & Wong scheme and works as follows. As shown in table 3.6, at every k-iteration, a valid core point is first determined and then the $IMWP^k$ problem is solved. Thirdly, an optimality Benders cut is generated from this resolution and applied to the MP^k. The next step is the SP^k resolution. If its solution is unfeasible, a feasible Benders cut is generated and the algorithm loops to the next iteration. Otherwise,

convergence is checked by means of equation (3.8). If the criterion is met, the algorithm stops. Otherwise, it jumps to the next iteration.

This scheme has two advantages compared with Magnanti & Wong's scheme. It allows a feasible solution to be obtained with only one iteration, and the resolution of $IMWP$ is much easier than the resolution of MWP, because of linking constraint (3.16). Moreover, the core point can be updated at every k-iteration with $k > 1$ by applying the following equation:

$$x_0^k = x_0^{k-1} \cdot \lambda + x^{k-1} \cdot (1 - \lambda),\ 1 > \lambda > 0 \tag{3.19}$$

which is a convex linear combination of the last core point found and MP^{k-1} variable values. The weight λ must be strictly positive, otherwise the operation will give the same core point. Additionally, λ must be strictly less than 1, otherwise we obtain the MP^{k-1} solution as a new core point, which is not a valid core point since it is not in the interior region of X. As stated by the author, a good choice is to set $\lambda = 0.5$.

1. Set $P \leftarrow Q \leftarrow \emptyset$, iteration $k \leftarrow 0$ and allowable error ϵ
2. Find a core point x_0^k.
3. Solve the dual of MWP^k with x_0^k.
4. $P \leftarrow P \cup \{\hat{\pi}^k\}$.
5. Solve the MP^k.
6. Solve the dual of SP^k with x^k
7. if dual of SP^k is unbounded then $Q \leftarrow Q \cup \{\overrightarrow{\pi}^k\}$ and go to Step 9, otherwise continue.
8. if $GAP < \epsilon$ then STOP, otherwise continue.
9. $k \leftarrow k + 1$ and go to Step 2.

Table 3.6: Pseudo-code of Papadakos' Scheme.

3.5.6 Bilevel programming and its solving techniques

This section is devoted to presenting a class of bilevel programming (BP) where the objective and constraints of both levels are linear. However, in the upper level there may be some discrete variables. The need for this kind of BP arises for a design model such as the one incorporating modal choice formulated in Chapter 7. In this model, the terms used in the objective function for passenger assignment and modal choice are different from those in the objective function indexing the quality of the design. This creates a need to formulate a bilevel programming problem. In the following, a brief introduction to mixed-integer linear bilevel programming ($MILBP$) is presented. The works on $MILBP$ that are most related to the problem formulated in Chapter 7 are discussed in particular. Then, in section 3.5.6.2, a solving technique suggested in [40] is developed. This technique is based on an adaptation of the Benders decomposition suited to the kind of $MILBP$ formulated in Chapter 7.

3.5.6.1 Introduction

The most fundamental form of two-level mathematical programming is the Stackelberg basic problem [177], where decision makers 1 (leader) and 2 (follower) correspond to the upper- and lower-level problems, respectively. The former has the following form:

$$\min_{\boldsymbol{x}\in\boldsymbol{X}} \boldsymbol{F}(\boldsymbol{x},\boldsymbol{y}^*) \tag{3.20}$$

$$\text{subject to } \boldsymbol{G}(\boldsymbol{x},\boldsymbol{y}^*) \leq \boldsymbol{0} \tag{3.21}$$

$$\boldsymbol{y}^* \in \boldsymbol{P}(\boldsymbol{x}) \tag{3.22}$$

where $\boldsymbol{y}^*$ denotes the response of the lower level, which must be held in its optimal solution set $\boldsymbol{P}(\boldsymbol{x})$. This set is also called the rational reaction set and has the following form:

$$\boldsymbol{P}(\boldsymbol{x}) = \left\{ \boldsymbol{y}^* \text{ solves} \;\middle|\; \boldsymbol{f}(\boldsymbol{x},\boldsymbol{y}^*) = \begin{array}{l} \min\limits_{\boldsymbol{y}\in\boldsymbol{Y}} f(\boldsymbol{x},\boldsymbol{y}) \\ \text{subject to } \boldsymbol{g}(\boldsymbol{x},\boldsymbol{y}) \leq \boldsymbol{0} \end{array} \right\} \tag{3.23}$$

where $\boldsymbol{x} \in \Re^p$ is the parameter (the leader's p-dimensional decision variable vector). A related set is the inducible region $\boldsymbol{IR}$= $\{(\boldsymbol{x},\boldsymbol{y}^*) \mid \boldsymbol{G}(\boldsymbol{x},\boldsymbol{y}) \leq \boldsymbol{0}, \boldsymbol{y}^* \in \boldsymbol{P}(\boldsymbol{x})\}$. It is assumed that for each decision taken by the leader, the follower has some room to respond. The rational set $\boldsymbol{P}(\boldsymbol{x})$ defines this response, whereas the inducible region $\boldsymbol{IR}$ represents the set which the leader may optimize.

The solution of problem (3.20) - (3.22) is in general not well-defined since $\boldsymbol{P}(\boldsymbol{x})$ is often multi-valued and discontinuous. In order to deal with that issue, there exist two approaches: the optimistic approach and the conservative one. In that work, we have considered the former, which establishes that the follower cooperates with the leader to minimize $\boldsymbol{F}(\boldsymbol{x},\boldsymbol{y})$ for a given set of rational reactions. This assumption is reasonable in our application since the objective function costs associated with the variables controlled by the follower are also considered in the leader, although in a different manner. The leader and the follower consider the passenger time costs in their objective functions. Additionally, the follower incorporates other costs which are related to the preferred mode of transportation and which are not contained in the objective function of the leader. However, the leader also takes into account the decision variables related to the preferred mode of transportation and sets them to the same values as the ones obtained by the follower. This is carried out by means of a constraint which links the optimal objective function values of the follower and the leader. The reader is referred to Chapter 7 for further details.

Assuming the cooperation of the follower, problem (3.20) - (3.22) is redefined as follows:

$$\min_{\boldsymbol{x}\in\boldsymbol{X}} \boldsymbol{F}(\boldsymbol{x},\boldsymbol{y}^*) \tag{3.24}$$

$$\text{subject to } \boldsymbol{G}(\boldsymbol{x},\boldsymbol{y}^*) \leq \boldsymbol{0} \tag{3.25}$$

$$\boldsymbol{y}^* \in \boldsymbol{P}(\boldsymbol{x}) = \left\{ \boldsymbol{y}^* \text{ solves} \;\middle|\; \boldsymbol{f}(\boldsymbol{x},\boldsymbol{y}^*) = \begin{array}{l} \min\limits_{\boldsymbol{y}\in\boldsymbol{Y}} \boldsymbol{f}(\boldsymbol{x},\boldsymbol{y}) \\ \text{subject to } \boldsymbol{g}(\boldsymbol{x},\boldsymbol{y}) \leq \boldsymbol{0} \end{array} \right\} \tag{3.26}$$

This problem conforms to the general case where the objective functions and constraint sets of both levels can be linear or non-linear. Moreover, decisional variables can be integer, continuous, or a mixture of both. In the mixed case, one subset of them is integerand the other one is continuous. The vast majority of works on the state of the art incorporate the linear and continuous case. A good review of the techniques used for the continuous linear and non-linear cases can be found in Colson *et al* [41]. As for the practical applications, we recommend the survey carried out by Saharidis *et al* [158].

In this research, we do not intend to carry out a general survey of BP, neither for solving techniques nor for applications. Instead, we focus on the nearest techniques which can be applied to the problem stated in

Chapter 7. They include those solutions for the mixed-integer linear bilevel problem ($MILBP$) with integer variables either on both levels or only on the upper level. The former is also included because the existing solving techniques can be easily adapted to the latter. It suffices to drop from their associated algorithms some steps which are related to the integrality of the lower level. The resulting method is faster to compute and thus remains a good candidate for our use.

In the rest of the subsection, we will review in detail the solving techniques for these two classes of $MILBPs$ in order to explain the motivation of developing a new methodology. This review is based partially on the surveys taken by Saharidis & Ierapetritou [156] and Saharidis *et al* [158], but without much detail; so the majority of the following references can also be found therein.

Solution approaches for $MILBP$ can be classified into the following four categories:

- Branch & bound (B & B) based approaches.
- Branch & cut (B & C) based techniques.
- Parametric programming.
- Mixed reformulation techniques, which convert KKT conditions to mathematical decompositions in order to split the initial $MILBP$ into two single-level problems.

Works related to exact and heuristic solution procedures based on the B & B technique include Bard and Moore [8], Wen & Yang [182] and Xu & Wang [188]. All of them are based on adaptations of the B & B techniques applied to mixed integer linear problems, but they differ in the way they compute the bounds and in the kind of subproblems they solve to obtain a bilevel feasible solution at a given node of the B & B tree.

Bard & Moore [8] solve up to three subproblems in a node. The first one is a continuous single level problem, which contains the leader and the follower constraint sets and evaluates only the leader objective function. The integrality of the integer variables is relaxed, but it includes some associated bounds computed in previous nodes. If this subproblem has a solution and its objective function value is greater than the best bilevel feasible solution found so far, the algorithm proceeds to solve the second subproblem. This one entails the resolution of the continuous version of the original $MILBP$, which is carried out in two steps. Firstly, it is reformulated as one discrete single-level problem by applying the KKT conditions in the same way as stated in [76]. Then, the resulting mixed integer linear problem is solved using a standard B & B procedure. If it has a solution and the relaxed discrete variables of the leader have integer values, the algorithm continues to solve the final subproblem. This one entails solving only the follower problem with the variables controlled by the leader and which are fixed to the values obtained by the second subproblem. The authors tried to develop some heuristics in order to reduce the size of the B & B tree by incorporating different policies in the branching. Despite their efforts, the resulting algorithm is extremely time consuming, even if we apply it to our case, where the resolution of the last subproblem can be skipped since the follower variables are all continuous. This drawback is due to the resolution of subproblem 2, which is not continuous, and demands a lot of time for larger instances.

Another Branch & Bound technique has been developed by Wen & Yang [182]. They focused on a particular version of an $MILBP$ where the lower level has no discrete variables and the upper one has only binary variable. Despite this simplification, its proposed algorithm can be easily adapted for tackling general integer variables. Unlike the work of Bard and Moore, they only solve one continuous single level problem at each node, except for the initial one, in which they solve two continuous single level problems. Whatever the case is, they employ only the continuous variables of the follower to solve the problems. To compute upper and lower bounds of the original $MILBP$, they combine duality theory with a right-hand

side perturbation of constraint sets and an explicit enumeration of the leader's integer values. The authors also developed a heuristic version of its B & B based on a judgment index, which is used to evaluate which variable is more suitable for branching. This index is calculated from the weighted estimated optimal solutions of the leader's decision variables, which are obtained by neglecting 1) the follower's objective function and 2) the leader's objective function. The weights depend on the estimated optimal solution obtained and the number of variables controlled by each of them; the higher the judgment index value of the upper level decision variable, the higher the priority for checking that variable. This algorithm seems more effective than that of Bard and Moore, since only one continuous single problem is solved at each node. However, if a subset of continuous variables can be included in the leader, the algorithm is not valid and its adaptation is not clear at all. Consequently, it does not seem to be useful for our case.

Quite recently, Xu & Wang [188] have developed more effective branch & bound rules, which seem to considerably reduce the search space by skipping solutions of the relaxed versions of the original $MILBP$, which are bilevel infeasible. Despite this promising feature, we cannot be sure whether or not the algorithm is useful since we were unable to view the details – the article is not yet available online and we could read only the abstract.

Regarding B & C techniques, we have found the work of DeNegre & Ralphs [54]. The authors were inspired by the work of Bard & Moore and developed a simpler solving schema to obtain bilevel feasible solutions for the $MILBP$. In this schema, the first and third subproblems of Bard & Moore are solved at each node, possibly more than once, and the resolution of the second subproblem is replaced with valid inequalities which obtain integer values of the discrete variables from the first subproblem. In that manner, the algorithm of Bard & Moore is less time consuming. Moreover, the authors also built valid inequalities using the solution of the third problem in order to obtain, if suitable, a different bilevel feasible solution in the same node. These inequalities are added to the first subproblem, which is solved again. This algorithm promises to be the most interesting one regarding B & B and B & C techniques. However, it is still immature. For instance, it is not clear when it is more suitable to branch or find another feasible solution (if it exists) in a given node.

Moving on to parametric programming techniques, Faisca *et al* [67] developed an iterative algorithm for separate resolution of one mixed integer linear problem and a multi-parametric continuous linear problem ($MPCLP$). The former uses the total set of constraints contained in the leader and the follower. However, the objective function includes only that of the leader. The $MPCLP$ is formed also by using the total set of constraints, but fixing the variables controlled by the leader to the values obtained in the first problem. This time, the objective function of the follower is evaluated and the problem is solved using the algorithm developed by the same authors in a previous work [59]. The resolution of this problem gives a set of feasible solutions for the follower and best one is chosen from among them. The performance of this algorithm depends heavily on the number of different continuous problems that must be solved at each iteration. This number, in turn, depends on the problem instance. Thus we cannot infer to what extent this algorithm is useful.

The last class of methods includes the works of Saharidis & Ierapetritou [156] and Gabriel *et al* [81]. Both combine reformulation techniques with mathematical decomposition techniques in order to split the initial $MILBP$ into two single-level problems. The differences rely mainly on the type of reformulation technique used and the way they split the initial $MILBP$.

Saharidis & Ierapetritou [156] used the classical Benders Decomposition [14] to split the $MILBP$ into a master problem and a subproblem. The former holds all the discrete variables and consists of relaxing the original $MILBP$, whereas the latter is a continuous bilevel problem whose solution gives a constraint (which may be a feasible, an optimal or an exclusive cut) to the master problem in order to drive it to the

optimal bilevel solution. The resolution of this subproblem is carried out in three steps. Firstly, the inner problem is substituted with its KKT optimality conditions applying [161]. Secondly, the resulting mixed integer linear subproblem is solved by means of the active set strategy of [86]. Its resolution gives information on the active constraints which are used to build the corresponding linear problem. The resolution of this problem (third step) is used to build the associated cut. This approach is the closest to ours and will be used to explain its development in the following subsection.

Gabriel *et al* [81] devise a similar approach to Saharidis & Ierapetritou, although it is heuristic, more complicated and less effective. Their algorithm also splits the original $MILBP$ into a master problem and a subproblem with identical structure. However, the way they solve the subproblem and the type of Benders cuts they employ is different. The subproblem is first reformulated as a mixed integer programming problem by means of the KKT conditions applying [76], and it is then solved with the B & B technique. The discrete variables are then fixed to the ones obtained and the resulting continuous problem is then solved. From this problem, the Benders cuts are inferred and added to the master problem. This algorithm has two drawbacks: 1) the resolution of the subproblem entails solving another integer problem (apart from the master problem) and 2) the values of the master problem variables (which are inputs to the subproblem) can render subproblem variables related to the values of the follower, which are not contained in the reaction set (3.23). The latter is due to explicit consideration of the subproblem's binary variables, which emerge when the KKT conditions are applied to turn the continuous bilevel problem into a single level problem. To overcome that limitation, the authors developed a set of heuristics based on decomposition of the master variables into regions where the subproblem's solutions are contained in (3.23). However, it entails solving several subproblems per iteration and adding an enormous set of constraints (including disjunctive) to the master problem in order to activate only those cuts related to the master variables under consideration. Consequently, this algorithm has also been discarded.

3.5.6.2 Benders Decomposition for a subclass of Mixed Integer Bilevel Problems

The approach used to solve the elastic demand version of the model is based on an adaptation of the Benders decomposition. This adaptation can tackle any linear $MILBP$ with only integer variables that are controlled by either the leader or outer problem. Moreover, the leader cannot have linking constraints (i.e., constraints involving variables of both outer and inner levels). On the other hand, it is necessary to assume that the subproblem has at least one feasible solution, whatever the master problem input is. Thus, only optimality cuts are generated. These three assumptions hold in our application since the leader is the only decision maker that controls the operator variables, of which some are discrete. Additionally, linking constraints only appear in the follower (in the form of link capacity and binding constraints). Moreover, the follower only contains the variables related to passengers, which are all continuous non-negative. The reader is referred to Chapter 7 for further details.

Our solving approach has some similarities to the work of Saharidis & Ierapetritou [156]. However, it differs in the way it solves the resulting subproblem and how the Benders cuts are generated. In the rest of the subsection, all the basic steps of the algorithm will be described and an illustrative example will be given as well.

Applying the aforementioned assumptions, the general form of the Stackelberg problem (3.24) - (3.26) can be turned into the following one:

$$\min_{\boldsymbol{x_1},\boldsymbol{x_2},\boldsymbol{y}} \boldsymbol{c_{11}x_1} + \boldsymbol{c_{12}x_2} + \boldsymbol{c_{13}y} \tag{3.27}$$

$$\text{s.t. } \boldsymbol{y} \in \boldsymbol{P}(\boldsymbol{x_1}, \boldsymbol{x_2})$$

$$\boldsymbol{x_1}, \boldsymbol{x_2} \in \boldsymbol{X} \cap \{\boldsymbol{x_1}, \boldsymbol{x_2} \mid \boldsymbol{P}(\boldsymbol{x_1}, \boldsymbol{x_2}) \neq \emptyset\} \triangleq \boldsymbol{Y}^0$$

$$\boldsymbol{x_1} \in Z^{m_1},\ \boldsymbol{x_2} \in \Re^{m_2}$$

$$\boldsymbol{y} \in \Re^{+,n}$$

where $\boldsymbol{x_1}$ is a m_1 - dimensional vector of discrete variables, $\boldsymbol{x_2}$ is a m_2 - dimensional vector of continuous variables, and $\boldsymbol{y}$ is a n - dimensional vector of continuous variables as well. The follower's reaction set $\boldsymbol{P}(\boldsymbol{x_1}, \boldsymbol{x_2})$ corresponds to the following mathematical programming problem:

$$\min_{\boldsymbol{y}} \boldsymbol{c_{21}x_1} + \boldsymbol{c_{22}x_2} + \boldsymbol{c_{23}y} \tag{3.28}$$

$$\text{s.t. } \boldsymbol{g}(\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{y}) \triangleq \boldsymbol{A_1x_1} + \boldsymbol{A_2x_2} + \boldsymbol{A_3y} \leq \boldsymbol{b}$$

$$\boldsymbol{y} \in \Re^{+,n}$$

Saharidis & Ierapetritou show that the problem (3.27) can be approached by considering in $\boldsymbol{Y}^0$ the constraint's subset, which is active in one of the optimal solutions. This constraint's subset is built iteratively by means of dual data inferred from the resolution of a subproblem (SP). This SP consists of a hierarchical problem, which is constructed by fixing the variables controlled by the outer problem ($\boldsymbol{x_1}\ \boldsymbol{x_2}$) to a resolution value provided by relaxing the original $MILBP$ This is called the Master Problem (MP). The MP and SP problems interact in a similar way as in the Classical Benders Decomposition. The only difference is that an additional type of cut, called the exclusive cut, can appear when the subproblem in a given iteration gives a new cut which does not improve the objective function of the last master problem resolution.

The proposed algorithm follows the same framework as in that of Saharidis & Ierapetritou: the MP approximates the original $MILBP$ and the SP represents the hierarchical problem parameterized by the MP variables coming from the current resolution. However, it differs from the way the SP is solved and in the form of the Benders cuts inferred from this SP. Thus, these features render an innovative algorithm for solving such a class of $MILBP$. In the rest of the subsection, this framework the distinct features will be formally stated.

According to Saharidis & Ierapetritou, problem (3.27) can be approximated by the following (MP):

$$\min_{\boldsymbol{x_1},\boldsymbol{x_2},w} \boldsymbol{c_{11}x_1} + \boldsymbol{c_{12}x_2} + w \tag{3.29}$$

$$\text{s.t. } w \in \boldsymbol{\Omega}(\boldsymbol{x_1}, \boldsymbol{x_2})$$

$$\boldsymbol{x_1}, \boldsymbol{x_2} \in \boldsymbol{X}$$

$$\boldsymbol{x_1} \in Z^{m_1},\ \boldsymbol{x_2} \in \Re^{m_2}$$

$$w \text{ free}$$

where the constraint set $\boldsymbol{\Omega}(\boldsymbol{x_1}, \boldsymbol{x_2})$ is built by Benders cuts, which are obtained by means of the resolution of the following SP:

$$\begin{aligned} &\min_{\boldsymbol{y}} \boldsymbol{c_{13}}\ \boldsymbol{y} \\ &\min_{\boldsymbol{y}} \boldsymbol{c_{23}}\ \boldsymbol{y} \\ &\text{s.t. } \boldsymbol{g}(\bar{\boldsymbol{x}}_1, \bar{\boldsymbol{x}}_2, \boldsymbol{y}) \triangleq \boldsymbol{A_3 y} \leq \boldsymbol{b} - \boldsymbol{A_1}\bar{\boldsymbol{x}}_1 - \boldsymbol{A_2}\bar{\boldsymbol{x}}_2 \\ &\quad \boldsymbol{y} \in \Re^{+,n} \end{aligned} \tag{3.30}$$

where $\bar{\boldsymbol{x}}_1$ and $\bar{\boldsymbol{x}}_2$ are parameters denoting the values of the previous resolution of the master problem (3.29).

This algorithm differs from that of Saharidis & Ierapetritou in the way we solve the subproblem (3.30) (which in our view is much simpler) and in the mathematical structure of $\boldsymbol{\Omega}(\boldsymbol{x_1}, \boldsymbol{x_2})$. Let us rewrite (3.30) as follows:

$$\begin{aligned} &\min_{\boldsymbol{y}} (\tilde{\boldsymbol{c}}_{\boldsymbol{13}} + \boldsymbol{c_{23}})\ \boldsymbol{y} \\ &\min_{\boldsymbol{y}} \boldsymbol{c_{23}}\ \boldsymbol{y} \\ &\text{s.t. } \boldsymbol{g}(\bar{\boldsymbol{x}}_1, \bar{\boldsymbol{x}}_2, \boldsymbol{y}) \triangleq \boldsymbol{A_3 y} \leq \boldsymbol{b} - \boldsymbol{A_1}\bar{\boldsymbol{x}}_1 - \boldsymbol{A_2}\bar{\boldsymbol{x}}_2 \\ &\quad \boldsymbol{y} \in \Re^{+,n} \end{aligned} \tag{3.31}$$

where $\tilde{\boldsymbol{c}}_{\boldsymbol{13}} + \boldsymbol{c_{23}} = \boldsymbol{c_{13}}$. This assumption is valid since any linear equation can be expressed as a linear combination of other linear equations. Now, the goal is to prove that the problem (3.31) can be solved in two steps with the help of duality theory.

Theorem. Problem (3.31) can be solved in two steps by means of the resolutions of two continuous problems called $SP1$ and $SP2$.

Proof. With the aid of duality theory, the inner problem of (3.31) (from now on called $SP2$) has the following equivalent dual form:

$$\begin{aligned} &\max_{\boldsymbol{\pi_2}} (\boldsymbol{b} - \boldsymbol{A_1}\bar{\boldsymbol{x}}_1 - \boldsymbol{A_2}\bar{\boldsymbol{x}}_2)\ \boldsymbol{\pi_2} \\ &\text{s.t. } \boldsymbol{A_3^T}\ \boldsymbol{\pi_2} \leq \boldsymbol{c_{23}} \\ &\quad \boldsymbol{\pi_2} \geq \boldsymbol{0} \end{aligned} \tag{3.32}$$

where $\boldsymbol{\pi_2}$ is the q - dimensional vector holding the dual variables and where q represents the number of rows of the matrix $\boldsymbol{A_3}$. This matrix holds the constraint's coefficients related to variables $\boldsymbol{y}$. Notice that the constraints region of this $SP2$ is independent from the value of the original variables $\boldsymbol{y}$. Thus, the optimal solution of this problem is also independent of the optimal solution of the outer problem in (3.31). The optimal solution of the dual of $SP2$ can be linked to the primal version by means of the strong duality theorem. It establishes that, in the optimum, the objective function values of both problems must coincide. Thus, the inner problem in (3.31) can be replaced with a linking constraint which relates the dual and primal variables. If we take advantage of this feature, problem (3.31) can be reformulated as follows:

$$\min_{\boldsymbol{y}}(\tilde{\boldsymbol{c}}_{\mathbf{13}} + \boldsymbol{c}_{\mathbf{23}})\ \boldsymbol{y} \tag{3.33}$$

$$\text{s.t. } \boldsymbol{c}_{\mathbf{3}}\ \boldsymbol{y} = (\boldsymbol{b} - \boldsymbol{A}_{\mathbf{1}}\bar{\boldsymbol{x}}_{\mathbf{1}} - \boldsymbol{A}_{\mathbf{2}}\bar{\boldsymbol{x}}_{\mathbf{2}})\ \boldsymbol{\pi}_{\mathbf{2}}\ \ (\psi)$$

$$\boldsymbol{g}(\bar{\boldsymbol{x}}_{\mathbf{1}}, \bar{\boldsymbol{x}}_{\mathbf{2}}, \boldsymbol{y}) \triangleq \boldsymbol{A}_{\mathbf{3}}\boldsymbol{y} \leq \boldsymbol{b}_{\mathbf{2}} - \boldsymbol{A}_{\mathbf{1}}\bar{\boldsymbol{x}}_{\mathbf{1}} - \boldsymbol{A}_{\mathbf{2}}\bar{\boldsymbol{x}}_{\mathbf{2}}\ \ (\boldsymbol{\pi}_{\mathbf{1}})$$

$$\boldsymbol{y} \in \Re^{+,n}$$

where $\boldsymbol{c}_{\mathbf{23}}\ \boldsymbol{y} = (\ \boldsymbol{b} - \boldsymbol{A}_{\mathbf{1}}\bar{\boldsymbol{x}}_{\mathbf{1}} - \boldsymbol{A}_{\mathbf{2}}\bar{\boldsymbol{x}}_{\mathbf{2}})\ \boldsymbol{\pi}_{\mathbf{2}}$ is the linking constraint. This problem, which will be called $SP1$ from now on, can be solved afterwards by using the dual variables of the $SP2$ as input. Thus, this method can be stated as reformulation technique that is alternative to those found in the state of the art (see [76], [161]). □

Having shown how to solve problem (3.30), it only remains to explain how the Benders cuts are obtained and how they are added to the master problem. In the original Benders decomposition, the cuts are related to one optimal solution of the dual form of the subproblem, and they are added by means of an inequality constraint, which establishes a bound to a free variable. In the following resolutions of the master problem, this variable takes the value of the highest bound from among the set of computed inequalities. In our case, we have to solve two subproblems: the $SP1$ and the $SP2$. Thus, we need to add two different types of inequality constraints, which will be denoted as h_k^1 and h_k^2, where k stands for the algorithm iteration. h_k^1 will establish a bound to the master's variable w_1 ($w_1 \geq h_k^1$), whereas h_k^2 will give a bound to the master's variable w_2 ($w_2 \geq h_k^2$). The expressions of h_k^1 and h_k^2 are computed as follows:

$$h_k^n = \begin{cases} (\boldsymbol{b} - \boldsymbol{A}_{\mathbf{1}}\bar{\boldsymbol{x}}_{\mathbf{1}} - \boldsymbol{A}_{\mathbf{2}}\bar{\boldsymbol{x}}_{\mathbf{2}})\ \boldsymbol{\pi}_{\mathbf{2}}^{\boldsymbol{k}} & \text{if } n = 2 \\ w_2\ \psi^k + (\boldsymbol{b}_{\mathbf{2}} - \boldsymbol{A}_{\mathbf{1}}\bar{\boldsymbol{x}}_{\mathbf{1}} - \boldsymbol{A}_{\mathbf{2}}\bar{\boldsymbol{x}}_{\mathbf{2}})\ \boldsymbol{\pi}_{\mathbf{1}}^{\boldsymbol{k}} & \text{if } n = 1 \end{cases} \tag{3.34}$$

where $\boldsymbol{\pi}_{\mathbf{2}}^{\boldsymbol{k}}$ are the variables coming from the resolution of (3.32) at iteration k, and $\boldsymbol{\pi}_{\mathbf{1}}^{\boldsymbol{k}}$, ψ^k are taken from the resolution of the dual form (3.33) at iteration k, whose mathematical programming problem is as follows:

$$\max_{\boldsymbol{\pi}_{\mathbf{1}}, \psi} (\boldsymbol{b} - \boldsymbol{A}_{\mathbf{1}}\bar{\boldsymbol{x}}_{\mathbf{1}} - \boldsymbol{A}_{\mathbf{2}}\bar{\boldsymbol{x}}_{\mathbf{2}})\ \boldsymbol{\pi}_{\mathbf{1}} + [(\boldsymbol{b} - \boldsymbol{A}_{\mathbf{1}}\bar{\boldsymbol{x}}_{\mathbf{1}} - \boldsymbol{A}_{\mathbf{2}}\bar{\boldsymbol{x}}_{\mathbf{2}})\ \boldsymbol{\pi}_{\mathbf{2}}]\ \psi \tag{3.35}$$

$$\text{s.t. } \tilde{\boldsymbol{A}}_{\mathbf{3}}[\psi\ \boldsymbol{\pi}_{\mathbf{2}}]^T \leq \tilde{\boldsymbol{c}}_{\mathbf{13}}$$

$$\boldsymbol{\pi}_{\mathbf{2}} \geq \mathbf{0}$$

$$\psi \text{ free}$$

where $\tilde{\boldsymbol{A}}_{\mathbf{3}} = [[\boldsymbol{c}_{\mathbf{23}}\ \mathbf{1}]^{\boldsymbol{T}}[\boldsymbol{A}_{\mathbf{3}}\ \mathbf{0}]^{\boldsymbol{T}}]$, ψ and $\boldsymbol{\pi}_{\mathbf{2}}$ are related to the constraints of the primal (3.33) (shown in parenthesis), and the weight $\boldsymbol{c}_{\mathbf{13}}$ (which takes part of the primal objective function in (3.33)) has been removed. The latter has been done because the original weight $\boldsymbol{c}_{\mathbf{13}} = \tilde{\boldsymbol{c}}_{\mathbf{13}} + \boldsymbol{c}_{\mathbf{23}}$ is achieved through the contribution of w_1 in the master's objective function, as well as the constraints set $h_k^1, \quad \forall\ k \in K$. The preliminary version of the master problem is formulated as follows:

$$\min_{\boldsymbol{x_1},\boldsymbol{x_2},w_1,w_2} \boldsymbol{c_{11}x_1} + \boldsymbol{c_{12}x_2} + w_1 + w_2 \tag{3.36}$$

$$\text{s.t. } w_n \geq h_k^n, \quad \forall n \in \{1,2\},\ k \in K$$

$$\boldsymbol{x_1}, \boldsymbol{x_2} \in \boldsymbol{X}$$

$$\boldsymbol{x_1} \in Z^{m_1},\ \boldsymbol{x_2} \in \Re^{m_2}$$

$$w_1,\ w_2 \text{ free}$$

where K stands for the number of Benders iterations carried out so far. This preliminary version of the master problem cannot be used since it is unbounded. Observe that for $n = 1$, the subset of constraints $w_2 \geq h_k^2$ can be inactive, i.e., allowing $w_2 \longrightarrow -\infty$. Thus, for $n = 2$, the other subset of constraints $w_1 \geq h_k^1$ is also unbounded since it depends on the value of w_1. To overcome that problem, we need to impose that at least one constraint of the type $w_2 \geq h_k^2$ is active. It is done by means of the following extra set of constraints:

$$\delta_k \leq 1 - \frac{1}{M_k}\left(w_2 - h_k^2\right), \quad \forall k \in K \tag{3.37}$$

$$\sum_{k \in K} \delta_k \geq 1 \tag{3.38}$$

where δ_k is a binary variable denoting whether the cut $n = 2$, computed at iteration k, is active and M_k is a large enough parameter to prevent the cut $n = 2, k$ from being active when $\delta_k = 0$. The last constraint (3.38) assures that at least one cut is active. Then, adding constraints (3.37) - (3.38) to the preliminary master problem (3.36) yields the following final master problem:

$$\min_{\boldsymbol{x_1},\boldsymbol{x_2},\boldsymbol{\delta},w_1,w_2} \boldsymbol{c_{11}x_1} + \boldsymbol{c_{12}x_2} + w_1 + w_2 \tag{3.39}$$

$$\text{s.t. } w_n \geq h_k^n, \quad \forall n \in \{1,2\},\ k \in K$$

$$\delta_k \leq 1 - \frac{1}{M_k}\left(w_2 - h_k^2\right), \quad \forall k \in K$$

$$\sum_{k \in K} \delta_k \geq 1$$

$$\boldsymbol{x_1}, \boldsymbol{x_2} \in \boldsymbol{X}$$

$$\boldsymbol{x_1} \in Z^{m_1},\ \boldsymbol{x_2} \in \Re^{m_2}$$

$$w_1,\ w_2 \text{ free}$$

$$\boldsymbol{\delta} \in \{0,1\}^{|K|}$$

To sum up, table 3.7 gives all the aforementioned steps of the algorithm plus the convergence check criterion (done in step 5).

1. Set Iteration $k \leftarrow 0$ and allowable error ϵ
2. Solve the final version of MP^k (3.39).
3. Solve the dual of $SP2^k$ (3.32) with $\boldsymbol{x_1^k}$, $\boldsymbol{x_2^k}$ fixed.
4. Solve the dual of $SP1^k$ (3.33) with $\boldsymbol{x_1^k}$, $\boldsymbol{x_2^k}$, $\boldsymbol{\pi_2}$ fixed.
5. if $GAP^k < \epsilon$ then STOP, otherwise continue.
6. Construct $h_k^n \ \forall n \in \{1,2\}$ (3.34) by means of $\boldsymbol{\pi_n^k}$, ψ^k.
7. $k \leftarrow k+1$ and go to Step 2.

Table 3.7: Pseudo-code of the Benders Scheme applied to mixed integer linear bilevel problems with only discrete variables in the outer level.

where the gap expression GAP^k is computed as follows:

$$GAP^k = \frac{(\boldsymbol{b} - \boldsymbol{A_1}\bar{\boldsymbol{x}}_{\boldsymbol{1}} - \boldsymbol{A_2}\bar{\boldsymbol{x}}_{\boldsymbol{2}})(\boldsymbol{\pi_1^k}\psi^k + \boldsymbol{\pi_2^k}) - (w_1^k + w_2^k)}{w_1^k + w_2^k} \tag{3.40}$$

In bi-level optimization, the leader examines the reactions of the follower for each feasible choice of its variables. Thus, restricting a variable which is controlled by the leader results in a restriction of the constraints region ($\boldsymbol{\Psi}$ = $\{(\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{y})$: $\boldsymbol{x_1}, \boldsymbol{x_2} \in \boldsymbol{X}$, $\boldsymbol{g}(\boldsymbol{x_1},\boldsymbol{x_2},\boldsymbol{y})\}$). Consequently in the new bi-level problem, the leader looks for the reaction of the follower in a region which is a restriction of $\boldsymbol{\Psi}(\boldsymbol{x_1}, \boldsymbol{x_2})$ = $\{\boldsymbol{y}$: $\boldsymbol{g}(\boldsymbol{x_1},\boldsymbol{x_2},\boldsymbol{y})\}$ or otherwise the resulted projection of the new constraint region on the leader's decision space is a restriction of the initial region ($\boldsymbol{\Psi}'(\boldsymbol{x_1}, \boldsymbol{x_2}) \subseteq \boldsymbol{\Psi}(\boldsymbol{x_1}, \boldsymbol{x_2})$). A direct result of this observation is that the follower's rational reaction set is a restriction of the initial one ($\boldsymbol{P}'(\boldsymbol{x_1}, \boldsymbol{x_2}) \subseteq \boldsymbol{P}(\boldsymbol{x_1}, \boldsymbol{x_2})$), and thus the new inducible region is a sub-set of the original inducible region IR = $\{$ $(\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{y})$: $\boldsymbol{x_1}, \boldsymbol{x_2} \in \boldsymbol{\Psi}(\boldsymbol{x_1}, \boldsymbol{x_2})$, $\boldsymbol{y} \in \boldsymbol{P}(\boldsymbol{x_1}, \boldsymbol{x_2})$ $\}$. Thus, the SP provides a valid cut and a valid upper bound.

To conclude the subsection, we give an illustrative example based on a modified example of Wen & Yang, which was used earlier by Saharidis & Ierapetritou [156]. It consists of the following $MILBP$:

$$\begin{aligned}
&\min_{x_2,y_2,y_3} F_1(x_2,y_2,y_3) = -60x_2 - 10y_2 - 7y_3 \\
&\text{s.t. } \min_{y_2,y_3} F_2(y_2,y_3) = -60y_2 - 8y_3 \\
&\quad \text{s.t. } g_1 : 10x_2 + 2y_2 + 3y_3 \leq 225 \\
&\quad\quad g_2 : 5x_2 + 3y_2 \leq 230 \\
&\quad\quad g_3 : 5x_2 + y_3 \leq 85 \\
&\quad x_2 \in \{0,1\},\ y_2, y_3 \geq 0
\end{aligned} \tag{3.41}$$

Before applying the algorithm shown in table 3.7, problem (3.41) needs to be decomposed. The initial master problem is built by dropping from (3.41) all data related to the follower (variables y_2, y_3, constraints g_1, g_2, g_3 and objective function $F_2(y_2, y_3)$). Moreover, variables w_1, w_2 related to the Benders cuts are added and constrained initially to a weight $M = 1200$ in order to bypass the unboundedness problem without constraining them so much that the optimal value is removed. As a result, we obtain the following:

$$\min_{x_2,w_1,w_2} -60x_2 + w_1 + w_2 \tag{3.42}$$

$$\text{s.t. } w_1, w_2 \in [-M, M]$$

$$x_2 \in \{0, 1\}$$

The subproblem is constructed by fixing variable x_2 in (3.41) to a given value and by dropping the weights associated with this variable in the objective functions, resulting in the following hierarchical problem:

$$\min_{y_2,y_3} -10y_2 - 7y_3 \tag{3.43}$$

$$\text{s.t. } \min_{y_2,y_3} -60y_2 - 8y_3$$

$$\text{s.t. } 2y_2 + 3y_3 \leq 225 - 10\bar{x}_2$$

$$3y_2 \leq 230 - 5\bar{x}_2$$

$$y_3 \leq 85 - 5\bar{x}_2$$

$$y_2, y_3 \geq 0$$

Now, taking $c_{23} = -60y_2 - 8y_3$, the primal of the $SP2$ can be written as follows:

$$\min_{y_2,y_3} -60y_2 - 8y_3 \tag{3.44}$$

$$\text{s.t. } 2y_2 + 3y_3 \leq 225 - 10\bar{x}_2 \ \ (\pi_{21})$$

$$3y_2 \leq 230 - 5\bar{x}_2 \ \ (\pi_{22})$$

$$y_3 \leq 85 - 5\bar{x}_2 \ \ (\pi_{23})$$

$$y_2, y_3 \geq 0$$

whose dual form is as follows:

$$\max_{\pi_{21},\pi_{22},\pi_{23}} (225 - 10\bar{x}_2)\pi_{21} + (230 - 5\bar{x}_2)\pi_{22} + (85 - 5\bar{x}_2)\pi_{23} \tag{3.45}$$

$$\text{s.t. } 2\pi_{21} + 3\pi_{22} \leq -60$$

$$3\pi_{21} + \pi_{23} \leq -8$$

$$\pi_{21}, \pi_{22}, \pi_{23} \leq 0$$

Moving on to the formulation of the $SP2$, taking $\tilde{c}_{13} + c_{23} = -10y_2 - 7y_3$, the weight $\tilde{c}_{13} = 50y_2 + y_3$, so $SP1$ can be written as follows:

$$\min_{y_2,y_3} 50y_2 + y_3 \tag{3.46}$$

$$\text{s.t.} -60y_2 - 8y_3 = (225 - 10\bar{x}_2)\pi_{21} + (230 - 5\bar{x}_2)\pi_{22} + (85 - 5\bar{x}_2)\pi_{23} \quad (\psi)$$

$$2y_2 + 3y_3 \leq 225 - 10\bar{x}_2 \quad (\pi_{11})$$

$$3y_2 \leq 230 - 5\bar{x}_2 \quad (\pi_{12})$$

$$3y_2 \leq 230 - 5\bar{x}_2 \quad (\pi_{12})$$

$$y_3 \leq 85 - 5\bar{x}_2 \quad (\pi_{13})$$

$$y_2, y_3 \geq 0$$

whose dual form is as follows:

$$\max_{\pi_{11},\pi_{12},\pi_{13},\psi} (225 - 10\bar{x}_2)\pi_{11} + (230 - 5\bar{x}_2)\pi_{12} + (85 - 5\bar{x}_2)\pi_{13} + \tag{3.47}$$

$$+ [(225 - 10\bar{x}_2)\pi_{21} + (230 - 5\bar{x}_2)\pi_{22} + (85 - 5\bar{x}_2)\pi_{23}]\psi$$

$$\text{s.t. } 2\pi_{11} + 3\pi_{12} - 60\psi \leq 50$$

$$3\pi_{11} + \pi_{13} - 8\psi \leq 1$$

$$\pi_{11}, \pi_{12}, \pi_{13} \geq 0$$

Having formulated all the problems involved in the algorithm, the iterative part begins. In the first iteration, the master problem (3.42) does not have any constraint which controls variables x_2 and, as it has an associated cost in the objective function, its optimal value for minimization is $x_2 = 0$. Moreover, variable cuts are set to the lower bounds $w_1 = -M$, $w_2 = -M$. Then, fixing $\bar{x}_2 = 0$, the $SP2^{(1}$ (3.45) is first solved. Its solution is given by $(\pi_{21}, \pi_{22}, \pi_{23}) = (-2.66667, -18.2222, 0)$, which is then incorporated into the linking constraint to solve the $SP1^1$ (3.47). Its solution is given by $(\pi_{11}, \pi_{12}, \pi_{13}) = (-2.07317, 0, 0)$ and $\psi^1 = -0.902439$ whose primal variables are $\boldsymbol{y}^{(1} = (76.6667, 23.8889)$. The last step of this first iteration is the gap evaluation by means of formula (3.40). It gives a value of $138,91\%$, which is larger than $\epsilon = 1e^{-}6$. Then, using the dual variables $\boldsymbol{\pi_2}$, $\boldsymbol{\pi_1}$, ψ of $SP2$ and $SP1$, respectively, we build the cuts h_1^1 and h_1^2 according to (3.34). Since it is the only cut, we do not include the additional constraints (3.37) - (3.38). Instead, we turn h_1^2 into an equality. Thus imposing that w_2 will take the value of h_1^2 parameterized by x_2. Now, we jump to the second iteration.

In the second iteration, the Master Problem faces this mathematical problem:

$$\min_{x_2,w_1,w_2} -60x_2 + w_1 + w_2 \tag{3.48}$$

$$\text{s.t. } w_1 \leq 20.7317x_2 - 0.902439w_2 - 466.463$$

$$w_2 = 117.778x_2 - 4791.11$$

$$w_1, w_2 \in [-M, M]$$

$$x_2 \in \{0, 1\}$$

which gives $x_2 = 1$, $w_1 = 3771.67$ and $w_2 = -4673.33$. Then, fixing $\bar{x}_2 = 2$, the $SP2^2$ is first solved. Its solution gives the same values $(\pi_{21}, \pi_{22}, \pi_{23}) = (-2.66667, -18.2222, 0)$, which are then incorporated into the linking constraint to solve the $SP1^2$. Its solution also provides the same values $(\pi_{11}, \pi_{12}, \pi_{13}) = (-2.07317, 0, 0)$ and $\psi^2 = -0.902439$ whose primal variables are, however, different, $\boldsymbol{y}^{(2} = (75, 21.6667)$. The evaluated gap is less than $1e^{-}06$, and thus the algorithm finishes by reporting the solution $x_2^* = 1$ and $\boldsymbol{y}^* = (75, 21.6667)$ with objective function values $F_1 = -961.667$ and $F_2 = 4673.33$. This solution is the same as that reported by Saharidis & Ierapetritou in [156], although they report erroneously $F_2 = -1011.69$ in the final results table. We think it is a mistake, since their description of the example gives the same value as ours.

Chapter 4

The corridor generation algorithm

This chapter presents the procedures devised for generating the pool of candidate corridors under the routing model $M3$ (see section 2.3.3.3 for further information). They include two main building blocks which compute at most K_r rectilinear shortest paths and K_c circular shortest paths, constrained to infrastructure budgetary and planning horizon limitations in a similar way as stated by equations (2.4) and (2.36). These building blocks consist of ad hoc implementations of Yen's k-shortest path algorithm [189], which uses as sub algorithms Dijkstra's shortest path algorithm [58] and/or Floyd & Warshall's all shortest paths algorithm [75], [181]. The structure of the chapter is as follows. Firstly, an overview of the whole set of procedures is given. Then, the two main building blocks for constructing the two types of corridor topologies are presented. Thirdly, the user's input rules are discussed. Finally, some computational results are included to demonstrate the need for using this routing approximation.

4.1 Overview of the algorithm

The methodology which generates the corridor pool, from which stretches and nodes are assigned to the lines, comprises two main building blocks: the rectilinear corridor generation procedure ($RCGP$) and the circular corridor generation procedure ($CCGP$). After having discarded some peculiarities, they rely on the interaction of the Dijkstra's shortest path algorithm [58] and a modification of Yen's k-shortest path algorithm [189]. The former is embedded into the latter in order to generate, at most, K_r and K_c feasible rectilinear and circular corridors, respectively, for every pair of nodes $i, j \in N_{TP}^N$, such that $i < j$, satisfying some constraints.

In the state of the art, there are many authors that use constraint satisfaction in computing k-shortest paths. To cite just a few of them, Androutsopoulos & Zografos [4] and the references herein limit the time length of each path according to a given time window. Jim *et al* [99] do not allow building paths whose node departure times are not within a predefined list of times. More recently, and in the context of Public Transportation systems, Van der Zijpp & Catalano [174] and their earlier works referenced herein apply some user behavioral rules to eliminate unwanted corridors.

In our work, we apply the following constraints:

- Corridors must accomplish a minimum service frequency.
- Their total construction costs do not exceed the infrastructure budget.

These constraints are quite similar to the mathematical programming constraints (2.4) and (2.36). However, they are established and/or interpreted slightly differently, due to the lack of some data which is only known during the optimization phase. As we cannot determine the amount of demand that the corridor will

carry, we only limit the maximum duration of one service in the corridor. As for the construction cost limitations, we take only into account those associated with the stretches and terminal nodes of the corridors. Furthermore, for the k-shortest corridors whose $k_r > 1$ and $k_c > 1$, we add the construction cost of the initial node from which the $k_r - 1$ or $k_c - 1$ shortest path is extended. The remaining node costs are not considered since we do not know a priori the role of the intermediate nodes of the corridor.

Apart from these model constraints, two additional ones called "user behavior rules" are verified. They have already been applied to the k-shortest path algorithms by Zijpp & Catalano [174]. As the authors say, they are based on practical observation of passenger choice preferences and may allow shortening the size of the corridor pool (Λ).

The first rule is related to the maximum allowable deviation cost that a subpath can have with respect to the shortest path cost of a given pair of nodes. From now on, we will refer to the deviated subpath as a detour, like in the authors' work [174]. For the sake of coherence, we will also introduce the same notation to explain this rule. Given the following subsequence of links of a certain subpath P^k_{i-j}: $\{a_{i,k}, a_{i+1,k}, ..., a_{j,k}\}$, which forms a detour between nodes i and j, it is said that P^k_{i-j} is not feasible if the following conditions hold:

$$\sum_{h=i}^{j} L(a_{h,k}) > \phi_{max} D\left[N^s(a_{i,k}), N^e(a_{i,k})\right] \tag{4.1}$$

where $L(a_{h,k})$ stands for a function which maps the link $a_{h,k}$ to its length and $N^s(a_{i,k})$ and $N^e(a_{i,k})$ are also mapping functions related to the start and end node of link $a_{h,k}$, respectively. $D[.,.]$ denotes the shortest path cost between nodes i and j. This rule is then parameterized by the weight ϕ_{max}, whose value must be positive and higher than 1. Schnabel and Lohse [159] recommend a value of $\phi_{max} = 1.4$ for urban networks and $\phi_{max} = 1.25$ for motorways.

The second rule is associated with the minimum allowable deviation cost that a subpath can have with respect to the shortest path cost of a given pair of nodes. This rule is complementary to the aforementioned one and works in a similar manner. Given subsequent links of a certain subpath P^k_{i-j}: $\{a_{i,k}, a_{i+1,k}, ..., a_{j,k}\}$, which forms a detour between nodes i and j, it is said that P^k_{i-j} overlaps P^1_{i-j} (it is not feasible) if the following two conditions hold:

$$\sum_{h=i}^{j} L(a_{h,k}) > D\left[N^s(a_{i,k}), N^e(a_{i,k})\right] \tag{4.2}$$

$$\sum_{h=i}^{j} L(a_{h,k}) < \Delta_{min} \tag{4.3}$$

Condition (4.2) simply states that P^k_{i-j} is in fact a detour, whereas condition (4.3) establishes that the cost of this detour is under a threshold value (Δ_{min}), relative to the whole cost of path k (P^k). This value is expressed here in terms of the path cost units for respecting the original authors' definition in [174]. However, for practical implementation, it can be defined as a portion of the P^k cost, like parameter ϕ_{max}. So, it should be multiplied by the P^k cost like in equation (4.1). Schnabel and Lohse [159] recommend a value of $\Delta_{min} = 0.5$, so that the resulting detours cannot overlap more than 50%.

Observe that in both rule definitions we have used the word "cost" instead of "time" for the sake of generality.

4.1.1 The rectilinear corridor generation procedure

The outline of the $RCGP$ procedure is depicted in Figure 4.1 and written in pseudo-code in table 4.1. Its implementation details can be followed throughout tables 4.2 - 4.3. The procedure consists of two main steps: the computing all shortest rectilinear paths between every pair of nodes $i, j \in N_{TP}^N$, and obtaining their following $K_r - 1$ shortest rectilinear paths. The former is done by means of an adaptation of the Floyd & Warshall algorithm (FWA) ([75], [181]), whereas the latter uses an adaptation of Yen's k-shortest path algorithm [189] for rectilinear corridors in collaboration with Dijkstra's shortest path algorithm [58]. In both phases, the aforementioned planning and construction constraints are verified. Additionally, in the second phase, the aforementioned user behavior rules are also taken into account.

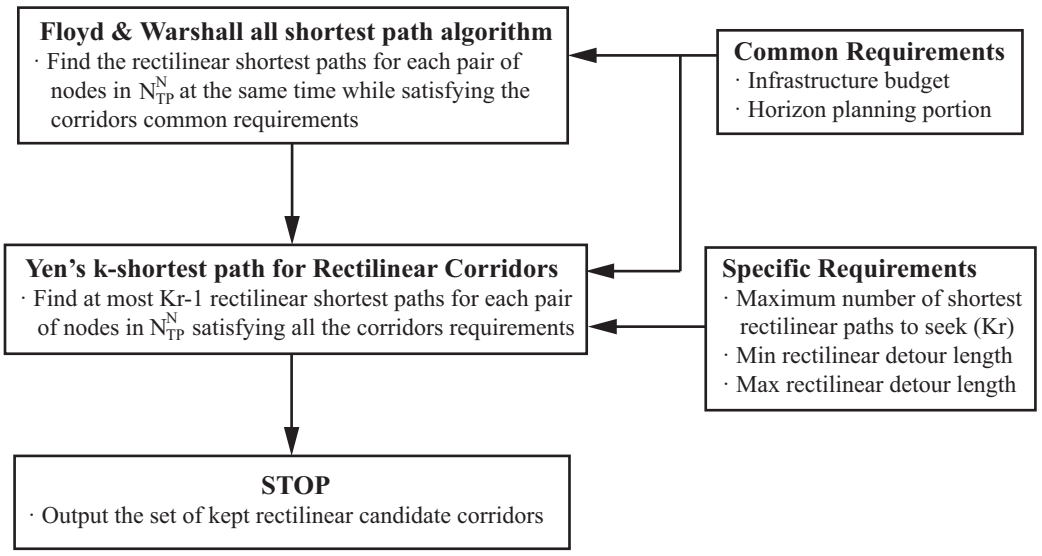


Figure 4.1: Overview of the rectilinear corridor generation procedure.

procedure RCGP (**in** $K_r,\ T,\ H,\ \bar{c}_{net},\ \Delta_{min}^r,\ \phi_{max}^r,$ **out** P)

$\tilde{T}(i,j) \leftarrow T(i,j) + T(j,i) \quad \forall (i,j) \in A_{TP}^N$

$P^1 \leftarrow$ Shortest Rectilinear Path for all pairs $i, j \in N_{TP}^N :\ i < j$ by calling

FWAllSP($\tilde{T}, \bar{c}_{net}, H, P^1$)

$\tilde{T}(i,j) \leftarrow \infty \quad \forall (i,j) \in A_{TP}^N$ **such that** $T(i,j) > \phi_{max} t(P_{ij}^1)$

for each $i, j \in N_{TP}^N$ **such that** $j > i$ **and** $P_{i,j}^1 \neq \emptyset$ **do**

$P_{ij} \leftarrow$ K-Shortest Rectilinear Path from i to j by calling

YSP4RC($K_r, \tilde{T}, P_{ij}^1, H, \bar{c}_{net}, \Delta_{min}^r, \phi_{max}^r, P_{ij}$)

end for

return P

end RCGP

Table 4.1: Pseudo-code of the rectilinear corridor generation procedure.

Observe that the $RCGP$ procedure properly initializes the transformed in-vehicle travel times matrix ($\tilde{T}$) before calling on subroutine **FWAllSP**, which implements the Floyd & Warshall algorithm. This initialization consists of merging the cell values of the upper and lower triangular matrices of the original in-vehicle travel times matrix (T). As a result, $\tilde{T}$ has duplicate values in the positions $\tilde{T}(i,j)$ and $\tilde{T}(j,i)$. Moreover, there is an update of this transformed matrix before going into the for loop. In this update, it applies the first user behavior rule and therefore eliminates those edges whose in-vehicle travel time costs exceed the upper rectilinear detour factor (ϕ^r_{max}). In the following subsections, the workings of each subroutine as well as its implementation details will be presented.

4.1.1.1 All shortest rectilinear paths procedure

The first phase of the $RCGP$ procedure computes all the shortest rectilinear paths for every pair of nodes $i,j \in N^N_{TP}$, such that $i < j$ immediately, due to an adaptation of the FWA algorithm [75], [181]. This algorithm works with an $N \times N$ matrix representing a directed graph $G = (N, A)$, where each cell element contains the shortest cost between its related pair of nodes, excluding the diagonal elements which are set to infinity to denote no connection. The algorithm uses mainly a triangular operation among two cell elements, which require two updates for their resolution: the pair of cell values involved, and the associated predecessor matrix cells. This complementary matrix stores all the trip data between each pair of nodes in $G = (V, E)$. For further details, the reader is referred to section 3.5.1 of chapter 3. In the following, we are going to explain the details of how the FWA was implemented for our proposes.

The subroutine **FWAllSP** in table 4.2 describes the FWA in pseudo-code form. Its adaptation includes a path recovery feature as well as the satisfaction of the aforementioned planning and construction constraints. Including the latter feature makes the algorithm somewhat innovative, although many other researchers employ constraint satisfaction in computing a preliminary set of paths. To mention some of them, Wei Fan & Machemehl [68] develop a pseudo-constrained k-shortest path algorithm in which they check minimal and maximal path distance constraints when computing a preliminary set of k-shortest paths in a similar way as the $RCGP$. Cipriani *et al* [30] carried out the same constraint checking in their sophisticated heuristic route generation algorithm ($HRGA$), which seeks to find two sets of paths: one satisfying the highest OD-demand pairs and other for carrying out the transfers among feeder lines.

The algorithm begins by initializing the matrix ST, which contains the minimum travel times traversing each $i,j \in N^N_{TP}$ in both directions. This is done in the following manner. Each cell $ST(i,j)$, such that $(i,j) \in A^N_{TP}$, is set to the travel time of the edge (i,j) and to its inverse (j,i), which are already held in the input matrix T. In contrast, if the cell position $(i,j) \notin A^N_{TP}$, we set it to infinity. To implement the path recovery feature, we also work with the predecessor matrix Pred, where each cell Pred(i,j) will contain the immediate predecessor of j in the shortest path from i to j. Consequently, we need also to initialize matrix Pred as follows. Each cell Pred(i,j) such that $(i,j) \in A^N_{TP}$ will set to i, whereas the cells Pred(i,j) such that $(i,j) \notin A^N_{TP}$ will be zero.

Having finished the initialization phase, the existence of intermediate nodes k (each origin i to each destination j) is verified in order to minimize the initial travel time from i to j. During that phase, we update the ST and Pred matrices according to the changes.

Finally, we recover all the feasible paths in list P^1. To do that, we first construct the temporary shortest path $P^{'}$ for each pair $i,j \in N^N_{TP} : i < j$. It is carried out as follows. We begin with the first predecessor node of the destination j, which is contained in Pred(i,j), and then we go backward until we reach the origin i. Secondly, the extreme nodes of $P^{'}$ are set as working nodes by means of mapping function NS. The planning and construction constraints, stated in section 4.1, are then verified in collaboration with the mapping functions $t(P^k)$ and $C(P^k)$, respectively. If they are verified successfully, we save the inverse as

procedure FWAllSP (**in** $T,\ \bar{c}_{net},\ H,$ **out** P)

$ST(i,j) \leftarrow T(i,j), \quad \forall (i,j) \in A^N_{TP},\ ST(i,j) \leftarrow \infty, \quad \forall (i,j) \notin A^N_{TP}$

Pred$(i,j) \leftarrow i, \quad \forall (i,j) \in A^N_{TP},\ Pred(i,j) \leftarrow 0, \quad \forall (i,j) \notin A^N_{TP}$

for each $k,i,j \in [1..|N^N_{TP}|]$ **such that** $i \neq j,\ k \neq i,\ k \neq j$ **do**

$t'_{ij} \leftarrow ST(i,k) + ST(k,j)$

if $t'_{ij} < ST(i,j)$ **then**

$ST(i,j) \leftarrow t'_{ij},\ Pred(i,j) \leftarrow$ Pred(k,j)

end if

end for

for each $i,j \in [1..|N^N_{TP}|]$ **such that** $i < j$ **do**

$P^1_{ij} \leftarrow \emptyset$

if $ST(i,j) \leq H$ **then**

$P' \leftarrow \{j\},\ k \leftarrow j$

while pred$(i,k) \neq i$ **do**

$k \leftarrow$ pred$(i,k),\ P' \leftarrow P' \cup \{k\}$

end while

$P' \leftarrow P' \cup \{i\},\ NS(R, P'_1, P'_t)$

if $C(P') \leq \bar{c}_{net}$ **then** $P^1_{i,j} \leftarrow$ reverse(P');

end if

end for

return P^1

end FWAllSP

Table 4.2: Pseudo-code of the adaption of the Floyd & Warshall all shortest paths algorithm.

the permanent shortest path for pair $i,j \in N^N_{TP}$ in sublist $P^1_{i,j}$.

4.1.1.2 Yen's k-shortest path algorithm for rectilinear corridors

The second phase of the $RCGP$ procedure computes the $K_r - 1$ (following) shortest rectilinear corridors for each feasible shortest rectilinear corridor found in the previous phase. It adapts Yen's k-shortest path algorithm [189] to cope with undirected graphs and to satisfy some constraints.

Subroutine **YSP4RC** of table 4.3 shows the details of this phase in pseudo-code. It employs: the in-vehicle travel times matrix $\tilde{T}$; list B, which contains all the computed feasible paths; and the sublist $Q \subset B$, which contains all the computed feasible not chosen as some k-shortest paths in P^k_{ij}.

The subroutine begins by initializing list Q to null and list B to the first shortest rectilinear path for a given pair $(i, j) \in N_{TP}^{N}$. Then, the iterative part of the algorithm starts. Every k-iteration seeks all the existing feasible new paths, such that they contain a common subpath (P_{1-i}^{k-1}) coming from the $k-1$ shortest path P^{k-1}, which itself starts at the source node P_1^{k-1} and ends at an intermediate node P_i^{k-1}. However, they differ from a detour subpath which starts at P_i^{k-1} and ends at terminal node P_t^{k-1}. Feasibility is checked by verifying if every new path satisfies the aforementioned planning and construction constraints, as well as the user input rules.

To compute these new subpaths, the method works with a copy of the in-vehicle travel times matrix ($T^{'}$). This copy may be modified according to the position of the intermediate node P_i^{k-1} (under process) in path P^{k-1} and the links held in the common subpath P_{1-i}^{k-1}. This modification is carried out in two steps:

1. The links adjacent to P_i^{k-1} are eliminated. These links are contained in the paths of list B. Their initial subpath from the first node B_j^1 to the intermediate node B_j^i overlaps with the initial subpath of $k-1$, from its first node P_1^{k-1} to the intermediate P_i^{k-1}.
2. The links adjacent to the nodes in the initial subpath of path $k-1$ are eliminated, from its first node P_1^{k-1} to the antecessor node of the intermediate node P_i^{k-1}.

These eliminations are carried out virtually by setting to infinity the costs associated with the links. Thus, it prevents them from being selected. The second step mentioned above is not specified in the original Yen algorithm [189], but it is required to deal with undirected graphs since it prevents more than one visit to any node already contained in the initial subpath P^{k-1}.

Having updated $T^{'}$ properly, it computes the shortest rectilinear path from P_i^{k-1} to P_t^{k-1} by calling on the subroutine **DSP** of table 4.4. It is an implementation of Dijkstra's algorithm [58] with path recovery. For a mathematical description of the subroutine, some notations are introduced. For node i, let A_i represent the arc adjacency list of node i. Let N denote the set containing all the nodes with permanent labels and $\bar{N}$ be the set containing all the nodes with temporary labels. Initially, every node has a temporary label and the time costs from the source node s to all other nodes are initialized to ∞. At each step, the algorithm chooses the node $i \in \bar{N}$ with the least temporary label time and makes it permanent. Then, it records its predecessor index and updates the temporary values of all nodes $j \in A_i$. This process is repeated until all nodes become permanent ones. Finally it outputs the shortest path P, from the source node s to the destination d by reversing the chain of the predecessor nodes, beginning with d and finishing in s.

If a new subpath (S) linking the intermediate node P_i^{k-1} to the final node P_t^{k-1} is found, the algorithm verifies whether S satisfies the user behavior rules (stated in section 4.1). Firstly, it checks the second rule, since this rule only needs one evaluation, as follows. It compares the travel time cost of the rectilinear detour ($t(S)$) with the travel time cost of the $k-1$ shortest path ($t(P^{k-1})$). If the $t(S)$ is equal to or higher than a given portion (Δ_{min}^{r}) of $t(P^{k-1})$, then it creates a new path R by appending S to the initial subpath $P_{1-(i-1)}^{k-1}$ and proceeds to verify the first rule as follows. For every pair of nodes r, s, such that r is held in the common subpath ($r \in P_{1-(i-1)}^{k-1}$) and s is contained in the detour subpath ($s \in S_{2-t}$), it compares the travel time cost from r to s in the new subpath ($t(R_{r-s})$) with the travel time cost of the shortest path from r to s ($t(P_{rs}^{1})$). If $t(R_{r-s})$ is equal to or lower than a given multiple (ϕ_{max}^{r}) of $t(P_{rs}^{1})$, the rule is satisfied. If any of these two rules are satisfied, path R is rejected and a new k-iteration is evaluated.

Having verified these rules, P_i^{k-1} is set as a working node by means of mapping function NS, and R is verified for whether or not it satisfies the planning and construction constraints, as stated in section 4.1. This is done by means of the mapping functions $t(R)$ and $C(R)$, respectively. If so, it is appended to lists Q and B. Otherwise, it is rejected.

procedure YSP4RC (**in** $K_r,\ T,\ P,\ \bar{h},\ \bar{c}_{net},\ \Delta^r_{min},\ \phi^r_{max},$ **out** P)

$Q \leftarrow \emptyset,\ B \leftarrow P$

for each $k \in [2...K_r]$ **do**

for each $i \in [1...|P^{k-1}| - 1]$ **do**

$T' \leftarrow T$

for each $j \in [1...|B|]$ **such that** $|B^j| \geq |P^{k-1}_{1-i}|$ **and** $B^j_{1-i} = P^{k-1}_{1-i}$ **do**

$T'(P^{k-1}_i, B^j_{i+1}) \leftarrow \infty,\ T'(B^j_{i+1}, P^{k-1}_i) \leftarrow \infty$

end for

$T'(r,s) \leftarrow \infty, \quad \forall (r,s) \in A^N_{TP},\ r < s :\ r \circ s \in P^{k-1}_{1-(i-1)}$

$S \leftarrow$ Shortest rectilinear path from P^{k-1}_i to P^{k-1}_t by calling **DSP**$(T', P^{k-1}_i, P^{k-1}_t, S)$

if $S \neq \emptyset$ **then**

if $t(S) < \Delta^r_{min} t(P^{k-1})$ **then** go to next $i \in [1...|P^{k-1}| - 1]$

$R \leftarrow P^{k-1}_{1-(i-1)} \cup S,\ NS(R, R_i)$

for each $r \in P^{k-1}_{1-(i-1)}$ **and** $s \in S_{2-t}$ **do**

if $t(R_{r-s}) > \phi^r_{max} t(P^1_{rs})$ **then** go to next $i \in [1...|P^{k-1}| - 1]$

end for

if $C(R) \leq \bar{c}_{net}$ **and** $t(R) \leq \bar{h}$ **then** $Q \leftarrow Q \cup \{R\},\ B \leftarrow B \cup \{R\}$

end if

end for

$P^k \leftarrow$ **Select** R with minimum $t(R)$ from Q, $Q \leftarrow Q - \{R\}$

end for

return P

end YSP4RC

Table 4.3: Pseudo-code of the adapted Yen's k-shortest path procedure for rectilinear corridors.

To complete the k^{th}-iteration, the shortest travel time is chosen from the list of candidate paths Q and it is then set to the k-shortest feasible rectilinear path. Finally, this chosen path is deleted from Q. The process is repeated at most $K - 1$ times, providing that the list Q is not empty when it tries to set a k-shortest path, with $k < K$. If so, the procedure **YSP4RC** is finished earlier.

4.1.2 The circular corridor generation procedure

The outline of the $CCGP$ procedure is depicted in Figure 4.2 and written down in pseudo-code in table 4.5. Its implementation details can be followed throughout tables 4.6 - 4.7 and 4.4. The procedure consists of two main steps: computing all shortest circular paths for every node $i \in N^N_{TP}$, and obtaining their following $K_c - 1$ shortest circular paths. The former is done by means of subroutine **CSC,** which drives the aforemen-

procedure DSP (in $T,\ s,\ d,$ **out** P)

$N \leftarrow \emptyset,\ \bar{N} \leftarrow N_{TP}^{N}$

$t_s \leftarrow 0,\ t_i \leftarrow \infty\ \ \forall i \in N_{TP}^{N} \backslash s$

$\text{pred}(s) \leftarrow 0,\ \text{pred}(i) \leftarrow -1\ \ \forall i \in N_{TP}^{N} \backslash s$

while $|N| < N_{TP}^{N}$ **do**

Select $i \in \bar{N}$ **such that** $t_i = \min\left\{t_j \mid \forall j \in \bar{N}\right\}$

$N \leftarrow N \cup i,\ \bar{N} \leftarrow \bar{N} - i$

for each $(i,j) \in A_i$ **do**

if $t_j > t_i + T(i,j)$ **then**

$t_j \leftarrow t_i + T(i,j),\ \text{pred}(j) \leftarrow i$

end if

end for

end while

$P \leftarrow \emptyset,\ P^{'} \leftarrow \{d\}$

$i \leftarrow d$

while $t_{\text{pred}(i)} < \infty$ **and** $\text{pred}(i) \neq s$ **do**

$i \leftarrow \text{pred}(i),\ P^{'} \leftarrow P^{'} \cup \{i\}$

end while

if $t_{\text{pred}(i)} < \infty$ **then**

$P^{'} \leftarrow P^{'} \cup \{s\},\ P \leftarrow \text{reverse}(P^{'})$

end if

return P

end DSP

Table 4.4: Pseudo-code of the adaptation of Dijkstra's shortest path algorithm.

tioned adaptation of Dijkstra's algorithm [58]; the latter uses an adaptation of Yen's k-shortest path algorithm [189] for circular corridors in collaboration with an adaptation of Dijkstra's algorithm. The subroutine **CSC** is used as well. In both phases, the planning and construction constraints (stated in section 4.1) are verified. Regarding the user behavior rules (also stated in section 4.1), the first one is also taken into account in both phases whereas the second is checked only in the second phase.

Observe that the $CCGP$ procedure properly initializes the transformed in-vehicle travel times matrix $\tilde{T}$ in two steps. Firstly, the cell values of the upper and lower triangular matrices of the original in-vehicle travel times matrix (T) are merged. As a result, the matrix $\tilde{T}$ has duplicate values in positions $\tilde{T}(i,j)$ and $\tilde{T}(j,i)$. Secondly, the first user behavior rule is verified, i.e., those edges whose in-vehicle travel time costs exceed the upper circular detour factor (ϕ_{max}^{c}) are eliminated. In the following subsections, the workings of each subroutine as well as its implementation details are presented.

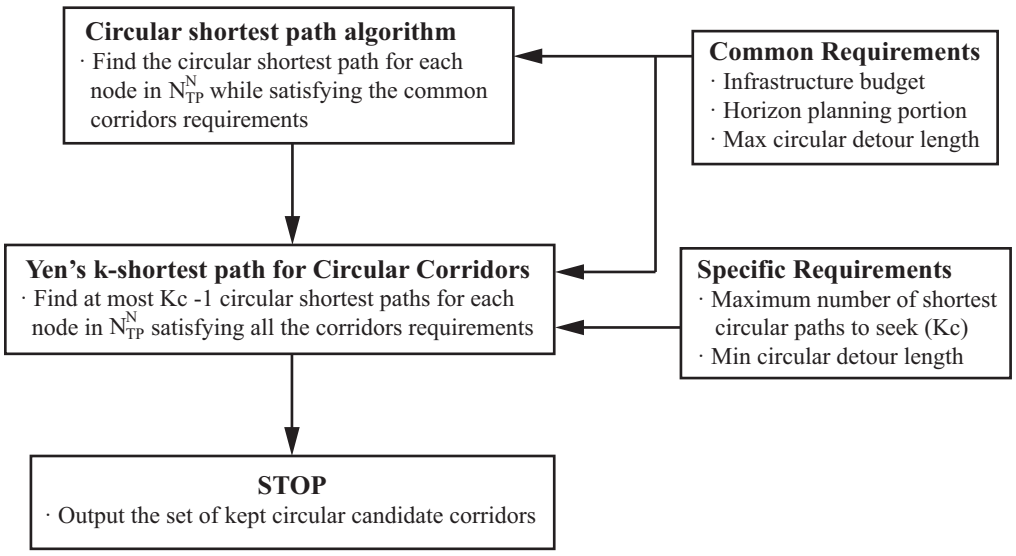


Figure 4.2: Overview of the circular corridor generation procedure.

procedure (**in** $K_c,\ T,\ P^1,\ \bar{c}_{net},\ \bar{h},\ \Delta^c_{min},\ \phi^c_{max},$ **out** P)

$\tilde{T}(i,j) \leftarrow T(i,j) + T(j,i) \quad \forall (i,j) \in A^N_{TP}$

$\tilde{T}(i,j) \leftarrow \infty \quad \forall (i,j) \in A^N_{TP}$ **such that** $T(i,j) > \phi^c_{max} t(P^1_{ij})$

for each $i \in N^N_{TP}$ **do**

$P^1_i \leftarrow$ Shortest Circular Path from i to i by calling **SCPath**($\tilde{T}, i, \bar{c}_{net}, \bar{h}, \phi^c_{max}, P^1_i$)

if $P^1_i \neq \emptyset$ **then**

$P_i \leftarrow$ Shortest K-Circular Paths from i to i by calling

YSP4CC($K_c, \tilde{T}, P^1_i, \bar{h}, \bar{c}_{net}, \Delta^c_{min}, \phi^c_{max}, P_i$)

end if

end for

return P

end

Table 4.5: Pseudo-code of the circular corridor generation procedure.

4.1.2.1 Circular shortest path algorithm

The first phase of the $CCGP$ procedure computes all the shortest circular paths for every node $i \in N^N_{TP}$ by means of subroutine **CSC** (shown in table 4.6), which drives the aforementioned subroutine **DSP**. It works as follows. Firstly, it initializes the list of candidate shortest circular paths Q to null, and stores all the nodes which are adjacent to node n in δn. We refer to n as the node from which the shortest circular path will start and end. Secondly, the iterative part of the procedure starts. It basically performs the following steps for every $i \in \delta_n$:

1. It duplicates the travel times matrix T into $T\prime$ and deletes from it the direct link (if it exists) that connects the pair i, n and its inverse.

2. It computes the temporary shortest rectilinear path from i to n by means of the subroutine **DSP** (shown in table 4.4) ,which is an adaptation of Dijkstra's shortest path algorithm, and stores it in S.
3. If S exists, it is verified that S meets the first user behavior rule (stated in section 4.1). If so, the new circular path given by $R = \{n\} \cup S$ is created and it is verified that R meets the planning and construction constraints (stated also in section 4.1). If all these requirements are satisfied, R is appended to the list Q. Otherwise, R is rejected.

Having finished that iterative part, we set the path held in Q with the minimum travel time to the shortest circular path P.

4.1.2.2 Yen's k-shortest path algorithm for circular corridors

The second phase of the $CCGP$ procedure computes the $K_c - 1$ (following) shortest circular corridors for each feasible shortest circular corridor found in the previous phase. This procedure adapts Yen's k-shortest path algorithm [189] to cope with undirected graphs and satisfy some constraints.

procedure CSC (**in** $T,\ n,\ \bar{c}_{net},\ \bar{h},\ \phi^c_{max},$ **out** P)
  $P \leftarrow \emptyset,\ Q \leftarrow \emptyset,\ k \leftarrow 1$
  $\delta_n \leftarrow \left\{ i \in N^N_{TP} \backslash n : \exists (i,n)\ \text{o}\ (n,i) \in A^N_{TP} \right\}$
  for each $i \in \delta_n$ **do**
    $T^{'} \leftarrow T,\ T^{'}(i,n) \leftarrow \infty,\ T^{'}(n,i) \leftarrow \infty$
    $S \leftarrow$ Shortest Rectilinear Path from i to n by calling **DSP**($T^{'}, i, n, S$)
    if $S \neq \emptyset$ **then**
      for each $s \in S$ **do**
        if $t(R_{1-s}) > \phi^c_{max} t(SP_{n,s})$ **then** go to next $i \in \delta_n$
      end for
      $R \leftarrow \{n\} \cup S,\ NS(R,n)$
      if $C(R) \leq \bar{c}_{net}$ **and** $t(R) \leq \bar{h}$ **then** $Q^k \leftarrow R,\ k \leftarrow k+1$
    end if
  end for
  if $Q \neq \emptyset$ **then** $P \leftarrow$ **Select** R with the minimum $t(R)$ in Q
  return P
end CSC

Table 4.6: Pseudo-code of the shortest circular path procedure.

Subroutine **YSP4CC** (written down in table 4.7) shows the pseudo-code details of this phase. The core of this method is quite similar to the procedure **YSP4RC** of table 4.3, i.e., it works with a transformed travel times matrix $\tilde{T}$, a list B containing all the computed feasible paths, and a sublist $Q \subset B$ containing all the computed feasible paths which have not been chosen as some k-shortest paths in P^k_{ij}. However, it differs from the way it changes the matrix $T^{'}$ and how it obtains the new subpath at each k-iteration.

procedure YSP4CC (**in** $K_c,\ T,\ P,\ \bar{h},\ \bar{c}_{net},\ \Delta^c_{min},\ \phi^c_{max},$ **out** P)

$Q \leftarrow \emptyset,\ B \leftarrow P$

for each $k \in [2...K_c]$ **do**

for each $i \in [1...|P^{k-1}| - 1]$ **do**

$T' \leftarrow T$

for each $j \in [1...|B|]$ **such that** $|B^j| \geq |P^{k-1}_{1-i}|$ **and** $B^j_{1-i} = P^{k-1}_{1-i}$ **do**

$T'(B^{k-1}_i, B^j_{i+1}) \leftarrow \infty\ T'(B^j_{i+1}, P^{k-1}_i) \leftarrow \infty$

end for

$T'(r,s) \leftarrow \infty \quad \forall (r,s) \in A^N_{TP},\ r < s :\ r \circ s \in P^{k-1}_{2-(i-1)}$

if $P^{k-1}_i = P^{k-1}_t$ **then**

$S \leftarrow$ Shortest circular path from P^{k-1}_i to t by calling **CSC**(T', P^{k-1}_i, S)

else

$S \leftarrow$ Shortest rectilinear path from P^{k-1}_i to t by calling **DSP**$(T', P^{k-1}_i, P^{k-1}_t, S)$

end if

if $S \neq \emptyset$ **then**

if $t(S) < \Delta^c_{min} t(P^{k-1})$ **then** go to next $i \in [1...|P^{k-1}| - 1]$

$R \leftarrow P^{k-1}_{1-(i-1)} \cup S,\ NS(R, R_i)$

for each $r \in P^{k-1}_{1-(i-1)}$ **and** $s \in S_{2-t}$ **do**

if $t(R_{r-s}) > \phi^c_{max} t(P^1_{rs})$ **then** go to next $i \in [1...|P^{k-1}| - 1]$

end for

if $C(R) \leq \bar{c}_{net}$ **and** $t(R) \leq \bar{h}$ **then** $Q \leftarrow Q \cup \{R\},\ B \leftarrow B \cup \{R\}$

end if

end for

$P^k \leftarrow$ **Select** R with minimum $t(R)$ from $Q,\ Q \leftarrow Q - \{R\}$

end for

return P

end YSP4CC

Table 4.7: Pseudo-code of the adapted Yen's k-shortest path procedure for circular corridors.

Recall from section 4.1.1.2 that the modification of the matrix T' involved two operations. On the one hand, we eliminated the links adjacent to the intermediate node P^{k-1}_i, belonging to the initial subpaths held in the list B which overlapped with the initial subpath P^{k-1}. On the other hand, we also eliminated every link adjacent to the nodes held in the initial subpath of P_{k-1}, except the last one. The latter is slightly modified in the following way. The links to be eliminated must be adjacent to the nodes of the subpath of P^{k-1}, beginning at P^{k-1}_2 and ending at P^{k-1}_{i-1}. In that manner, it can find a subpath S with the last link connecting

the intermediate node P_i^{k-1} to its terminal node S_t.

The way it obtains the new subpath depends on the relationship of the intermediate node with the terminal node of path P_{k-1}. If they are the same node, which occurs when it looks for a subpath starting at the first node P_1^{k-1}, it uses the aforementioned procedure **CSC**, since the searched subpath is circular. Otherwise, if they are different, it calls the procedure **DSP** since the searched subpath is rectilinear.

We refer the reader to the explanation of the $RCGP$ procedure in section 4.1.1.2 to understand the rest of the steps that we have skipped for the sake of simplification. Notice that, some nomenclature should be changed. For instance, when referring to Δ^r_{min}, one should change it to Δ^c_{min}. Analogously, Φ^r_{max} should be replaced with Φ^c_{max}.

In the appendix, we present an illustrative example where the circular shortest path algorithm and the Yen's k-shortest path algorithm for circular corridors are applied to a small network.

4.2 Preliminary computational results

This section is devoted to demonstrating how efficient the algorithm is in finding a very reduced set of candidate corridors, so long as the user behavior rules (stated in section 4.1) are well parameterized. This study is carried out by applying different configurations of these rules to two test networks. The first one is a complete 6-node railway network, shown in Figure 8.1; whereas the other is the 9-node and 26-link railway network depicted in Figure 8.2. We refer the reader to section 8.1.1 for further details regarding the input parameters associated with both networks.

Table 4.8 shows the resulting sizes of the corridor pools. The first column denotes the network used. Label $N1$ stands for the 6-node network, whereas label $N2$ is related to the 9-node network. The second column indicates the parameterizations of the user behavior rules. We have chosen three types of scenarios: the first consists of disabling the effect of these rules, whereas the second and third scenarios enable the effect of the user behavior rules. The second scenario applies the configuration suggested by Schnabel and Lohse [159], whereas the third one uses an accurate setup. The following two columns contain the resulting values of the K parameter for the rectilinear and circular types of corridors, respectively. The next two columns stand for the total number of corridors found for each topology. The last column denotes the whole number of corridors found (circular + rectilinear). Observe that we haven't mentioned the setup of the infrastructure budgetary and planning horizon constraints, since they aren't active.

Results show that the corridor pool size decreases because the applied configuration is more accurate. Moreover, as the network size increases, the accurate setup reduces the pool more and more. Furthermore, the suggested configuration of Schnabel and Lohse [159] approaches this pool. The latter happens partially because of the connectivity grade of the network. Observe that the 6-node network is complete, whereas the 9-node network is quite sparse.

Regarding the search for the K parameter, we have proceeded as follows. For the first and third scenarios of each network, the $RCGP$ and $CCGP$ procedures are repeated as long as the whole set of optimal rectilinear and circular corridors remain unfound. These corridors are provided by the resolution of the $RTNPD$ model under the routing formulations $M1$ or $M2$ applied to the instances shown in table 2.1. Consequently, several calls are made on these procedures. Initially, we begin with low values of K, which are iteratively incremented by one unit. On the other hand, in the second scenario, we proceed slightly differently. We also made several calls to these procedures but relaxed the stopping condition. Thus, the final set of corridors does not necessarily contain all the optimal corridors. This situation arises in both networks, since the Schnabel and Lohse configuration is so constrained that no feasible corridors (corridors satisfying the user

Network	Scenarios	K_r	K_c	$\|\Lambda^r\|$	$\|\Lambda^c\|$	$\|\Lambda\|$
N1	$\Delta^r_{min} = \Delta^c_{min} = 0$ $\phi^r_{max} = \phi^c_{max} = \infty$	45	256	675	1536	2211
	$\Delta^r_{min} = \Delta^c_{min} = 0.5$ $\phi^r_{max} = \phi^c_{max} = 1.4$	8	6	89	17	106
	$\Delta^r_{min} = 0.19,\ \Delta^c_{min} = 0.24$ $\phi^r_{max} = 13,\ \phi^c_{max} = 14$	45	237	675	1292	1967
N2	$\Delta^r_{min} = 0$ $\phi^r_{max} = \infty$	18	-	355	-	355
	$\Delta^r_{min} = 0.5$ $\phi^r_{max} = 1.4$	7	-	136	-	136
	$\Delta^r_{min} = 0.55$ $\phi^r_{max} = 5.8$	17	-	295	-	295

Table 4.8: Corridor pool sizes obtained by applying different configurations of the user behavior rules.

behavior rules) exist, neither for $K_r > 8$ and $K_c > 6$ in the 6-node network nor for $K_r > 7$ in the 9-node one.

To conclude this section, table 4.9 shows the influence of the corridor pool size in the quality of the solution, as well as in the performance. The first column stands for the network identifier. The second column denotes the experiment identifier, whereas the third column denotes the modified parameter of each experiment, i.e., the number of lines under consideration. The fourth column indicates the routing model. The fifth column contains the size of the corridor pool, which is associated with a configuration of the user behavior rules. The next two columns contain the values of the global objective function and the user objective function. The seventh column shows the computing time. Finally, the last column gives the gap between the routing model solutions M1-2 and M3 under the different configurations of the user behavior rules. Observe that, for the first and reference row of the gap computation for each experiment and network, we have chosen the optimal solution given by the routing model with non-fixed corridors and with the lowest computing time (see table 2.1).

Results show that, in general, the more we reduce the size of the corridor pool, the less time we need to reach optimality. Moreover, the time gaps become wider as the network size increases. On the other hand, we observe that for the scenarios of type 2, whose corridor pools do not include the whole set of optimal line corridors stated by routing models $M1$-$M2$, their solutions are not so far from the optimum. However, the reduction in computing time is much higher. Consequently, it seems quite reasonable to apply the proposed configuration of Schnabel and Lohse for the user behavior rules.

Network	Exp.	$\|L\|$	R.M.	Λ	F.Obj	F. Users	T_{CPU}	Gap
N1	1	2	M1	-	184034	183225	1	-
			M3	2211	184034	183225	5	0 %
				106	216749	216206	1	18 %
				1967	184034	183225	4	0 %
	2	3	M1	-	157036	155962	42	-
			M3	2211	157036	155962	93	0 %
				106	188964	188232	3	20%
				1967	157036	155962	133	0 %
	3	4	M1	-	147612	146471	1586	-
			M3	2211	147612	146471	7234	0 %
				106	174004	173131	33	18%
				1967	147612	146471	5557	0 %
N2	1	2	M2	-	782332	781481	3	-
			M3	355	782332	781481	5	0 %
				136	784851	784156	2	$< 1\%$
				295	782332	781481	4	0 %
	2	3	M2	-	758634	757608	1125	-
			M3	355	758634	757608	173	0 %
				136	764078	763161	10	$< 1\%$
				295	758634	757608	104	0 %
	3	4	M2	-	742175	741038	5001	-
			M3	355	742175	741038	6710	0 %
				136	750325	749317	105	1 %
				295	742175	741038	3208	0 %

Table 4.9: Influence of the corridor pool size for the experiments performed in the test networks.

Chapter 5

The line splitting algorithm

This chapter presents the line splitting algorithm (LSA), which is a heuristic framework for dealing with the resolution of multiple lines. It basically consists of the resolution of a series of $RTNPD$ instances, where only a single line can be constructed. The structure of the chapter is as follows. Firstly, a formal description of the LSA algorithm is given. Then, we conduct a test in which the two LSA variants are compared with the direct resolution of the model using all the routing model formulations. In this way, we demonstrate why this algorithm is worth using.

5.1 Motivation

The motivation for developing the line splitting algorithm (LSA) comes from the excessive amount of time taken to solve an instance where more than one line can be constructed, even if the network size is small (see the preliminary results section 2.4 in the Modeling Approach chapter). Furthermore, we have found that the increased time is not linear as the number of lines under construction grows.

Roughly speaking, the LSA works as follows. It splits the $RTNPD$ problem resolution into L_{max} subproblems, where L_{max} stands for the maximum number of lines under construction. Each subproblem considers only one single line under construction and a full/partial assignment of the total OD-demand according to the iteration number and the value of L_{max}. The remaining previously constructed lines play the role of fictitious working lines. Having solved that subproblem, some sets and parameters related to the infrastructure resources are updated according to the subproblem solution. In that update, a new line can also be transformed if it is constructed as a working line for the next subproblem's resolutions. The last solved subproblem contains a heuristic solution of the model.

The goal of the LSA is to solve the $RTNPD$ model efficiently, keeping a good tradeoff between the solution quality and the solving time. The main idea of the heuristic is based on previous computational experiences, as we have found relatively slow times in reaching the optimal solution in cases where only one line was constructed at low demand.

5.1.1 The working of the algorithm

The big picture of the LSA is depicted in Figure 5.1, where a detailed description of its implementation is shown in Tables 5.1 and 5.2. The main routine of the algorithm is the *LSA* procedure, described in Table 5.1. It takes as inputs all data sets and parameters which conform to the public transportation graph (G_{TP}), the OD-demand (g), and the maximum number of lines which can be constructed (L_{max}). An additional parameter δ_p is included to establish how the demand will be assigned.

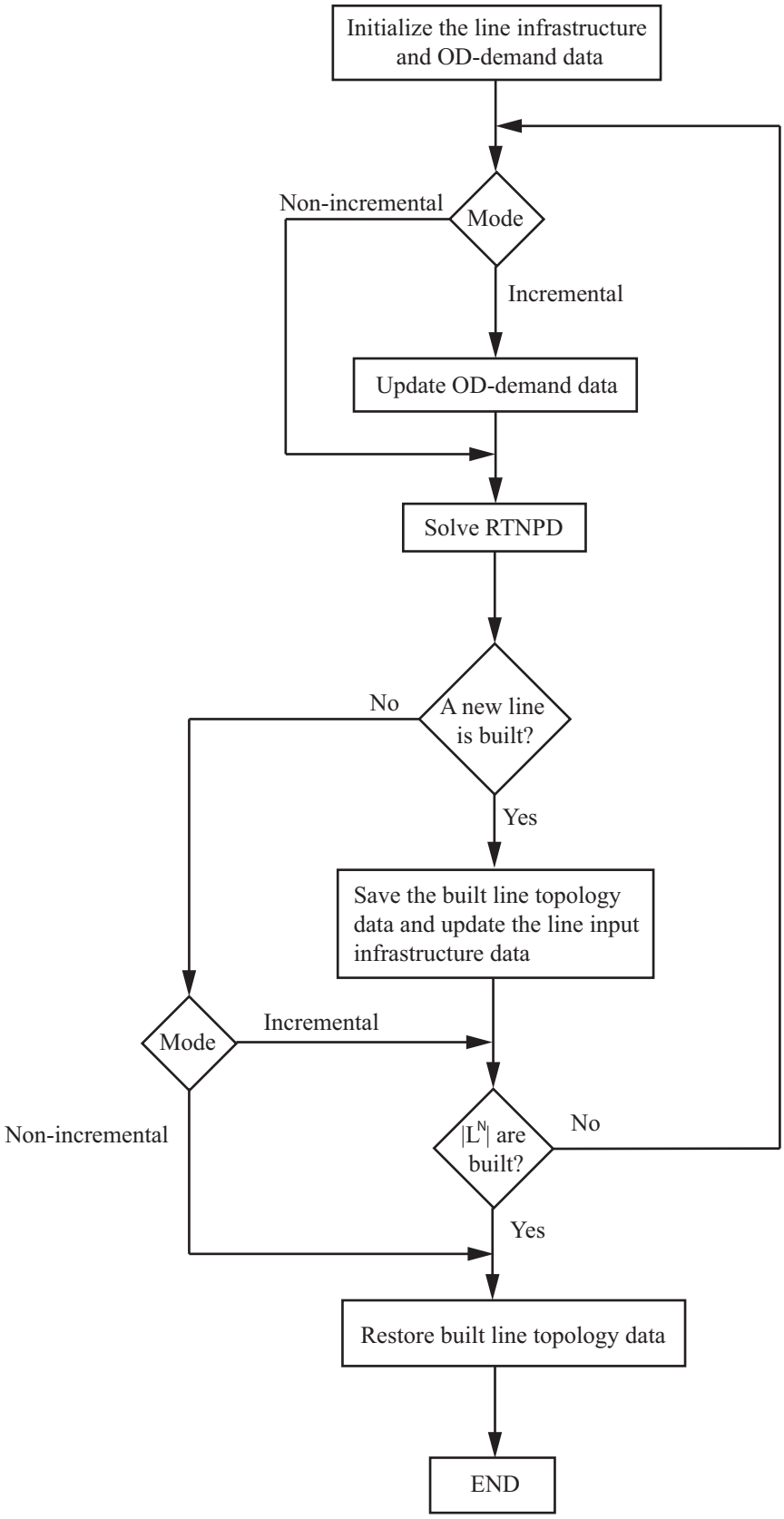


Figure 5.1: Overview of the line splitting algorithm.

The LSA procedure can be split into two phases. The first phase consists of initializing the data sets and parameters related to the network layout, which will be modified at each iteration of the algorithm, as well as initializing the incremental load portion Δg_p, which will be used later to update the passengers demand. The second phase is an iterative process where a series of subproblems are solved. Each subproblem consists of a mathematical model like the one presented in section 2.3, where only a new line can be constructed according to the nodes and stretches given by sets N_{TP}^N and A_{TP}^N. The rest of the lines included in the resolution are treated as working lines, having arisen from the heuristic input or from the lines constructed in previous subproblem resolutions (fictitious working lines).

Having solved each subproblem, a check is performed on whether or not there are any fictitious working lines with no services. If so, the construction costs of the infrastructure resources and the infrastructure budget are updated according to the stretches and stops held exclusively in these lines. Finally, the data sets related to these lines are dropped. These steps are carried out by means of the procedure $DelNoServLines$, shown in table 5.2.

Having finished that step, it is verified whether or not a new line has been constructed. If so, we make a backup of the current routing decision variables and update some network infrastructure data according to the current subproblem resolution. This update includes the transformation of the new constructed line into a working one, as well as the reduction of the infrastructure budget –provided that the cost is positive for some used stretch or bus stop, as established by sets N_{TP}^N and A_{TP}^N. Finally, the costs of the infrastructure resources used exclusively in this line are set to zero. In that manner, their construction costs are not taken into account in the next subproblem resolutions.

That second phase is repeated until the maximum number of iterations (L_{max}) is reached. Additionally, if $\delta_p = 1$, it is also verified whether a new line has been constructed. If some of these requirements are not met, the LSA finishes and the current solution of the k^{th} resolution of the subproblem becomes the final solution.

Notice that at each subproblem resolution the number of vehicles and its services performed at every line are recomputed so that the previous vehicles and services assigned to it will not be taken into account.

5.1.2 LSA Variants

To date, two variants of the LSA have been implemented. The difference between them lies in the amount of demand to be satisfied in each subproblem resolution. That amount is set by means of the δ_p parameter. For $\delta^p = 1$, the full origin-destination passenger demand is assigned, whereas for $\delta^p = 0$, only a portion of that demand is assigned, according to L_{max} and the current k^{th} iteration. For instance, suppose that a maximum of 10 lines can be constructed ($L_{max} = 10$). If we are at the first iteration, 10% of the demand will be assigned to the lines; whereas, if we are at the fifth one, 50% of the demand will be satisfied. Clearly, these variants modify the heuristic's behavior. However, they only take effect when we set $L_{max} > 1$.

procedure LSA (**in** G_{TP}, g, L_{max}, δ_p, **out** G_{TP})

$L^{E(1} \leftarrow L^E, \;\; L^{N(1} \leftarrow \{|L^{E(1}| + 1\}, \;\; L^{(1} \leftarrow \{L^{N(1} \cup L^{E(1}\}$

$c_i^{c(1} \leftarrow c_i^c \;\; \forall i \in N_{TP}^N, \quad c_{ij}^{c(1} \leftarrow c_{ij}^c \;\; \forall (i,j) \in A_{TP}^N, \quad \bar{c}_{net}^{(1} \leftarrow \bar{c}_{net}$

$N_{TP}^{S(1} \leftarrow N_{TP}^S, \;\; N_{TP}^{P(1} \leftarrow N_{TP}^P, \;\; A_{TP}^{(1} \leftarrow A_{TP}$

$\Delta g_p \leftarrow \frac{g_p}{L_{max}}, \;\; g_p^{(0} \leftarrow \delta_p \cdot g_p \;\; \forall p \in O$

For $k := 1$ **to** L_{max} **do**

$g_p^{(k} \leftarrow g_p^{(k-1} + (1 - \delta_p) \cdot \Delta g_p \;\; \forall p \in O$

Solve $RTNPD^{(k}$

If $\delta_p = 0$ **then** $G_{TP}^{(k} \leftarrow$ **DelNoServLines**$(G_{TP}^{(k}, L^{E(0})$

If "A new line is constructed" **then**

Backup x_{ij}, y_i, x^l, x_{ij}^l, $\tilde{y}_i^l$, y_i^l

$$\bar{c}_{net}^{(k+1} \leftarrow \bar{c}_{net}^{(k} - \sum_{(i,j) \in A_{TP}^N} c_{ij}^{c(k} \cdot x_{ij} - \sum_{i \in N_{TP}^N} c_i^{c(k} \cdot y_i$$

$$L^{E(k+1} \leftarrow L^{E(k} \bigcup \left\{ l \in L^{N(k} \right\}$$

$$N_{TP}^{S(k+1} \leftarrow N_{TP}^{S(k} \bigcup \left\{ i \in N_{TP}^N \mid y_i = 1 \right\}$$

$$N_{TP}^{P(k+1} \leftarrow N_{TP}^{P(k} \bigcup \left\{ i \in N_{TP}^N \mid y_i^l = 1, \; \tilde{y}_i^l = 0, \; l \in L^{N(k} \right\}$$

$$A_{TP}^{(k+1} \leftarrow A_{TP}^{(k} \bigcup \left\{ (i,j) \in A_{TP}^N \mid x_{ij} = 1 \right\}$$

$$c_i^{c(k+1} \leftarrow \begin{cases} c_i^{c(k} & \text{if} \;\; y_i = 0 \\ 0 & \text{if} \;\; y_i = 1 \end{cases} \quad \forall i \in N_{TP}^N$$

$$c_{ij}^{c(k+1} \leftarrow \begin{cases} c_{ij}^{c(k} & \text{if} \;\; x_{ij} = 0 \\ 0 & \text{if} \;\; x_{ij} = 1 \end{cases} \quad \forall (i,j) \in A_{TP}^N$$

End If

$L^{N(k+1} \leftarrow \{|L^{E(k+1}| + 1\}, \;\; L^{(k+1} \leftarrow \{L^{N(k+1} \cup L^{E(k+1}\}$

If $\delta_p = 1$ **and** "No new line is constructed" **then** **Exit** for loop

end for

$G_{TP}^F \leftarrow$ **Get the Final Public Network** $\left(G_{TP}^{(k}, \text{Routing variables} \right)$

Return G_{TP}^F

end LSA

Table 5.1: Pseudo-code of the main procedure of the line splitting algorithm.

procedure DelNoServLines (**in** G_{TP}, $L^{E(0}$, **out** G_{TP})

$$\tilde{L}_0^E \leftarrow \left\{ l \in L^E \backslash L^{E(0} \text{ such that } f^l \leq \epsilon \right\}$$

$$\tilde{L}_1^E \leftarrow \left\{ l \in L^E \backslash L^{E(0} \text{ such that } f^l > \epsilon \right\}$$

$$\tilde{N}_{TP}^S \leftarrow \left\{ i \in \bigcup_{l \in \tilde{L}_0^E} N_{TP}^S(l) \text{ such that } i \notin \bigcup_{l \in \tilde{L}_1^E} N_{TP}^S(l) \right\}$$

$$\tilde{A}_{TP} \leftarrow \left\{ (i,j) \in \bigcup_{l \in \tilde{L}_0^E} A_{TP}(l) \text{ such that } (i,j) \notin \bigcup_{l \in \tilde{L}_1^E} A_{TP}(l) \right\}$$

$$c_i^c \leftarrow c_i^{c(0} \quad \forall i \in \tilde{N}_{TP}^S, \quad c_{ij}^c \leftarrow c_{ij}^{c(0} \quad \forall (i,j) \in \tilde{A}_{TP}$$

$$\bar{c}_{net} \leftarrow \bar{c}_{net} + \sum_{(i,j) \in \tilde{A}_{TP}} c_{ij}^c + \sum_{i \in \tilde{N}_{TP}^S} c_i^c$$

$$N_{TP}^S(l) \leftarrow \emptyset, \quad N_{TP}^P(l) \leftarrow \emptyset, \quad A_{TP}(l) \leftarrow \emptyset, \quad L^E \leftarrow L^E \backslash \{l\} \quad \forall l \in \tilde{L}_0^E$$

Return G_{TP}

end DelNoServLines

Table 5.2: Pseudo-code of the procedure DelNoServLines.

5.2 Performance Test

In order to show the efficiency of the LSA algorithm compared to the resolution of the $RTNPD$ model without line splitting, two test networks have been used. The first one is a complete 6-node railway network, shown in Figure 8.1, whereas the other one is a 9-node and 26-link railway network, as depicted in Figure 8.2. We refer the reader to section 8.1.1 for further details regarding the input parameters associated with both networks.

Tables 5.3 - 5.4 show the main results from applying the three routing models (see sections 2.3.3.1 - 2.3.3.3 for further details). The model has been formulated in AMPL, using directly the CPLEX v.12.4.0 solver. So the SBD decomposition has not been used. Regarding the LSA algorithm, it has been coded in MATLAB R2012b, and successive calls on AMPL were made in order to solve each $RTNPD$ iteration. The used machine was an R5500 work station with Intel(R) Xeon(R) processor, CPU E5645 2.40 GHz and 48 Gbits of RAM.

In both tables, the column $Exp.$ stands for the experiment range. The next column, $|L^N|$, is the maximum number of lines which can be constructed, whereas column $R.M$ shows which routing model has been used by denoting one of the following tags: $M1$, $M2$ and $M3$. The $M1$ and $M2$ tags stand for the models with non-fixed corridors. The difference lies in the way they treat the sublines. The former do not include them in the formulation from the very beginning. Instead, they are added dynamically as needed. The latter treat them in a static fashion. To see the details of their formulations, see sections 2.3.3.1 and 2.3.3.2, respectively. The $M3$ tag refers to the model with fixed corridors (see section 2.3.3.3 for further details). The next column, $Method$, shows which technique has been used by denoting one of the following tags: $Heu1$, $Heu2$ and $Exact$. The two first tags stand for the incremental and fixed demand variants of the LSA algorithm, respectively; whereas the latter refers to the direct resolution of the model. Columns $F.Obj$ and $F.Users$ show the total objective function and the user costs (2.2) weighted by β, respectively. Column

Exp.	$\|L^N\|$	R.M.	Method	F.Obj	F.Users	T_{CPU}	Gap	N_{SEC}
1	2	M1	Exact	184034	183225	1	-	0
			Heu1	193177	192354	2	5%	0
			Heu2	193177	192354	2	5%	0
		M2	Exact	184034	183225	2	-	-
			Heu1	193177	192354	1	5%	-
			Heu2	193177	192354	1	5%	-
		M3	Exact	184034	183225	5	-	-
			Heu1	193177	192354	2	5%	-
			Heu2	193177	192354	2	5%	-
2	3	M1	Exact	157036	155962	42	-	0
			Heu1	171305	170340	2	9%	0
			Heu2	169361	168370	2	8%	1
		M2	Exact	157036	155962	48	-	-
			Heu1	171305	170340	2	9%	-
			Heu2	169361	168370	2	8%	-
		M3	Exact	157036	155962	129	-	-
			Heu1	171305	170340	3	9%	-
			Heu2	172368	171358	3	10%	-
3	4	M1	Exact	147612	146471	1586	-	0
			Heu1	158104	156999	3	7%	1
			Heu2	153275	152135	3	4%	0
		M2	Exact	147612	146471	2543	-	-
			Heu1	158104	156999	2	7%	-
			Heu2	153275	152135	2	4%	-
		M3	Exact	147612	146471	3847	-	-
			Heu1	158337	157318	5	7%	-
			Heu2	155648	154516	4	5%	-

Table 5.3: Results for the experiments performed in the 6-node network using all the routing models.

$\boldsymbol{T_{CPU}}$ reports the total number of elapsed seconds for the method used in the experiment. The next column, Gap, reports the difference between the heuristic variant and the exact objective functions, respectively. Finally, column $\boldsymbol{N_{SEC}}$ shows the number of subline elimination constraints (2.13) added to the model $M1$ in order to obtain a valid line trace.

Results show that in both networks, the exact method can reach the optimum within a short time with $|L| = 2$, for $|L| \geq 2$. The LSA allows us to obtain quickly feasible solutions within acceptable gaps. We do not report experiments with values of $|L| > 4$, since the direct resolution did not reach optimality within the one-day time limit; thus, comparing the exact non-optimal solutions with the LSA variants would be incorrect. On the the hand, determining which variant performs better is unclear, since their gaps are very close and the averages are quite similar. Moreover, these gaps seem to be 0 for the 9-node network, whereas for the 6-node network they are worse. We hope that the gaps will be very small for bigger networks, such as those with 9 nodes. Regarding the SECs, it seems that they tend to emerge more and more as the size of the network grows. Although, they seldom emerge when applying the LSA variants. Finally, the use of fixed corridors also does not seem to affect the quality of the LSA solution.

Exp.	$\|L^N\|$	R.M.	Method	F.Obj	F.Users	T_{CPU}	Gap	N_{SEC}
1	2	M1	Exact	782332	781481	8	-	3
			Heu1	784616	783653	1	0.3%	0
			Heu2	784598	783635	4	0.3%	0
		M2	Exact	782332	781481	3	-	-
			Heu1	784616	783653	1	0.3%	-
			Heu2	784598	783635	1	0.3%	-
		M3	Exact	782332	781481	5	-	-
			Heu1	784616	783653	2	0.3%	-
			Heu2	784598	783635	2	0.3%	-
2	3	M1	Exact	758634	757608	7386	-	19
			Heu1	760084	759061	2	0.2%	0
			Heu2	760067	759043	4	0.2%	3
		M2	Exact	758634	757608	1125	-	-
			Heu1	760084	759061	2	0.2%	-
			Heu2	760067	759043	2	0.2%	-
		M3	Exact	758634	757608	488	-	-
			Heu1	760084	759061	3	0.2%	-
			Heu2	760067	759043	3	0.2%	-
3	4	M1	Exact	-	-	> 86400	-	≥ 11
			Heu1	745527	744396	3	0.5%	0
			Heu2	742181	741038	5	0.0008%	3
		M2	Exact	742175	741038	5001	-	-
			Heu1	745527	744396	3	0.5%	-
			Heu2	742181	741038	3	0.0008%	-
		M3	Exact	742175	741038	3440	-	-
			Heu1	745527	744396	4	0.5%	-
			Heu2	742181	741038	3	0.0008%	-

Table 5.4: Results for the experiments performed in the 9-node network using all the routing models.

Chapter 6

Application of the Benders decomposition

This chapter presents the Specialized Benders decomposition (SBD) applied to the inelastic version of the $RTNPD$ model. The SBD includes the enhancements of J. Benders [14], Wong & Magnanti [125], N. Papadakos [147] and McDaniel & Devine [136]. These are explained in detail in sections 3.5.4 and 3.5.5 of chapter 3. Moreover, some ad hoc techniques are also developed. The model has two decision levels: first, the system operator builds the network layout and determines the line frequencies; second, passengers choose their itineraries. To apply the SBD, we split the mathematical programming problem associated with the $RTNPD$ model into two problems: a master problem (MP), which defines a feasible network and assigns frequencies to the constructed lines; and a subproblem (SP), which assigns the demand to this network. The SBD iterates between the MP and SP problems to find the optimal solution. At each k-iteration, the dual variables of the SP define the optimality of the Benders cuts (OBC), which are added as new constraints to the MP. The process continues until it converges under convex assumptions verified by the model. The structure of the chapter is as follows. Firstly, the SP and MP problems are presented as well as the classical Benders scheme (CBS) (J. Benders [14]), in which they interact. Secondly, the enhancements [125], [147], [136] are introduced, along with some ad hoc techniques applied to the CBS. Finally, we report some computational results which justify including these enhancements.

6.1 Master Problem

The decisional variables related to the network design and determination of line frequencies are difficult variables in the problem. Thus, once they are fixed, the problem is easy to solve. In this way, the MP arises as:

$$z_{MP} = \min \quad (1-\alpha)\, z_{op} + \omega \tag{6.1}$$

subject to

Network design constraints (2.4)-(2.7)

Line frequency setting constraints (2.33)-(2.36) and (2.42)-(2.45)

Breaking symmetry constraints (2.59)

Relaxing integrality constraints (2.60)-(2.63) if routing model M1 is chosen or, (2.60)-(2.63) if routing model M2 is chosen or, (2.60)-(2.61) otherwise

Routing constraints (2.8)-(2.12) if routing model M1 is chosen or, (2.8)-(2.9), (2.19)-(2.22), (2.24)-(2.26) if routing model M2 is chosen or, (2.27)-(2.29) otherwise

and the following optimality Benders cut (OBC):

$$\omega \geq \sum_{p\in O}\sum_{i\in N}\left(\sum_{l\in L_i^N}\left[(\pi_{i,p}^{l,k}-\gamma_{i,p}^{l,k})\ \tilde{y}_i^l-\pi_{i,p}^{l,k}\ y_i^l\right]+t_i^p\ \beta_{i,p}^k\right)-q\sum_{(i,j)\in A_{TP}}\sum_{l\in L_{ij}}\tilde{f}_{ij}^l\ \tau_{ij}^{l,k},\quad \forall k\in\Omega \tag{6.2}$$

where k stands for the Benders iteration; ω is the Benders variable associated with the OBC cuts and dual variables $\beta_{i,p}^k$, $\gamma_{i,p}^{l,k}$, $\pi_{i,p}^{l,k}$; and $\tau_{ij}^{l,k}$ are related to the passenger flow balance constraints (2.50) and (2.54)-(2.56), respectively.

6.2 Benders Subproblem

The Benders subproblem (SP) assumes that the multicommodity network topology and the line frequencies are known. Both are found by previously solving the master problem (MP). The SP is always feasible for every MP solution since the G_{COM} network connects every pair of nodes $i,j\in N_{TP}$. Consequently, no feasibility Benders cut (FBC) will be needed in the MP. The primal form of the SP is then defined by the non-negative passenger flows $v_{ij}^{p,l}$, $\tilde{v}_{ij}^{p,l}$ and u_{ij}^p throughout the global network G as follows:

$$z_{SP}=\min\quad \alpha\cdot z_{pax} \tag{6.3}$$

$s.t.$

$$\sum_{l\in L_i}\left(\sum_{j\in l_{a(i)}}v_{ij}^{p,l}-\sum_{j\in l_{y(i)}}v_{ji}^{p,l}\right)+\sum_{j\in A_{COM}(i)}u_{ij}^p-\sum_{j\in A_{COM}(i)}u_{ji}^p=t_i^p\quad\left(\beta_{i,p}^k\right),\ \forall i\in N,\ p\in O \tag{6.4}$$

$$v_{inv^+(i)}^{p,l}=v_{y(i)}^{p,l}+\sum_{j\in A^l_{x^+(i)}}\left(v_{ij}^{p,l}+\phi_{ij}^{p,l}\right)\quad\left(\rho_{i,p}^{l+,k}\right),\ \forall l\in L,\ i\in N_{TP}^{S+}(l),\ p\in O \tag{6.5}$$

$$v_{inv^-(i)}^{p,l}=v_{a(i)}^{p,l}+\sum_{j\in A^l_{x^-(i)}}\left(v_{ji}^{p,l}+\phi_{ji}^{p,l}\right)\quad\left(\rho_{i,p}^{l-,k}\right),\ \forall l\in L,\ i\in N_{TP}^{S-}(l),\ p\in O \tag{6.6}$$

$$v_{inv^+(i)}^{p,l}=v_{inv^-(i)}^{p,l}\quad\left(\chi_{j,p}^{l,k}\right),\ \forall\, l\in L^E,\ i\in N_{TP}^P(l),\ p\in O \tag{6.7}$$

$$\sum_{(r,s)\in A_{xya}^N(i)}v_{rs}^{p,l}\leq\tilde{y}_i^l\quad\left(\gamma_{i,p}^{l,k}\right),\ \forall l\in L^N,\ i\in N_{TP}^N,\ p\in O \tag{6.8}$$

$$\sum_{(r,s)\in A_x^N(i)}\tilde{v}_{rs}^{p,l}\leq(y_i^l-\tilde{y}_i^l)\quad\left(\pi_{s,p}^{l,k}\right),\ \forall l\in L^N,\ i\in N_{TP}^N,\ p\in O \tag{6.9}$$

$$\sum_{p\in O}g_p\cdot v_{inv(i,j)}^{p,l}\leq q\cdot\tilde{f}_{ij}^l\quad\left(\tau_{ij}^{l,k}\right),\ \forall(i,j)\in A_{TP},\ l\in L_{ij} \tag{6.10}$$

where the SP objective function (6.3) consists of the passenger travel times contained in z_{pax} (defined by equation (2.3)) and equations (6.4) - (6.10) correspond to the passenger flow balance constraints (2.50) - (2.56). Moreover, to the right of them, we show their corresponding dual variables.

Applying dual theory, we obtain the dual form of the SP as follows:

$$\Omega^k_{SP} = \max_{\beta,\rho,\chi,\gamma,\pi,\tau} \sum_{p\in O}\sum_{i\in N}\left(\sum_{l\in L^N_i}\left[(\pi^{l,k}_{i,p}-\gamma^{l,k}_{i,p})\ \tilde{y}^l_i - \pi^{l,k}_{i,p}\ y^l_i\right] + t^p_i\ \beta^k_{i,p}\right) - q\sum_{(i,j)\in A_{TP}}\sum_{l\in L_{ij}}\tilde{f}^l_{ij}\ \tau^{l,k}_{ij} \quad (6.11)$$

$s.t.$

$$\beta^k_{i,p} - \beta^k_{j,p} \le \alpha\ \theta\ g_p\ t^{COM}_{ij}, \quad \forall (i,j)\in A_{COM},\ p\in O \quad (6.12)$$

$$\rho^{l+,k}_{i,p} + \rho^{l-,k}_{j,p} + g_p\ \tau^{l,k}_{ij} \le \alpha\ \theta\ g_p\ t^{TP}_{ij}, \quad \forall l\in L^N,\ (i,j)\in A^N_{TP},\ p\in O \quad (6.13)$$

$$\rho^{l+,k}_{i,p} + \rho^{l-,k}_{j,p} + g_p\ \tau^{l,k}_{ij} \le \alpha\ \theta\ g_p\ t^{TP}_{ij}, \quad \forall l\in L^E,\ (i,j)\in A^l_{TP},\ i,j\in N^S_{TP}(l),\ p\in O \quad (6.14)$$

$$\rho^{l-,k}_{i,p} + \chi^{l,k}_{j,p} + g_p\ \tau^{l,k}_{ij} \le \alpha\ \theta\ g_p\ t^{TP}_{ij}, \quad \forall l\in L^E,\ (i,j)\in A^l_{TP},\ i\in N^{S-}_{TP}(l)\ j\in N^P_{TP}(l),\ p\in O \quad (6.15)$$

$$\rho^{l+,k}_{i,p} - \chi^{l,k}_{j,p} + g_p\ \tau^{l,k}_{ij} \le \alpha\ \theta\ g_p\ t^{TP}_{ij}, \quad \forall l\in L^E,\ (i,j)\in A^l_{TP},\ i\in N^P_{TP}(l)\ j\in N^{S+}_{TP}(l),\ p\in O \quad (6.16)$$

$$\chi^{l,k}_{i,p} - \chi^{l,k}_{j,p} + g_p\ \tau^{l,k}_{ij} \le \alpha\ \theta\ g_p\ t^{TP}_{ij}, \quad \forall l\in L^E,\ (i,j)\in A^l_{TP},\ i,j\in N^P_{TP}(l),\ p\in O \quad (6.17)$$

$$\gamma^{l,k}_{i,p} - \beta^k_{i,p} - \rho^{l+,k}_{j,p} \le \alpha\ \theta\ g_p\ t_y, \quad \forall l\in L,\ i\in N_{TP}(l),\ \exists (j,i)\in A^l_y,\ p\in O \quad (6.18)$$

$$\beta^k_{i,p} - \rho^{l-,k}_{j,p} + \gamma^{l,k}_{i,p} \le \alpha\ \theta\ g_p\ t_a, \quad \forall l\in L,\ i\in N_{TP}(l),\ \exists (i,j)\in A^l_a,\ p\in O \quad (6.19)$$

$$\gamma^{l,k}_{s,p} - \rho^{l+,k}_{i,p} - \rho^{l-,k}_{j,p} \le \alpha\ \theta\ g_p\ t_x, \quad \forall (l,i,j,s,p)\in \Pi^1 \quad (6.20)$$

$$-\rho^{l+,k}_{i,p} - \rho^{l-,k}_{j,p} \le \alpha\ \theta\ g_p\ t_x, \quad \forall (l,i,j,s,p)\in \Pi^2 \quad (6.21)$$

$$\pi^{l,k}_{s,p} - \rho^{l+,k}_{i,p} - \rho^{l-,k}_{j,p} \le 0, \quad \forall (l,i,j,s,p)\in \Pi^3 \quad (6.22)$$

where dual variables $\beta^k_{i,p}$, $\rho^{l+,k}_{i,p}$, $\rho^{l-,k}_{i,p}$, $\chi^{l,k}_{j,p}$, $\gamma^{l,k}_{i,p}$, $\pi^{l,k}_{s,p}$ and $\tau^{l,k}_{ij}$ are associated with the aforementioned passenger flow balance constraints (6.4) - (6.10), respectively, and constraint sets Π^1, Π^2 and Π^3 hold the following elements:

$$\Pi^1 = \Big\{(l,i,j,s,p)\ \mid\ l\in L^N,\ i\in N^{S+}_{TP},\ j\in N^{S-}_{TP},\ s\in N^N_{TP},\ \exists (i,s)\in A^N_y, \ldots \quad (6.23)$$

$$\ldots \exists (s,j)\in A^N_a,\ p\in O\Big\} \quad (6.24)$$

$$\Pi^2 = \Big\{(l,i,j,s,p)\ \mid\ l\in L^E,\ i\in N^{S+}_{TP}(l),\ j\in N^{S-}_{TP}(l),\ \exists (i,s)\in A^l_y,\ \exists (s,j)\in A^l_a,\ p\in O\Big\} \quad (6.25)$$

$$\Pi^3 = \Big\{(l,i,j,s,p)\ \mid\ l\in L^E,\ i\in N^{S+}_{TP}(l),\ j\in N^{S-}_{TP}(l),\ \exists (i,s)\in A^N_y,\ \exists (s,j)\in A^N_a,\ p\in O\Big\} \quad (6.26)$$

6.3 Classical Benders scheme

The classical Benders decomposition scheme (CBS) (J. Benders [14]) is the widest used scheme for applying Benders' decomposition. At each k-iteration, the MP problem is solved first, and then the SP problem. Initially, the MP has no Benders cuts (BC), and each time, the SP is solved, a new BC is added to the MP. Since our SP is always feasible no matter what the MP solution is, only the OBC type of BC cuts are included. It can be checked easily by verifying that the A_{COM} set holds all the possible links from a pair of nodes $(i,j)\in A$, so that every o-d pair $w\in W$ can find a path even if no lines are constructed. Solving both problems verifies whether or not the Benders gap is less or equal to an allowable error ϵ. If so, we finish the algorithm and report the current MP and SP solutions as optimal. Otherwise, it jumps to the next iteration until the desired gap is reached.

The CBS scheme converges to optimality in a finite number of iterations, provided that the original problem has an optimal solution and that the convergence gap is small enough. Table 6.1 shows that this scheme for our model proposes where k stands for the iteration's counter; whileβ^k, π^k, γ^k and τ^k are the

dual variable vectors associated with the primal SP constraints (2.50) and (2.54)-(2.56), respectively, at iteration k. They are used to construct a new OBC cut given by equation (6.2).

1. **Set** $\Omega \leftarrow \emptyset$, $UB \leftarrow \infty$, iteration $k \leftarrow 0$; and allowable error ϵ
2. **Solve** MP^k.
3. **Update** $LB \leftarrow (1-\alpha) \cdot z_{op}^k + \omega^k$.
4. **Solve** DSP^k **with** $(y^k, \tilde{y}^k, \tilde{f}^k)$.
5. **Update** $UB \leftarrow \min\left\{UB, (1-\alpha) \cdot z_{op}^k + z_{SP}^k\right\}$.
6. **if** $\dfrac{UB - LB}{LB} < \epsilon$ **then** STOP, **otherwise** continue.
7. **Generate** OBC^k **with** $(\beta^k, \pi^k, \gamma^k, \tau^k)$.
8. $k \leftarrow k+1$ and go to Step 2.

Table 6.1: Pseudo-code of the Classic Benders Scheme.

The CBS scheme is not suitable to our model, since the duality of the SP suffers from degeneration. Thus, it entails a high number of iterations to reach optimality or pseudo-optimality. In the next two sections, we will discuss this topic in more detail and give alternative schemes for speeding up its resolution.

6.4 Accelerating Benders' convergence

In this section, two alternative Benders schemes will be introduced. The former is based on the Magnanti and Wong scheme (MWS) in [125], whereas the latter is related to a practical enhancement of this scheme [147]. Both are quite similar, but the difference lies on the order in which the MP, SP, and the Magnanti & Wong problem (MWP) are solved, as well as the mathematical structure of the MWP being considered.

6.4.1 The Magnanti & Wong scheme

Magnanti & Wong [125] studied the problem of degeneration in the dual form of the SP and proposed a modification of the aforementioned CBD scheme to overcome that problem. The idea is based on Pareto-optimality applied to the OBC cuts. A pareto-optimal cut renders the tightest cut to the MP (see proof of section 2 in page 470 of [125]), and thus it reduces the number of Benders iterations.

To explain what a pareto-optimal cut is, some basic notation and definitions need to be introduced. Let's say that cut (6.27) dominates or is stronger than cut (6.28) if (6.29) is met for all $(y, \tilde{y}, \tilde{f}) \in Y$ and with a strict inequality for at least one point $(y, \tilde{y}, \tilde{f}) \in Y$. Set Y is made up of points satisfying the master problem constraints, except for the OBC cuts. Then, a cut is called pareto-optimal if no cut dominates it.

$$\omega \geq \sum_{p \in O} \sum_{i \in N} \left(\sum_{l \in L_i^N} \left[(\pi_{i,p}^{l,1} - \gamma_{i,p}^{l,1})\ \tilde{y}_i^l - \pi_{i,p}^{l,1}\ y_i^l \right] + t_i^p\ \beta_{i,p}^1 \right) - q \sum_{(i,j) \in A_{TP}} \sum_{l \in L_{ij}} \tilde{f}_{ij}^l\ \tau_{ij}^{l,1} \tag{6.27}$$

$$\omega \geq \sum_{p \in O} \sum_{i \in N} \left(\sum_{l \in L_i^N} \left[(\pi_{i,p}^{l,2} - \gamma_{i,p}^{l,2})\ \tilde{y}_i^l - \pi_{i,p}^{l,2}\ y_i^l \right] + t_i^p\ \beta_{i,p}^2 \right) - q \sum_{(i,j) \in A_{TP}} \sum_{l \in L_{ij}} \tilde{f}_{ij}^l\ \tau_{ij}^{l,2} \tag{6.28}$$

$$\sum_{p\in O}\sum_{i\in N}\left(\sum_{l\in L_i^N}\left[(\pi_{i,p}^{l,1}-\gamma_{i,p}^{l,1})\ \tilde{y}_i^l-\pi_{i,p}^{l,1}\ y_i^l\right]+t_i^p\ \beta_{i,p}^1\right)-q\sum_{(i,j)\in A_{TP}}\sum_{l\in L_{ij}}\tilde{f}_{ij}^l\ \tau_{ij}^{l,1}\geq \tag{6.29}$$

$$\sum_{p\in O}\sum_{i\in N}\left(\sum_{l\in L_i^N}\left[(\pi_{i,p}^{l,2}-\gamma_{i,p}^{l,2})\ \tilde{y}_i^l-\pi_{i,p}^{l,2}\ y_i^l\right]+t_i^p\ \beta_{i,p}^2\right)-q\sum_{(i,j)\in A_{TP}}\sum_{l\in L_{ij}}\tilde{f}_{ij}^l\ \tau_{ij}^{l,2},\quad \forall\ y_i^l,\tilde{y}_i^l,\tilde{f}_{ij}^l\in Y$$

To generate pareto-optimality cuts, Magnanti & Wong [125] define the notion of core points. A point $(y^0,\tilde{y}^0,\tilde{f}^0)\in Y$ is a core point if $(y^0,\tilde{y}^0,\tilde{f}^0)\in ri(Y^c)$, where $ri(Y^c)$ and Y^c are, respectively, the relative interior and the convex hull of set Y. This core point is then used in association with the master decisional variables and the optimal objective function value of the SP to solve the Magnanti & Wong Problem (MWP), which generates the pareto-optimal cut. The MWP problem structure for our model is as follows:

$$\max_{\beta,\rho,\chi,\gamma,\pi,\tau}\ \sum_{p\in O}\sum_{i\in N}\left(\sum_{l\in L_i^N}\left[(\pi_{i,p}^{l,k}-\gamma_{i,p}^{l,k})\ \tilde{y}_i^{l,0}-\pi_{i,p}^{l,k}\ y_i^{l,0}\right]+t_i^p\ \beta_{i,p}^k\right)-q\sum_{(i,j)\in A_{TP}}\sum_{l\in L_{ij}}\tilde{f}_{ij}^{l,0}\ \tau_{ij}^{l,k} \tag{6.30}$$

$s.t.$

$$\sum_{p\in O}\sum_{i\in N}\left(\sum_{l\in L_i^N}\left[(\pi_{i,p}^{l,k}-\gamma_{i,p}^{l,k})\ \tilde{y}_i^l-\pi_{i,p}^{l,k}\ y_i^l\right]+t_i^p\ \beta_{i,p}^k\right)-q\sum_{(i,j)\in A_{TP}}\sum_{l\in L_{ij}}\tilde{f}_{ij}^l\ \tau_{ij}^{l,k}=\Omega_{SP}^k \tag{6.31}$$

and subproblem dual constraints $(6.12)-(6.22)$

where parameters $y_i^{l,0}$, $\tilde{y}_i^{l,0}$ and $\tilde{f}_{i,j}^{l,0}$ conform to the core point. Observe that the mathematical structure of the MWP is quite similar to the dual form of the subproblem, except that the additional constraint (6.31) is included to ensure that an extreme point from the set of optimal solutions to the DSP^k will be chosen. Moreover, the objective function (6.30) compares all possible cuts at a point $(y^0,\tilde{y}^0,\tilde{f}^0)$ in the feasible region of the master problem.

The resulting iterative scheme among the MP, SP and MWP problems is shown in table 6.2. Unlike the CBS, this scheme includes steps 7 and 8, which are needed to determine the dual variables that will form part of the next pareto-optimal Benders cut. The supra-index 0 is used to distinguish the core-point variables from those pertaining to resolution of the current master problem.

Despite the good convergence properties of this schema, two important pitfalls arise: finding a core point and resolution of the MWP problem. To overcome the first issue, we have to develop a problem-specific method to determine a core point, which is in general non-trivial. Regarding the second issue, the amount of time needed to solve the MWP to optimality can considerably increase the time per iteration, such that the overall time to reach optimality could not be decreased significantly when compared to the CBS scheme. In the next section, we will present a variation of the Magnanti & Wong scheme, which overcomes the second issue.

6.4.2 Papadakos' scheme

Papadakos [147] revised the method of finding core points and discovered an easier way to determine them. His main idea is inspired by the Magnanti & Wong proof of a pareto-optimality cut [125] and demonstrates

1. **Set** $\Omega \leftarrow \emptyset$, $UB \leftarrow \infty$, iteration $k \leftarrow 0$; and allowable error ϵ
2. **Solve** MP^k.
3. **Update** $LB \leftarrow (1-\alpha) \cdot z_{op}^k + \omega^k$.
4. **Solve** DSP^k **with** $(y^k, \tilde{y}^k, \tilde{f}^k)$.
5. **Update** $UB \leftarrow \min\left\{UB, (1-\alpha) \cdot z_{op}^k + z_{SP}^k\right\}$.
6. **if** $\dfrac{UB - LB}{LB} < \epsilon$ **then** STOP, **otherwise** continue.
7. **Find a new Magnanti Wong Point** $(y^0, \tilde{y}^0, \tilde{f}^0)$.
8. **Solve** MWP^k **with** $(y^0, \tilde{y}^0, \tilde{f}^0)$.
9. **Generate** OBC^k **with** $(\beta^k, \pi^k, \gamma^k, \tau^k)$.
10. $k \leftarrow k+1$ and go to Step 2.

Table 6.2: Pseudo-code of the Magnanti & Wong scheme.

that the MWP problem can be reduced to the same mathematical structure of the SP, but by replacing the MP variables with those coming from the core point. He called this resulting problem the independent Magnanti & Wong problem ($IMWP$), since it does not depend on the SP solution. The author also showed that it is more convenient to solve this $IMWP$ problem before the SP, since it gives a valid Benders cut to the MP from the first iteration and, thus, it reduces more and more the number of iterations to reach convergence.

As a result, he developed a new Benders scheme which we will call from now on the Papadakos scheme (PS). Table 6.3 shows an adaptation of the PS for our proposes, where we have basically dropped consideration of feasibility Benders cuts. Observe that this scheme also includes a fast way to find core points, provided that we have an initial one. The author proves that by means of a convex combination of the core point and the decisional variables of the MP at the current iteration (see step 10), one can find a new valid core point for the next iteration. The parameter λ is a weight whose value must be within the interval $(0, 1)$, where 0 and 1 values are not within the parameter dominion. Notice that, for $\lambda = 1$, we obtain the same core point, whereas for $\lambda = 0$, we obtain an invalid core point, since it is not in the $ri(Y^c)$. Thus, if the initial core point is far away from the facets of Y^c, a good choice is to set $\lambda = 0.5$. In the next section, we will present a straightforward method to find initial core points.

6.4.3 Finding an initial core point

The contributions of Magnanti & Wong and Papadakos ([125], [147]) to the Benders decomposition depend heavily on determining an appropriate initial core point. As stated by the authors, it is not always possible to find an exact core point. However, in those cases approximations can be used. Fortunately, in our model it is possible to find an initial exact core point by means of the procedure **FindICP,** shown in table 6.4. It basically seeks two opposed feasible solutions $(\tilde{y}, \tilde{f}) \in Y^c$ and then applies a convex combination, like in the core point update. To find each point, we call the subroutine **BuiltLines,** which is driven by the RoutingModel, ConstCriterion and FreqCriterion input parameters. The former indicates the type of routing model used, whereas the rest contain the selected criteria for the routing of lines under construction as well as the frequency assignment of the whole set of lines.

The subroutine **BuiltLines** is then split into two phases which are carried out in a sequential manner. The

1. **Set** $\Omega \leftarrow \emptyset$; $UB \leftarrow \infty$; iteration $k \leftarrow 0$; and allowable error ϵ
2. **Find an Initial Core Point** $(y^0, \tilde{y}^0, \tilde{f}^0)$.
3. **Solve** $IMWP^k$ **with** $(y^0, \tilde{y}^0, \tilde{f}^0)$.
4. **Generate** OBC^k **with** $(\beta^k, \pi^k, \gamma^k, \tau^k)$.
5. **Solve** MP^k.
6. **Update** $LB \leftarrow (1-\alpha) \cdot z_{op}^k + \omega^k$.
7. **Solve** DSP^k **with** $(y^k, \tilde{y}^k, \tilde{f}^k)$.
8. **Update** $UB \leftarrow \min\{UB, (1-\alpha) \cdot z_{op}^k + z_{SP}^k\}$.
9. **if** $\frac{UB - LB}{LB} < \epsilon$ **then** STOP, **otherwise** continue.
10. **Update** $(y^0, \tilde{y}^0, \tilde{f}^0) \leftarrow \lambda \cdot (y^0, \tilde{y}^0, \tilde{f}^0) + (1-\lambda) \cdot (y^k, \tilde{y}^k, \tilde{f}^k)$
11. $k \leftarrow k+1$ and go to Step 3.

Table 6.3: Pseudo-code of the Papadakos' Scheme.

first phase corresponds to either line routing or selecting the line corridors, depending on the RoutingModel parameter; whereas the second phase is associated with the line frequency setting. The details of these procedures as well as the related subroutines can be found in appendix section 9.3.

6.5 Decreasing the number of integral Master Problem resolutions

In this section, we present other extensions of the CBS scheme, which allow us to decrease consumption time in solving the integral MP problem. The first one is inspired by the work of McDaniel & Devine [136], whereas the rest are ad hoc techniques which were presented earlier in [124]. These additional extensions are absolutely compatible with the MWS and PS schemes presented so far. Mixing all these enhancements have provided the specialized Benders scheme (SBD), which have been applied successfully to versions of the model, both with and without elastic demand.

6.5.1 The McDaniel and Devine scheme

McDaniel & Devine [136] developed a two-phase Benders scheme. In the first phase, the integer variables corresponding to the MP problem are relaxed, and thus the resulting problem is much easier to solve. The BC cuts coming from the resolution of the SP in this phase are called the relaxed Benders cuts (RBC), which are also valid cuts for the integral MP. Once the relaxed Benders gap is reached, the first phase finishes and the second phase starts. Initially, integrality is imposed on the MP integer variables, and at every iteration of this phase the MP is solved using all the RBC computed in the previous phase in addition to those obtained in this one. As a result, the authors [136] demonstrated that the number of integral MP resolutions decreased considerably by solving several known difficult problems in the state of the art.

Table 6.5 shows the specialized integration of the McDaniel & Devine scheme (DDS), with the PS scheme added for our proposes. The main differences regarding the PS scheme lie in the operations carried out in steps 6 and 11. In the former, the integrality of the MP integer variables are activated or relaxed according to the parameter $phase$, which denotes the phase in which the algorithm is working. The latter checks if the convergence criterion is met according to the parameter $phase$ as well. This parameter is up-

procedure FindICP $\big($**in** RoutingModel, $\lambda,\ \bar{A},$ **out** $y^0,\ \tilde{y}^0,\ f^0\big)$

$f^{l,0} \leftarrow 0, \quad \forall l \in L^E$

$f_{ij}^{l,0} \leftarrow 0, \quad \forall (i,j) \in A_{TP}^N,\ l \in L^N$

$y_i^{l,0} \leftarrow 0, \quad \forall (i,j) \in A_{TP}^N,\ l \in L^N$

$\tilde{y}_i^{l,0} \leftarrow 0, \quad \forall (i,j) \in A_{TP}^N,\ l \in L^N$

ConstCriterion ← Minimize construction & maintenance costs

FreqCriterion ← Assign maximum frequency

$(N,\ A,\ f)$ ← Get a line configuration following the set criterion by calling

BuiltLines $\big($RoutingModel, ConstCriterion, FreqCriterion, $\bar{A},\ N,\ A,\ f\big)$

$f^{l,0} \leftarrow \lambda \cdot f^l, \quad \forall l \in L^E$

$f_{ij}^{l,0} \leftarrow \lambda \cdot f^l, \quad \forall l \in L^N,\ (i,j) \in A^l$

$y_i^{l,0} \leftarrow \lambda, \quad \forall l \in L^N,\ i \in N^l$

$\tilde{y}_i^{l,0} \leftarrow \lambda, \quad \forall l \in L^N,\ i \in N^l$

ConstCriterion ← Maximize construction & maintenance costs

FreqCriterion ← Assign minimum frequency

$(N,\ A,\ f)$ ← Get a line configuration following the set criterion by calling

BuiltLines $\big($RoutingModel, ConstCriterion, FreqCriterion, $\bar{A},\ N,\ A,\ f\big)$

$f^{l,0} \leftarrow f^{l,0} + (1-\lambda) \cdot f^l, \quad \forall l \in L^E$

$f_{ij}^{l,0} \leftarrow f_{ij}^{l,0} + (1-\lambda) \cdot f^l, \quad \forall l \in L^N,\ (i,j) \in A^l$

$y_i^{l,0} \leftarrow y_i^{l,0} + (1-\lambda), \quad \forall l \in L^N,\ i \in N^l$

$\tilde{y}_i^{l,0} \leftarrow \tilde{y}_i^{l,0} + (1-\lambda), \quad \forall l \in L^N,\ i \in N^l$

return $y^0,\ \tilde{y}^0,\ f^0$

end FindICP

Table 6.4: Pseudo-code of the procedure FindICP.

dated after having met the convergence criterion of the first phase. Notice also that the computation of the initial core point is performed only once, at the first iteration of phase 1.

6.5.2 Ad hoc techniques

The aforementioned scheme renders the best convergence performance and the least consuming time, as far as we know. However, some additional problem-dependent enhancements can be made. If we work with

1. **Set** $\Omega \leftarrow \emptyset$; $UB \leftarrow \infty$; iteration $k \leftarrow 0$; allowable error ϵ; and phase $\leftarrow 1$

2. **Find an Initial Core Point** $(y^0, \tilde{y}^0, \tilde{f}^0)$.

3. **Solve** $IMWP^k$ **with** $(y^0, \tilde{y}^0, \tilde{f}^0)$.

4. **Generate** Ω^k **with** $(\beta^k, \pi^k, \gamma^k, \tau^k)$.

5. **If** phase = 1 **then** continue, **otherwise** goto Step 6b.

6a. **Remove** integrality for variables $\tilde{y}_i^l$, b^l **and**

x_{ij}^l **if** routing models M1 or M2 are used

δ_c^l **otherwise**

Go to step 7

6b. **Activate** integrality for variables $\tilde{y}_i^l$, b^l **and**

x_{ij}^l **if** routing models M1 or M2 are used

δ_c^l **otherwise**

7. **Solve** MP^k

8. $LB \leftarrow (1-\alpha) \cdot z_{op}^k + \omega^k$.

9. **Solve** DSP^k **with** $(y^k, \tilde{y}^k, \tilde{f}^k)$.

10. $UB \leftarrow \min\left\{UB, (1-\alpha) \cdot z_{op}^k + z_{SP}^k\right\}$.

11. **if** phase = 2 **and** $\dfrac{UB - LB}{LB} < \epsilon$ **then** STOP

if phase = 1 **and** $\dfrac{UB - LB}{LB} < \epsilon$ **then** phase $\leftarrow 2$.

12. **Update** $(y^0, \tilde{y}^0, \tilde{f}^0) \leftarrow \lambda \cdot (y^0, \tilde{y}^0, \tilde{f}^0) + (1-\lambda) \cdot (y^k, \tilde{y}^k, \tilde{f}^k)$

13. $k \leftarrow k+1$ and goto Step 3.

Table 6.5: Pseudo-code denoting integration of the McDaniel & Devine and Papadakos schemes.

the routing model M3 (see section 2.3.3.3 for further details), having solved the relaxed version of the MP allows us to easily obtain a rounded feasible solution. It is sufficient to drop from the corridor pool Λ all corridors whose δ_c^l is less than a given ξ, yielding to a very reduced $\tilde{\Lambda}$ set, which is then applied to the resolution of an integral version of the MP. As a result, the RBD cuts turn into OBC cuts, which will tighten more and more the integral MP of phase 2. Moreover, the optimal solution obtained in phase 1 is also feasible for the second phase.

Table 6.6 shows the specialized Benders Scheme (SBS), which has been used to solve the medium-sized network presented in chapter 8. The only difference in the fusion of DDS and PS schemes (see Table 6.5) lies in the Macro step 6a, which represents the set of operations that allows us to obtain a rounded solution when solving the relaxed version of the MP in phase 1.

Like in the CBS, the convergence of the SBS scheme is also guaranteed. Notice that if the integral version of the MP has a solution (which it does, since the set of feasible solutions is not null), its LP relaxation must exhibit some solution as well.

1. **Set** $\Omega \leftarrow \emptyset$; $UB \leftarrow \infty$; iteration $k \leftarrow 0$; allowable error ϵ; and phase $\leftarrow 1$

2. **Find an Initial Core Point** $(y^0, \tilde{y}^0, \tilde{f}^0)$.

3. **Solve** $IMWP^k$ **with** $(y^0, \tilde{y}^0, \tilde{f}^0)$.

4. **Generate** Ω^k **with** $(\beta^k, \pi^k, \gamma^k, \tau^k)$.

5. **If** phase = 1 **then** continue, **otherwise** goto Step 6b.

6a. **Remove** integrality for variables $\tilde{y}_i^l$, δ_c^l, b^l.

Solve MP^k **with** all $c \in \Lambda$.

Activate integrality for variables $\tilde{y}_i^l$, δ_c^l, b^l.

Solve MP^k **with** all $c \in \Lambda$ such that $\delta_c^l \geq \xi$, goto Step 7.

6b. **Solve** MP^k **with** all $c \in \Lambda$.

7. $LB \leftarrow (1-\alpha) \cdot z_{op}^k + \omega^k$.

8. **Solve** DSP^k **with** $(y^k, \tilde{y}^k, \tilde{f}^k)$.

9. $UB \leftarrow \min\left\{UB, (1-\alpha) \cdot z_{op}^k + z_{SP}^k\right\}$.

10. **if** phase = 2 **and** $\dfrac{UB - LB}{LB} < \epsilon$ **then** STOP

if phase = 1 **and** $\dfrac{UB - LB}{LB} < \epsilon$ **then** phase $\leftarrow 2$.

11. **Update** $(y^0, \tilde{y}^0, \tilde{f}^0) \leftarrow \lambda \cdot (y^0, \tilde{y}^0, \tilde{f}^0) + (1-\lambda) \cdot (y^k, \tilde{y}^k, \tilde{f}^k)$

12. $k \leftarrow k+1$ and goto Step 3.

Table 6.6: Pseudo-code of the Specialized Benders Scheme.

6.6 Preliminary computational results

In this section, we show some preliminary computational results to justify inclusion of the enhancements to the classical Benders decomposition. To do that, two test networks have been used. The first one is a complete 6 node railway network, shown in Figure 8.1 and tagged as N_1; whereas the other is a 9 node and 26 link railway network, depicted in Figure 8.2 and tagged as N_2. Since the SBS scheme needs to use the routing model M3, this one has been chosen for carrying out these experiments. Moreover, we include only one instance for each network with $|L^N| = 2$, since the CBS and/or the MWS schemes converge neither to zero nor to small gaps for $L^N > 2$ within a reasonable horizon time. We refer the reader to section 8.1.1 for further details regarding the input parameters associated with both networks.

Table 6.7 shows the general results of applying each scheme. The contents of each column are as follows. Column 1 denotes the network used. Column 2 indicates the Benders scheme applied. Column 3 shows the global objective function value. Column 4 shows the overall CPU time in seconds. Column 5 represents the Benders gap if the optimum is not reach. Finally, column 6 indicates the total number of Benders iterations.

The general results justify to some extent the gain CPU time from applying each enhancement. Clearly, the CBS and MWS schemes have a much slower convergence than the others. What's more, the CBS scheme is not capable of reaching optimality or near-optimality within a reasonable time. What's more, the Benders gaps are huge (44.65 % and 118,14 %). However, having applied the other enhancements,

Network	BD. Scheme	F.Obj.	Time	Gap	Iter.
N_1	CBS	230196	1498	44.65 %	356
	MWS	184034	503	0	132
	PS		152	0.0 %	53
	DDS		157	0.0 %	99
	SBS		99	0.0 %	67
N_2	CBS	836156	143	118.14 %	100
	MWS	787938	933	0.06 %	144
	PS	782332	123	0.0 %	42
	DDS		83	0.0 %	99
	SBS		43	0.0 %	68

Table 6.7: General results for the test networks applying each Benders scheme.

convergence improves greatly. On the other hand, introducing the two-phase schemes do not significantly improve N_1. However, in N_2, they are worth using. In chapter 8 we will show that applying these schemes to medium- and large-sized-networks yields better improvements.

For the sake of completion, we give more detailed results in the following tables. In 6.8, we show the time spent on each type of problem in the CBS scheme. The second and third columns, labeled $\boldsymbol{T_{MP}}$ and $\boldsymbol{T_{SP}}$, stand for the whole time spent on the MP and SP problems. It clearly indicates that the vast majority of time is spent on the MP problem, which contains complicated integer variables.

Network	$\boldsymbol{T_{MP}}$	$\boldsymbol{T_{SP}}$	$\boldsymbol{T_{total}}$
N_1	1467	31	1498
N_2	139	5	143

Table 6.8: Detailed results for the Classical Benders scheme.

Table 6.9 is similar to the previously mentioned table, but for the MWS and PS schemes. The extra column, labeled as $\boldsymbol{T_{IMWP}}$ / $\boldsymbol{T_{MWP}}$, denotes the whole time spent on the $IMWP$ or MWP problems, depending on the applied scheme. As stated before, the MP problem consumes the vast majority of time used by the Benders decomposition. However, we can also observe that the portion of time spent on the MWP problem in the MWS scheme is considerably higher than the time consumed by the $IWMP$ problem in the PS scheme. The convergence is also much better in the PS scheme.

Network	BD. Scheme	$\boldsymbol{T_{MP}}$	$\boldsymbol{T_{(I)MWP}}$	$\boldsymbol{T_{SP}}$	$\boldsymbol{T_{total}}$
N_1	MWS	499	15	4	503
	PS	145	5	2	152
N_2	MWS	915	18	12	933
	PS	117	2	4	123

Table 6.9: Detailed results for the Magnanti & Wong and Papadakos schemes.

Table 6.10 splits the general results of DDS and SBS schemes into each phase. The meaning of each column label is the same as those already mentioned. Its results show that the vast majority of time is spent in the integrality phase (phase 2), despite the lower number of Benders iterations. Moreover, rounding off the relaxed MP solution does not seem to greatly increase the time spent in the relaxation phase (phase 0). Furthermore, it provides a better start to the integrality phase. Consequently, its time decreases considerably.

What's more, for the N_2 with only one iteration in phase 2, it reaches optimality.

Network	BD. Scheme	Phase 1				Phase 2				T_{Total}
		F.Obj.	Time	Gap	Iter.	F.Obj.	Time	Gap	Iter.	
N_1	DDS	159091	32	0.0 %	65	184034	125	0.0 %	34	157
	SBS	193411	38	0.0 %	65	184034	63	0.0 %	14	99
N_2	DDS	738188	17	0.0 %	70	782332	66	0.0 %	28	83
	SBS	802678	40	0.0 %	67	782332	3	0.0 %	1	43

Table 6.10: General results of the Mc Daniel & Devine and Specialized Benders schemes per phase.

Finally, table 6.11 shows the resolution details for the whole MP problem in the SBS scheme. The column labeled $\boldsymbol{T_{\tilde{MP}}}$ stands for the whole time spent by the reduced integral master problem, whereas columns $\tilde{\mathbf{\Lambda}}$ and $\mathbf{\Lambda}$ represent, respectively, the average size of the reduced pool of corridors used per line in the reduced integral master problem, and the size of the total pool of corridors.

Networks	$T_{\tilde{MP}}$	T_{MP}	T_{total}	$\tilde{\mathbf{\Lambda}}$	$\mathbf{\Lambda}$
N_1	26	56	82	15	2235
N_2	27	3	29	17	355

Table 6.11: Detailed results for resolving the master problem in phase 1 of the Specialized Benders Scheme.

Results show that most of the increase in computational time is due to applying the rounding-off operation to the relaxed MP solution. However, as pointed out in the discussion results of the previous table, that increase is permissible, since the amount of time taken by the integrality phase decreases considerably. On the other hand, the size of the pool of corridors used per line in this rounding-off step is quite small. In the N_1 instance, only 7 out of 2235 corridors are used; whereas in the N_2 instance, only 6 out of 355 corridors are.

We conclude this section by showing the evolution of the Benders gap and the global objective function 2.1 per scheme and network in Figures 6.1 and 6.2, respectively. Focusing on the gap evolution, the traces show that, whatever the scheme used, the gap is not monotonous decreasing. However, it converges (near-)zero for all the schemes, except for the CBS, where it gets stuck on the$\log gap \approx 0$ and $\log gap \approx 1.2$ values, respectively. On the other hand, the convergence rates are pretty good for the PS, DDS and SBS schemes, although the MWS convergence rate is very slow. Moving on to the objective evolution, the scenario is quite similar: it also converges, but the irregularity of the trace is bigger. For the cases of the DDS and SBS schemes, there is a sudden slope due to the transition of phase 1 to phase 2.

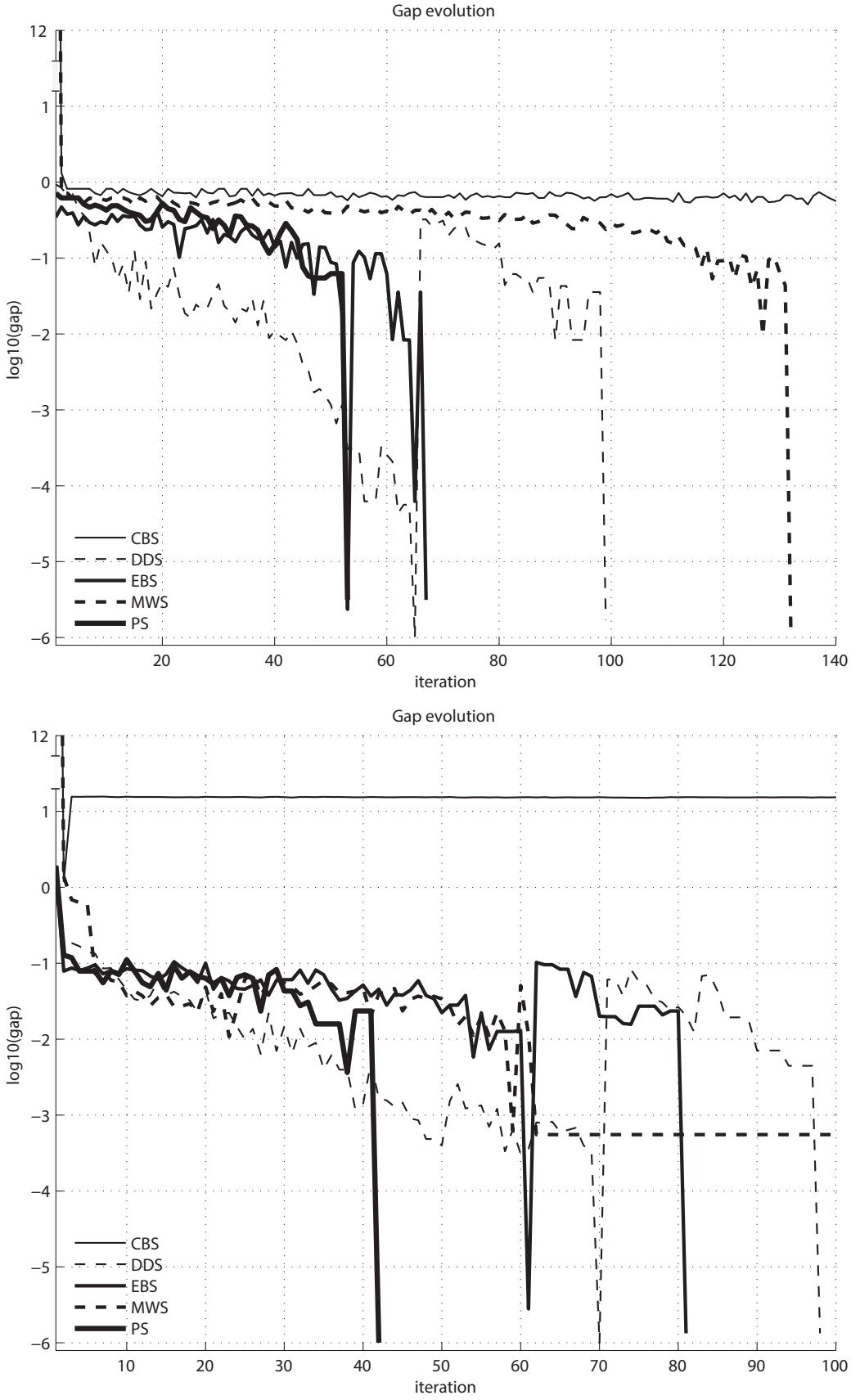


Figure 6.1: Evolution of the Benders gap for the different Benders schemes. At the top, the evolution traces for the 6NMAR network. At the bottom, the evolution traces for the 9NMAR network.

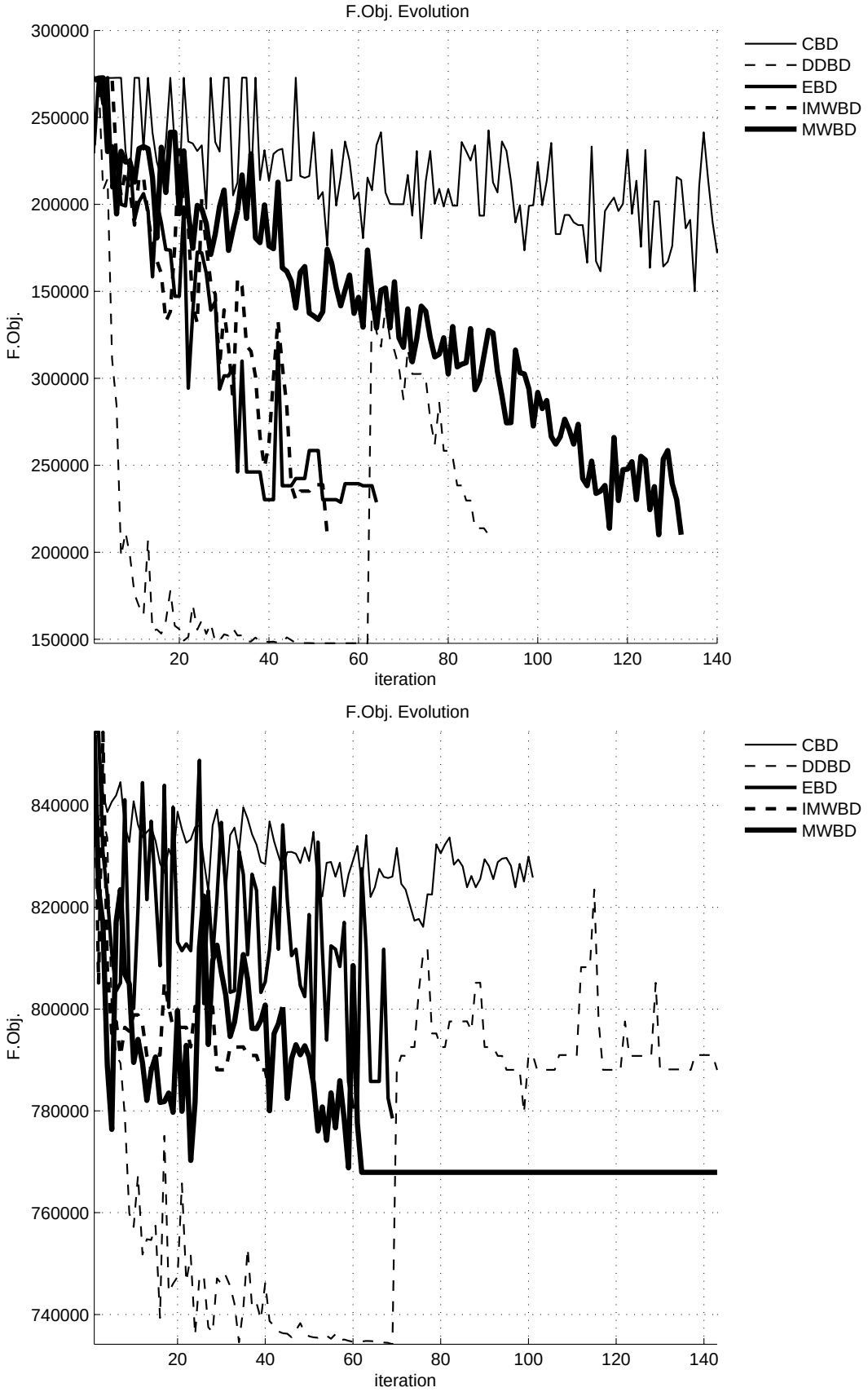


Figure 6.2: Evolution of the global objective function for the different Benders schemes. At the top, the evolution traces for the 6NMAR network. At the bottom, the evolution traces for the 9NMAR network.

Chapter 7

Modeling Approach with Elastic Demand

This chapter presents a combined modal splitting assignment model, as described by Oppenheim [145]. This model substitutes the assignment demand submodel shown in section 2.3.5 of chapter 2, and allows incorporation of elasticity in the demand of the model. Combined modal splitting-assignment models are non-linear in nature. To circumvent this difficulty, and to formulate the model into a linear integer programming framework, piecewise linearization has been carried out on the entropy terms that appear in these kinds of models. Moreover, including this submodel into the global model entails solving a subclass of mixed integer bilevel problems where discrete variables are present only in the upper level. Very few related and well developed works have been found on the state of the art and, thus, we were motivated to develop a new innovative technique for solving BLP problems whose structure adjusts to the elastic demand model developed in this chapter. The details of this algorithm, as well as a review of the related state of the art, can be found in chapter 3. For the sake of simplicity, here we have devoted ourselves only to showing the application of this algorithm. The structure of the chapter is as follows. Firstly, some mathematical notation will be introduced. Secondly we will demonstrate how to turn the inelastic demand model into an elastic one by means of the KKT optimality conditions. Thirdly, we will show how to efficiently solve the resulting model by adapting the algorithm shown in section 3.5.6.2, in order to take advantage of Magnanti & Wong and Papadakos' ideas [125], [147]. Finally, we include some preliminary results in order to show the validity of the algorithm.

7.1 Notation

In this subsection, we define all the mathematical elements that are specific to this chapter:

M	Set of modes under consideration (i.e., $M = \{TP, PRI\}$).
m	Identifier of a mode (i.e., $m \in \{TP, PRI\}$).
R_w	Set of intervals in which the entropy function for the o-d pair w has been discretized.
r	Identifier of an interval.
U_w^m	Value of the utility function of a user belonging to the o-d pair w that wants to use the mode m.
ϕ_w^m	Value of the entropy function of a user belonging to the o-d pair w that wants to use the mode m.
d_w^{PRI}	Average distance traveled by a user belonging to the o-d pair w that uses the private mode.
t_w^m	Average time of a user belonging to the o-d pair w that wants to use the mode m.
β_w^{PRI}	Parking cost at origin $p(w)$ and destination $q(w)$ of a user belonging to the o-d pair w that uses the private mode.
β_w^{TP}	Fare cost of a user belonging to the o-d pair w that uses the public transportation mode.
θ_w	A factor that weighs the average times of the modes $m \in M$.
γ_w	A factor that weighs the average traveled distance for a user belonging to the o-d pair w that uses the private mode.
g_w^m	Number of trips of the OD-pair w assigned to the mode m.

7.2 Modal Utilities

Up to now the whole amount of demand, given by the OD trip matrix, has been assigned to the public transportation network. This is not realistic since in practice there are other modes of transportation which are in competition. For instance, in a city like Barcelona there are bus and underground networks which can move people from its origin to its destination. Moreover, there is also a roadway network for those travelers who prefer to use their private car or taxi. The choice among these modes depends on several parameters, like fares, travel time and comfort.

To start, we have chosen only two modes which will be in competition. Let's say the public transportation mode (TP) and the private one (PRI). The choice between them will be based on the so-called utility functions given by equations (7.1) - (7.2), which capture the passenger parameters used for making their decision.

$$U_w^{PRI} = -\beta_w^{PRI} - \theta_w \cdot t_w^{PRI} - \gamma_w^{PRI} \cdot d_w^{PRI} \tag{7.1}$$

$$U_w^{TP} = -\beta_w^{TP} - \theta_w \cdot t_w^{TP} \tag{7.2}$$

Both utilities take into account two aspects: the monetary costs and the travel times of a given OD-pair w. For the private utility function (7.1), the monetary costs are characterized by β_w^{Pri} and $\gamma_w \cdot d_w^{Pri}$, which denote the parking fares in the origin and destination of the od-pair w and the cost of the consumed fuel, respectively. The latter is proportional to the travel distance (d_w^{Pri}); thus it is weighted by parameter γ_w which stands for the cost of the fuel per unit of distance. The travel time is captured by t_w^{Pri} and it is weighted by θ_w in order to render an economical sense.

Regarding the public transportation utility function (7.2), the monetary cost is characterized by β_w^{TP}, which denotes the public transportation fare, whereas travel time is defined by t_w^{TP}. Again, to render an economical sense for time, we weight it with the same θ_w parameter, since we consider that a user of the same od-pair gives the same value of time no matter what mode is used.

All the aforementioned parameters are considered as inputs to the model except for public transportation times t_w^{TP}, which depend on decisional variables related to the line topology and the passenger assignment. In the next section, we will discuss how to incorporate them into the model by considering that the demand is split into these models by the logit distribution functions given by equations (7.3) - (7.4), which are characterized by the aforementioned utilities.

$$g_w^{PRI} = g_w \cdot \frac{1}{1 + \exp^{(U_w^{TP} - U_w^{PRI})}} \tag{7.3}$$

$$g_w^{TP} = g_w \cdot \frac{1}{1 + \exp^{(U_w^{PRI} - U_w^{TP})}} \tag{7.4}$$

7.3 KKT Conditions for the Passenger Assignment Model

In this section we present the modifications that the passenger assignment model (PAM) (or, in other words, the primal version of the SP (6.3) - (6.10)) requires for splitting passenger demands in the same way as equations (7.3)-(7.4) do. Our deductions will be based on a reduced version of the SP, in which we will obtain the lagrangian function and then apply the KKT conditions.

First of all, we notice that since the MP problem determines which nodes are assigned to the constructed lines as well as their role (variables y_i^l and $\tilde{y}_i^l$, respectively), the flow blinding constraints (6.8) - (6.9) simply give us the in-station exchange flows (variables $v_{rs}^{w,l}$ and $\tilde{v}_{rs}^{w,l}$) which are active. Observe that, when in (6.9) $y_i^l - \tilde{y}_i^l = 0$, the $v_{rs}^{w,l}$ flows that are adjacent to node i ($(r,s) \in A_{xya}^N(i)$) are set automatically to zero, since they are defined as non-negative. Additionally, when in (6.8) $\tilde{y}_i^l = 1$, the $\tilde{v}_{rs}^{w,l}$ flows that are adjacent to node i ($(r,s) \in A_x^N(i)$) are set to zero by definition as well. Finally, the non-zero variables are simply bound to 1, which is trivial since that value cannot be exceeded by any flow variable satisfying constraints (6.4)-(6.7). Consequently, we can drop these non-active flow variables and the flow blinding constraints (6.8) - (6.9) from the model without loss of generality. In that manner, it allows us to work with a more compact formulation.

Having applied that variable and constraint reduction to the PAM (6.3) - (6.10), we arrive at the mathematical formulation of a Min Cost Capacitated Network flow problem ($MCCNFP$):

$$\min_{\boldsymbol{x}} \quad \boldsymbol{c}^T \, \boldsymbol{x} \tag{7.5}$$

$s.t.$

$$\boldsymbol{A} \, \boldsymbol{x}^p = \boldsymbol{b}^p, \quad \forall p \in O \quad (\boldsymbol{\lambda}^p) \tag{7.6}$$

$$\boldsymbol{x}^p \geq \boldsymbol{0}, \quad \forall p \in O \quad (\boldsymbol{\mu}^p) \tag{7.7}$$

$$\sum_{p \in O} \boldsymbol{x}^p \leq \bar{\boldsymbol{x}} \quad (\boldsymbol{\nu}) \tag{7.8}$$

where $\boldsymbol{x}$ is a vector holding the flow variables expressed by origins $p \in O$: u_{ij}^p, $v_{ij}^{p,l}$ and $\tilde{v}_{ij}^{p,l}$. Their dominion will be expressed from now on in the interval $(0, g_p)$. Constraints (7.6) are related to (6.4)-(6.7), whereas constraints (7.8) are equivalent to (6.10). The right hand side vector $\boldsymbol{b^p} = \{\boldsymbol{b_i^p} \mid \forall \boldsymbol{i} \in \boldsymbol{N_{TP}}\}$ captures the result of the flow balance for a given origin $p \in O$ and its expression is as follows:

$$b_i^p = \begin{cases} \sum\limits_{q \in D_p} g_{pq} & \text{if} \quad i = p(w) \\ -g_{pi} & \text{if} \quad i \neq p(w),\ i \in D_p \\ 0 & \text{if} \quad i \notin D_p \end{cases}, \quad \forall p \in O \tag{7.9}$$

The lagrangian function of problem (7.5)-(7.8) is obtained by appending to the objective function (7.5) the constraints (7.6)-(7.8) which are weighted by the lagrangian multipliers (the dual variables which are written next to the constraints). Having applied this step, we obtain the following (using vectorial notation):

$$\boldsymbol{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}, \boldsymbol{\nu}) = \boldsymbol{c}^T \boldsymbol{x} - \boldsymbol{\lambda}^T (\boldsymbol{A} \, \boldsymbol{x} - \boldsymbol{b}) - \boldsymbol{\mu}^T \boldsymbol{x} - \boldsymbol{\nu}^T \left(\bar{x} - \sum_{p \in O} \boldsymbol{x}^p \right) \tag{7.10}$$

Now, applying optimality conditions to (7.10), we deduce the relationship of the primal and dual variables with the time utility parameter t_w^{PRI}. Regarding the stationary condition, it establishes that $\boldsymbol{\nabla_x L(x,\lambda,\mu,\nu)} = \boldsymbol{0}$. Therefore, we obtain the following equation:

$$\boldsymbol{c} = \boldsymbol{A}^T \boldsymbol{\lambda} + \boldsymbol{\mu} - \boldsymbol{\nu} \tag{7.11}$$

which shows a relationship between the travel time costs $\boldsymbol{c}$ and dual variables $\boldsymbol{\lambda}^p$, $\boldsymbol{\mu}^p$ and $\boldsymbol{\nu}$, where the following complementarities (7.12) - (7.15) must hold between primal variables $\boldsymbol{x}$ and dual variables ($\boldsymbol{\lambda}$, $\boldsymbol{\mu}$, $\boldsymbol{\nu}$):

$$\boldsymbol{\mu}^p > \mathbf{0} \longrightarrow \boldsymbol{x}^p = \mathbf{0}, \quad \forall p \in O \tag{7.12}$$

$$\boldsymbol{x}^p > \mathbf{0} \longrightarrow \boldsymbol{\mu}^p = \mathbf{0}, \quad \forall p \in O \tag{7.13}$$

$$\boldsymbol{\nu}_{ij} > \mathbf{0} \longrightarrow \boldsymbol{x}_{ij} = \bar{\boldsymbol{x}}, \quad \forall (i,j) \in A \tag{7.14}$$

$$\boldsymbol{x}_{ij} > \bar{\boldsymbol{x}} \longrightarrow \boldsymbol{\nu}_{ij} = \mathbf{0}, \quad \forall (i,j) \in A \tag{7.15}$$

The incident matrix $\boldsymbol{A} = \{\boldsymbol{a_j} \mid j \in J\}$, where J is the total number of columns, has a well known structure. Each column vector $\boldsymbol{a_j}$ is made of zeros, except for two positions where there are 1 and -1 coefficients. Thus, the product $(\boldsymbol{\lambda^p})^T\ \boldsymbol{a_p}$ can be rewritten as $\lambda_j^p - \lambda_i^p,\ (i,j) \in A$. Let us neglect $\boldsymbol{\nu}$ from (7.11) for the moment and consider a single link (i,j), where some positive flow goes through without reaching the link's capacity. Under this situation, the following holds:

$$c_{ij} = \lambda_j^p - \lambda_i^p \tag{7.16}$$

where c_{ij} is related to the cost of traveling throughout the link (i,j).

Using formulas (7.16) and (7.11) in collaboration with relationships (7.12) - (7.15), it is possible to infer a close formula which associates link costs with dual variables. To demonstrate that, we will present two illustrative examples.

Let us consider the path solution for a given pair $(1,4) \in W$, as depicted in Figure 7.1. In this case, the total travel unit cost ($C_{1,4}$) for the pair $(1,4)$ is given by the sum of the costs associated with each traversed link (drawn in continuous trace) within the path $\mathcal{K}_{(1,4)}$ with origin at node 1 and destination at node 4:

$$C_{1,4} = \sum_{(i,j) \in \mathcal{K}_{(1,4)}} c_{ij} = c_{1,12-} + c_{12-,21+} + c_{21+,24-} + c_{24-,42+} + c_{42+,4} \tag{7.17}$$

and, if we substitute each c_{ij} with its corresponding expression in (7.16), we have:

$$C_{1,4} = \lambda_{12-}^1 - \lambda_1^1 + \lambda_{21+}^1 - \lambda_{12-}^1 + \lambda_{24-}^1 - \lambda_{21+}^1 + \lambda_{42+}^1 - \lambda_{24-}^1 + \lambda_4^1 - \lambda_{42+}^1 = \lambda_4^1 - \lambda_1^1 \tag{7.18}$$

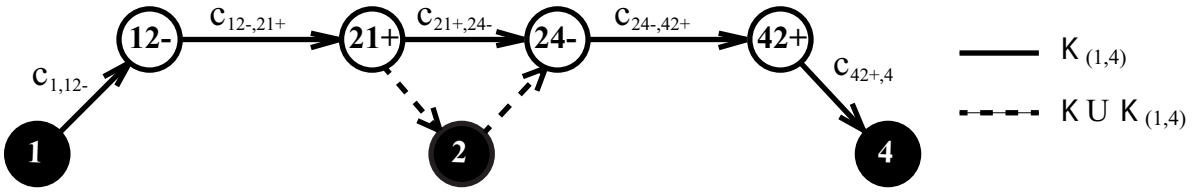


Figure 7.1: Example of a single path traversed by the OD-pair $(1,4)$.

Consequently, the total travel time cost ($c_{p,q}$) for an od-pair $(p,q) \in W$ which uses a single path can be computed by:

$$c_{p,q} = \lambda_q^p - \lambda_p^p \tag{7.19}$$

Let us now consider a more general example. Figure 7.2 depicts three different paths which are traversed by the same od-pair $(1,4) \in W$ whose layout is as follows:

$$\mathcal{K}^1_{(1,4)} = \{(1, 14-), (14-, 41+), (41+, 4)\} \tag{7.20}$$

$$\mathcal{K}^2_{(1,4)} = \{(1, 12-), (12-, 21+), (21+, 24-), (24-, 42+), (42+, 4)\} \tag{7.21}$$

$$\mathcal{K}^3_{(1,4)} = \{(1, 12-), (12-, 21+), (21+, 2), (2, 4)\} \tag{7.22}$$

Clearly, condition (7.14) happens in some links of paths $\mathcal{K}^1_{(1,4)}$ and $\mathcal{K}^2_{(1,4)}$. Otherwise, the pair $(1, 4)$ would be assigned to a single path (the shortest travel time cost one). Then, equation (7.19) must include dual variables $\nu_{i,j}$ for each in-vehicle link held in any path (i.e., $\forall (i, j) \in \mathcal{K}^{IN}_{(1,4)}$), which play the role of added costs. Carrying out this change, we have:

$$C^1_{1,4} = \lambda^1_4 - \lambda^1_1 - \nu_{14-,41+} \tag{7.23}$$

$$C^2_{1,4} = \lambda^1_4 - \lambda^1_1 - (\nu_{12-,21+} + \nu_{24-,42+}) \tag{7.24}$$

$$C^3_{1,4} = \lambda^1_4 - \lambda^1_1 - \nu_{12-,21+} \tag{7.25}$$

Suppose that, the travel time cost of paths ($c^i_{1,4},\ \forall i \in \{1, 2, 3\}$) verify that $c^1_{1,4} < c^2_{1,4} < c^3_{1,4}$. Thus, since the objective function has a minimization criterion, the passenger assignment will be carried out as follows. Paths $\mathcal{K}^1_{(1,4)}$ and $\mathcal{K}^2_{(1,4)}$ will carry as many passengers as follows, thus $\nu_{i,j} > 0$ for some or all $(i, j) \in \mathcal{K}^1_{(1,4)} \bigcup \mathcal{K}^2_{(1,4)}$), whereas path $\mathcal{K}^3_{(1,4)}$ will carry a low number of passengers, so $\nu_{i,j} = 0,\ \ \forall (i, j) \in \mathcal{K}^1_{(1,4)}$. This simplifies equations (7.23) - (7.25) as follows:

$$C^1_{1,4} = \lambda^1_4 - \lambda^1_1 - \nu_{14-,41+} \tag{7.26}$$

$$C^2_{1,4} = \lambda^1_4 - \lambda^1_1 - \nu_{24-,42+} \tag{7.27}$$

$$C^3_{1,4} = \lambda^1_4 - \lambda^1_1 \tag{7.28}$$

Consequently, equation (7.19) is also useful for assigning multiple path demands, since it denotes the travel time cost of the longest path. Therefore, it will be used as the unit travel time cost in the public transportation mode (t^{TP}_w) for evaluating its utility according to (7.2).

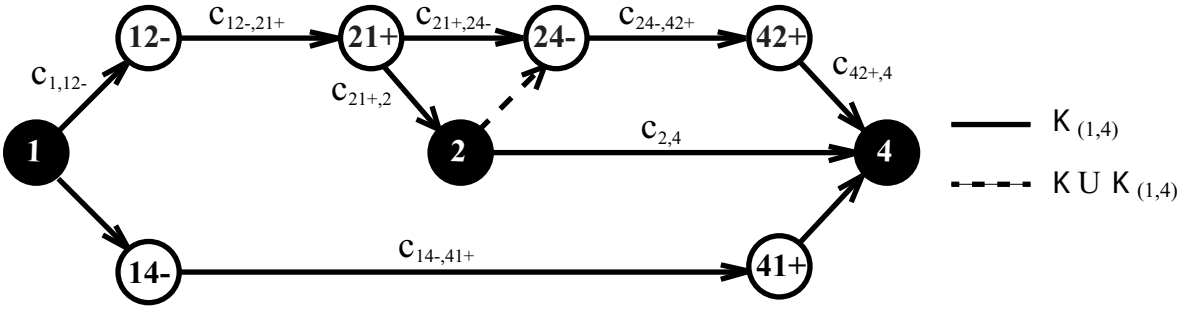


Figure 7.2: Example of multiple paths traversed by the OD-pair $(1, 4)$.

7.3.1 Accommodating elastic demand

The passenger objective function (2.2) evaluates only the passenger's travel time costs in the public transportation network. However, we now want to generalize it for splitting bimodal demand, where the modes

under consideration include the public transportation mode (TP) and the private mode (PRI). The amount of demand each mode uses will be computed indirectly through evaluation of the modal utilities (7.1) and (7.2) in the logit functions (7.3) and (7.4), respectively. This mechanism can be implemented by using an entropy function, which has a strong relationship with the modal distribution function and which also models it without explicit knowledge of the involved modal costs. Extra modal flows provide additional help. Following this subsection, we will describe the changes required by the static passenger assignment model for including these features.

The right hand side vector of the public transportation generalized flow balance ($\boldsymbol{b}^p$) is constant as long as we know beforehand the total demand flow originating at node p ($g_p = \sum_{q \in D_p} g_{p,q}$), which covers the entire public transportation network. However, we now only know that the total od-pair demand $g_w,\ \forall \in W$ will be split into two components: the demand flow traveling in private mode (g_w^{PRI}) and the demand flow going through the public transportation network (g_w^{TP}). To overcome that indetermination, we first define g_w^{TP} and g_w^{PRI} as modal demand flow variables, whose values must be within the interval $(0, g_w)$. Then we add the following constraints:

$$g_w^{TP} + g_w^{PRI} = g_w \quad (\xi^w), \quad \forall w \in W \tag{7.29}$$

$$g_w^{TP},\ g_w^{PRI} \geq 0, \quad \forall w \in W \tag{7.30}$$

which balance the modal demand flow for each OD demand pair. Finally, the expression of the right hand side vector $\boldsymbol{b}^{\boldsymbol{p}}$ is replaced with the following:

$$b_i^p = \begin{cases} \sum\limits_{q \in D_p} g_{p,q}^{TP} & \text{if} \quad i = p(w) \\ -g_{pi}^{TP} & \text{if} \quad i \neq p(w),\ i \in D_p \\ 0 & \text{if} \quad i \notin D_p \end{cases}, \quad \forall p \in O \tag{7.31}$$

Observe that this right side term is no longer constant, since it depends on decision variables g_w^{TP}. Now, let's move on to the changes in the passenger objective function. The entropy functions (E_w^{PRI} and E_w^{TP}) have the following form:

$$E_w^{PRI} = g_w^{PRI} \cdot \left(\ln g_w^{PRI} - 1\right) \tag{7.32}$$

$$E_w^{TP} = g_w^{TP} \cdot \left(\ln g_w^{TP} - 1\right) \tag{7.33}$$

which were introduced by Wilson [184]. The objective function ϕ can be built using these entropy functions in collaboration with the modal utilities (U_w^{PRI} and U_w^{TP}) as follows:

$$\phi = \sum_{w \in W} \sum_{m \in M} \left(U_w^m \cdot g_w^m + E_w^m\right) \tag{7.34}$$

Then, substituting functions E_w^m with (7.32) - (7.33) and U_w^m with (7.1) - (7.2), then arranging each term properly, we arrive at the following expression for ϕ:

$$\phi = \sum_{w \in W} \left[g_w^{PRI} \cdot \left(\ln g_w^{PRI} - \beta_w^{PRI} - \theta_w \cdot t_w^{PRI} - \gamma_w^{PRI} \cdot d_w^{PRI} - 1\right) + \right. \tag{7.35}$$

$$\left. + g_w^{TP} \cdot \left(\ln g_w^{TP} - \beta_w^{TP} - \theta_w \cdot t_w^{TP} - 1\right)\right]$$

This equation cannot be directly used as the passenger objective function for the elastic demand, since the unit travel time t_w^{TP} depends on the dual variables λ_q^p, λ_p^p, according to equation (7.19). However, this inconvenience can be easily overcome by replacing the term $\theta_w \cdot t_w^{TP} \cdot g_w^{TP}$ with the passenger traveling time costs (2.2). Let us continue to consider the passenger flows throughout the public transportation network, expressed in terms of origins by the vector $\boldsymbol{x}^p$. Then:

$$\phi = \sum_{w \in W} \left[g_w^{PRI} \cdot \left(\ln g_w^{PRI} - \beta_w^{PRI} - \theta_w \cdot t_w^{PRI} - \gamma_w^{PRI} \cdot d_w^{PRI} - 1 \right) + \right. \tag{7.36}$$

$$\left. + g_w^{TP} \cdot \left(\ln g_w^{TP} - \beta_w^{TP} - 1 \right) \right] + \sum_{p \in O} \boldsymbol{c}^T \boldsymbol{x}^p$$

where $\boldsymbol{c}$ is a vector of the link traveling time costs, which includes properly arranged weights θ_w. Finally, let us divide (7.36) by $\dfrac{1}{\theta_w}$, arriving at the final passenger objective function for the elastic demand:

$$z_{pax} = \sum_{w \in W} \frac{1}{\theta_w} \left[g_w^{PRI} \cdot \left(\ln g_w^{PRI} - \beta_w^{PRI} - \theta_w \cdot t_w^{PRI} - \gamma_w^{PRI} \cdot d_w^{PRI} - 1 \right) + \right. \tag{7.37}$$

$$\left. + g_w^{TP} \cdot \left(\ln g_w^{TP} - \beta_w^{TP} - 1 \right) \right] + \sum_{p \in O} \tilde{\boldsymbol{c}}^T \boldsymbol{x}^p$$

where $\tilde{\boldsymbol{c}}$ stands for the vector of the modified link travel time costs. The combined modal splitting assignment model will be then stated as a minimized objective function (7.37) subject to (6.3) - (6.10), (7.29) and (7.30). The following proposition proves that a logit modal split is attained by the optimal solutions of this model:

Proposition (validity of objective function (7.37)). Equations (7.3) and (7.4) provide a logit modal split that can be reproduced by the minimized objective function (7.37) with decision variables $\boldsymbol{x}$, $\boldsymbol{g_{PRI}}$ and $\boldsymbol{g_{TP}}$; subject to the link-flow balance constraints (7.6), (7.7), capacity constraints (7.8), modal split constraints (7.29), (7.30).

Proof: Consider the following lagrangian function 7.38 for the minimization problem of the statement:

$$\boldsymbol{L}(\boldsymbol{x}, \boldsymbol{g}, \boldsymbol{\lambda}, \boldsymbol{\mu}, \boldsymbol{\nu}, \boldsymbol{\xi}) = z_{pax} - \boldsymbol{\lambda}^T \left(\boldsymbol{A}\, \boldsymbol{x} - \boldsymbol{b} \right) - \boldsymbol{\mu}^T \boldsymbol{x} - \boldsymbol{\nu}^T \left(\bar{\boldsymbol{x}} - \sum_{p \in O} \boldsymbol{x}^p \right) - \tag{7.38}$$

$$- \boldsymbol{\xi}^T \left(\boldsymbol{g}^{TP} + \boldsymbol{g}^{PRI} - \boldsymbol{g} \right)$$

where the dual variable vector $\boldsymbol{\xi}$ is associated with the demand flow balance constraints (7.29). If we now apply the stationary condition to the demand mode flow variables g_w^{TP} and g_w^{PRI}, i.e., $\boldsymbol{\nabla}_{\boldsymbol{g}^{TP}}\, \boldsymbol{L}(\boldsymbol{x}, \boldsymbol{g}, \boldsymbol{\lambda}, \boldsymbol{\mu}, \boldsymbol{\nu}) = \boldsymbol{0}$ and $\boldsymbol{\nabla}_{\boldsymbol{g}^{PRI}}\, \boldsymbol{L}(\boldsymbol{x}, \boldsymbol{g}, \boldsymbol{\lambda}, \boldsymbol{\mu}, \boldsymbol{\nu}) = \boldsymbol{0}$, we obtain the following equations:

$$\frac{1}{\theta^w} \cdot \left[\ln g_w^{TP} + \beta_w^{TP} \right] - \xi^w + t_{p(w),q(w)}^{TP} = 0 \tag{7.39}$$

$$\frac{1}{\theta^w} \cdot \left[\ln g_w^{PRI} - \tilde{U}_w^{PRI} \right] - \xi^w = 0 \tag{7.40}$$

where $\tilde{U}_w^{PRI}$ is a slight modification of the private utility function (7.1) in which all terms have been divided by the weight θ^w. Rearranging the terms of equations (7.39) - (7.40), we can obtain the following relationships between the utility functions and variables g_w^m, ξ^w:

$$\tilde{U}_w^{TP} = \frac{1}{\theta^w} \ln g_w^{TP} - \xi^w \tag{7.41}$$

$$U_w^{PRI} = \frac{1}{\theta^w} \ln g_w^{PRI} - \xi^w \tag{7.42}$$

where $\tilde{U}_w^{TP}$ is similar to the public transportation utility function (7.2) in which all terms have been divided by the weight θ^w. Isolating the g_w^{TP} and g_w^{PRI} variables from equations (7.41)-(7.42), we obtain:

$$g_w^{TP} = \exp^{\theta^w \cdot (\xi^w + \tilde{U}_w^{TP})} \tag{7.43}$$

$$g_w^{PRI} = \exp^{(\theta^w \cdot \xi^w + U_w^{PRI})} \tag{7.44}$$

and if we undo the parenthesis in (7.43) and multiply all the elements by θ^w, we arrive at the following expression, which depends on the standard form of the utility functions:

$$g_w^{TP} = \exp^{(\theta^w \cdot \xi^w + U_w^{TP})} \tag{7.45}$$

$$g_w^{PRI} = \exp^{(\theta^w \cdot \xi^w + U_w^{PRI})} \tag{7.46}$$

Finally, if we compute the ratio expressions $\frac{g_w^{TP}}{g_w}$ and $\frac{g_w^{PRI}}{g_w}$ by using (7.45)-(7.46) in collaboration with (7.29) - (7.30), we arrive at the following equations:

$$g_w^{PRI} = g_w \cdot \frac{1}{1 + \exp^{(U_w^{TP} - U_w^{PRI})}} \tag{7.47}$$

$$g_w^{TP} = g_w \cdot \frac{1}{1 + \exp^{(U_w^{PRI} - U_w^{TP})}} \tag{7.48}$$

which are in fact the logit distributions functions (7.3)-(7.4). Thus, we have proved that the objective function (7.37) is valid. □

For computational implementation, we do not work directly with (7.37), since the modal entropy functions (7.32) - (7.33) are non-linear. However, as they are convex, we can linearize them with piecewise linear functions as follows. Let us denote ϕ_w^{PRI} and ϕ_w^{TP} as continuous free variables which approximate the real values of (7.32) - (7.33). Then the linear approximation can be implemented by means of the following set of constraints:

$$\phi_w^{PRI} - b_r^w \cdot g_w^{PRI} \geq a_r^w, \quad \forall\, r \in R_w,\ w \in W \tag{7.49}$$

$$\phi_w^{TP} - b_r^w \cdot g_w^{TP} \geq a_r^w, \quad \forall\, r \in R_w,\ w \in W \tag{7.50}$$

where b_r^w and a_r^w are the coefficients of the piecewise linear functions that linearize (7.32) - (7.33) at interval $r \in R_w$. Now, if we replace the terms in (7.37) (which are related to (7.32) - (7.33)) with variables ϕ_w^{PRI} and ϕ_w^{TP}, we arrive at the following passenger objective function, which has been implemented:

$$z_{pax} = \sum_{w \in W} \frac{1}{\theta_w} \left[\phi_w^{PRI} - g_w^{PRI} \cdot \left(\beta_w^{PRI} + \theta_w \cdot t_w^{PRI} + \gamma_w^{PRI} \cdot d_w^{PRI} \right) + \right. \tag{7.51}$$

$$\left. + \phi_w^{TP} - g_w^{TP} \cdot \beta_w^{TP} \right] + \sum_{p \in O} \tilde{\boldsymbol{c}}^T \, \boldsymbol{x}^p$$

For the sake of simplicity, we have skipped over the details for obtaining the coefficients related to piecewise linear equations (b_r^w and a_r^w). The interested reader is referred to appendix section 9.3.

7.4 Formulating the bimodal splitting of demand flow

The model with elastic demand cannot be formulated by directly incorporating the aforementioned changes into the mathematical programming problem shown in section 2.3. Notice that the modal entropy functions (7.32) - (7.33) do not give any currency units, and thus they cannot be integrated into the global objective function (2.1). Moreover, it expresses a tradeoff between two agents: operators and users, whereas the entropy function provides a way of choosing the transportation mode, which has nothing to do with the objective function's goal.

This conflict can be overcome by formulating the network design model with elastic demand as a subclass of a mixed-integer linear bilevel problem ($MILBP$), where only the upper level controls discrete variables. Following this subsection, we will provide: its mathematical formulation, a summary of its solving algorithm (shown in detail in section 3.5.6.2), and an enhanced version based on the ideas of Magnanti & Wong [125] and Papadakos [147].

A subclass of $MILBP$, where only the upper level controls discrete variables, can be formulated as follows:

$$\min_{\boldsymbol{x_1}, \boldsymbol{x_2} \in \boldsymbol{X}, \boldsymbol{y}} \boldsymbol{c_{11}} \boldsymbol{x_1} + \boldsymbol{c_{12}} \boldsymbol{x_2} + \boldsymbol{c_{13}} \boldsymbol{y} \tag{7.52}$$

$$\text{s.t.} \min_{\boldsymbol{y}} \boldsymbol{c_{21}} \boldsymbol{x_1} + \boldsymbol{c_{22}} \boldsymbol{x_2} + \boldsymbol{c_{23}} \boldsymbol{y}$$

$$\text{s.t. } \boldsymbol{g}(\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{y}) \triangleq \boldsymbol{A_1} \boldsymbol{x_1} + \boldsymbol{A_2} \boldsymbol{x_2} + \boldsymbol{A_3} \boldsymbol{y} \leq \boldsymbol{b}$$

$$\boldsymbol{x_1} \in Z^{m_1}, \ \boldsymbol{x_2} \in \Re^{m_2}$$

$$\boldsymbol{y} \in \Re^{+,n}$$

Regarding the elastic demand model, the m_1 - dimensional vector $\boldsymbol{x_1}$ is related to the network layout variables $\boldsymbol{\delta}$, $\tilde{\boldsymbol{y}}$ and planning variables $\boldsymbol{b}$, whereas the m_2 - dimensional vector $\boldsymbol{x_2}$ is associated with the remaining layout variables $\boldsymbol{x}$, $\boldsymbol{y}$ and remaining planning variables $\boldsymbol{f}$, Δb. Finally, the n - dimensional vector $\boldsymbol{y}$ holds all the variables related to passenger flows and modal choices: $\boldsymbol{u}$, $\boldsymbol{v}$, $\tilde{\boldsymbol{v}}$, $\boldsymbol{g^{TP}}$, $\boldsymbol{g^{PRI}}$, $\boldsymbol{\phi^{TP}}$, $\boldsymbol{\phi^{PRI}}$. Thus, the upper level objective function optimizes the global function (2.1), whereas the lower level function optimizes the modal demand splitting function (7.51). Regarding constraint sets, $\boldsymbol{X}$ symbolizes the independent operator constraint set, whereas $\boldsymbol{g}(\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{y})$ contain the interdependent constraint set between operators and passengers.

In section 3.5.6.2, it is demonstrated that the $MILBP$ (7.52) can be solved by means of an algorithm which splits this problem into two simpler problems. One of them is called the master problem, which relaxes the original $MILBP$. Its mathematical structure is as follows:

$$\min_{\boldsymbol{x_1},\boldsymbol{x_2},\boldsymbol{\delta},w_1,w_2} \boldsymbol{c_{11}x_1} + \boldsymbol{c_{12}x_2} + w_1 + w_2 \tag{7.53}$$

$$\text{s.t. } w_1 \geq \psi\, w_2 + h_k^1, \quad \forall k \in K$$

$$w_2 \geq h_k^2, \quad \forall k \in K$$

$$\delta_k \leq 1 - \frac{1}{M_k}\left(w_2 - h_k^2\right), \quad \forall k \in K$$

$$\sum_{k \in K} \delta_k \geq 1$$

$$\boldsymbol{x_1}, \boldsymbol{x_2} \in \boldsymbol{X}$$

$$\boldsymbol{x_1} \in Z^{m_1},\ \boldsymbol{x_2} \in\ Re^{m_2}$$

$$w_1,\ w_2 \text{ free}$$

$$\boldsymbol{\delta} \in \{0,1\}^{|K|}$$

which controls the operator variables as well as variables w_1, w_2, $\boldsymbol{\delta}$. These are related to the Benders cuts h_k^n as well as an extra set, which is used to assure that at least one h_k^2 is active. The Benders cuts are used to iteratively build the active interdependencies in the upper and lower level problems by means of resolving the second subproblem. It consists of a hierarchical problem with the following mathematical structure:

$$\min_{\boldsymbol{y}} \left(\tilde{\boldsymbol{c}}_{\boldsymbol{13}} + \boldsymbol{c_{23}}\right) \boldsymbol{y} \tag{7.54}$$

$$\min_{\boldsymbol{y}}\ \boldsymbol{c_{23}}\ \boldsymbol{y}$$

$$\text{s.t. } \boldsymbol{g}(\bar{\boldsymbol{x}}_{\boldsymbol{1}}, \bar{\boldsymbol{x}}_{\boldsymbol{2}}, \boldsymbol{y}) \triangleq \boldsymbol{A_3 y} \leq \boldsymbol{b} - \boldsymbol{A_1}\bar{\boldsymbol{x}}_{\boldsymbol{1}} - \boldsymbol{A_2}\bar{\boldsymbol{x}}_{\boldsymbol{2}}$$

$$\boldsymbol{y} \in \Re^{+,n}$$

where both objective functions are related to passenger costs. The difference relies on the inclusion of costs $\tilde{\boldsymbol{c}}_{\boldsymbol{13}}$ in the upper level objective function. Regarding the elastic demand model, $\boldsymbol{c_{23}}$ are related to the passenger travel time costs, whereas $\tilde{\boldsymbol{c}}_{\boldsymbol{13}}$ are associated to the remaining utility function costs and the entropy function costs.

Problem (7.54) can be solved in two steps by resolving two subproblems called $SP1$ and $SP2$. The latter is related to the inner problem, and thus its mathematical structure is as follows:

$$\min_{\boldsymbol{y}}\ \boldsymbol{c_{23}}\ \boldsymbol{y} \tag{7.55}$$

$$\text{s.t. } \boldsymbol{g}(\bar{\boldsymbol{x}}_{\boldsymbol{1}}, \bar{\boldsymbol{x}}_{\boldsymbol{2}}, \boldsymbol{y}) \triangleq \boldsymbol{A_3 y} \leq \boldsymbol{b} - \boldsymbol{A_1}\bar{\boldsymbol{x}}_{\boldsymbol{1}} - \boldsymbol{A_2}\bar{\boldsymbol{x}}_{\boldsymbol{2}}\ \ (\boldsymbol{\pi_2})$$

$$\boldsymbol{y} \in \Re^{+,n}$$

The optimal dual variables of this problem ($\boldsymbol{\pi_2}$) are used to build a linking constraint which takes the place of the inner problem in (7.54) and allows the formulation of the $SP1$ problem as follows:

$$\min_{\boldsymbol{y}} \ \tilde{\boldsymbol{c}}_{\boldsymbol{13}}\boldsymbol{y} \tag{7.56}$$

$$\text{s.t. } \boldsymbol{c_{23}y} = (\boldsymbol{b} - \boldsymbol{A_1}\bar{\boldsymbol{x}}_{\boldsymbol{1}} - \boldsymbol{A_2}\bar{\boldsymbol{x}}_{\boldsymbol{2}})\boldsymbol{\pi_2} \ \ (\psi)$$

$$\boldsymbol{g}(\bar{\boldsymbol{x}}_{\boldsymbol{1}}, \bar{\boldsymbol{x}}_{\boldsymbol{2}}, \boldsymbol{y}) \triangleq \boldsymbol{A_3 y} \leq \boldsymbol{b} - \boldsymbol{A_1}\bar{\boldsymbol{x}}_{\boldsymbol{1}} - \boldsymbol{A_2}\bar{\boldsymbol{x}}_{\boldsymbol{2}} \ \ (\boldsymbol{\pi_1})$$

$$\boldsymbol{y} \in \Re^{+,n}$$

where cost $\boldsymbol{c_{23}}$ has been removed. This operation is valid since the master objective function includes variable w_1, which approaches the expression of $\tilde{\mathbf{c}}_{\mathbf{13}} + \mathbf{c_{23}y}$ by means of cut constraints $h_k^1, \ \ \forall\ k \in K$. The derivation of these two subproblems is based on duality theory, and the proof can be found in section 3.5.6.2.

Now, if we incorporate the changes mentioned in the previous subsection into the elastic passenger assignment model, problems $SP1$ and $SP2$ can be explicitly formulated by (7.73) - (7.83) and (7.63) - (7.72), respectively. Notice that in parenthesis on the right hand side of the constraints, we have written its dual variables. Moreover the linking constraint (7.74) in $SP1$ has been expressed by using the optimal primal value of $SP2$ (z^*_{SP2}), which is equivalent to the dual one, according to the strong duality theorem.

Problems (7.73) - (7.83) and (7.63) - (7.72) have been expressed in their primal form instead of in dual. This is because of formulation issues, since it is easier to state them in the primal version and pass them to a commercial optimization software like CPLEX, which provides both the primal and dual variables as a solution.

Given the dual variables associated with the independent right side constraint terms (7.74) - (7.83) and (7.64) - (7.72) of the SP1 and SP2 problems, respectively, the optimality Benders cuts can be explicitly formulated as follows:

$$h_k^n = \sum_{p\in O}\sum_{i\in N}\sum_{l\in L_i^N}\left[(c_{\pi_{i,p,l}^{k,n}} - c_{\gamma_{i,p,l}^{k,n}})\tilde{y}_i^l - c_{\pi_{i,p,l}^{k,n}} y_i^l\right] + \sum_{w\in W}\left(c_{\xi_w^{k,n}} + c_{\nu_w^{k,n}}\right) - \sum_{(i,j)\in A_{TP}}\sum_{l\in L_{ij}} c_{\tau_{ij,l}^{k,n}}\tilde{f}_{ij}^l \tag{7.57}$$

where n and k are indexes denoting the subproblem and iteration of the algorithm, respectively, and $c_{\pi_{i,p,l}^{k,n}}$, $c_{\gamma_{i,p,l}^{k,n}}$, $c_{\xi_w^{k,n}}$, $c_{\nu_w^{k,n}}$ and $c_{\tau_{ij,l}^{k,n}}$ are coefficient functions of the master variables involved in the bender's cut associated with the problem's dual variables. Their mathematical expressions are as follows:

$$c_{\pi_{i,p,l}^{k,n}} = g_p \cdot \pi_{i,p}^{l,k,n} \tag{7.58}$$

$$c_{\gamma_{i,p,l}^{k,n}} = g_p \cdot \gamma_{i,p,l}^{k,n} \tag{7.59}$$

$$c_{\xi_w^k} = g_w \cdot \xi_w^{k,n} \tag{7.60}$$

$$c_{\nu_w^k} = \sum_{r\in R^w} a_r^w \sum_{m\in M} \nu_{r,w,m}^{k,n} \tag{7.61}$$

$$c_{\tau_{ij,l}^{k,n}} = q \cdot \tau_{ij,l}^{k,n} \tag{7.62}$$

$$z_{SP2} = \min_{v,\tilde{v},u,g,\phi} \quad \sum_{w\in W} \frac{1}{\theta_w}\left(\sum_{m\in M}\phi_w^m - \tilde{U}_w^{PRI} g_w^{PRI} + \beta_w^{TP} g_w^{TP}\right) + \tag{7.63}$$
$$+ \sum_{p\in O}\sum_{(i,j)\in A}\left(\sum_{l\in L_{ij}} t_{ij}^{TP} v_{ij}^{p,l} + t_{ij}^{COM} u_{ij}^{p}\right)$$

$s.t.$

$$\sum_{l\in L_i}\left(\sum_{j\in l_{a(i)}} v_{ij}^{p,l} - \sum_{j\in l_{y(i)}} v_{ji}^{p,l}\right) + \sum_{j\in A_{COM}(i) j\in A_{COM}(i)} u_{ij}^{p} - \sum u_{ji}^{p} - b_i^p = 0 \quad \left(\beta_{i,p}^{k,2}\right), \quad \forall i \in N,\ p \in O \tag{7.64}$$

$$v_{inv^+(i)}^{p,l} - v_{y(i)}^{p,l} - \sum_{j\in A^l_{x^+(i)}} \left(v_{ij}^{p,l} + \Psi_{ij}^{p,l}\right) = 0 \quad \left(\rho_{i,p}^{l+,k,2}\right), \quad \forall l \in L,\ i \in N_{TP}^{S+}(l),\ p \in O \tag{7.65}$$

$$v_{inv^-(i)}^{p,l} - v_{a(i)}^{p,l} - \sum_{j\in A^l_{x^-(i)}} \left(v_{ji}^{p,l} + \Psi_{ji}^{p,l}\right) = 0 \quad \left(\rho_{i,p}^{l-,k,2}\right), \quad \forall l \in L,\ i \in N_{TP}^{S-}(l),\ p \in O \tag{7.66}$$

$$v_{inv^+(i)}^{p,l} - v_{inv^-(i)}^{p,l} = 0 \quad \left(\chi_{j,p}^{l,k,2}\right), \quad \forall l \in L^E,\ i \in N_{TP}^{P}(l),\ p \in O \tag{7.67}$$

$$\sum_{(r,s)\in A_{xya}^N(i)} v_{rs}^{p,l} \le \tilde{y}_i^l \quad \left(\gamma_{i,p,l}^2\right), \quad \forall l \in L^N,\ i \in N_{TP}^N,\ p \in O \tag{7.68}$$

$$\sum_{(r,s)\in A_x^N(i)} \tilde{v}_{rs}^{p,l} \le y_i^l - \tilde{y}_i^l \quad \left(\pi_{i,p,l}^2\right), \quad \forall l \in L^N,\ i \in N_{TP}^N,\ p \in O \tag{7.69}$$

$$\sum_{p\in O} v_{inv(i,j)}^{p,l} \le q \cdot \tilde{f}_{ij}^l \quad \left(\tau_{ij,l}^2\right), \quad \forall (i,j) \in A_{TP},\ l \in L_{ij} \tag{7.70}$$

$$g_w^{TP} + g_w^{PRI} = g_w \quad \left(\xi_2^w\right), \quad \forall w \in W \tag{7.71}$$

$$\phi_w^m - b_r^w \cdot g_w^m \ge a_r^w \quad \left(\nu_{r,w,m}^2\right), \quad \forall w \in W,\ r \in R_w,\ m \in M \tag{7.72}$$

$$z_{SP1} = \min_{v,\tilde{v},u,g,\phi} \quad -\sum_{w\in W}\frac{1}{\theta_w}\sum_{m\in M}\phi_w^m \tag{7.73}$$

$s.t.$

$$\sum_{w\in W}\frac{1}{\theta_w}\left(\sum_{m\in M}\phi_w^m - \tilde{U}_w^{PRI}g_w^{PRI} + \beta_w^{TP}g_w^{TP}\right) + \tag{7.74}$$

$$+\sum_{p\in O}\sum_{(i,j)\in A}\left(\sum_{l\in L_{ij}} t_{ij}^{TP}v_{ij}^{p,l} + t_{ij}^{COM}u_{ij}^{p}\right) = z_{SP2}^{*} \quad (\eta)$$

$$\sum_{l\in L_i}\left(\sum_{j\in l_{a(i)}} v_{ij}^{p,l} - \sum_{j\in l_{y(i)}} v_{ji}^{p,l}\right) + \sum_{j\in A_{COM}(i)j\in A_{COM}(i)} u_{ij}^{p} - \sum u_{ji}^{p} - b_i^p = 0 \quad \left(\beta_{i,p}^{k,1}\right), \ \forall i\in N,\ p\in O \tag{7.75}$$

$$v_{inv^+(i)}^{p,l} - v_{y(i)}^{p,l} - \sum_{j\in A^l_{x^+(i)}}\left(v_{ij}^{p,l} + \Psi_{ij}^{p,l}\right) = 0 \quad \left(\rho_{i,p}^{l+,k,1}\right), \ \forall l\in L,\ i\in N_{TP}^{S+}(l),\ p\in O \tag{7.76}$$

$$v_{inv^-(i)}^{p,l} - v_{a(i)}^{p,l} - \sum_{j\in A^l_{x^-(i)}}\left(v_{ji}^{p,l} + \Psi_{ji}^{p,l}\right) = 0 \quad \left(\rho_{i,p}^{l-,k,1}\right), \ \forall l\in L,\ i\in N_{TP}^{S-}(l),\ p\in O \tag{7.77}$$

$$v_{inv^+(i)}^{p,l} - v_{inv^-(i)}^{p,l} = 0 \quad \left(\chi_{j,p,l}^{k,1}\right), \ \forall l\in L^E,\ i\in N_{TP}^{P}(l),\ p\in O \tag{7.78}$$

$$\sum_{(r,s)\in A^N_{xya}(i)} v_{rs}^{p,l} \le \tilde{y}_i^l \quad \left(\gamma_{i,p,l}^{1}\right), \ \forall l\in L^N,\ i\in N_{TP}^{N},\ p\in O \tag{7.79}$$

$$\sum_{(r,s)\in A^N_{x}(i)} \tilde{v}_{rs}^{p,l} \le y_i^l - \tilde{y}_i^l \quad \left(\pi_{i,p,l}^{1}\right), \ \forall l\in L^N,\ i\in N_{TP}^{N},\ p\in O \tag{7.80}$$

$$\sum_{p\in O} v_{inv(i,j)}^{p,l} \le q\cdot \tilde{f}_{ij}^{l} \quad \left(\tau_{ij,l}^{1}\right), \ \forall (i,j)\in A_{TP},\ l\in L_{ij} \tag{7.81}$$

$$g_w^{TP} + g_w^{PRI} = g_w \quad (\xi_1^w), \ \forall w\in W \tag{7.82}$$

$$\phi_w^m - b_r^w\cdot g_w^m \ge a_r^w \quad \left(\nu_{r,w,m}^{1}\right), \ \forall w\in W,\ r\in R_w,\ m\in M \tag{7.83}$$

Finally, using the cut expression (7.57), we can explicitly formulate the relaxed $MIBLP$ (master problem) as follows:

$$z_{MP} = \min \quad (1-\alpha)\ z_{op} + \alpha\ (\omega_1 + \omega_2) \tag{7.84}$$

subject to

Network design constraints (2.4)-(2.7)

Line frequency setting constraints (2.33)-(2.36) and (2.42)-(2.45)

Breaking symmetry constraints (2.59)

Relaxing integrality constraints (2.60)-(2.63) if routing model M1 is chosen or, (2.60)-(2.63) if routing model M2 is chosen or, (2.60)-(2.61) otherwise

Routing constraints (2.8)-(2.12) if routing model M1 is chosen or, (2.8)-(2.9), (2.19)-(2.22), (2.24)-(2.26) if routing model M2 is chosen or, (2.27)-(2.29) otherwise

and the following optimality Benders cuts:

$$\omega_1 \geq \eta\, \omega_2 + h_k^1, \quad \forall k \in \Omega \tag{7.85}$$

$$\omega_2 \geq h_k^2, \quad \forall k \in \Omega \tag{7.86}$$

$$\delta_k \leq 1 - \frac{1}{M_k}\left(\omega_2 - h_k^2\right), \quad \forall k \in \Omega \tag{7.87}$$

$$\sum_{k \in \Omega} \delta_k \geq 1 \tag{7.88}$$

where constraints (7.85) - (7.86) are related to Benders cuts coming from SP1 and SP2 problems, respectively. Moreover, constraints (7.87) - (7.88) assure that, at least one Benders cut (7.86) is active. δ_k is a binary variable denoting whether the cut (7.86) computed at iteration k is active, and M_k is a parameter big enough to prevent the cut (7.86) from being active when $\delta_k = 0$. This mechanism is required because of the unbounded nature of (7.86), as stated in section 3.5.6.2. Finally, z_{op} is related to the operator objective function (2.3).

The master problem (7.84) and subproblems (7.73) - (7.83) and (7.63) - (7.72) interact according to the scheme shown in table 3.7 of section 3.5.6.2. This scheme, which is a direct adaptation of the classical Benders decomposition [14], is not suitable for efficiently solving the elastic demand model. The reasons are related to the degeneration of the dual forms of (7.63), (7.73), from which the inferred Benders cuts are weak, in the sense that they do not constrain very much the variables w_1 and w_2. Thus, it requires an exorbitant number of Benders iterations to reach (near-) optimality.

This limitation can be overcome by applying pareto-optimality to the Benders cuts (7.85) - (7.86). The development of this theory was carried out by Magnanti & Wong [125], and practical improvements were performed by Papadakos [147]. A detailed explanation of both can be found in section 3.5.5 of chapter 3. Here, we will give only the final result for the sake of simplification. Let us denote $(y^0, \tilde{y}^0, \tilde{f}^0)$ as a point in the relative interior of the convex hull of set Y ($ri(Y^c)$), where Y is made of points satisfying all master problem constraints except (7.85) - (7.88). Then the pareto-optimal Benders cuts can be formulated by changing function (7.57) as follows:

$$h_k^n = \sum_{p \in O} \sum_{i \in N} \sum_{l \in L_i^N} \left[(c_{\pi_{i,p,l}^{k,n,0}} - c_{\gamma_{i,p,l}^{k,n,0}}) \tilde{y}_i^{l,0} - c_{\pi_{i,p,l}^{k,n,0}} y_i^{l,0} \right] + \sum_{w \in W} \left(c_{\xi_w^{k,n,0}} + c_{\nu_w^{k,n,0}} \right) - \sum_{(i,j) \in A_{TP}} \sum_{l \in L_{ij}} c_{\tau_{ij,l}^{k,n,0}} \tilde{f}_{ij}^{l,0} \tag{7.89}$$

The dual variables $\pi_{i,p,l}^{k,n,0}$, $\gamma_{i,p,l}^{k,n,0}$, $\xi_w^{k,n,0}$, $\nu_w^{k,n,0}$ and $\tau_{ij,l}^{k,n,0}$, which are used to compute the coefficients,

come from the resolution of the independent Magnanti & Wong Problem. This problem has the same mathematical structure of (7.54), and thus it can be solved by means of (7.73) - (7.83) and (7.63) - (7.72), but replacing $(y, \tilde{y}, f)$ with $(y^0, \tilde{y}^0, \tilde{f}^0)$.

Table 7.1 shows the interaction of these problems, where in steps 3 and 7 the independent Magnanti & Wong Problem and subproblem are solved in two phases. The point $(y^0, \tilde{y}^0, \tilde{f}^0) \in ri(Y^c)$ is updated by means of a linear convex combination of the current point and the variables obtained from resolving the last master problem(see step 11). Furthermore, we also include the McDaniel & Devine [136] enhancement and the ad hoc techniques presented in subsections 6.5.1 and 6.5.2, respectively, of chapter 6. The former allows us to quickly obtain valid Benders cuts by solving a relaxed master problem in order to decrease the number of integral master problem resolutions. Relaxation is done by dropping the integrality requirements in the discrete variables. The latter allows us to obtain a feasible solution (of the integral master problem) from the optimal solution found in that relaxed master problem, such that a feasible and possibly near optimal solution can be found after having finished phase 0.

1. **Set** $UB \leftarrow \infty$; iteration $k \leftarrow 0$; allowable error ϵ; and phase $\leftarrow 1$

2. **Find an Initial Core Point** $(y^0, \tilde{y}^0, \tilde{f}^0)$.

3. **Solve** $SP2^k$ $(7.63) - (7.72)$ **with** $(y^0, \tilde{y}^0, \tilde{f}^0)$.

   Solve $SP1^k$ $(7.73) - (7.83)$ **with** $(y^0, \tilde{y}^0, \tilde{f}^0),\ z^{*,k}_{IMWP2}$.

4. **Generate** h^n_k **with** $(\pi^{k,0}_n,\ \gamma^{k,0}_n,\ \tau^{k,0}_n,\ \xi^{k,0}_n,\ \nu^{k,0}_n),\ \ \forall\, n \in \{1, 2\}$.

5. **If** phase = 1 **then** continue, **otherwise** goto Step 6b.

6a. **Remove** integrality for variables $\tilde{y}^l_i$, δ^l_c, b^l, δ_k.

   Solve MP^k (7.84) **with** all $c \in \Lambda$.

   Activate integrality for variables $\tilde{y}^l_i$, δ^l_c, b^l, δ_k.

   Solve MP^k (7.84) **with** all $c \in \Lambda$ such that $\delta^l_c \geq \xi$, goto Step 7.

6b. **Solve** MP^k (7.84) **with** all $c \in \Lambda$.

7. $LB \leftarrow (1 - \alpha) \cdot z^k_{op} + \alpha \cdot (\omega^k_1 + \omega^k_2)$.

8. **Solve** $SP2^k$ $(7.63) - (7.72)$ **with** $(y^k, \tilde{y}^k, \tilde{f}^k)$.

   Solve $SP1^k$ $(7.73) - (7.83)$ **with** $(y^k, \tilde{y}^k, \tilde{f}^k),\ z^{*,k}_{SP2}$.

9. $UB \leftarrow \min\left\{UB, (1 - \alpha) \cdot z^k_{op} + z^k_{SP1} + z^k_{SP2}\right\}$.

10. **if** phase = 2 **and** $\dfrac{UB - LB}{LB} < \epsilon$ **then** STOP

    if phase = 1 **and** $\dfrac{UB - LB}{LB} < \epsilon$ **then** phase $\leftarrow 2$.

11. **Update** $(y^0, \tilde{y}^0, \tilde{f}^0) \leftarrow \lambda \cdot (y^0, \tilde{y}^0, \tilde{f}^0) + (1 - \lambda) \cdot (y^k, \tilde{y}^k, \tilde{f}^k)$

12. $k \leftarrow k + 1$ and goto Step 3.

Table 7.1: Pseudo-code of the Specialized Benders Scheme under elastic demand.

7.5 Preliminary computational results

In this section, we show some preliminary computational results to verify the model under elastic demand by means of the solving scheme shown in table 7.1 using $\lambda = 0.5$. Two test networks have been used. The first one is a complete 6-node railway network shown in Figure 8.1, whereas the other one is a 9-node and 26-link railway network depicted in Figure 8.2. Since the ad hoc techniques explained in subsection 6.5.2 need to use the routing submodel M3 (see section 2.3.3.3), this submodel has been chosen for carrying out these experiments. We refer the reader to section 8.1.1 for further details regarding the input parameters associated with both networks.

The first table 7.2 shows the general results . All mathematical programming formulations have been coded in AMPL and solved using a CPLEX v.12.4.0 solver in an R5500 working station with Intel(R) processor Xeon(R), CPU E5645 2.40 GHz and 48 Gbytes of RAM. In the table, the column **Network** denotes the test network used. Label $N1$ refers to the 6-node network and label $N2$ to the 9-node network. The next column, $|\boldsymbol{L}|$, stands for the maximum number of lines to be constructed, whereas column $|\boldsymbol{\Lambda}|$ reports the number of fixed corridors in the routing submodel $M3$. Column **Method** denotes the solving scheme used where BD refers to the Benders Decomposition scheme 7.1 without line splitting; $BD + LSA1$ relates to the Benders Decomposition driven by the Line Splitting Algorithm under the non-incremental load variant, and $BD + LSA2$ stands for the Benders Decomposition driven by the Line Splitting Algorithm under the incremental load variant. Column **F.Obj.** shows the best global objective function value found in each Benders phase. Column $\boldsymbol{T_{CPU}}$ reports the total number of elapsed seconds in each Benders phase for the LSA scheme used in the experiment. Column **Gap** $\%$ denotes the final relative gap percentage in each Benders phase, and the last column, **Iter,** shows the total number of iterations in each Benders phase. In those cells with two values separated by a hyphen, the left one refers to phase 0 of the Benders scheme 7.1 and the right one to phase 1.

Network	$\lvert\boldsymbol{L}\rvert$	$\lvert\boldsymbol{\Lambda}\rvert$	Method	F.Obj.	$\boldsymbol{T_{CPU}}$	Gap $\%$	Iter
N1	2	1967	BD	32463 - 30456	955 - 64569	0.15 - 0.55	198 - 108
			BD + LSA1	31062 - 31073	16 - 164	0.66 - 0.11	50 - 48
			BD + LSA2	31062 - 31073	23 - 85	0.68 - 0.31	66 - 34
	3	1967	BD	28934 - 28020	1897 - 85308	0.09 - 7.57	247 - 119
			BD + LSA1	30019 - 29357	15 - 24	0.73 - 0.27	68 - 52
			BD + LSA2	30507 - 29130	28811 - 33	0.72 - 0.27	3475 - 20
	4	1967	BD	34231 - ∞	86900 - 0	11 - ∞	292 - 0
			BD + LSA1	27500 - 27578	31 - 187	0.6 - 0.3	84 - 62
			BD + LSA2	27040 - 26258	21635 - 118	0.8 - ∞	3198 - 55
N2	2	295	BD	173373 - 173216	87 - 87157	0.8 - 2	64 - 381
			BD + LSA1	171723 - 170254	9 - 22	0.4 - 0.6	31 - 24
			BD + LSA2	171696 - 170671	61 - 391	0.7 - 0.07	124 - 124
	3	295	BD	172125 - 169741	35726 - 51413	1.9 - 6.8	314 - 99
			BD + LSA1	168754 - 168477	15 - 24	0.3 - 0.5	50 - 28
			BD + LSA2	171091 - 168863	16 - 327	0.6 - 0.6	53 - 106
	4	295	BD	173782 - ∞	87208 - 0	1.3 - ∞	488 - 0
			BD + LSA1	167805 - 167671	16 - 61	0.3 - 0.6	52 - 47
			BD + LSA2	167680 - 167293	19 - 410	0.5 - 0.6	66 - 155

Table 7.2: General results for the experiments performed in the test networks.

Results show that only instances with two lines under construction can be solved to optimality without

using the Line Splitting Algorithm (LSA). However, it takes 18.2 hours to solve the 6-node network instance, whereas the 9-node network instance reaches the time limit of 24 hours with a relative gap of 2 % in phase 1. As for the BD driven by the LSA, it seems more appropriate to use the non-incremental variant since its solving time is much lower than the incremental variant while maintaining practically the same good solution. The gap reported by the LSA for these instances is the average sum of the final Benders gap at each LSA iteration. Focusing now on the Benders phases, phase 0 gives a solution which is close to the one found in phase 2, no matter what the solving scheme is. Furthermore, the elapsed time is clearly much lower than the one of the phase 1.

The following table 7.3 holds the time distribution among the three problems involved in the Benders Decomposition, where new columns $\boldsymbol{T_{MP}}$, $\boldsymbol{T_{SP}}$ and $\boldsymbol{T_{IMWP}}$ stand for the time consumed in solving the master problem (MP), the subproblem (SP) and the Independent Magnanti & Wong ($IMWP$), respectively. We can see that practically all the CPU time is spent on the MP resolution for both Benders phases.

Network	$\|\boldsymbol{L}\|$	**Method**	$\boldsymbol{T_{MP}}$	$\boldsymbol{T_{SP}}$	$\boldsymbol{T_{IMWP}}$	$\boldsymbol{T_{CPU}}$
N1	2	BD	930 - 64552	8 - 4	17 - 13	955 - 64569
		BD + LSA1	11 - 159	2 - 2	4 - 3	16 - 164
		BD + LSA2	16 - 82	2 - 1	5 - 2	23 - 85
	3	BD	1867 - 85280	10 - 4	21 - 24	1897 - 85308
		BD + LSA1	6 - 19	3 -2	6 - 3	15 - 24
		BD + LSA2	28565 - 30	121 - 1	125 - 2	28811 - 33
	4	BD	86856 - 0	10 - 0	34 - 0	86900 - 0
		BD + LSA1	19 - 179	4 -3	7 - 5	31 - 187
		BD + LSA2	21363 - 110	133 - 3	139 - 5	21635 - 118
N2	2	BD	74 - 87103	3 - 15	10 - 39	87 - 87157
		BD + LSA1	4 - 18	1 - 1	4 - 3	9 - 22
		BD + LSA2	43 - 370	6 - 8	12 - 13	61 - 391
	3	BD	35654 - 51385	15 - 4	57 - 23	35726 - 51413
		BD + LSA1	6 - 19	3 - 2	6 - 3	15 - 24
		BD + LSA2	6 - 306	3 - 7	7 - 15	16 - 327
	4	BD	87092 - 0	22 - 0	93 - 0	87208 - 0
		BD + LSA1	6 - 50	3 - 4	7 - 7	16 - 61
		BD + LSA2	9 - 380	3 - 9	7 - 21	19 - 410

Table 7.3: Details of the time distribution for elastic demand in the test networks.

Next table 7.4 shows the time distribution within the MP resolution for phase 0 where new columns $\boldsymbol{T_{MP1}}$ and $\boldsymbol{T_{MP2}}$ stand for the time spent in solving the relaxed and the reduced master problems, respectively. Results show that in all instances where the total MP solving time is significantly high (see last column), the reduced MP takes much more time to solve than the relaxed MP.

To conclude the performance analysis, we include tables 7.5 and 7.6 to see the time distribution in the SP and the $IMWP$ problems. On table 7.5, new columns $\boldsymbol{T_{SP1}}$, $\boldsymbol{T_{SP2}}$, denote the computing times in seconds of the subproblems (7.73) - (7.83) and (7.63) - (7.72), respectively. Analogously, in table 7.6, new columns $\boldsymbol{T_{IMWP1}}$ and $\boldsymbol{T_{IMWP2}}$ refer to (7.73) - (7.83) and (7.63) - (7.72), respectively, but they are parameterized by the core point $(y^0, \tilde{y}^0, \tilde{f}^0)$.

Results of table 7.5 show that the SP solving time is higher in phase 0 and that both subproblems consume a similar amount of time in both phases. Regarding the results in table 7.6, they do not clarify in which phase the $IWMP$ solving time is higher, but the $IWMP1$ seems to take more time to be solved.

Network	$\|L\|$	Method	T_{MP1}	T_{MP2}	T_{MP}
N1	2	BD	59.4	860.6	930
		BD + LSA1	5.5	5.5	11
		BD + LSA2	6	10	16
	3	BD	153	1714	1867
		BD + LSA1	5	1	6
		BD + LSA2	3094	25471	28565
	4	BD	292	86564	86856
		BD + LSA1	9	10	19
		BD + LSA2	4503	16860	21363
N2	2	BD	4.5	69.5	74
		BD + LSA1	1	3	4
		BD + LSA2	5	38	43
	3	BD	107	35547	35654
		BD + LSA1	1	5	6
		BD + LSA2	1	5	6
	4	BD	273	86819	87092
		BD + LSA1	1.1	4.9	6
		BD + LSA2	1.7	7.3	9

Table 7.4: Master problem time distribution for the first phase under elastic demand for the test networks.

Network	$\|L\|$	Method	T_{SP1}	T_{SP2}	T_{SP}
N1	2	BD	4.3 - 0.75	3.7 - 3.25	8 - 4
		BD + LSA1	1 - 1	1 - 1	2 - 2
		BD + LSA2	1 - 0.5	1 - 0.5	2 - 1
	3	BD	5 - 2	5 - 2	10 - 4
		BD + LSA1	1 - 0.5	2 - 1.5	3 - 2
		BD + LSA2	70 - 0.5	51 - 0.5	121 - 1
	4	BD	6 - 0	4 - 0	10 - 0
		BD + LSA1	2 - 1.5	2 - 1.5	4 - 3
		BD + LSA2	64 - 1.6	69 - 1.4	133 - 3
N2	2	BD	1.3 - 7.6	2.7 - 7.4	3 - 15
		BD + LSA1	0.5 - 0.5	0.5 - 0.5	1 - 1
		BD + LSA2	3.6 - 4.1	2.4 - 3.9	6 - 8
	3	BD	8 - 2.5	7 - 1.5	15 - 4
		BD + LSA1	1.5 - 1	1.5 - 1	3 - 2
		BD + LSA2	1.3 - 3.3	1.7 - 3.7	3 - 7
	4	BD	10 - 0	12 - 0	22 - 0
		BD + LSA1	1.4 - 1.9	1.6 - 2.1	3 - 4
		BD + LSA2	1.7 - 5	1.3 - 4	3 - 9

Table 7.5: Subproblem time distribution under elastic demand for the test networks.

Network	$\|L\|$	Method	T_{IMWP1}	T_{IMWP2}	T_{IMWP}
N1	2	BD	11.5 - 4	5.5 - 9	17 - 13
		BD + LSA1	2.5 - 2	1.5 - 1	4 - 3
		BD + LSA2	3 - 1	2 - 1	5 - 2
	3	BD	15 - 5	6 - 19	21 - 24
		BD + LSA1	2 - 1	4 - 2	6 - 3
		BD + LSA2	70 - 1.1	55 - 0.9	125 - 2
	4	BD	23 - 0	11 - 0	34 - 0
		BD + LSA1	4 - 3	3 - 2	7 - 5
		BD + LSA2	66 - 73	3.1 - 1.9	139 - 5
N2	2	BD	7 - 23	3 - 16	10 - 39
		BD + LSA1	2.2 - 1.5	1.7 - 1.5	4 - 3
		BD + LSA2	6.7 - 7.5	5.3 - 5.5	12 - 13
	3	BD	37 - 15	20 - 8	57 - 23
		BD + LSA1	3.5 - 1.7	2.5 - 1.3	6 - 3
		BD + LSA2	4.1 - 8.5	2.9 - 6.5	7 - 15
	4	BD	58.5 - 0	34.5 - 0	93 - 0
		BD + LSA1	4 - 3.5	3 - 3.5	7 - 7
		BD + LSA2	5.1 - 13	1.9 - 8	7 - 21

Table 7.6: Independent Magnanti & Wong problem time distribution under elastic demand for the test networks.

Moving on to an analysis of the solution, Figures 7.3 - 7.5 show the routing configuration for all instances performed with the 6-node and 9-node networks with the inelastic and elastic demand variants of the model. In this way, we can carry out a comparison of both demand variants. First, Figure 7.3 depicts those solutions related to the 6-node network with two and three lines under construction. We can see that the line segment configurations are the same for the two demand variants, and that lines $L1$ and $L2$ coincide in all instances (i.e., $L1 : 2-5-6-4-5-2$ and $L2 : 1-3-5-2-6-4-1$). However, regarding the role of the nodes, some differences appear. For instance, line $L2$ of the instance with two lines under construction using elastic demand has nodes 3 and 4, which do not provide service. The same line in the instance with three lines under construction using elastic demand has nodes 5 and 4, which do not perform service as well.

Next, Figure 7.4 shows those solutions related to the 6-node network with four lines under construction and to the 9-node network with two lines under construction. Focusing first on the instances associated with the 6-node network, we can see that there is no coincidence among the line's segment configurations in the demand variants, although the number of topological combinations is the same: three rectilinear lines and one circular (line $L4$ in the inelastic variant and line $L2$ in the elastic one). Furthermore, all nodes performed service no matter what the demand variant is. Moving on to the instance related to the 9-node network, we can see that line segments coincide partially in both demand variants: line $L1$ is the same (i.e., $4-6-9$) whereas $L2$ is completely different ($1-2-3-5-4-7-6-8$ in the inelastic demand variant and $1-3-5-4-7$ for the elastic one). Moreover, in the elastic demand variant, node 6 does not perform service on line $L1$, and nodes 3 and 4 do not carry out service on line $L2$.

Last, Figure 7.5 depicts those solutions related to the 9-node network with three and four lines under construction. We can see that in instances with three lines under construction, lines $L1$ and $L2$ have the same line segments and the nodes play the same role as in lines $L1$ and $L2$ in instances with two lines under construction (see previous Figure 7.4). However line $L3$ is completely different (i.e., $L3 : 1-2-3-5-4-7-6-8$ in the elastic demand variant and $L3 : 7-4-2-1-3-5-6-8$ in the inelastic demand one). Moreover, in the elastic demand variant, half of the involved nodes do not perform service $(2, 4, 5, 6)$. As for the instances

with four lines under construction, there is no coincidence regarding the line segment configurations as well as the node's role (i.e., $L1:4-6-9$, $L2:2-3-5-4$, $L3:1-3-2-4-7$ and $L4:2-1-3-4-7-6-8$ with the inelastic demand variant, whereas $L1:1-3-5-4-6-9$, $L2:7-4-6-9$, $L3:5-6-8$ and $L4:9-6-7-4-3-2-1$ with the inelastic demand variant). Moreover, in the elastic demand variant, node 6 on line $L1$ does not perform service, nor do nodes 3 and 4 on line $L4$.

Regarding the analysis of the planning solution, table 7.7 gives the details of using the planning resources for the inelastic and elastic demand models, allowing a comparison of their solutions for the same instances. The solution data is taken from the best solution found among the various solving schemes performed. In the table, the column **Demand** specifies the demand model used, whereas the columns $\boldsymbol{NL}$ and $\boldsymbol{NLC}$ report the number of built lines and the ones that have a circular topology. The next two columns, $\boldsymbol{NV}$ and $\boldsymbol{NZ}$, stand for the number of assigned vehicles and their corresponding services per horizon time carried out. The remaining columns report some measures regarding the level of line utilization. The first two columns ($\overline{\boldsymbol{U}}_{\boldsymbol{a}}$ and $\hat{\boldsymbol{U}}_{\boldsymbol{a}}$) show the average and maximum used capacity of the links. This capacity corresponds to input parameter q_{ij}, which stands for the maximum number of vehicles per time unit allowed through the link (i,j). The other two columns ($\overline{\boldsymbol{U}}_{\boldsymbol{l}}$ and $\hat{\boldsymbol{U}}_{\boldsymbol{l}}$) refer to the capacity of the average and maximum lines used, computed as the total number of vehicles per time unit carried out, f^l, times the maximum vehicle capacity, q_l.

Results show that the elastic demand variant uses one less vehicle than the inelastic demand variant. However, the number of services carried out is quite similar, excluding the instance with four lines under construction for both networks. In the variant related to the 6-node network, vehicles perform more services under elastic demand. With the 9-node network, vehicles perform more services under inelastic demand. As for the utilization indicators, they are both rather similar under elastic demand, except for the $\overline{\boldsymbol{U}}_{\boldsymbol{a}}$ measure, which differs significantly in some instances.

Next, Table 7.8 contains the modal demand splitting solutions for the inelastic and elastic demand variants. New columns $\boldsymbol{G^{TP}}$, $\boldsymbol{G^{COM}}$ and $\boldsymbol{G^{PRI}}$ report the portions of demand assigned to the modes of public transportation, complementary and private, respectively; whereas column $\boldsymbol{G}$ contains the total amount of demand in absolute values (i.e., $G^{TP}+G^{COM}+G^{PRI}$). Results show that in the elastic demand variant, passengers do not use the complementary networks at all and instead prefer to go by private mode. Furthermore, the number of users going through the public transportation network drops dramatically, although it increases with the number of lines under construction.

Complementary to this table, we provide the following table 7.9, which shows the details of the demand assignment throughout the public transportation network for the inelastic and elastic demand variants. The new column $\boldsymbol{G_0^{TP}}$ stands for the total amount of demand that does not use any complementary links (no walking), whereas column $\boldsymbol{G_n^{TP}}$ shows the demand throughout the complementary links. The last column $\boldsymbol{G^{TP}}$ denotes the total demand assigned to the public transportation network in absolute values. Results show that, in both demand variants, complementary links are seldom used. Thus, the demand is served directly by the public transportation lines.

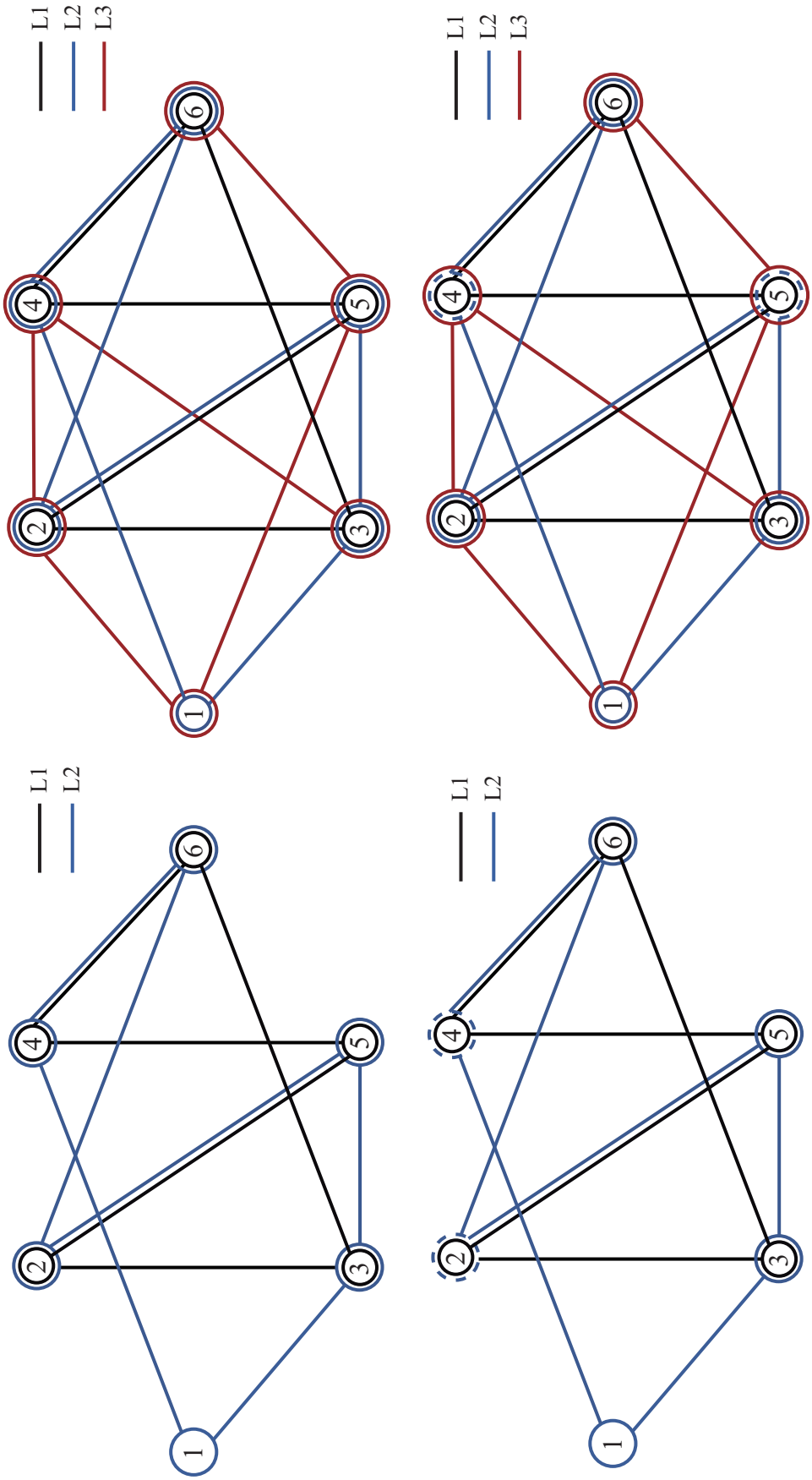


Figure 7.3: Layout configuration for the 6-node network with two and three lines under construction. Top-left: the layout for the model under inelastic demand with two lines under construction. Bottom-left: the layout for the the model under elastic demand with two lines under construction. Top-right: the layout for the model under inelastic demand with three lines under construction. Bottom-right: the layout for the the model under elastic demand with three lines under construction.

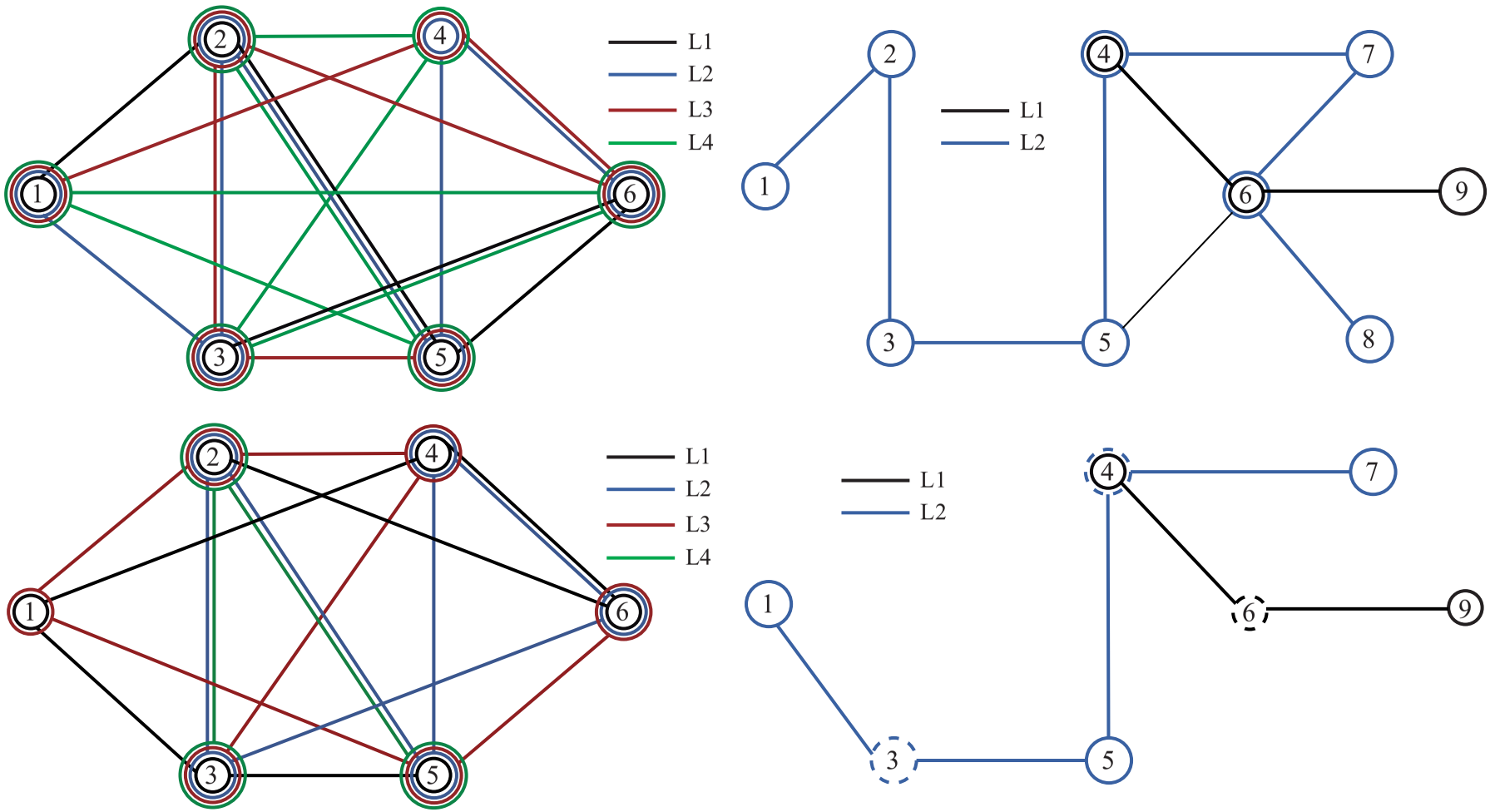


Figure 7.4: Layout configuration for the 6-node and 9-node networks. Top-left: the layout for the model under inelastic demand with four lines under construction for the 6-node network. Bottom-left: the layout for the the model under elastic demand with four lines under construction for the 6-node network. Top-right: the layout for the model under inelastic demand with two lines under construction for the 9-node network. Bottom-right: the layout for the the model under elastic demand with two lines under construction for the 9-node network.

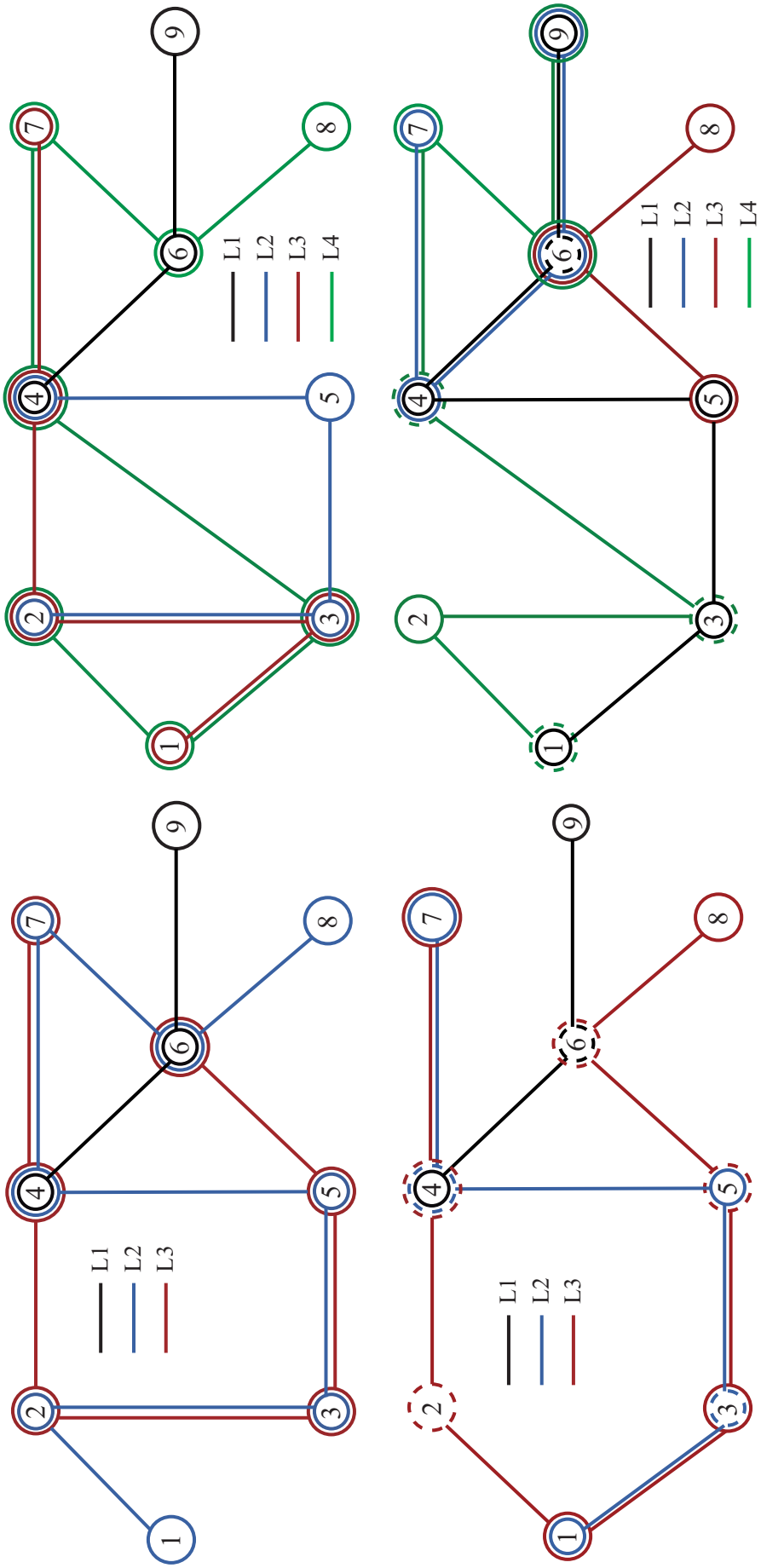


Figure 7.5: Layout configuration for the 9-node network with three and four lines under construction. Top-left: the layout for the model under inelastic demand with three lines under construction. Bottom-left: the layout for the the model under elastic demand with three lines under construction. Top-right: the layout for the model under inelastic demand with four lines under construction. Bottom-right: the layout for the the model under elastic demand with four lines under construction.

Network	$\|L\|$	Demand	NL	NLC	NV	NZ	$\overline{U}_{ij}$	$\hat{U}_{ij}$	$\overline{U}_{l}$	$\hat{U}_{l}$
N1	2	Inelastic	2	2	5	36	77 %	100 %	100 %	100 %
		Elastic	2	2	4	36	77 %	100 %	100 %	100 %
	3	Inelastic	3	2	6	45	81 %	100 %	100 %	100 %
		Elastic	3	2	5	45	81 %	100 %	100 %	100 %
	4	Inelastic	4	1	6	54	70 %	100 %	100 %	100 %
		Elastic	4	1	6	63	79 %	100 %	100 %	100 %
N2	2	Inelastic	2	0	4	45	69 %	100 %	100 %	100 %
		Elastic	2	2	4	54	100 %	100 %	100 %	100 %
	3	Inelastic	3	0	5	54	66 %	100 %	100 %	100 %
		Elastic	3	3	4	54	67 %	100 %	100 %	100 %
	4	Inelastic	4	0	6	72	75 %	100 %	100 %	100 %
		Elastic	4	4	5	48	67 %	100 %	93 %	100 %

Table 7.7: Details of resources used for inelastic and elastic demand in the test networks.

Network	$\|L\|$	Demand	NL	G^{TP}	G^{COM}	G^{PRI}	G
N1	2	Inelastic	2	80 %	20 %	0 %	177000
		Elastic	2	30 %	0 %	70 %	
	3	Inelastic	3	81 %	19 %	0 %	
		Elastic	3	40 %	0 %	60 %	
	4	Inelastic	4	80 %	20 %	0 %	
		Elastic	4	52 %	0 %	48 %	
N2	2	Inelastic	2	91 %	9 %	0 %	373000
		Elastic	2	7 %	0 %	93 %	
	3	Inelastic	3	91 %	9 %	0 %	
		Elastic	3	8 %	0 %	92 %	
	4	Inelastic	4	91 %	9 %	0 %	
		Elastic	4	12 %	0 %	88 %	

Table 7.8: Details of modal demand splitting for inelastic and elastic demand in the test networks.

Network	$\|L\|$	Demand	NL	G_0^{TP}	G_n^{TP}	G^{TP}
N1	2	Inelastic	2	78 %	2 %	80 %
		Elastic	2	30 %	0 %	30 %
	3	Inelastic	3	79 %	2 %	81 %
		Elastic	3	40 %	0 %	40 %
	4	Inelastic	4	79.7 %	0.3 %	80 %
		Elastic	4	52 %	0 %	52 %
N2	2	Inelastic	2	88 %	2 %	91 %
		Elastic	2	6.98 %	0.02 %	7 %
	3	Inelastic	3	90 %	1 %	91 %
		Elastic	3	7.95 %	0.05 %	8 %
	4	Inelastic	4	90 %	1 %	91 %
		Elastic	4	11.83 %	0.17 %	12 %

Table 7.9: Details of the demand assignment throughout the public transportation network for inelastic and elastic demand in the test networks.

We conclude this section by showing the evolution of the Benders relative gap and the global objective function (GOF) under the different solving schemes and for all the experiments carried out in the 6-node and 9-node networks throughout figures 7.6 - 7.43. Roughly speaking, we can observe that the GOF evolves rather hesitantly in all situations, whereas the gap evolution has many plateaus in the centered regions. Despite these irregularities, the Benders gap, as well as the GOF, converge to a value as long as the time horizon is wide enough. This condition does not hold, for instance, in the experiments with three and four lines under construction on both networks, which are solved by means of the BD without LSA. Furthermore, in the first iteration of the LSA incremental load variant, this also happens in the same experiments for the 6-node network. This is due to its non-polinomial time performance. It seems that the LSA with a non-incremental variant shows a certain regularity: the convergence rate is similar in every LSA iteration, unlike the non-incremental variant where the first LSA iteration takes a lot of time to converge.

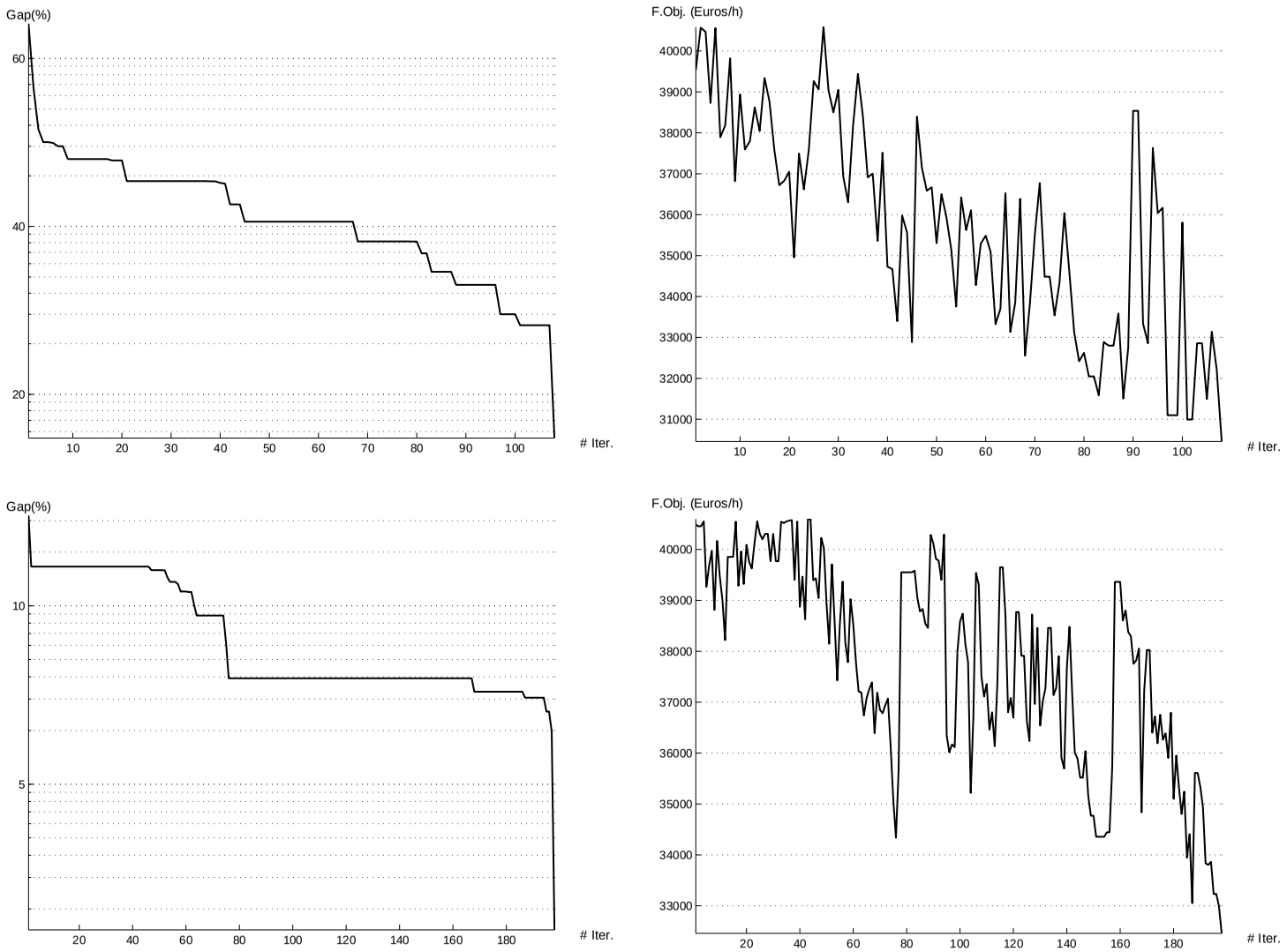


Figure 7.6: Graphs for evolution of the Benders scheme applied to the model under elastic demand with the 6-node network and two lines under construction. Top-left: evolution of the Benders gap for the first phase. Bottom-left: evolution of the Benders gap for the second phase. Top-right: evolution of the global objective function for the first phase. Bottom-right: evolution of the global objective function for the second phase.

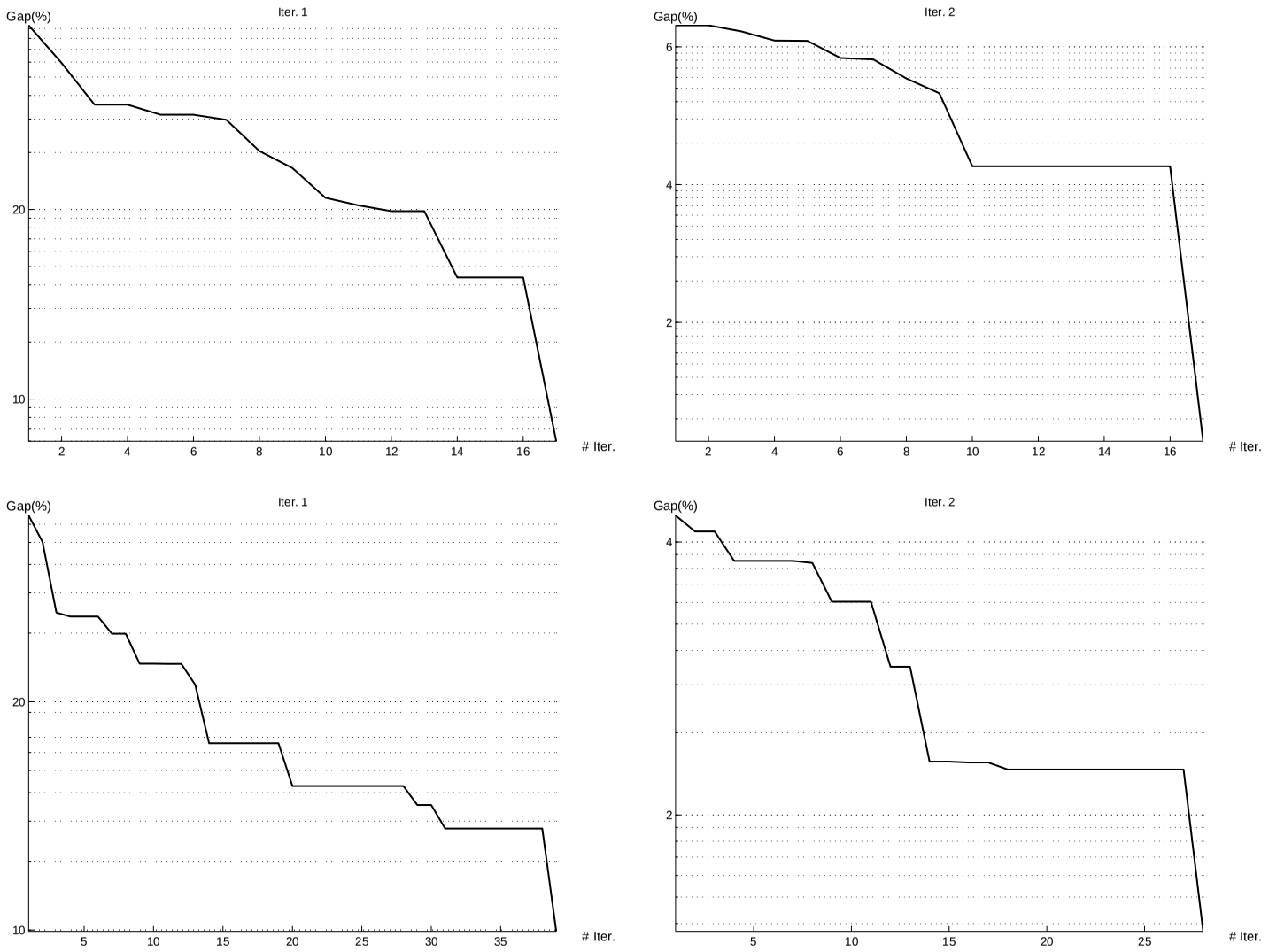


Figure 7.7: Graphs for evolution of the Benders scheme (BD) driven by the line splitting algorithm (LSA) without incremental load variant applied to the model under elastic demand with the 6-node network and two lines under construction. Top-left: gap evolution for the first iteration of the LSA in the first phase of the BD. Bottom-left: gap evolution for the first iteration of the LSA in the second phase of the BD. Top-right: gap evolution for the second iteration of the LSA in the first phase of the BD. Bottom-right: gap evolution for the second iteration of the LSA in the second phase of the BD.

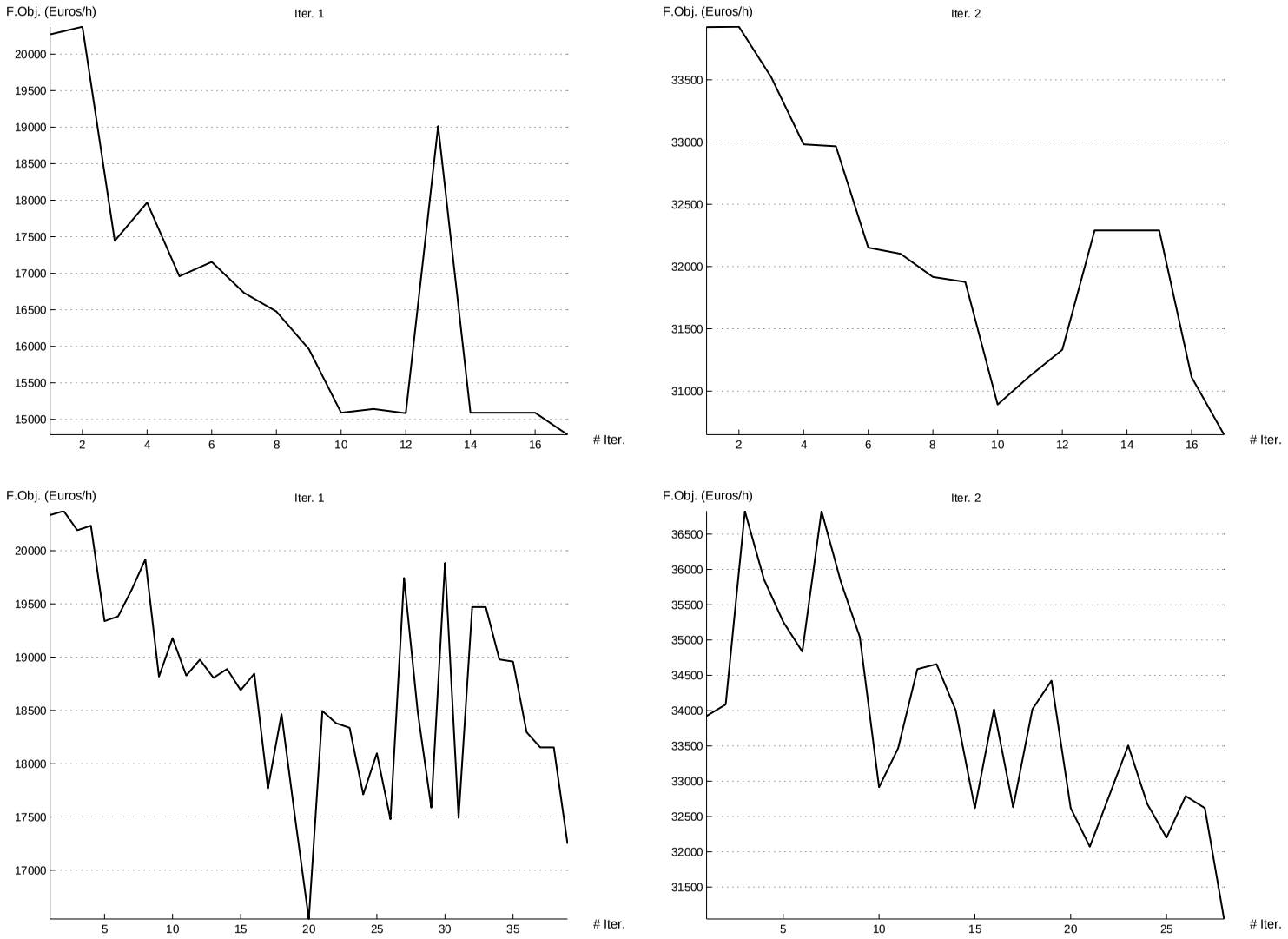


Figure 7.8: Graphs showing the evolution of the objective function for the Benders scheme (BD) driven by the line splitting algorithm (LSA) without incremental load variant applied to the model under elastic demand with the 6-node network and two lines under construction. Top-left: objective function evolution for the first iteration of the LSA in the first phase of the BD. Bottom-left: objective function evolution for the first iteration of the LSA in the second phase of the BD. Top-right: objective function evolution for the second iteration of the LSA in the first phase of the BD. Bottom-right: objective function evolution for the second iteration of the LSA in the second phase of the BD.

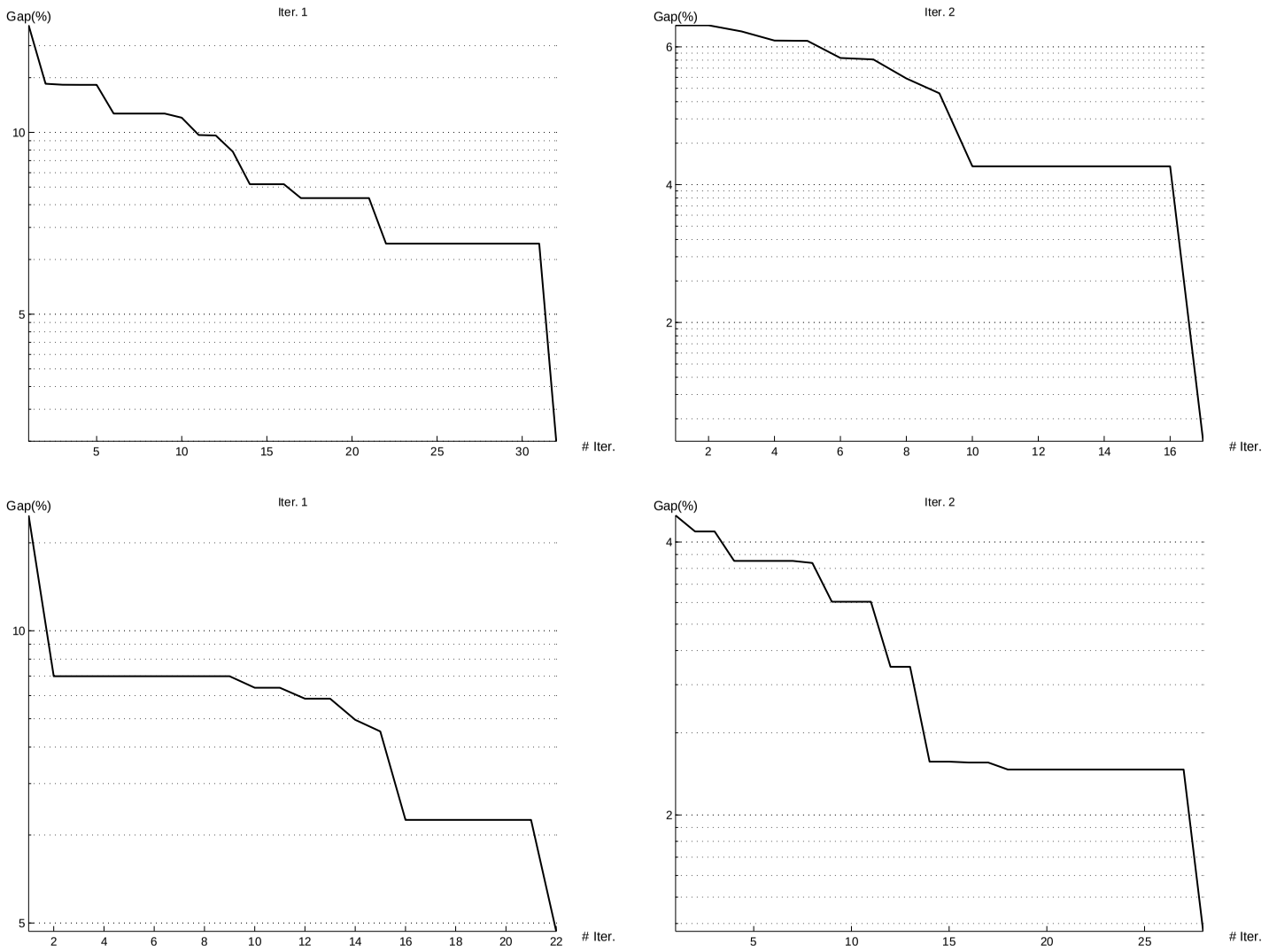


Figure 7.9: Graphs for evolution of the Benders scheme (BD) driven by the line splitting algorithm (LSA) with incremental load variant applied to the model under elastic demand with the 6-node network and two lines under construction. Top-left: gap evolution for the first iteration of the LSA in the first phase of the BD. Bottom-left: gap evolution for the first iteration of the LSA in the second phase of the BD. Top-right: gap evolution for the second iteration of the LSA in the first phase of the BD. Bottom-right: gap evolution for the second iteration of the LSA in the second phase of the BD.

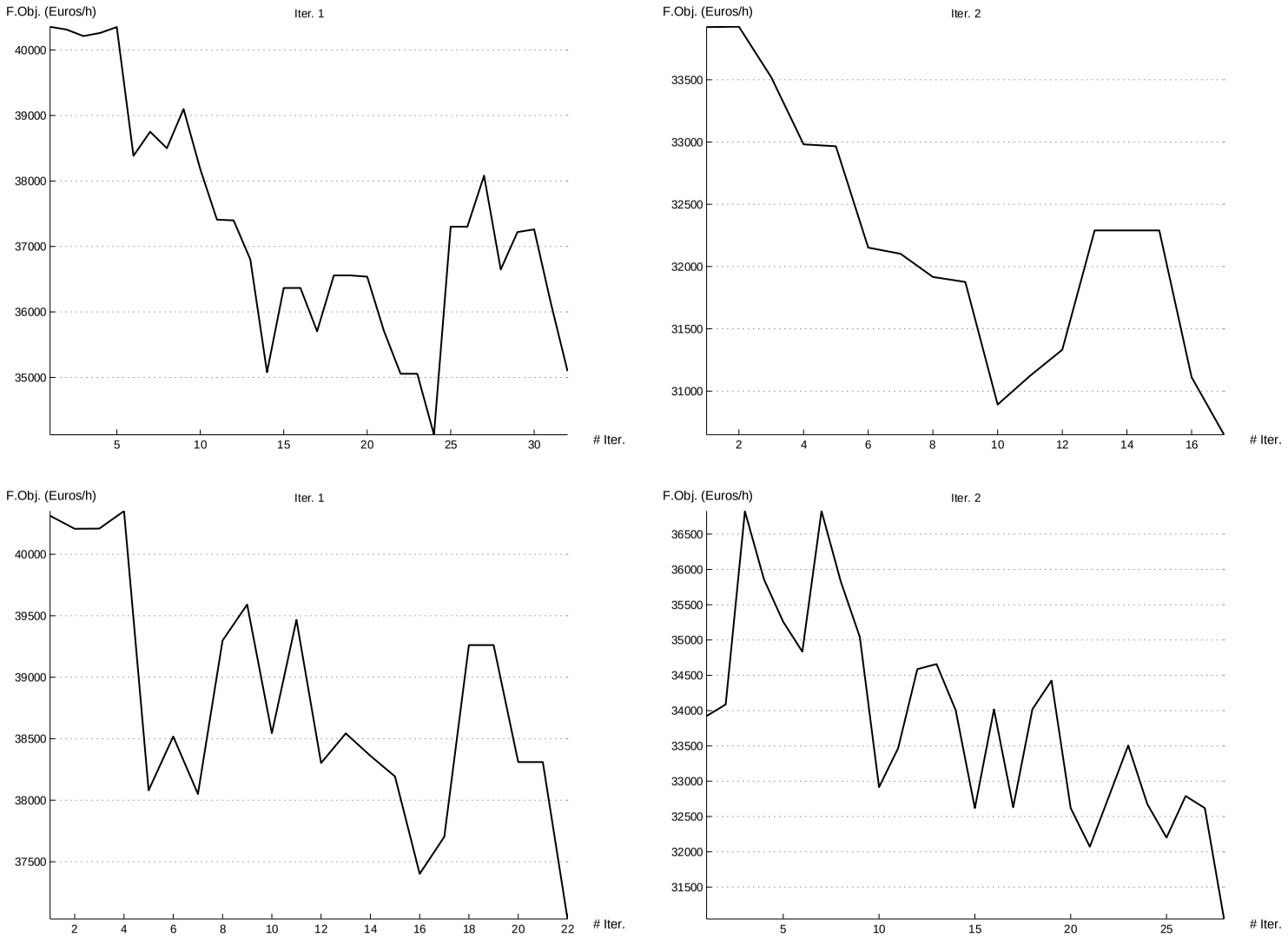


Figure 7.10: Graphs showing the evolution of the objective function for the Benders scheme (BD) driven by the line splitting algorithm (LSA) with incremental load variant applied to the model under elastic demand with the 6-node network and two lines under construction. Top-left: objective function evolution for the first iteration of the LSA in the first phase of the BD. Bottom-left: objective function evolution for the first iteration of the LSA in the second phase of the BD. Top-right: objective function evolution for the second iteration of the LSA in the first phase of the BD. Bottom-right: objective function evolution for the second iteration of the LSA in the second phase of the BD.

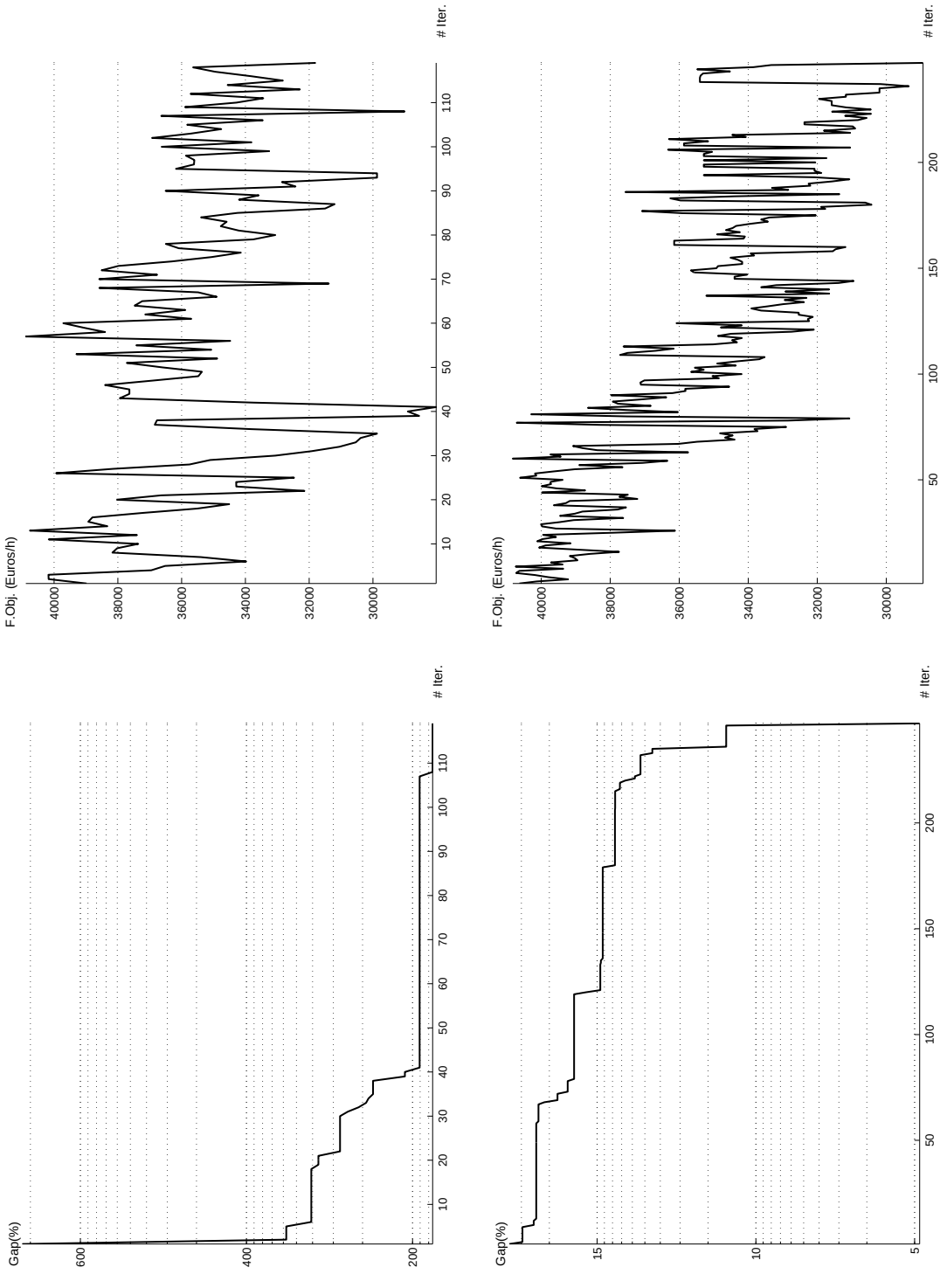


Figure 7.11: Graphs showing the evolution of the objective function for the Benders scheme applied to the model under elastic demand with the 6-node network and three lines under construction. Top-left: evolution of the Benders gap for the first phase. Bottom-left: evolution of the Benders gap for the second phase. Top-right: evolution of the global objective function for the first phase. Bottom-right: evolution of the global objective function for the second phase.

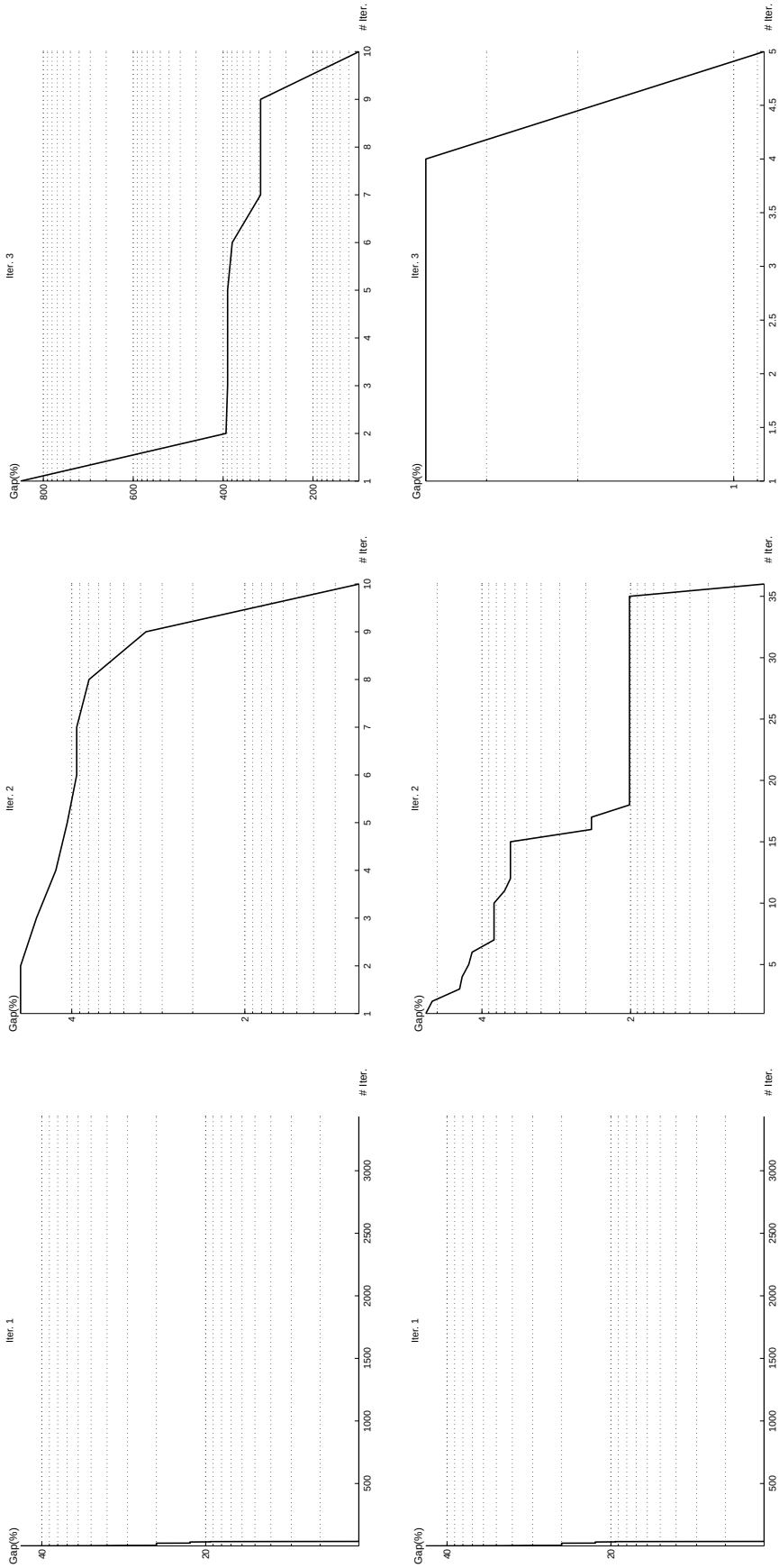


Figure 7.12: Graphs for evolution of the Benders scheme (BD) driven by the line splitting algorithm (LSA) without incremental load variant applied to the model under elastic demand with the 6-node network and three lines under construction. Top-left: gap evolution for the first iteration of the LSA in the first phase of the BD. Bottom-left: gap evolution for the first iteration of the LSA in the second phase of the BD. On the top-center, gap evolution for the third iteration of the LSA in the first phase of the BD. On the bottom-center, gap evolution for the third iteration of the LSA in the second phase of the BD. Top-right: gap evolution for the third iteration of the LSA in the first phase of the BD. Bottom-right: gap evolution for the third iteration of the LSA in the second phase of the BD.

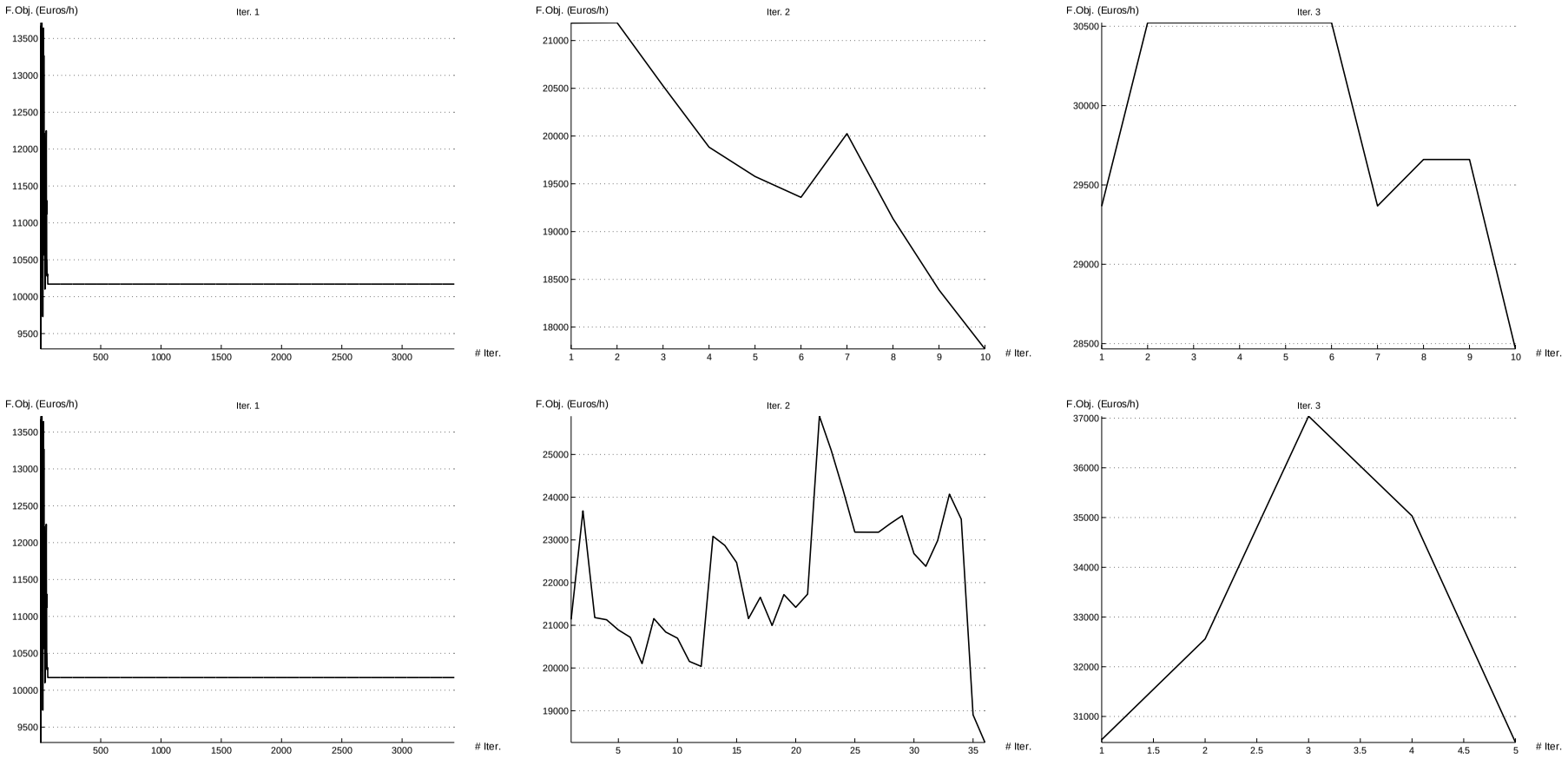
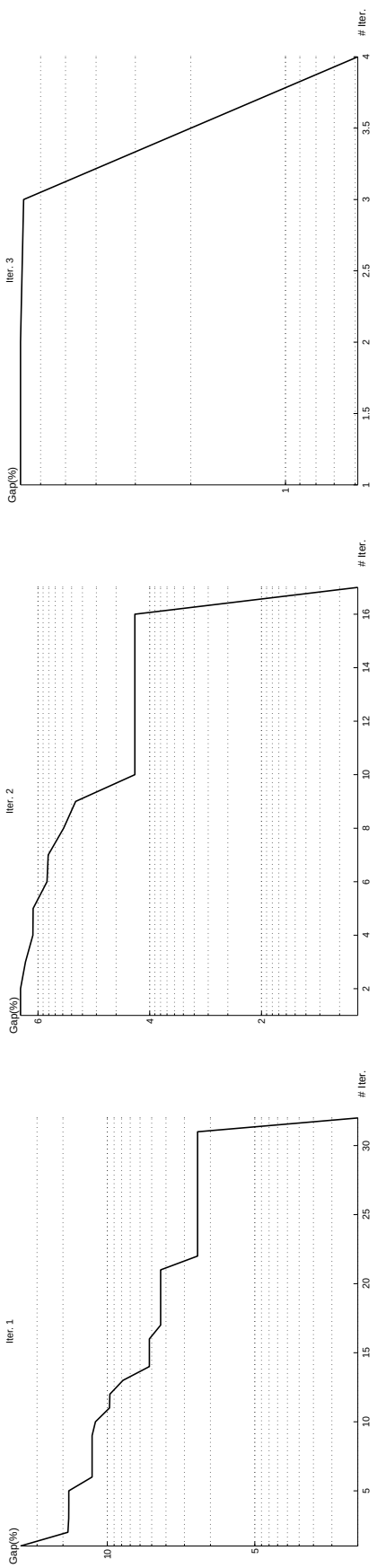
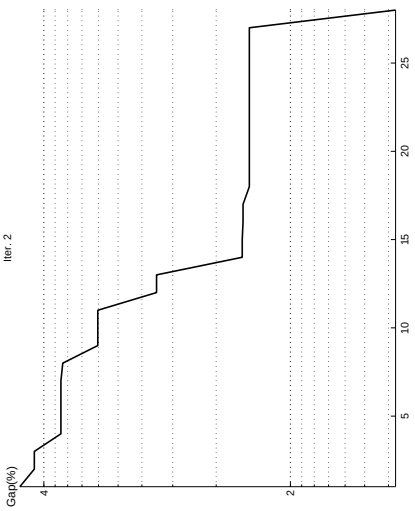


Figure 7.13: Graphs showing the evolution of the objective function for the Benders scheme (BD) driven by the line splitting algorithm (LSA) without incremental load variant applied to the model under elastic demand with the 6-node network and three lines under construction. Top-left: objective function evolution for the first iteration of the LSA in the first phase of the BD. Bottom-left: objective function evolution for the first iteration of the LSA in the second phase of the BD. On the top-center, objective function evolution for the third iteration of the LSA in the first phase of the BD. On the bottom-center, objective function evolution for the third iteration of the LSA in the second phase of the BD. Top-right: objective function evolution for the third iteration of the LSA in the first phase of the BD. Bottom-right: objective function evolution for the third iteration of the LSA in the second phase of the BD.





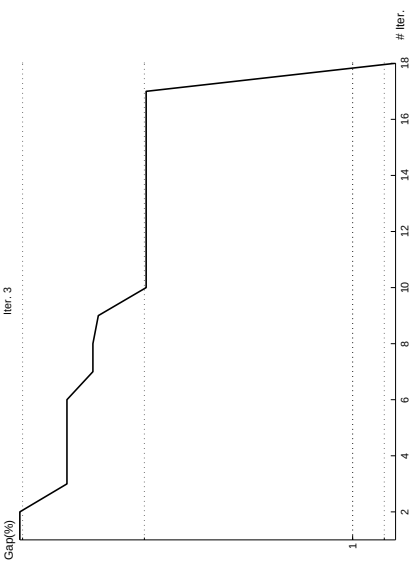


Figure 7.14: Graphs for evolution of the Benders scheme (BD) driven by the line splitting algorithm (LSA) with incremental load variant applied to the model under elastic demand with the 6-node network and three lines under construction. Top-left: gap evolution for the first iteration of the LSA in the first phase of the BD. Bottom-left: gap evolution for the first iteration of the LSA in the second phase of the BD. On the top-center, gap evolution for the third iteration of the LSA in the first phase of the BD. On the bottom-center, gap evolution for the third iteration of the LSA in the second phase of the BD. Top-right: gap evolution for the third iteration of the LSA in the first phase of the BD. Bottom-right: gap evolution for the third iteration of the LSA in the second phase of the BD.

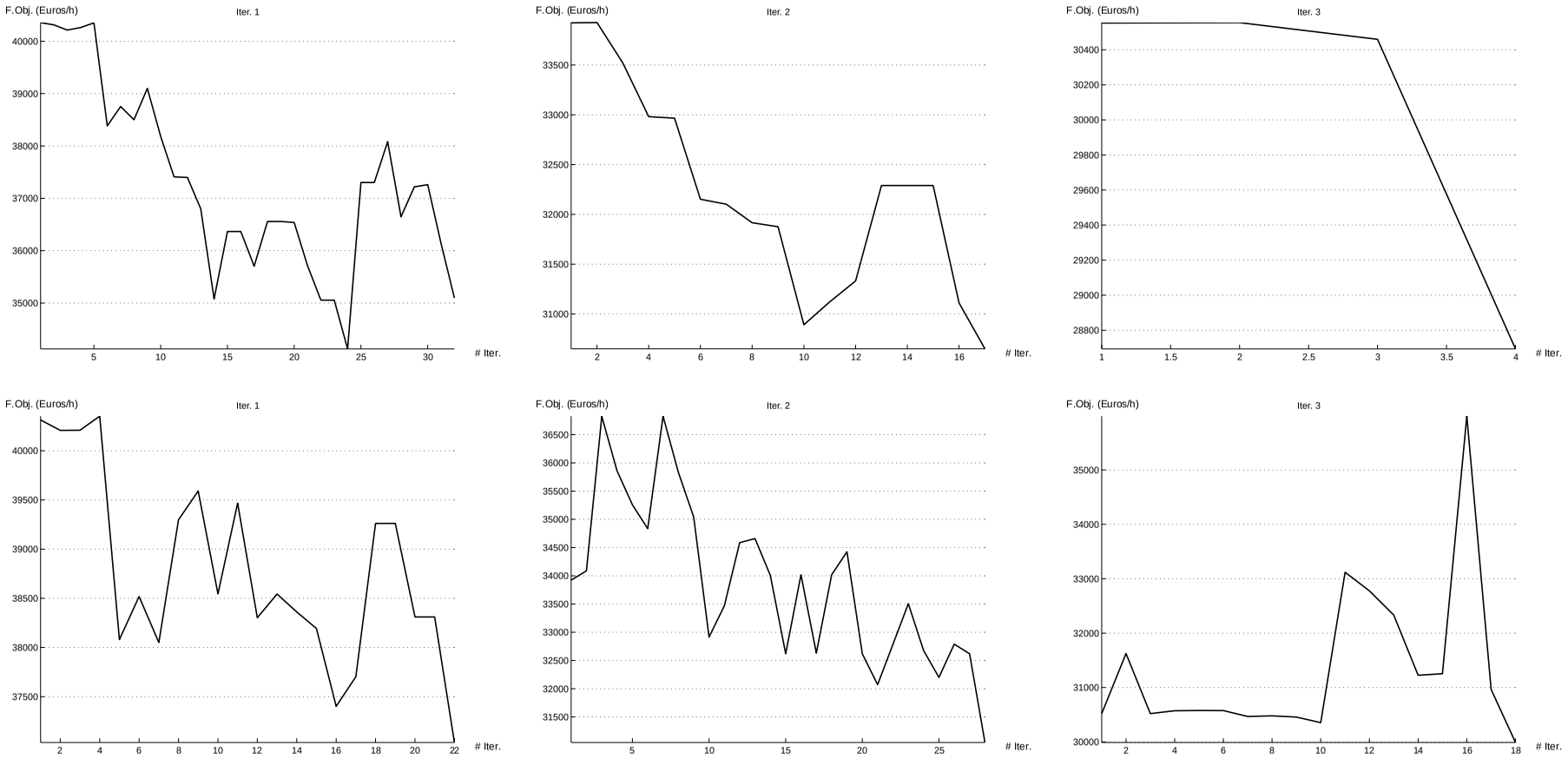


Figure 7.15: Graphs showing the evolution of the objective function for the Benders scheme (BD) driven by the line splitting algorithm (LSA) with incremental load variant applied to the model under elastic demand with the 6-node network and three lines under construction. Top-left: objective function evolution for the first iteration of the LSA in the first phase of the BD. Bottom-left: objective function evolution for the first iteration of the LSA in the second phase of the BD. On the top-center, objective function evolution for the third iteration of the LSA in the first phase of the BD. On the bottom-center, objective function evolution for the third iteration of the LSA in the second phase of the BD. Top-right: objective function evolution for the third iteration of the LSA in the first phase of the BD. Bottom-right: objective function evolution for the third iteration of the LSA in the second phase of the BD.

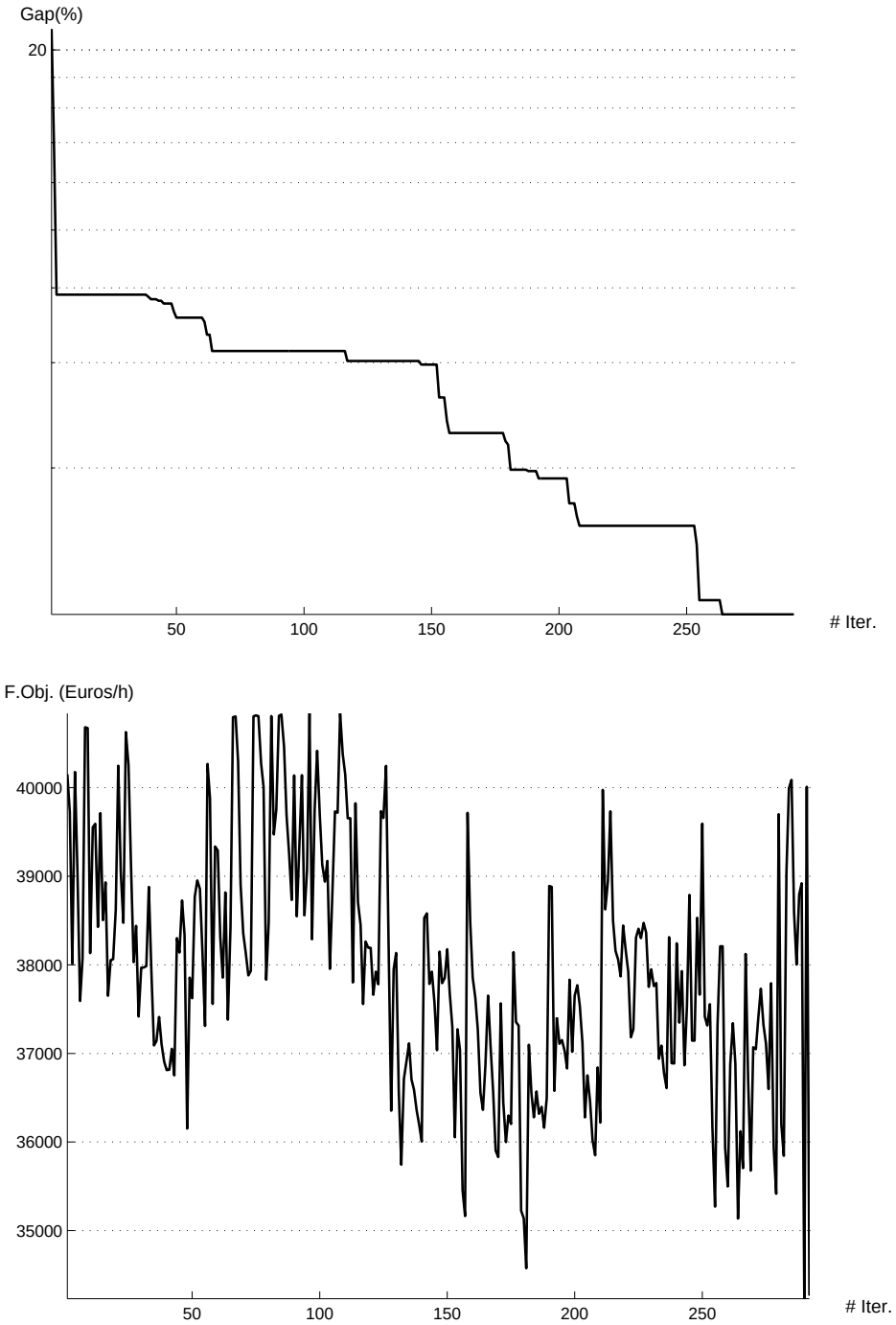


Figure 7.16: Graphs for evolution of the Benders scheme applied to the model under elastic demand with the 6-node network and four lines under construction. At the top, evolution of the Benders gap for the first phase. On the bottom, evolution of the global objective function for the first phase.

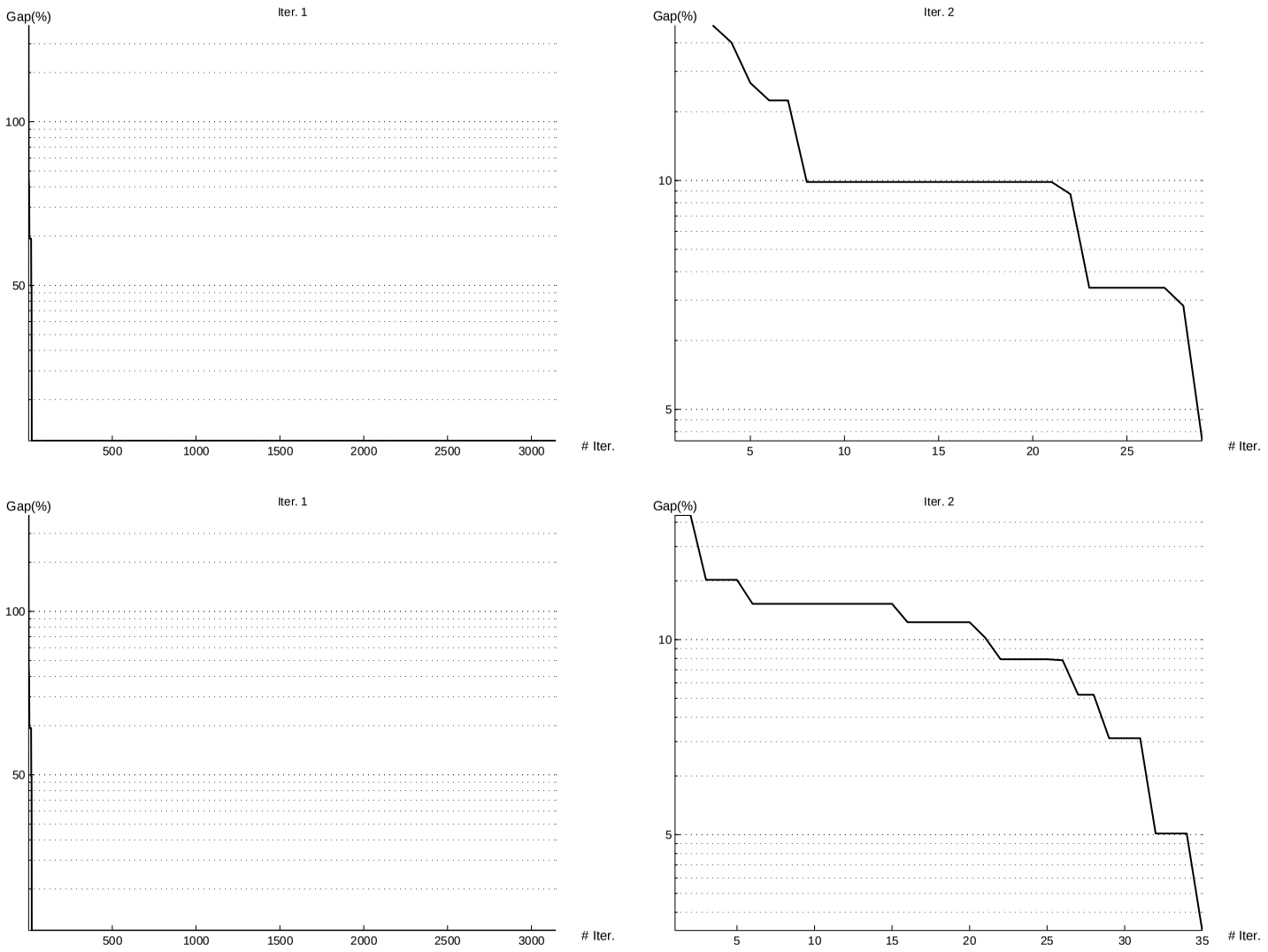


Figure 7.17: Graphs for evolution of the Benders scheme (BD) driven by the line splitting algorithm (LSA) without incremental load variant applied to the model under elastic demand with the 6-node network and four lines under construction. Top-left: gap evolution for the first iteration of the LSA in the first phase of the BD. Bottom-left: gap evolution for the first iteration of the LSA in the second phase of the BD. Top-right: gap evolution for the second iteration of the LSA in the first phase of the BD. Bottom-right: gap evolution for the second iteration of the LSA in the second phase of the BD.

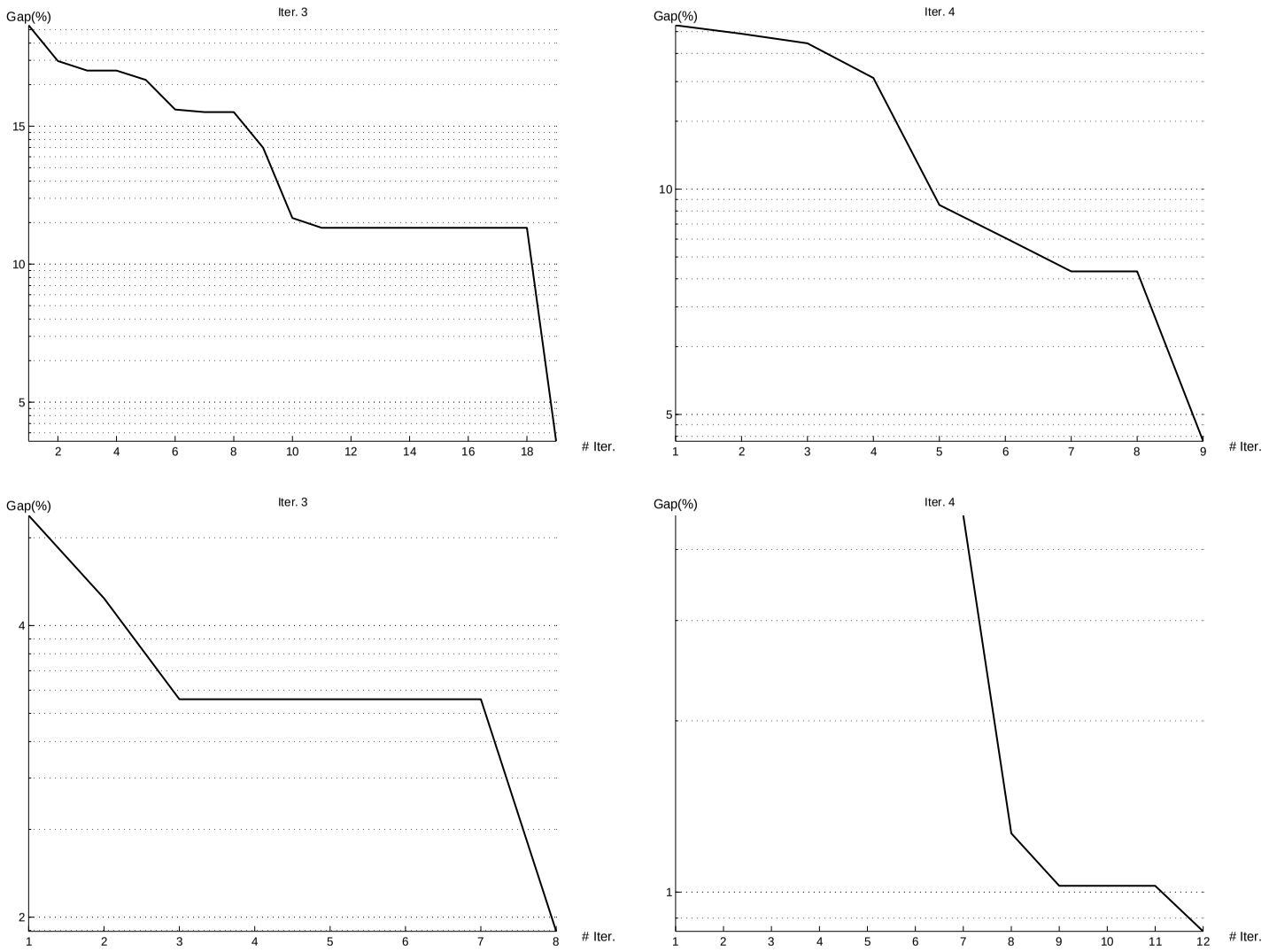


Figure 7.18: Graphs for evolution of the Benders scheme (BD) driven by the line splitting algorithm (LSA) without incremental load variant applied to the model under elastic demand with the 6-node network and four lines under construction. Top-left: gap evolution for the third iteration of the LSA in the first phase of the BD. Bottom-left: gap evolution for the third iteration of the LSA in the second phase of the BD. Top-right: gap evolution for the fourth iteration of the LSA in the first phase of the BD. Bottom-right: gap evolution for the fourth iteration of the LSA in the second phase of the BD.

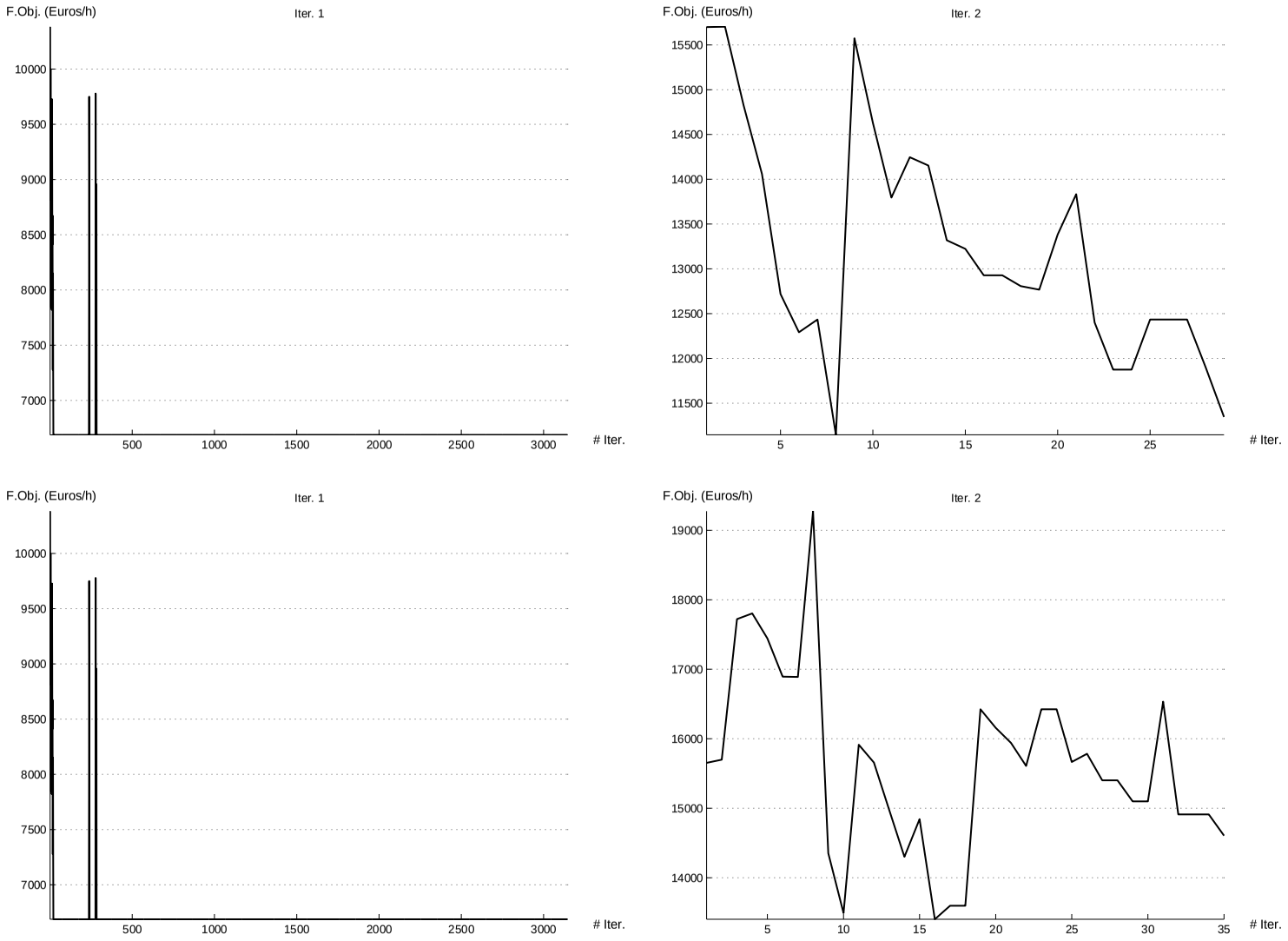


Figure 7.19: Graphs showing the evolution of the objective function for the Benders scheme (BD) driven by the line splitting algorithm (LSA) without incremental load variant applied to the model under elastic demand with the 6-node network and four lines under construction. Top-left: objective function evolution for the first iteration of the LSA in the first phase of the BD. Bottom-left: objective function evolution for the first iteration of the LSA in the second phase of the BD. Top-right: objective function evolution for the second iteration of the LSA in the first phase of the BD. Bottom-right: objective function evolution for the second iteration of the LSA in the second phase of the BD.

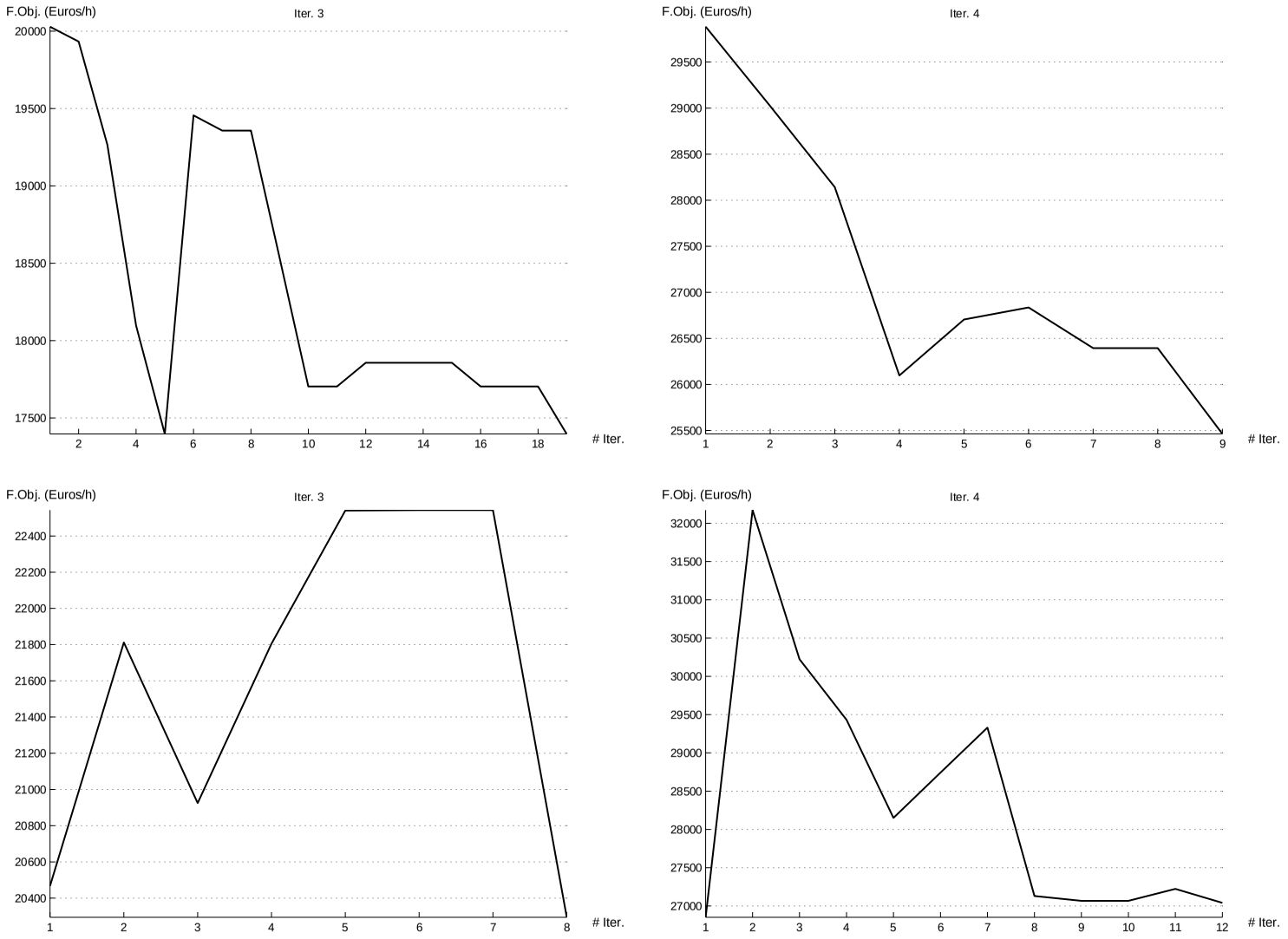


Figure 7.20: Graphs showing the evolution of the objective function for the Benders scheme (BD) driven by the line splitting algorithm (LSA) without incremental load variant applied to the model under elastic demand with the 6-node network and four lines under construction. Top-left: objective function evolution for the third iteration of the LSA in the first phase of the BD. Bottom-left: objective function evolution for the third iteration of the LSA in the second phase of the BD. Top-right: objective function evolution for the fourth iteration of the LSA in the first phase of the BD. Bottom-right: objective function evolution for the fourth iteration of the LSA in the second phase of the BD.

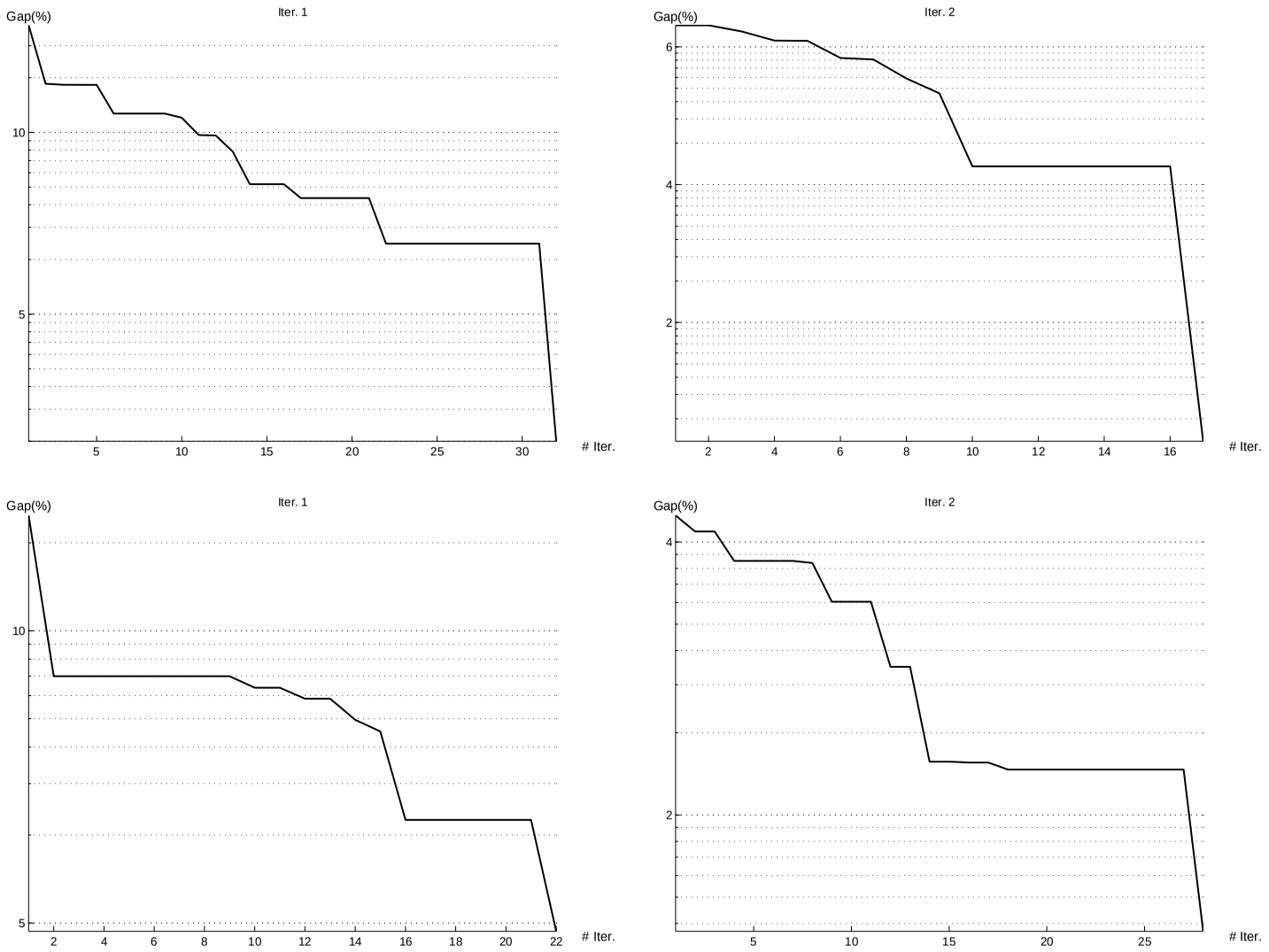


Figure 7.21: Graphs for evolution of the Benders scheme (BD) driven by the line splitting algorithm (LSA) with incremental load variant applied to the model under elastic demand with the 6-node network and four lines under construction. Top-left: gap evolution for the first iteration of the LSA in the first phase of the BD. Bottom-left: gap evolution for the first iteration of the LSA in the second phase of the BD. Top-right: gap evolution for the second iteration of the LSA in the first phase of the BD. Bottom-right: gap evolution for the second iteration of the LSA in the second phase of the BD.

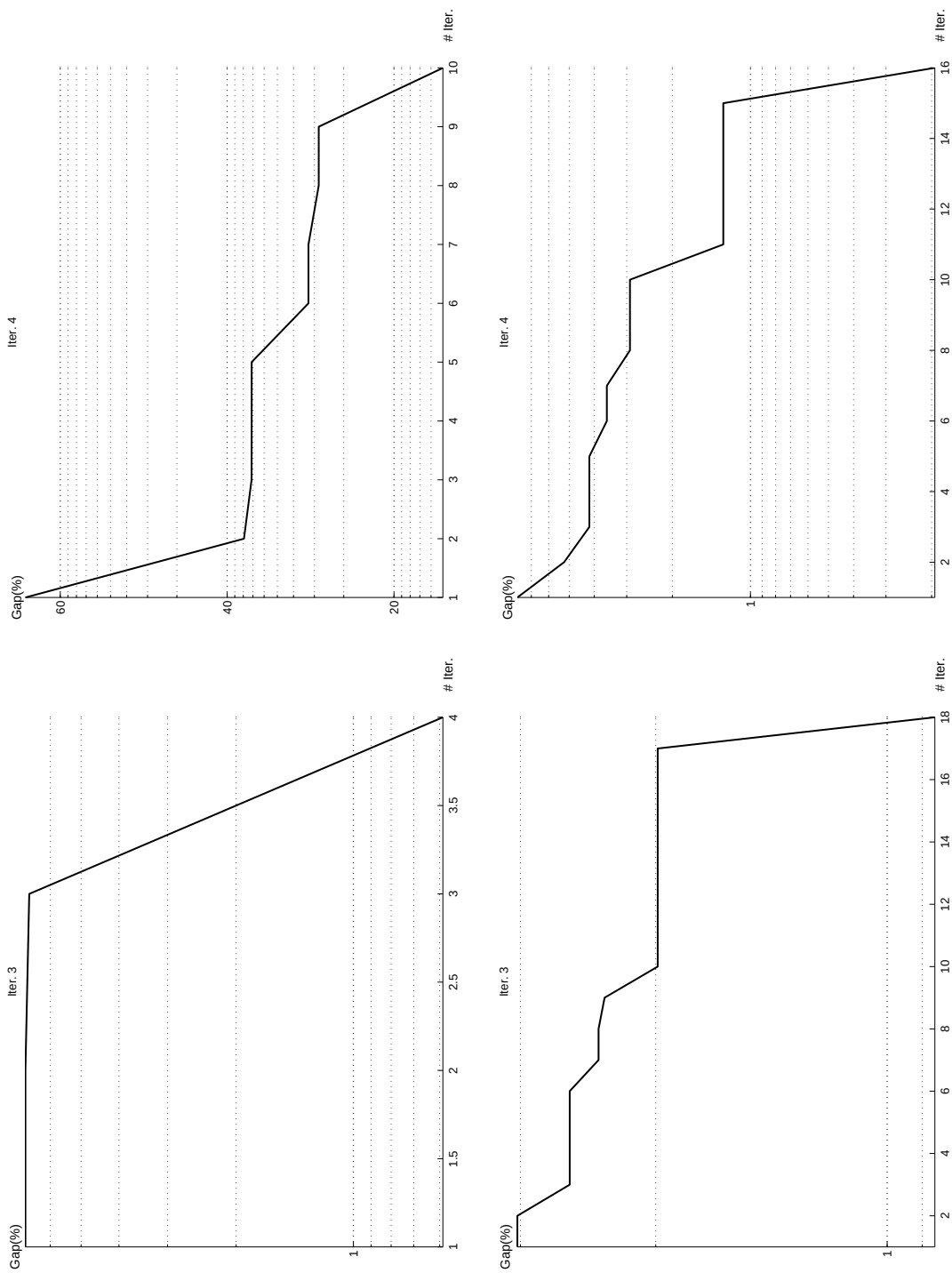


Figure 7.22: Graphs for evolution of the Benders scheme (BD) driven by the line splitting algorithm (LSA) with incremental load variant applied to the model under elastic demand with the 6-node network and four lines under construction. Top-left: gap evolution for the third iteration of the LSA in the first phase of the BD. Bottom-left: gap evolution for the third iteration of the LSA in the second phase of the BD. Top-right: gap evolution for the fourth iteration of the LSA in the first phase of the BD. Bottom-right: gap evolution for the fourth iteration of the LSA in the second phase of the BD.

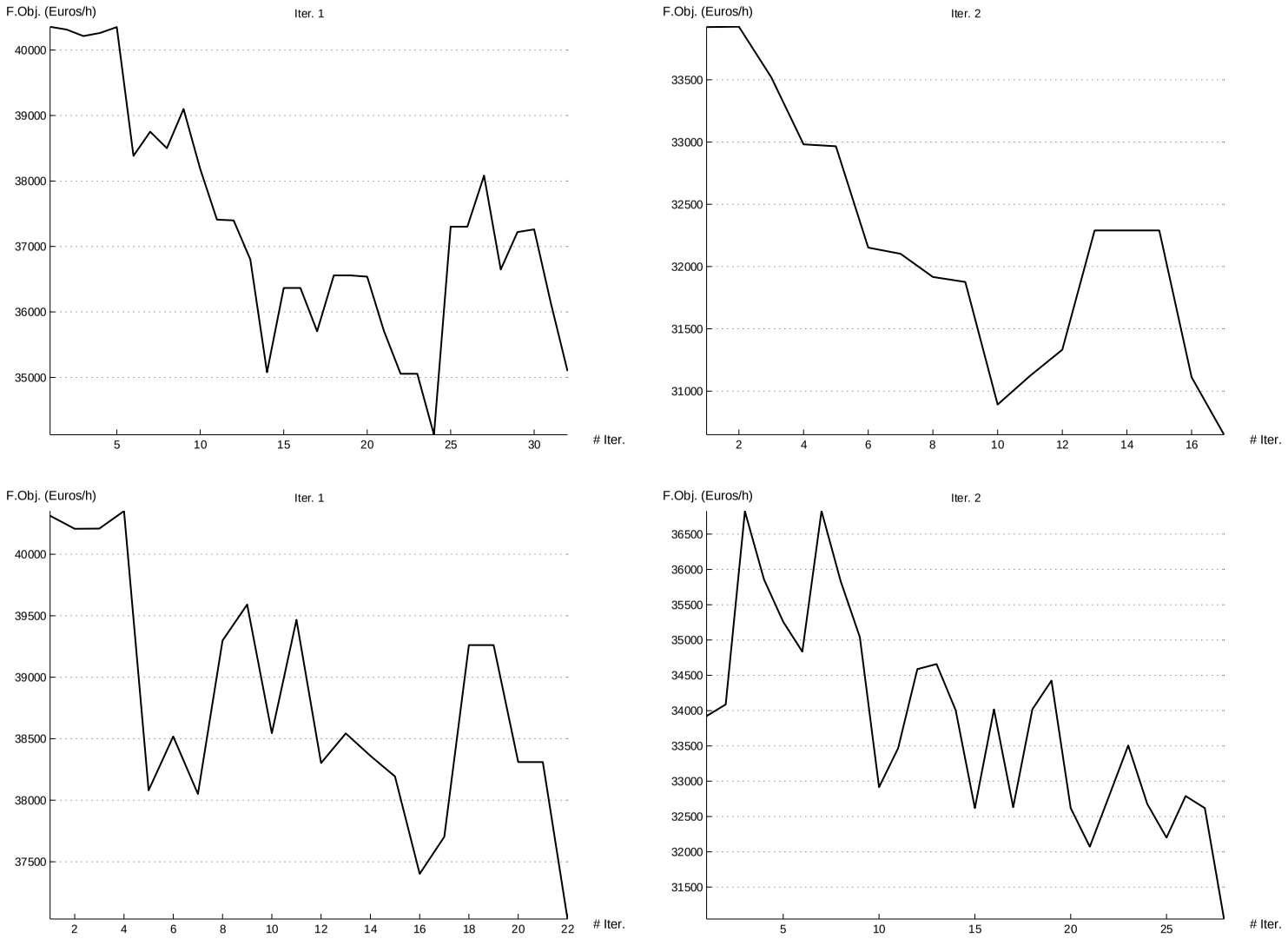


Figure 7.23: Graphs showing the evolution of the objective function for the Benders scheme (BD) driven by the line splitting algorithm (LSA) with incremental load variant applied to the model under elastic demand with the 6-node network and four lines under construction. Top-left: objective function evolution for the first iteration of the LSA in the first phase of the BD. Bottom-left: objective function evolution for the first iteration of the LSA in the second phase of the BD. Top-right: objective function evolution for the second iteration of the LSA in the first phase of the BD. Bottom-right: objective function evolution for the second iteration of the LSA in the second phase of the BD.

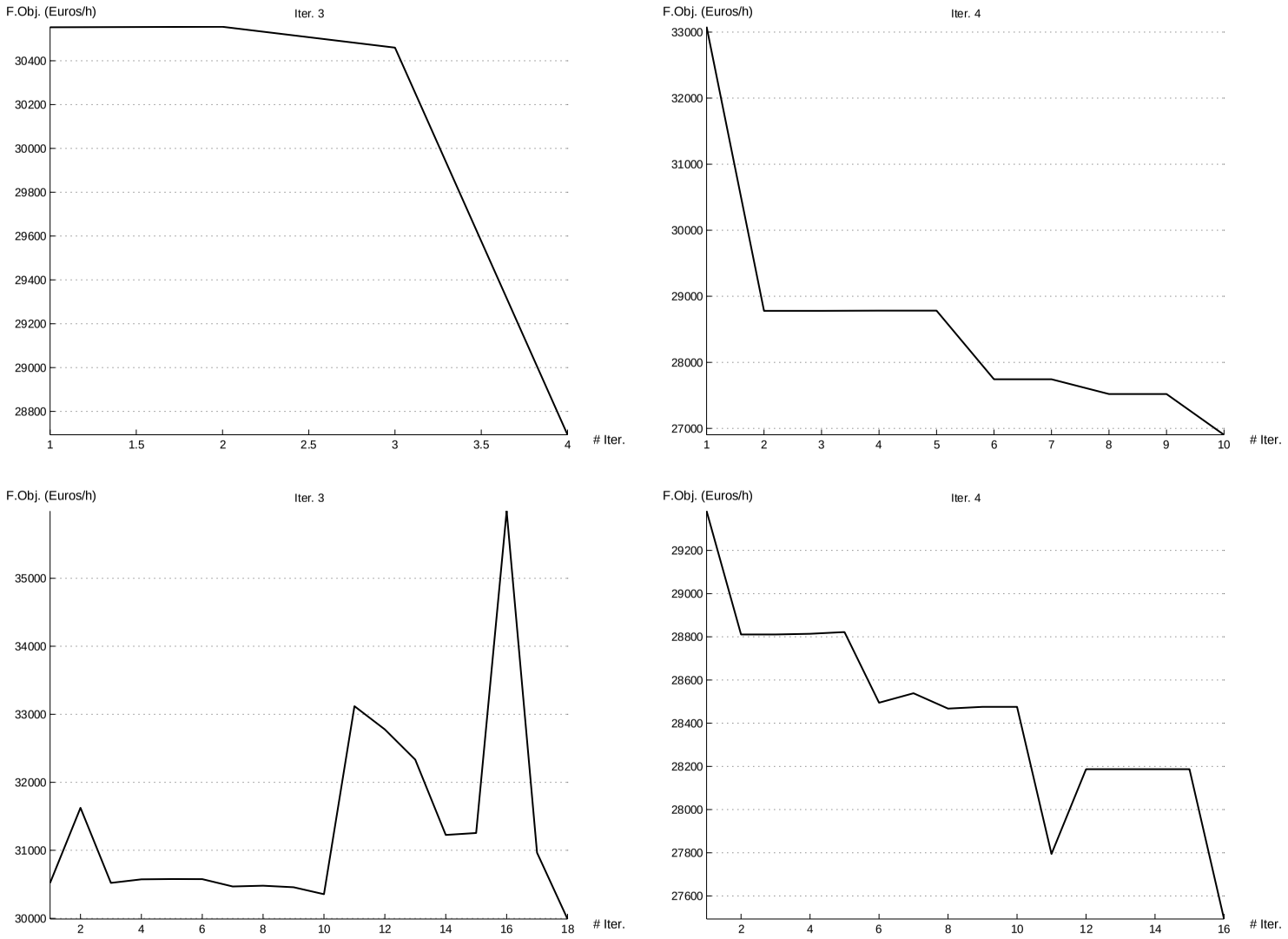


Figure 7.24: Graphs showing the evolution of the objective function for the Benders scheme (BD) driven by the line splitting algorithm (LSA) with incremental load variant applied to the model under elastic demand with the 6-node network and four lines under construction. Top-left: objective function evolution for the third iteration of the LSA in the first phase of the BD. Bottom-left: objective function evolution for the third iteration of the LSA in the second phase of the BD. Top-right: objective function evolution for the fourth iteration of the LSA in the first phase of the BD. Bottom-right: objective function evolution for the fourth iteration of the LSA in the second phase of the BD.

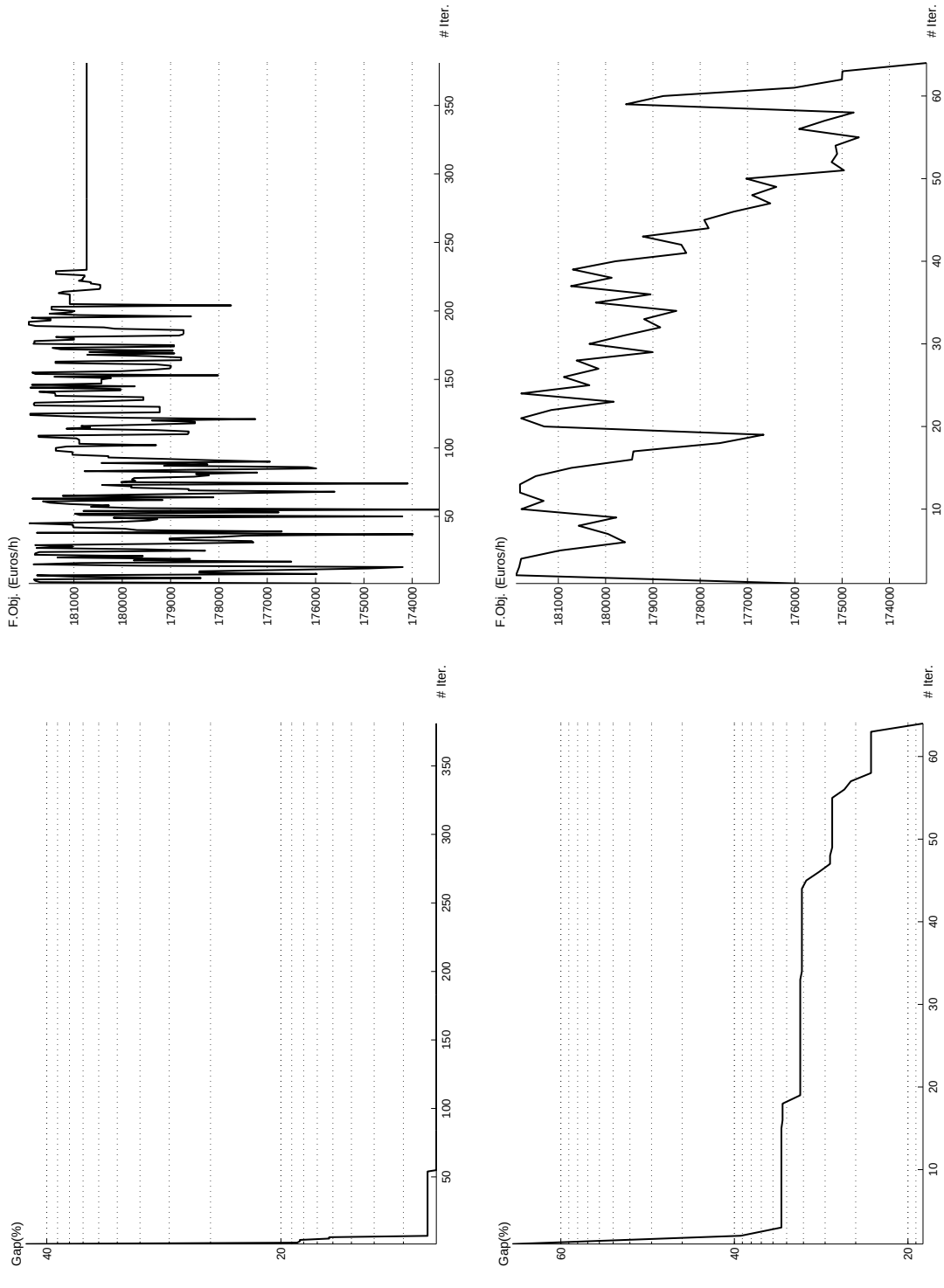


Figure 7.25: Graphs for evolution of the Benders scheme applied to the model under elastic demand with the 9-node network and two lines under construction. Top-left: evolution of the Benders gap for the first phase. Bottom-left: evolution of the Benders gap for the second phase. Top-right: evolution of the global objective function for the first phase. Bottom-right: evolution of the global objective function for the second phase.

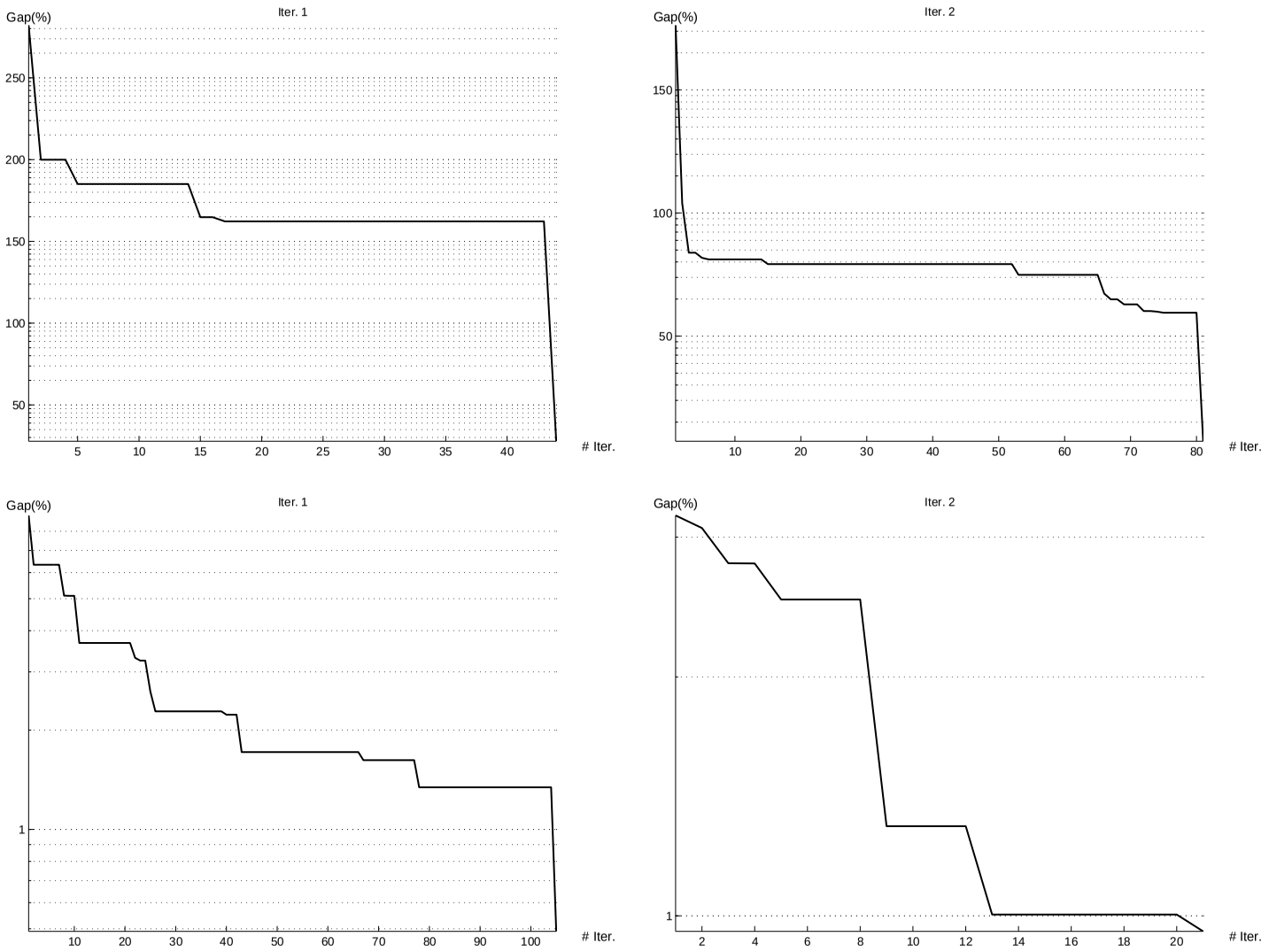


Figure 7.26: Graphs for evolution of the Benders scheme (BD) driven by the line splitting algorithm (LSA) without incremental load variant applied to the model under elastic demand with the 9-node network and two lines under construction. Top-left: gap evolution for the first iteration of the LSA in the first phase of the BD. Bottom-left: gap evolution for the first iteration of the LSA in the second phase of the BD. Top-right: gap evolution for the second iteration of the LSA in the first phase of the BD. Bottom-right: gap evolution for the second iteration of the LSA in the second phase of the BD.

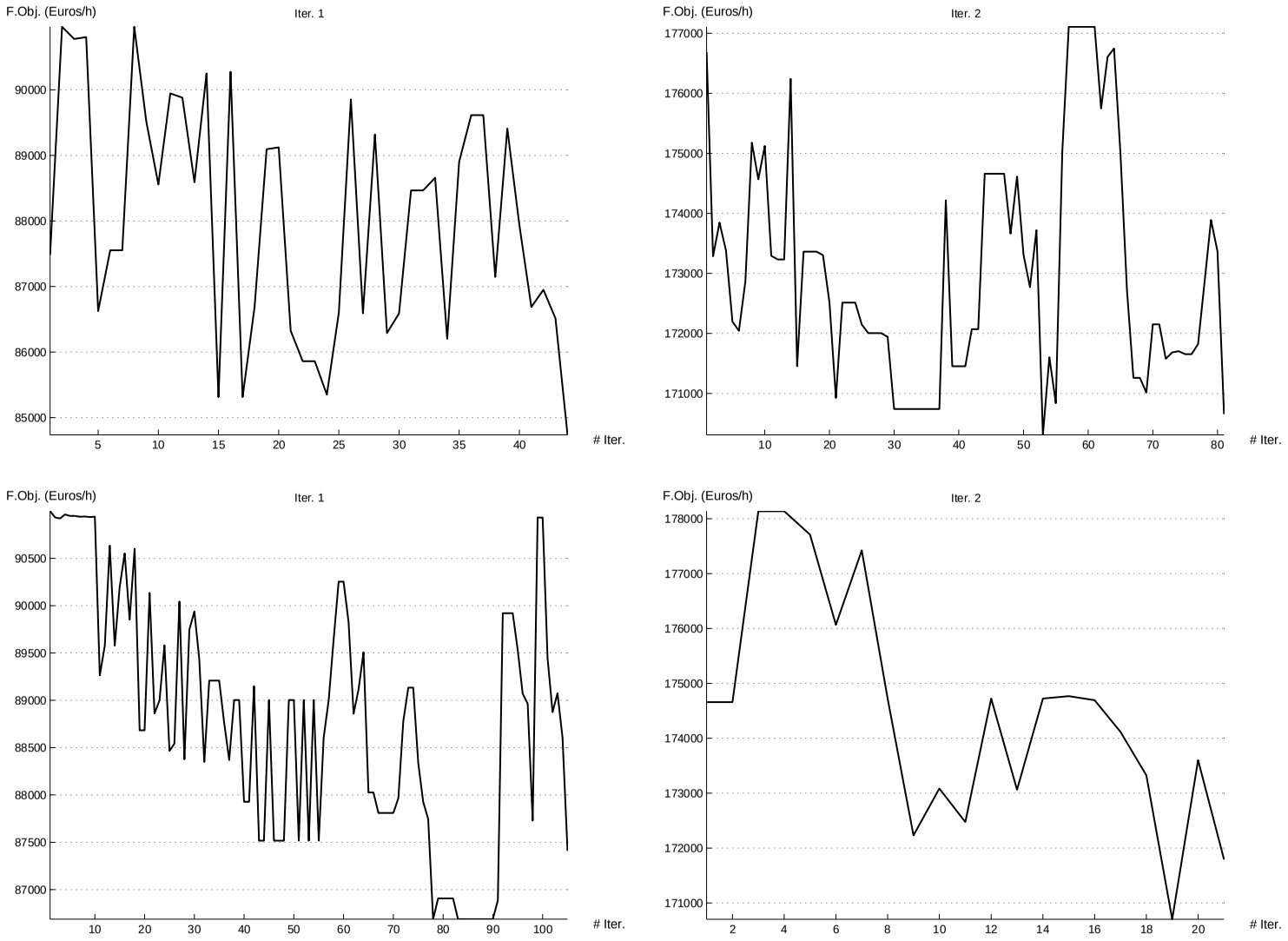


Figure 7.27: Graphs showing the evolution of the objective function for the Benders scheme (BD) driven by the line splitting algorithm (LSA) without incremental load variant applied to the model under elastic demand with the 9-node network and two lines under construction. Top-left: objective function evolution for the first iteration of the LSA in the first phase of the BD. Bottom-left: objective function evolution for the first iteration of the LSA in the second phase of the BD. Top-right: objective function evolution for the second iteration of the LSA in the first phase of the BD. Bottom-right: objective function evolution for the second iteration of the LSA in the second phase of the BD.

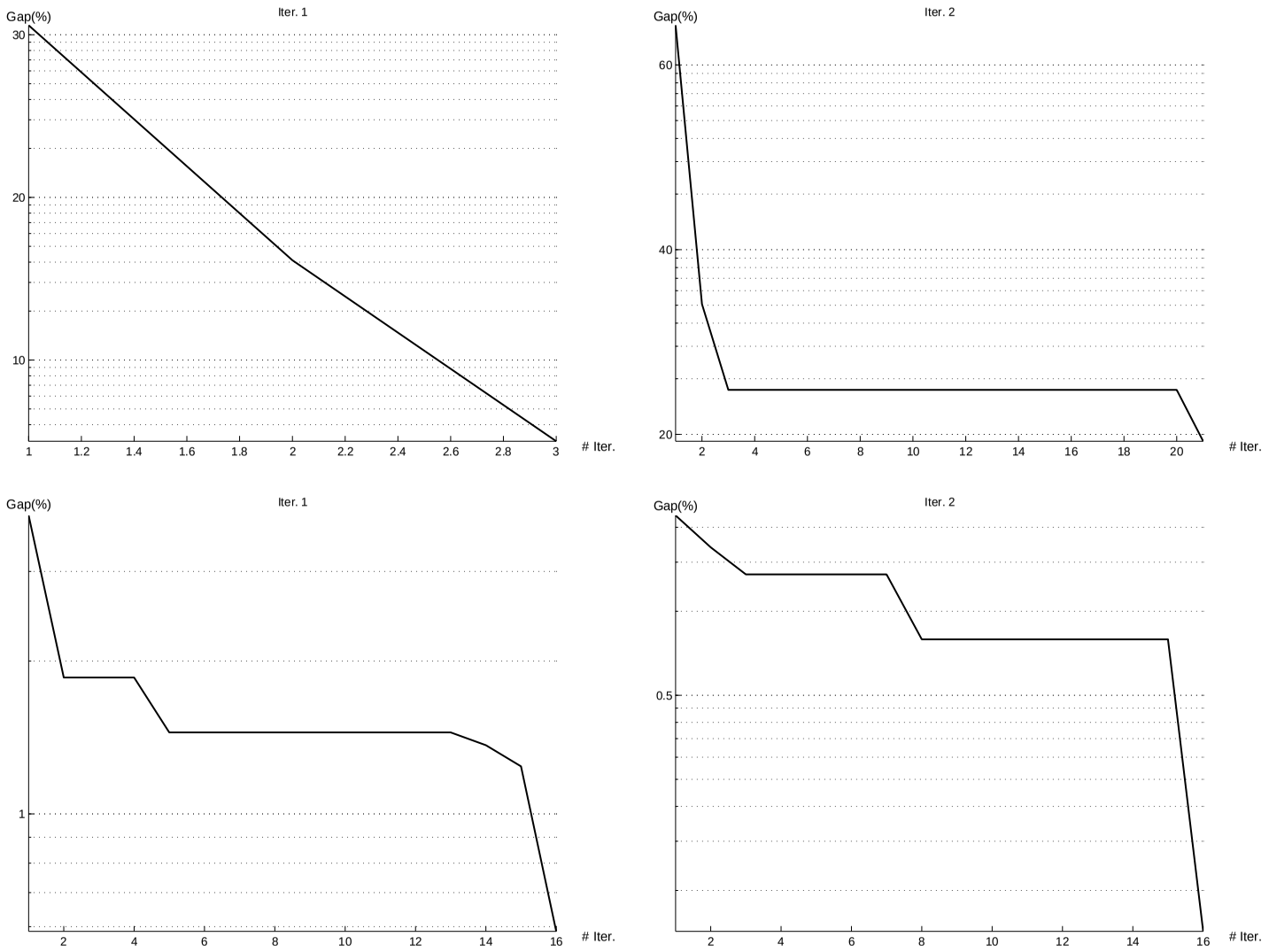


Figure 7.28: Graphs for evolution of the Benders scheme (BD) driven by the line splitting algorithm (LSA) with incremental load variant applied to the model under elastic demand with the 9-node network and two lines under construction. Top-left: gap evolution for the first iteration of the LSA in the first phase of the BD. Bottom-left: gap evolution for the first iteration of the LSA in the second phase of the BD. Top-right: gap evolution for the second iteration of the LSA in the first phase of the BD. Bottom-right: gap evolution for the second iteration of the LSA in the second phase of the BD.

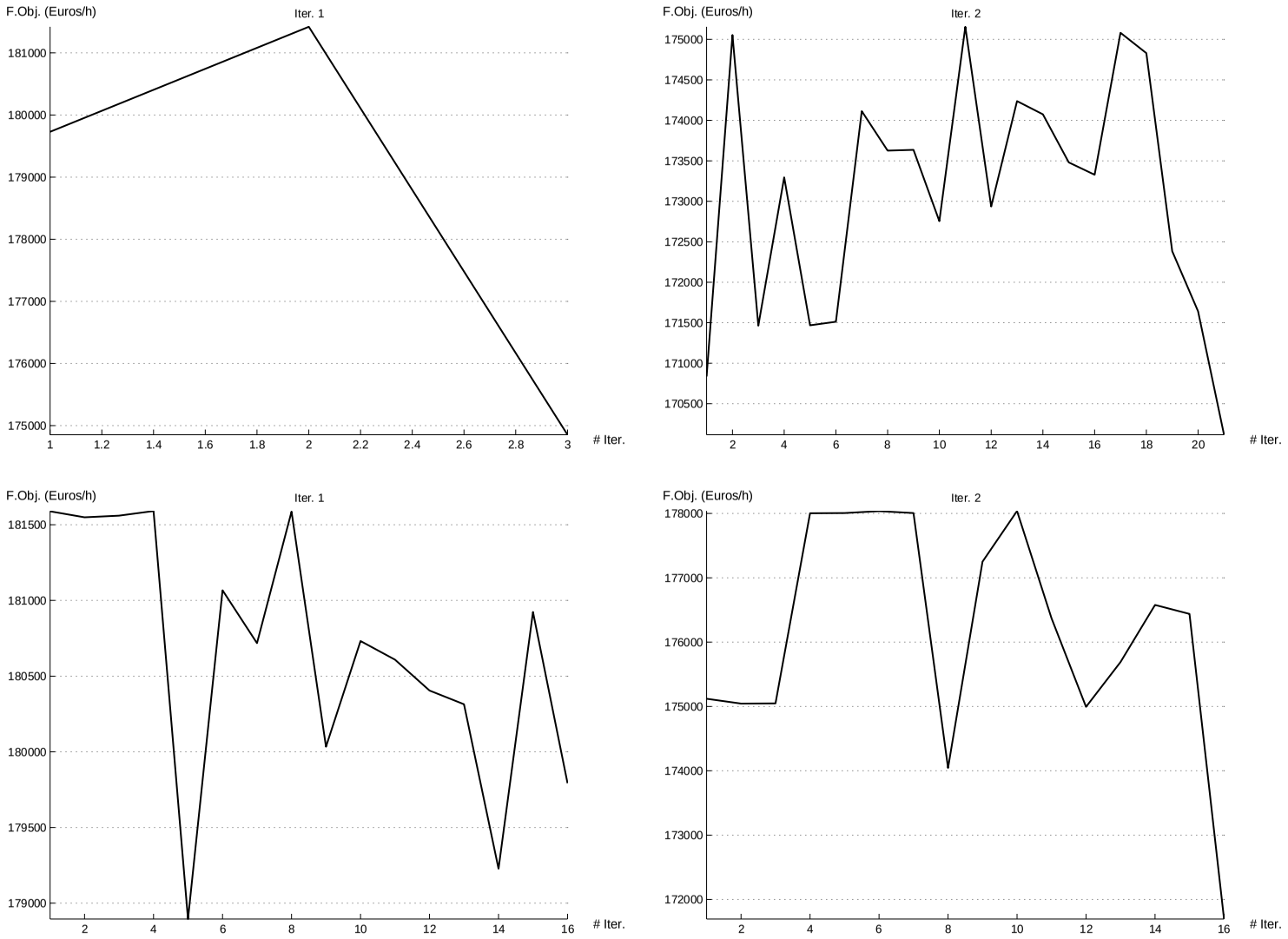


Figure 7.29: Graphs showing the evolution of the objective function for the Benders scheme (BD) driven by the line splitting algorithm (LSA) with incremental load variant applied to the model under elastic demand with the 9-node network and two lines under construction. Top-left: objective function evolution for the first iteration of the LSA in the first phase of the BD. Bottom-left: objective function evolution for the first iteration of the LSA in the second phase of the BD. Top-right: objective function evolution for the second iteration of the LSA in the first phase of the BD. Bottom-right: objective function evolution for the second iteration of the LSA in the second phase of the BD.

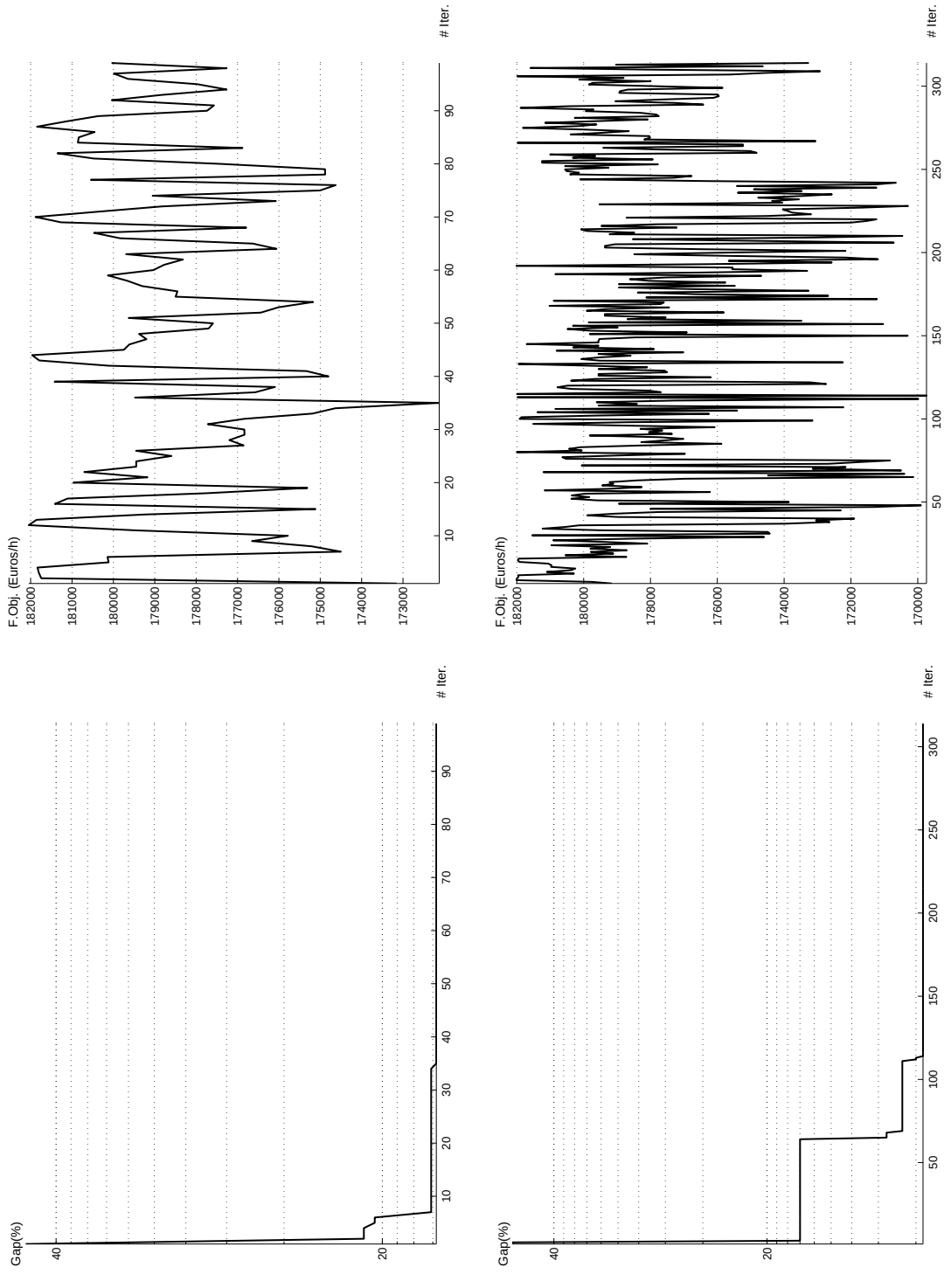


Figure 7.30: Graphs for evolution of the Benders scheme applied to the model under elastic demand with the 9-node network and three lines under construction. Top-left: evolution of the Benders gap for the first phase. Bottom-left: evolution of the Benders gap for the second phase. Top-right: evolution of the global objective function for the first phase. Bottom-right: evolution of the global objective function for the second phase.

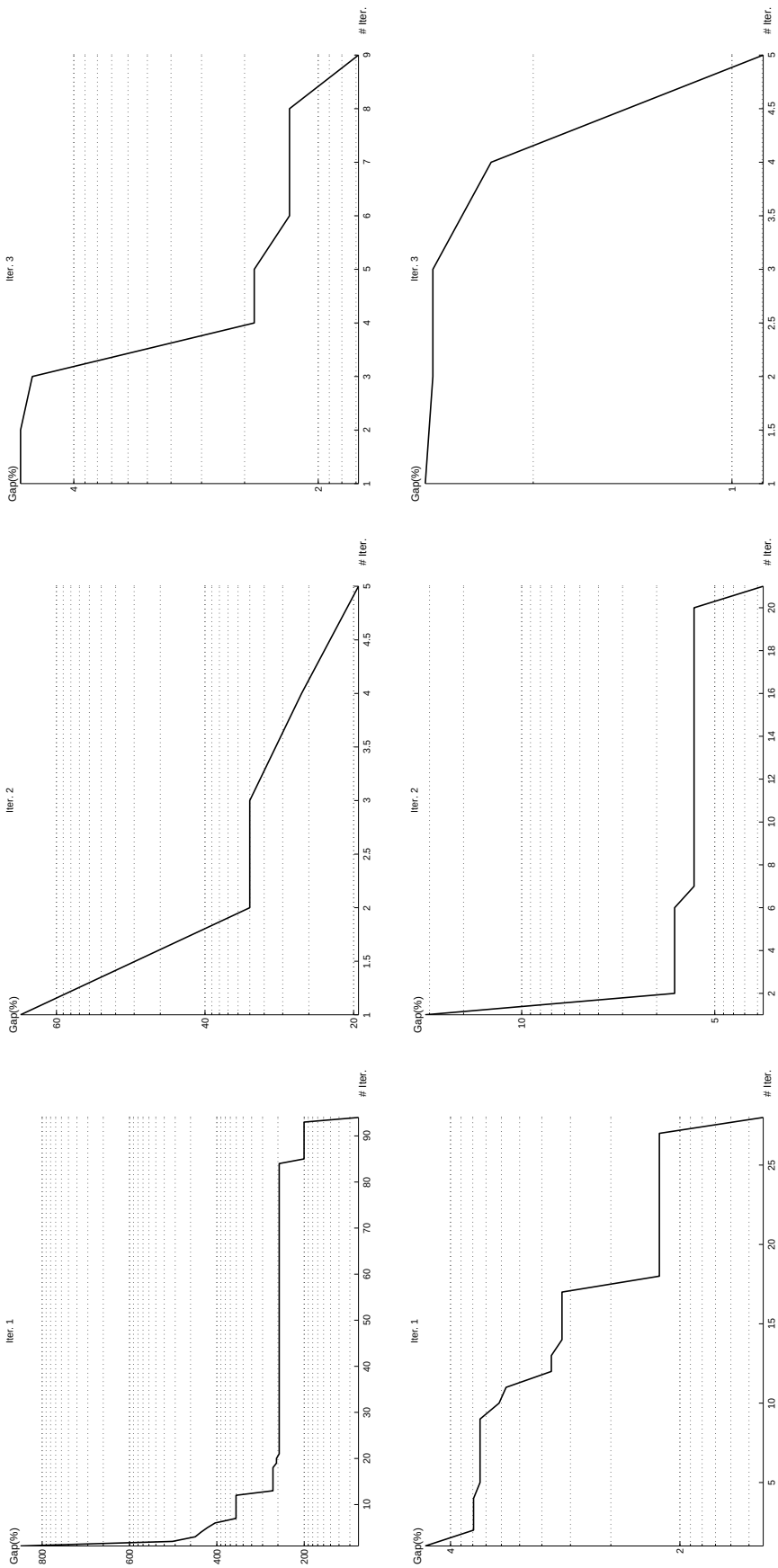


Figure 7.31: Graphs for evolution of the Benders scheme (BD) driven by the line splitting algorithm (LSA) without incremental load variant applied to the model under elastic demand with the 9-node network and three lines under construction. Top-left: gap evolution for the first iteration of the LSA in the first phase of the BD. Bottom-left: gap evolution for the first iteration of the LSA in the second phase of the BD. On the top-center, gap evolution for the third iteration of the LSA in the first phase of the BD. On the bottom-center, gap evolution for the third iteration of the LSA in the second phase of the BD. Top-right: gap evolution for the third iteration of the LSA in the first phase of the BD. Bottom-right: gap evolution for the third iteration of the LSA in the second phase of the BD.

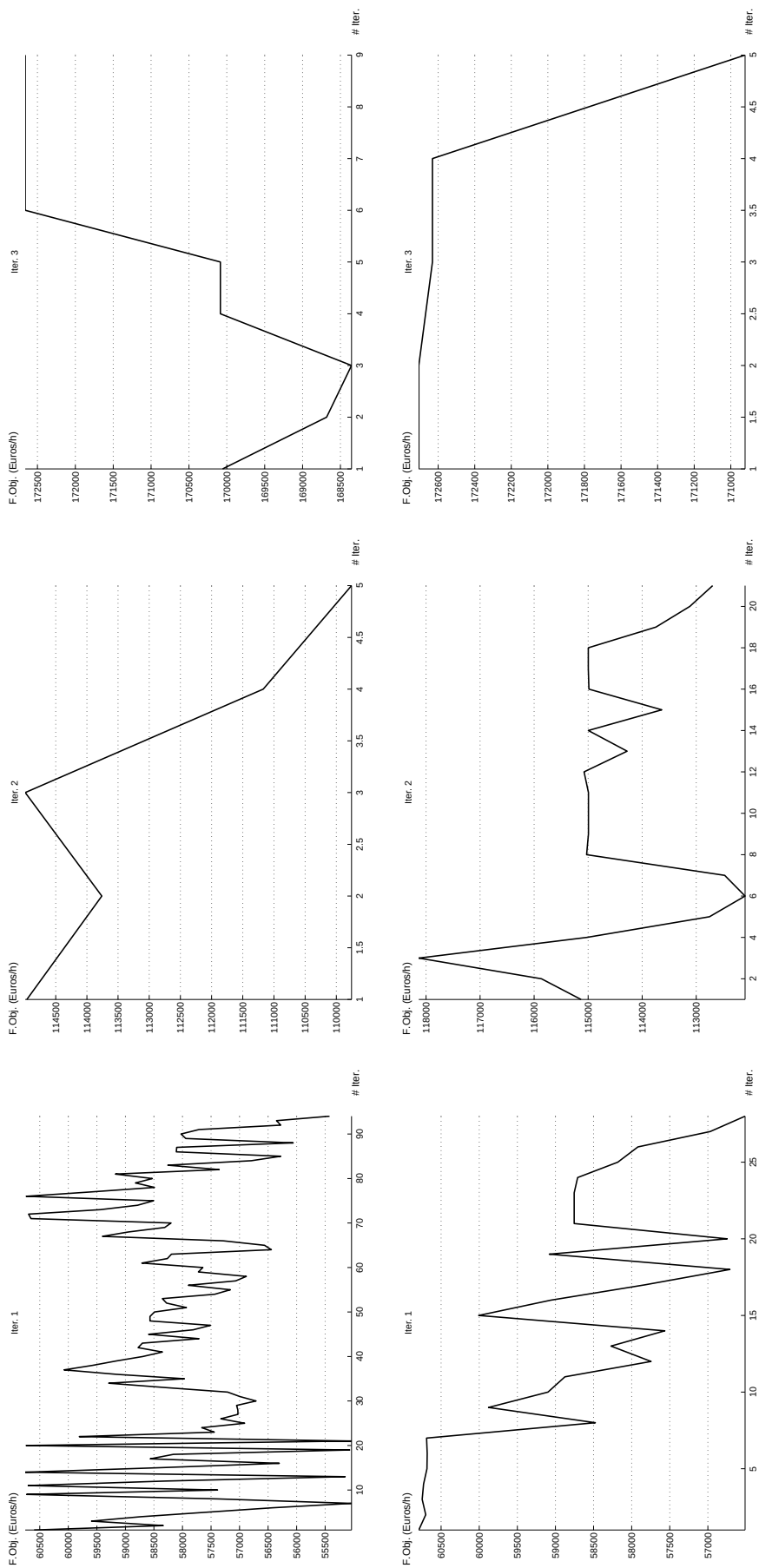


Figure 7.32: Graphs showing the evolution of the objective function for the Benders scheme (BD) driven by the line splitting algorithm (LSA) without incremental load variant applied to the model under elastic demand with the 9-node network and three lines under construction. Top-left: objective function evolution for the first iteration of the LSA in the first phase of the BD. Bottom-left: objective function evolution for the first iteration of the LSA in the second phase of the BD. On the top-center, objective function evolution for the third iteration of the LSA in the first phase of the BD. On the bottom-center, objective function evolution for the third iteration of the LSA in the second phase of the BD. Top-right: objective function evolution for the third iteration of the LSA in the first phase of the BD. Bottom-right: objective function evolution for the third iteration of the LSA in the second phase of the BD.

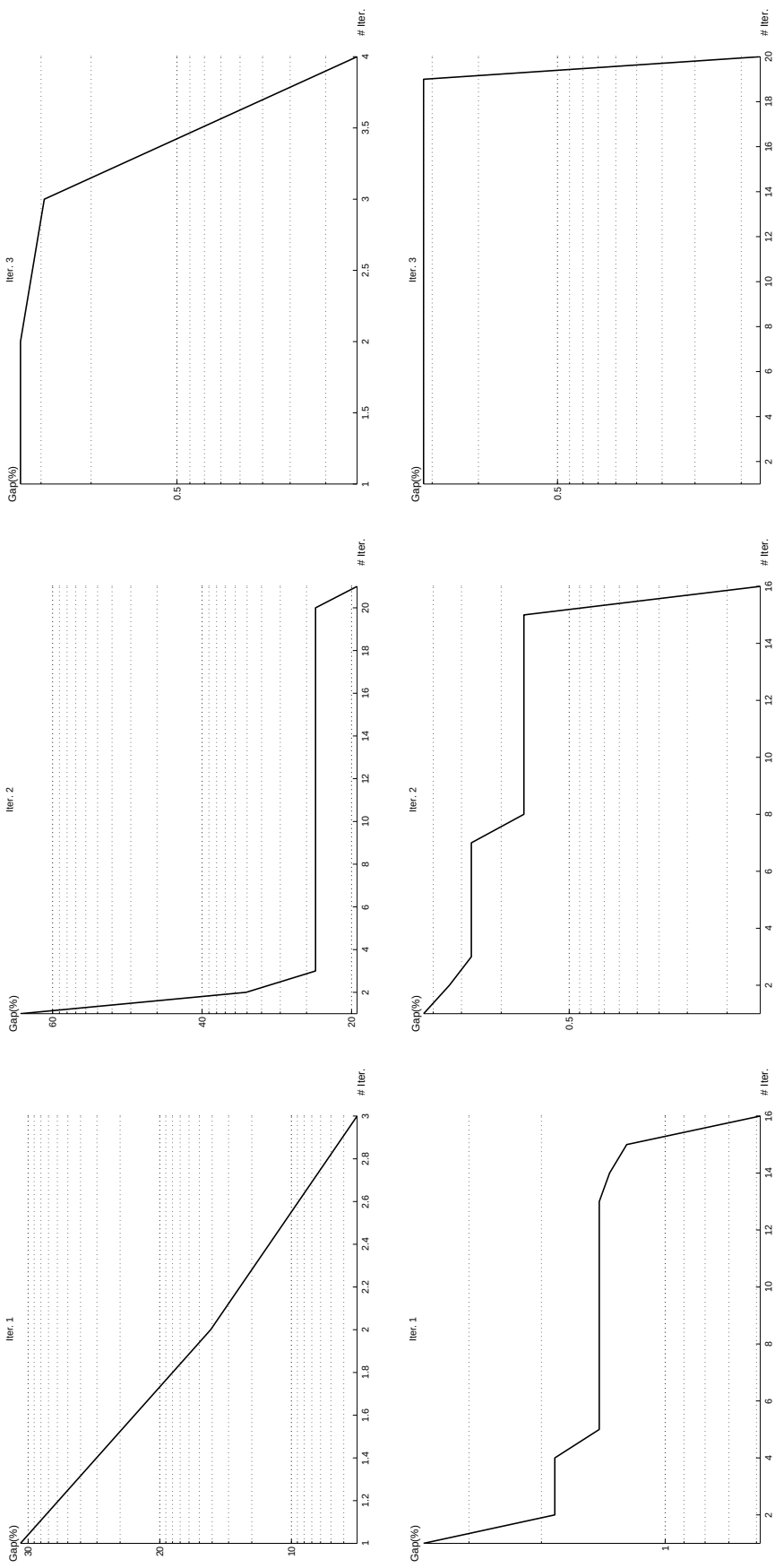


Figure 7.33: Graphs for evolution of the Benders scheme (BD) driven by the line splitting algorithm (LSA) with incremental load variant applied to the model under elastic demand with the 9-node network and three lines under construction. Top-left: gap evolution for the first iteration of the LSA in the first phase of the BD. Bottom-left: gap evolution for the first iteration of the LSA in the second phase of the BD. On the top-center, gap evolution for the third iteration of the LSA in the first phase of the BD. On the bottom-center, gap evolution for the third iteration of the LSA in the second phase of the BD. Top-right: gap evolution for the third iteration of the LSA in the first phase of the BD. Bottom-right: gap evolution for the third iteration of the LSA in the second phase of the BD.

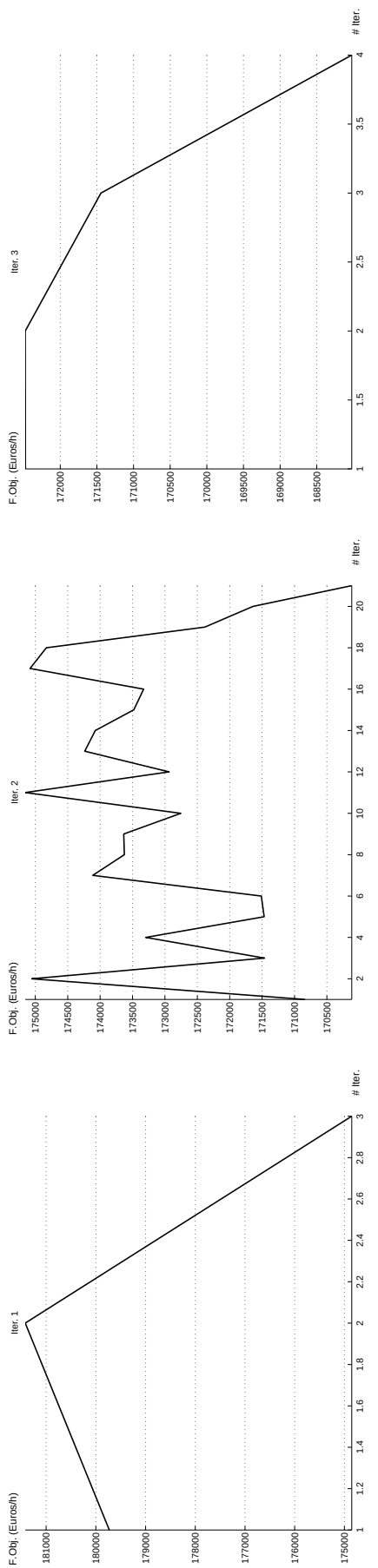


Figure 7.34: Graphs showing the evolution of the objective function for the Benders scheme (BD) driven by the line splitting algorithm (LSA) with incremental load variant applied to the model under elastic demand with the 9-node network and three lines under construction. Top-left: objective function evolution for the first iteration of the LSA in the first phase of the BD. Bottom-left: objective function evolution for the first iteration of the LSA in the second phase of the BD. On the top-center, objective function evolution for the third iteration of the LSA in the first phase of the BD. On the bottom-center, objective function evolution for the third iteration of the LSA in the second phase of the BD. Top-right: objective function evolution for the third iteration of the LSA in the first phase of the BD. Bottom-right: objective function evolution for the third iteration of the LSA in the second phase of the BD.

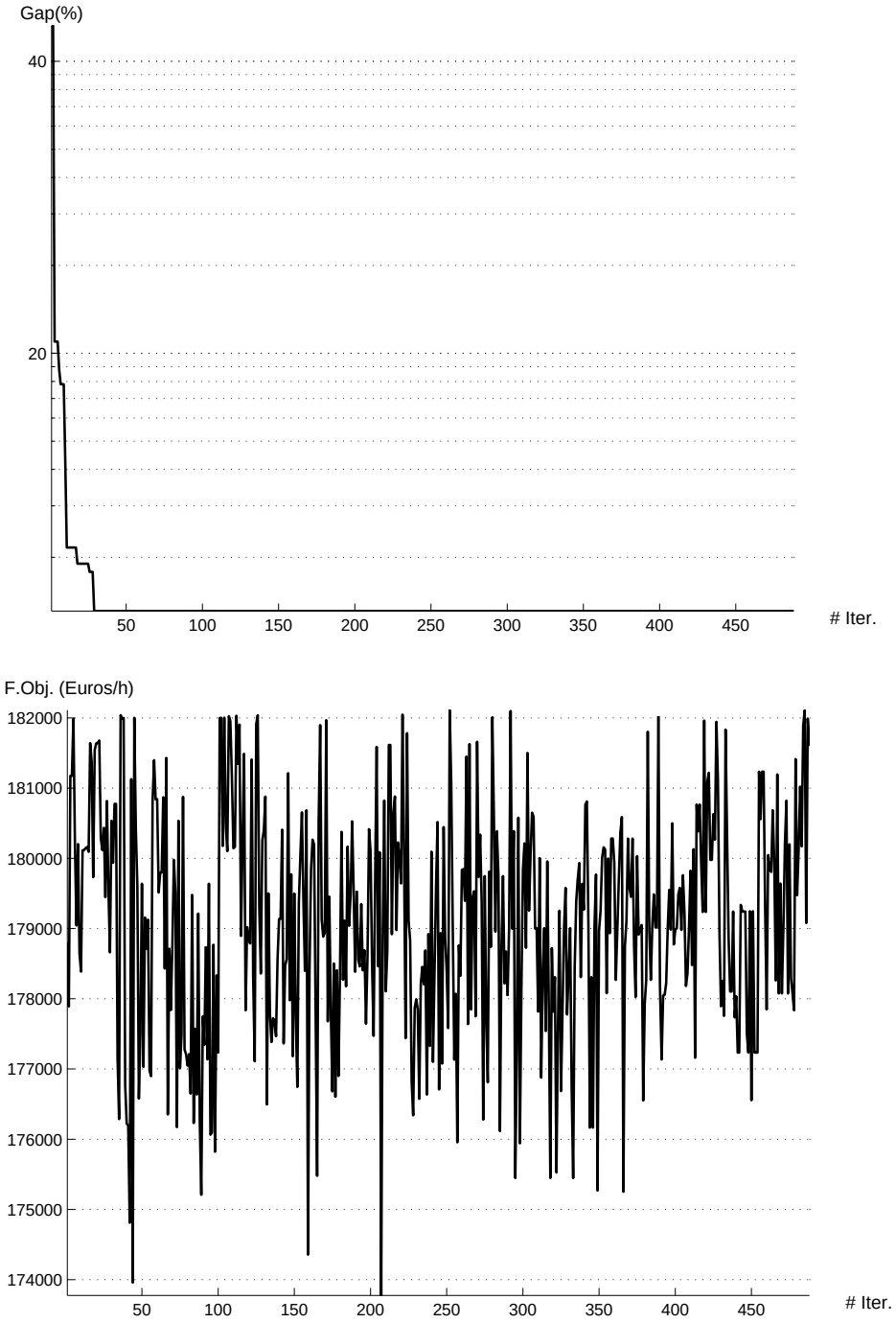


Figure 7.35: Graphs for evolution of the Benders scheme applied to the model under elastic demand with the 9-node network and four lines under construction. At the top, evolution of the Benders gap for the first phase. On the bottom, evolution of the global objective function for the first phase.

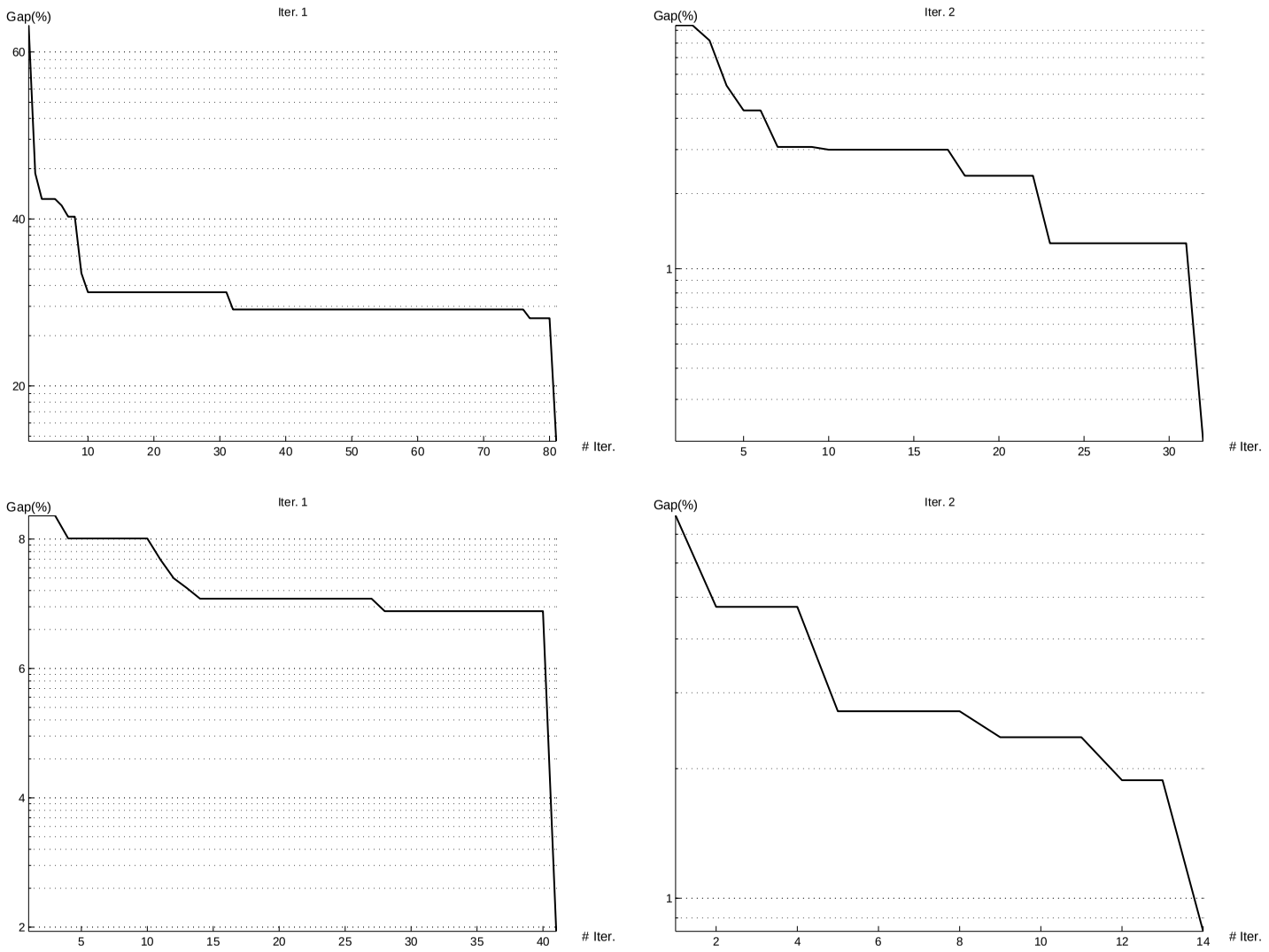


Figure 7.36: Graphs for evolution of the Benders scheme (BD) driven by the line splitting algorithm (LSA) without incremental load variant applied to the model under elastic demand with the 9-node network and four lines under construction. Top-left: gap evolution for the first iteration of the LSA in the first phase of the BD. Bottom-left: gap evolution for the first iteration of the LSA in the second phase of the BD. Top-right: gap evolution for the second iteration of the LSA in the first phase of the BD. Bottom-right: gap evolution for the second iteration of the LSA in the second phase of the BD.

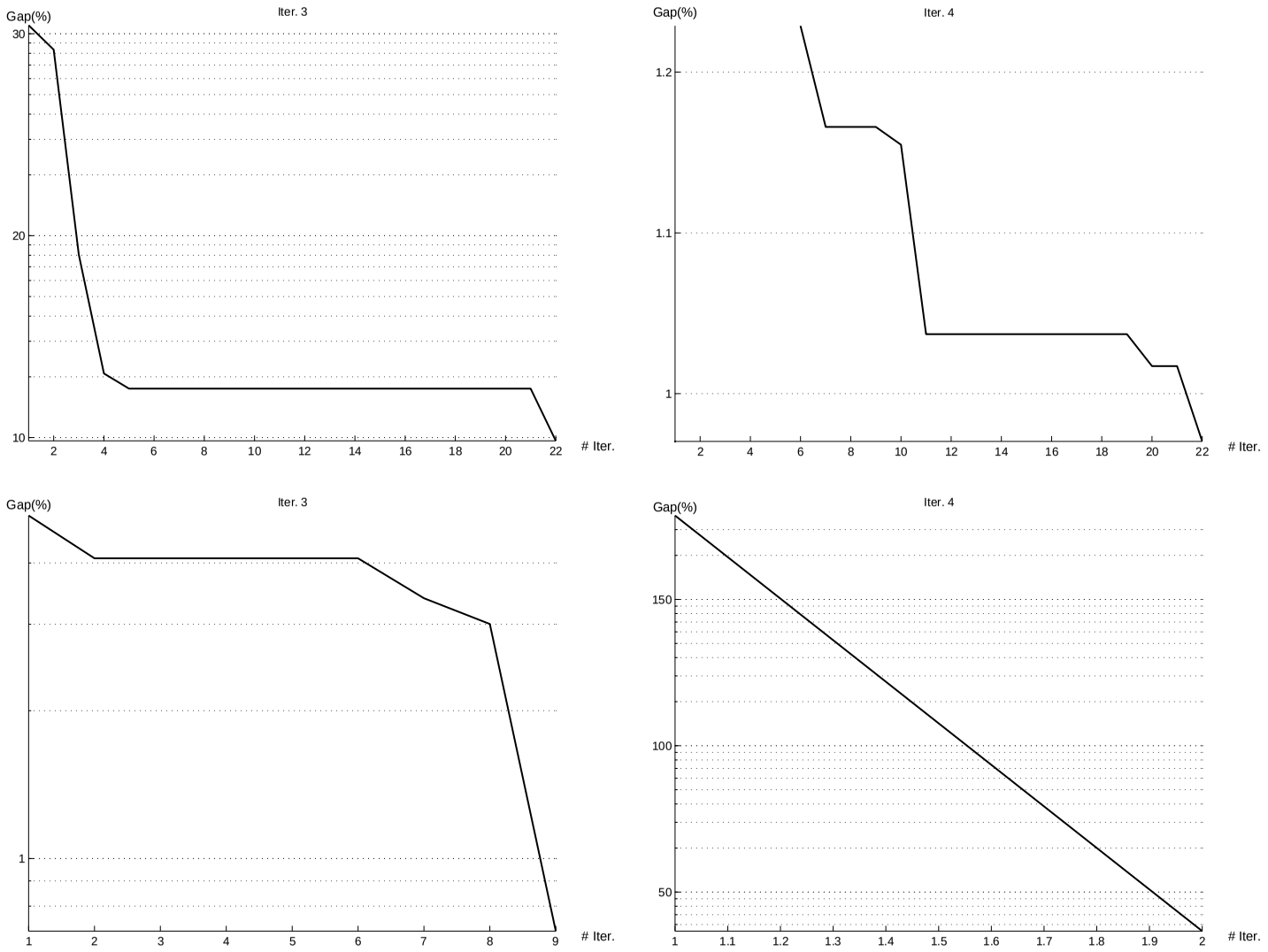


Figure 7.37: Graphs for evolution of the Benders scheme (BD) driven by the line splitting algorithm (LSA) without incremental load variant applied to the model under elastic demand with the 9-node network and four lines under construction. Top-left: gap evolution for the third iteration of the LSA in the first phase of the BD. Bottom-left: gap evolution for the third iteration of the LSA in the second phase of the BD. Top-right: gap evolution for the fourth iteration of the LSA in the first phase of the BD. Bottom-right: gap evolution for the fourth iteration of the LSA in the second phase of the BD.

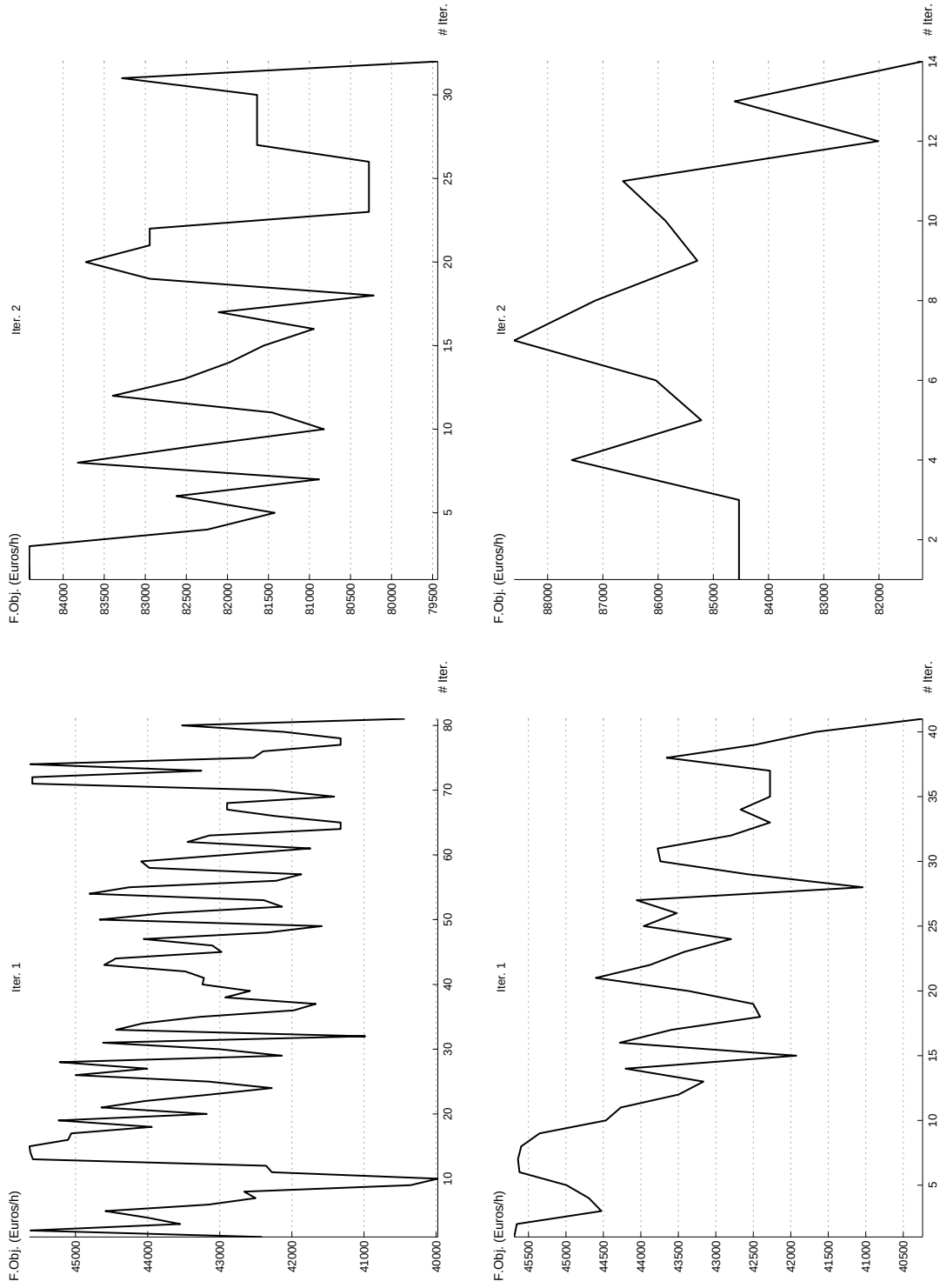


Figure 7.38: Objective function evolution graphs for the Benders scheme (BD) driven by the line splitting algorithm (LSA) without incremental load variant applied to the model under elastic demand with the 9-node network and four lines under construction. Top-left: objective function evolution for the first iteration of the LSA in the first phase of the BD. Bottom-left: objective function evolution for the first iteration of the LSA in the second phase of the BD. Top-right: objective function evolution for the second iteration of the LSA in the first phase of the BD. Bottom-right: objective function evolution for the second iteration of the LSA in the second phase of the BD.

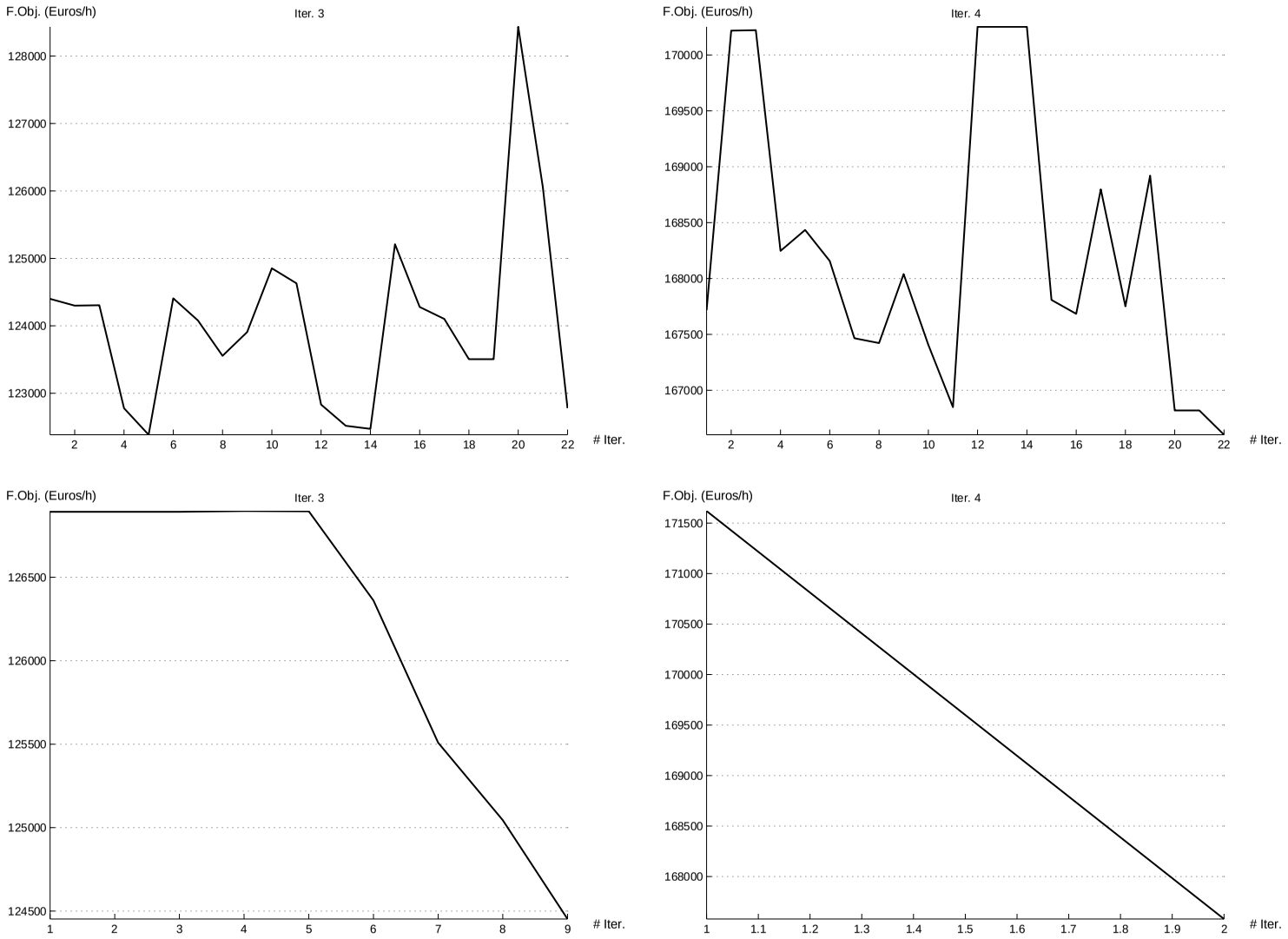


Figure 7.39: Graphs showing the evolution of the objective function for the Benders scheme (BD) driven by the line splitting algorithm (LSA) without incremental load variant applied to the model under elastic demand with the 9-node network and four lines under construction. Top-left: objective function evolution for the third iteration of the LSA in the first phase of the BD. Bottom-left: objective function evolution for the third iteration of the LSA in the second phase of the BD. Top-right: objective function evolution for the fourth iteration of the LSA in the first phase of the BD. Bottom-right: objective function evolution for the fourth iteration of the LSA in the second phase of the BD.

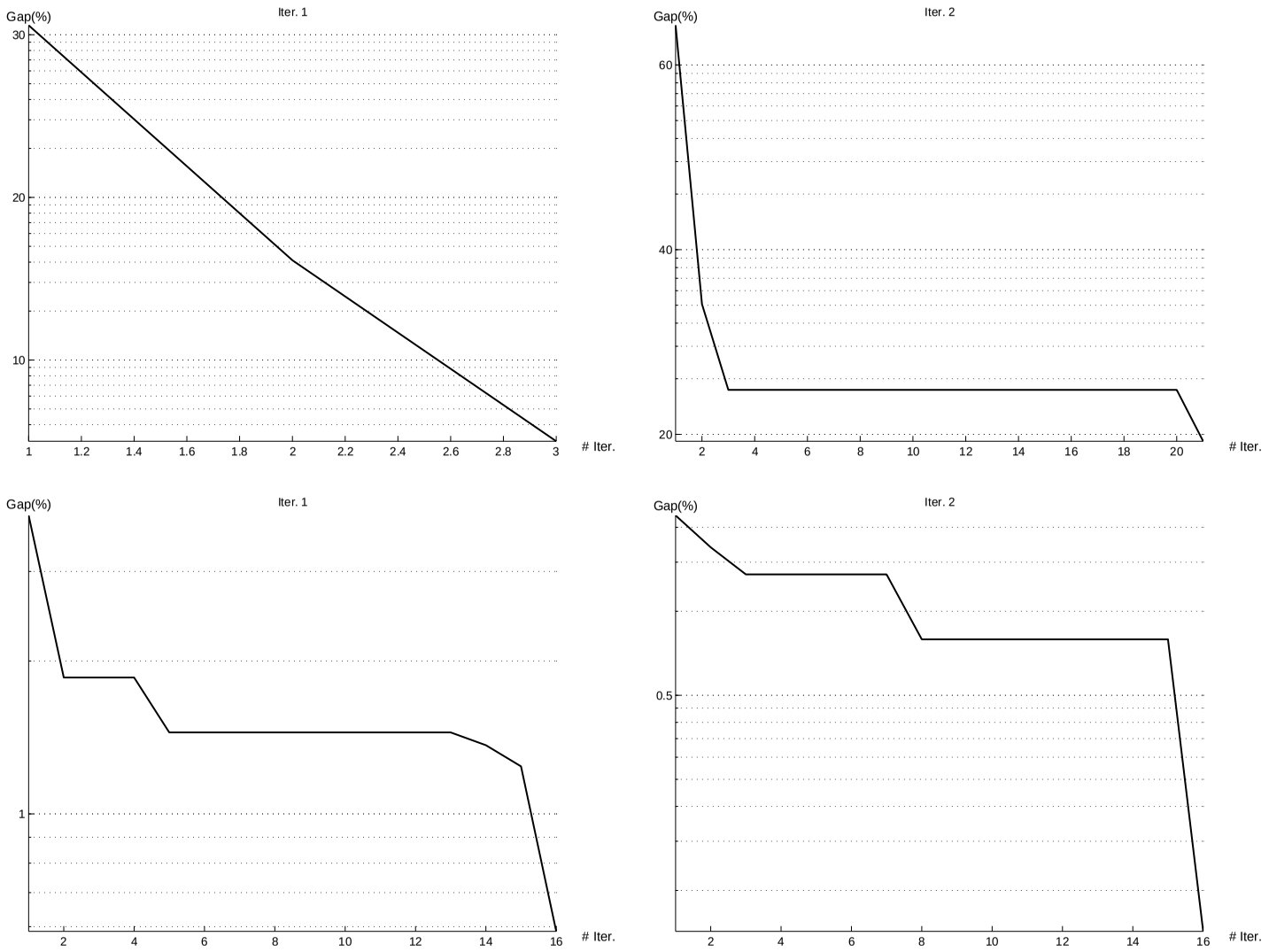


Figure 7.40: Gap Evolution graphs for the Benders scheme (BD) driven by the line splitting algorithm (LSA) with incremental load variant applied to the model under elastic demand with the 9-node network and four lines under construction. Top-left: gap evolution for the first iteration of the LSA in the first phase of the BD. Bottom-left: gap evolution for the first iteration of the LSA in the second phase of the BD. Top-right: gap evolution for the second iteration of the LSA in the first phase of the BD. Bottom-right: gap evolution for the second iteration of the LSA in the second phase of the BD.

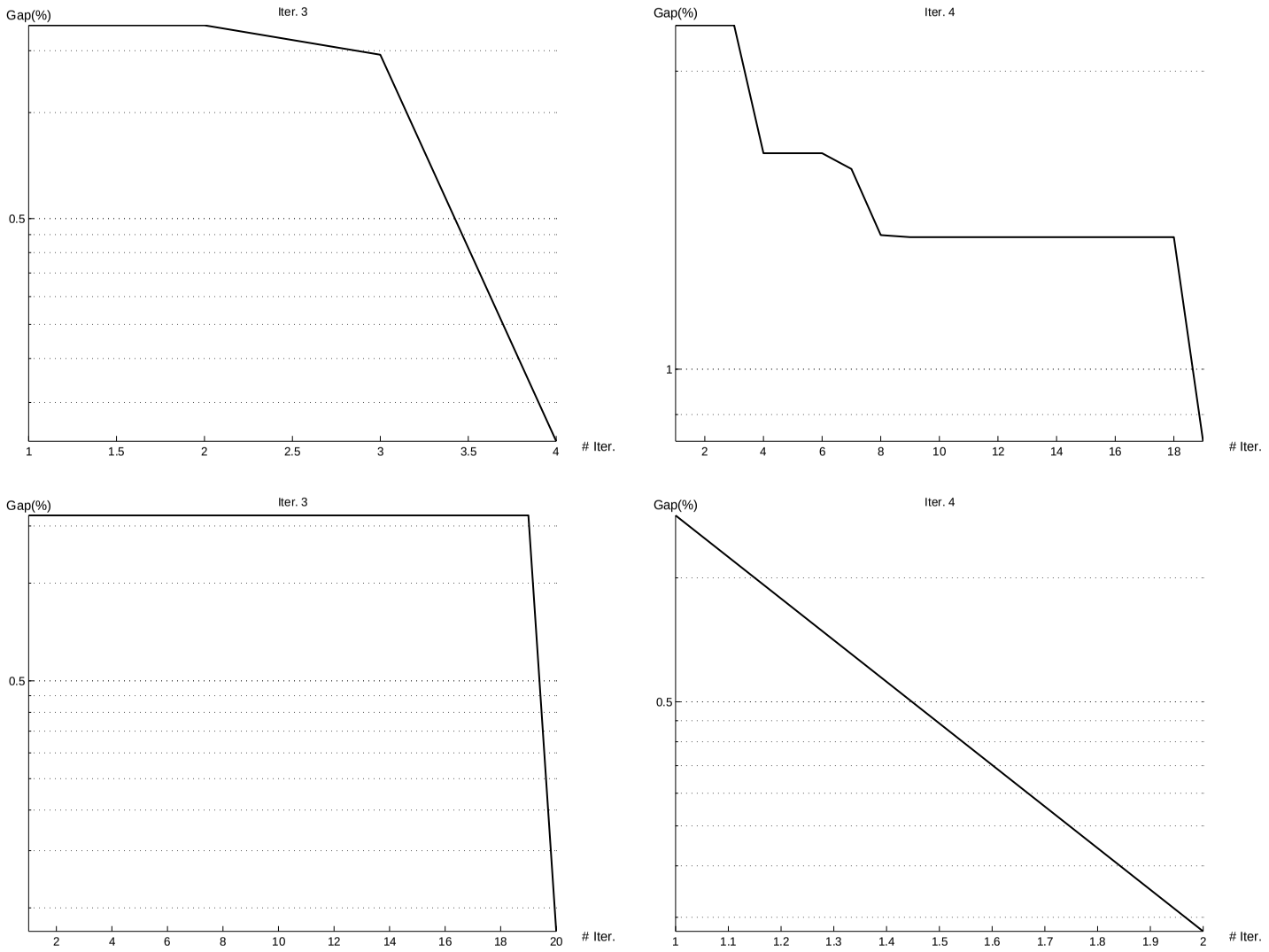


Figure 7.41: Graphs for evolution of the Benders scheme (BD) driven by the line splitting algorithm (LSA) with incremental load variant applied to the model under elastic demand with the 9-node network and four lines under construction. Top-left: gap evolution for the third iteration of the LSA in the first phase of the BD. Bottom-left: gap evolution for the third iteration of the LSA in the second phase of the BD. Top-right: gap evolution for the fourth iteration of the LSA in the first phase of the BD. Bottom-right: gap evolution for the fourth iteration of the LSA in the second phase of the BD.

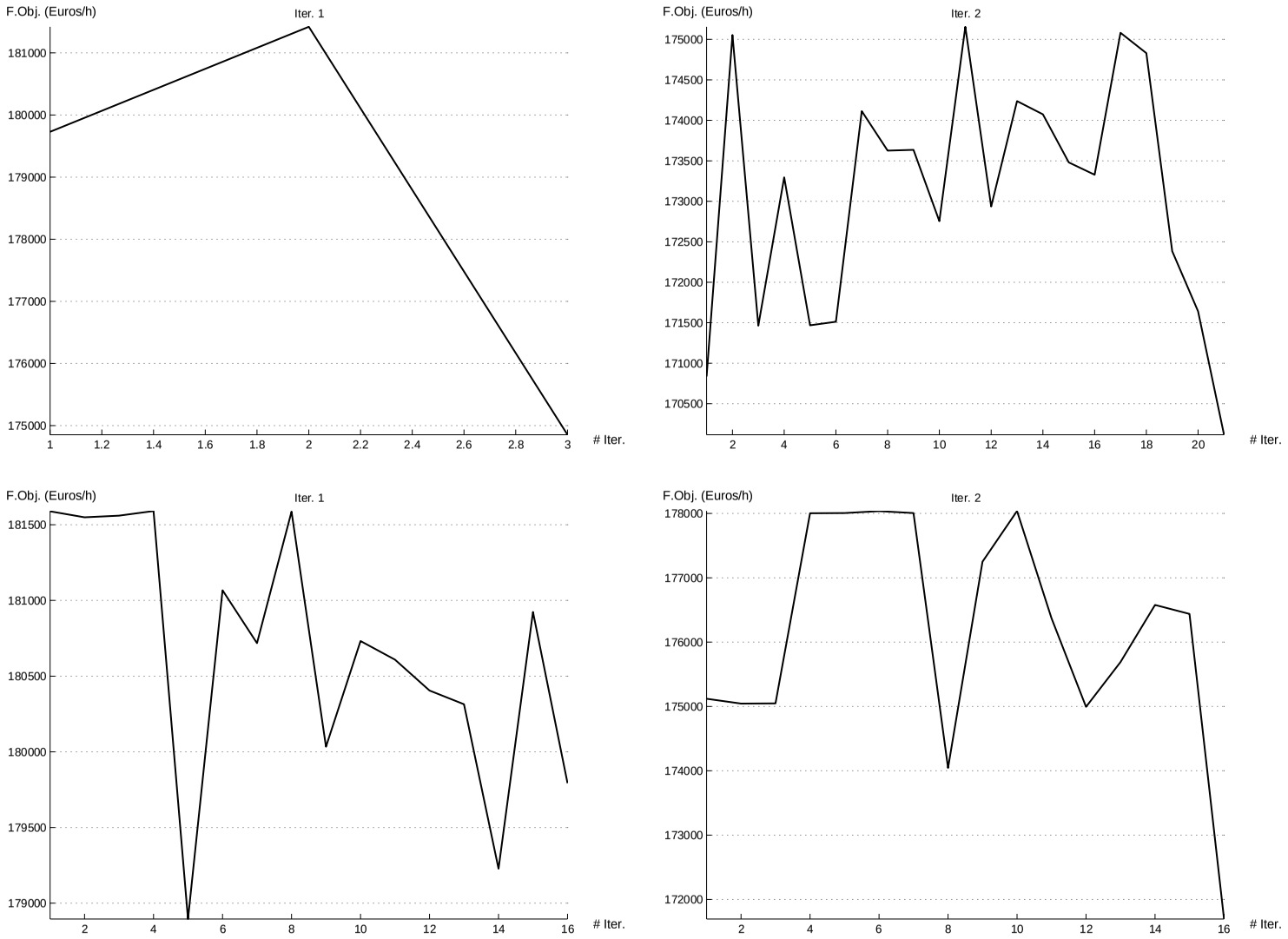


Figure 7.42: Graphs showing the evolution of the objective function for the Benders scheme (BD) driven by the line splitting algorithm (LSA) with incremental load variant applied to the model under elastic demand with the 9-node network and four lines under construction. Top-left: objective function evolution for the first iteration of the LSA in the first phase of the BD. Bottom-left: objective function evolution for the first iteration of the LSA in the second phase of the BD. Top-right: objective function evolution for the second iteration of the LSA in the first phase of the BD. Bottom-right: objective function evolution for the second iteration of the LSA in the second phase of the BD.

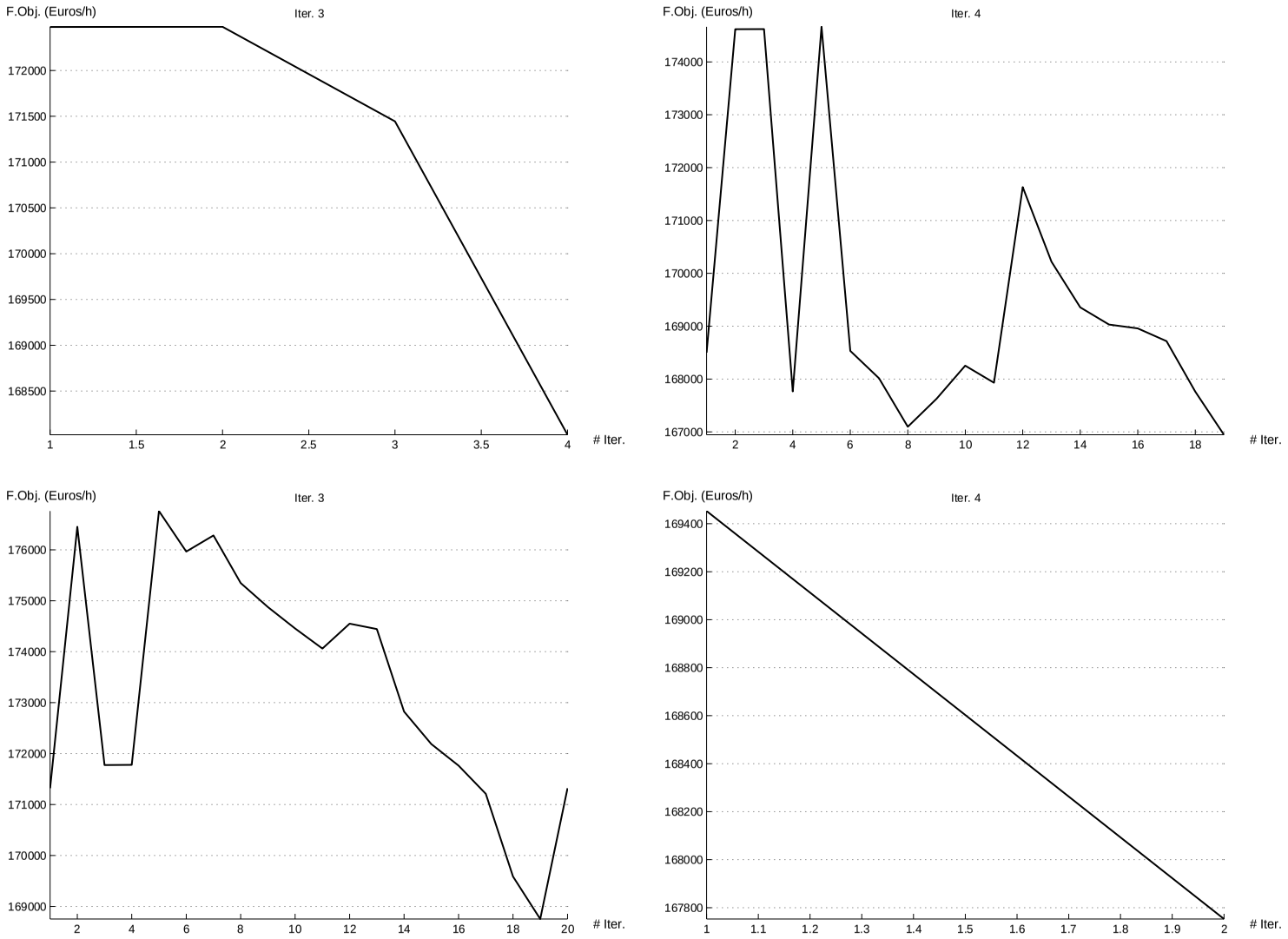


Figure 7.43: Graphs showing the evolution of the objective function for the Benders scheme (BD) driven by the line splitting algorithm (LSA) with incremental load variant applied to the model under elastic demand with the 9-node network and four lines under construction. Top-left: objective function evolution for the third iteration of the LSA in the first phase of the BD. Bottom-left: objective function evolution for the third iteration of the LSA in the second phase of the BD. Top-right: objective function evolution for the fourth iteration of the LSA in the first phase of the BD. Bottom-right: objective function evolution for the fourth iteration of the LSA in the second phase of the BD.

Chapter 8

Experimentation

This chapter presents the description of the two test networks which have been used throughout the previous chapters. It also includes two case study networks from which comprehensive experiments have been carried out using the routing model $M3$ and the inelastic demand version of the model. They demonstrate that the model is capable of solving to (near-)optimality real sized-networks in a reasonable computational time. The structure of the chapter is as follows. Firstly, we describe the test networks, including all the input data; secondly, we introduce the two case studies together with comprehensive results. Finally, we give general conclusions and recommendations on when and how to efficiently use each solving technique, base on the results analysis.

8.1 Experimental networks

This section is devoted to describing the whole set of networks applied to the $RTNPD$ model. They include two small test networks: a complete 6-node network and a sparse 9-node network, and two case study networks: a medium-sized network which represents the urban area of Seville (a not so small Spanish city located on the very south of the country), and a large-sized network which represents the urban area of Santiago de Chile (the capital city of Chile). In the following subsections, we will introduce each group of networks.

The results shown in the Seville and Santiago de Chile networks are not intended to be used in the current working transportation systems for the following reasons. Firstly, the input tuning data does not correspond to the one used by the operators and, secondly, the models employed are academic, and thus they lack certain practical and important considerations. We have simply developed approximate models whose accuracy is limited to the authors' knowledge and the available horizon time of this work. By using similar data, we are able to analyze hypothetical scenarios which have the same computational difficulty as real scenarios. The authors do not advise applying these models to the actual networks.

8.1.1 Test networks

In order to evaluate the model using the solving techniques explained throughout the previous chapters, two test networks have been proposed. The first one is a complete 6-node railway network shown in Figure 8.1 whereas the other one is a 9-node and 26-link railway network depicted in Figure 8.2. The latter has also been used in Laporte *et al* 2010 [109], but with some of the inputs changed and/or not considered. For that reason, we also report the whole set of parameters for this network. In both figures, there is a four component vector $(c_{ij}^m, d_{ij}^{TP}, c_{ij}^f, \bar{f}_{ij})$ attached to each link. The meaning of each vector component, from left to right, is as follows: the stretch allocation cost, the traveling distance for both ways, the service cost, and the link capacity (maximum number of vehicles per hour). Additionally, we find a number next to the node representing the station allocation cost (c_i^m). Infrastructure construction costs (c_{ij}^c and c_i^c) are not included

since they are considered to be 10 times the associated allocation costs.

The amount of demand is 177 and 373 thousand passengers per hour, respectively, and they are split into each possible node pair combination as follows:

$$G(W)^{6N} = \begin{pmatrix} - & 3 & 9 & 7 & 5 & 4 \\ 4 & - & 7 & 9 & 3 & 6 \\ 18 & 5 & - & 10 & 8 & 3 \\ 7 & 3 & 4 & - & 8 & 6 \\ 7 & 5 & 3 & 3 & - & 7 \\ 9 & 1 & 8 & 8 & 5 & - \end{pmatrix},$$

$$G(W)^{9N} = \begin{pmatrix} - & 3 & 9 & 7 & 5 & 4 & 2 & 2 & 2 \\ 4 & - & 4 & 9 & 3 & 6 & 1 & 3 & 3 \\ 10 & 7 & - & 10 & 8 & 3 & 3 & 3 & 4 \\ 7 & 3 & 4 & - & 8 & 6 & 7 & 6 & 6 \\ 5 & 5 & 3 & 3 & - & 7 & 4 & 6 & 3 \\ 9 & 1 & 8 & 8 & 5 & - & 4 & 10 & 7 \\ 3 & 2 & 8 & 7 & 5 & 5 & - & 6 & 5 \\ 2 & 3 & 4 & 6 & 7 & 9 & 6 & - & 7 \\ 2 & 3 & 4 & 6 & 4 & 7 & 5 & 7 & - \end{pmatrix}$$

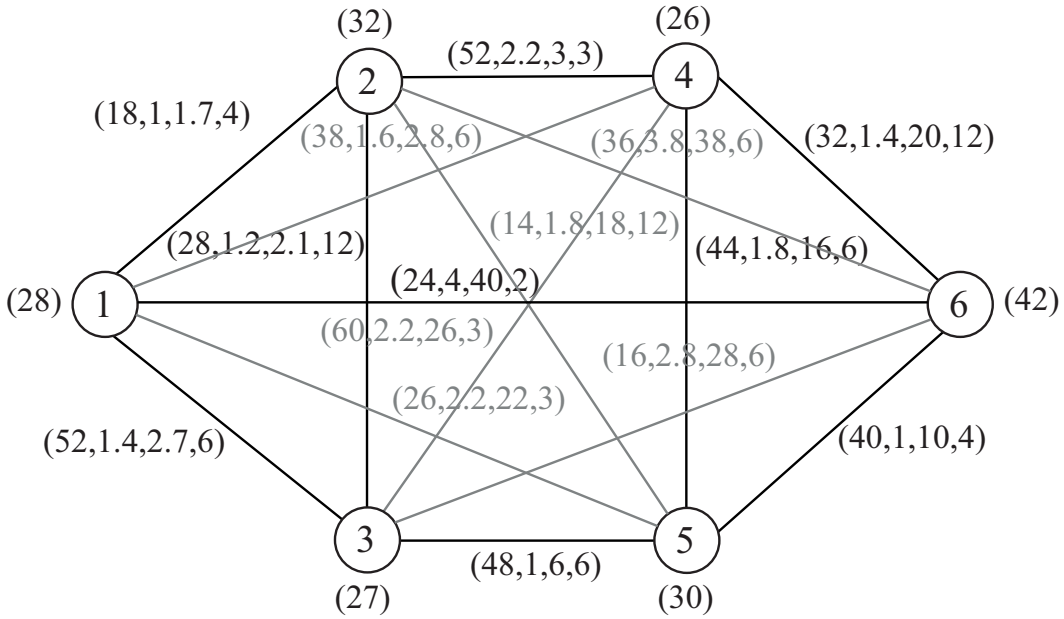


Figure 8.1: 6-node Railway Network.

The walking costs from node to node have been computed as the minimum road street distance divided by an average passenger's speed. The resulting matrices are as follows:

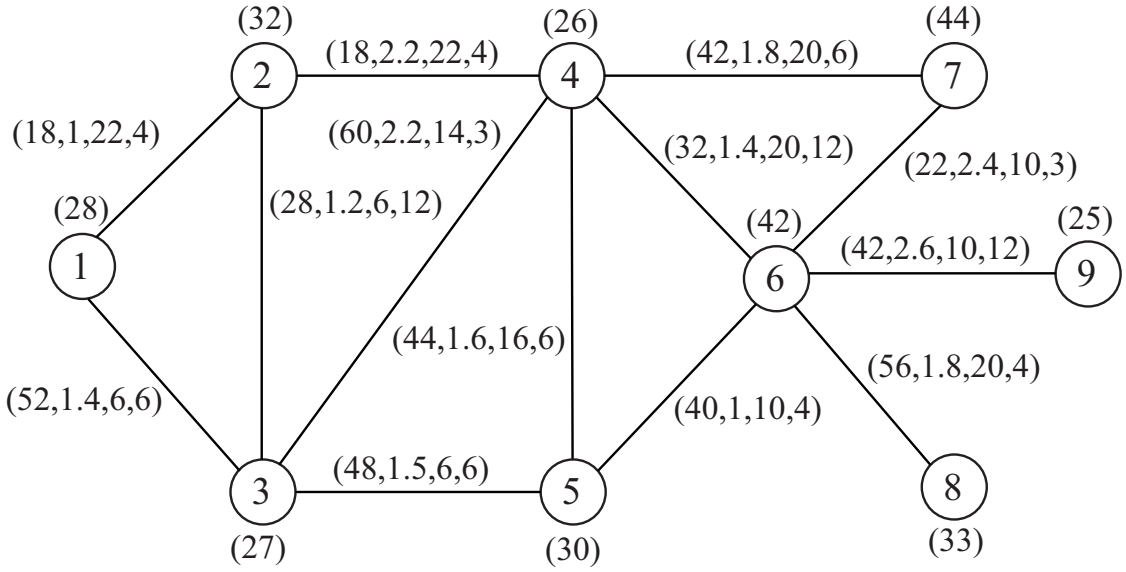


Figure 8.2: 9-node Railway Network.

$$T^{6N}_{COM} = \begin{pmatrix} - & 1.6 & 0.8 & 2 & 1.6 & 2.5 \\ 2 & - & 0.9 & 1.2 & 1.5 & 2.5 \\ 1.5 & 1.4 & - & 1.3 & 0.9 & 2 \\ 1.9 & 2 & 1.9 & - & 1.8 & 2 \\ 3 & 1.5 & 2 & 2 & - & 1.5 \\ 2.1 & 2.7 & 2.2 & 1 & 1.5 & - \end{pmatrix},$$

$$T^{9N}_{COM} = \begin{pmatrix} - & 1.6 & 0.8 & 2 & 1.6 & 2.5 & 4 & 3.6 & 4.6 \\ 2 & - & 0.9 & 1.2 & 1.5 & 2.5 & 3.2 & 3.5 & 4.5 \\ 1.5 & 1.4 & - & 1.3 & 0.9 & 2 & 3.3 & 2.9 & 3.9 \\ 1.9 & 2 & 1.9 & - & 1.8 & 2 & 2 & 3.8 & 4.1 \\ 3 & 1.5 & 2 & 2 & - & 1.5 & 3 & 2 & 3 \\ 2.1 & 2.7 & 2.2 & 1 & 1.5 & - & 2.5 & 3 & 2.5 \\ 3.9 & 3.9 & 3.9 & 2 & 3 & 2.5 & - & 2.5 & 2.5 \\ 5 & 3.5 & 4 & 4 & 2 & 3 & 2.5 & - & 2.5 \\ 4.6 & 4.5 & 4 & 3.5 & 3 & 2.5 & 2.5 & 2.5 & - \end{pmatrix}$$

The remaining parameters are the same for both networks. The planning features are as follows. The planning horizon ($\bar{h}$) is set to 3 hours, the total vehicle's capacity (q) to 550 passengers, and its average working speed (without considering service time at stops) rises to 80 km/h. The available vehicle fleet (B) is set to 20 vehicles and all of them have the same capacities. Moreover, each one has a setting cost (c^s_b) of 80 currency units and their average service time per station (t_s) is considered to be 1 minute.

Regarding the passenger features, the boarding and waiting-in-vehicle times per passenger unit (t_a and t_x, respectively) have been established at 9 seconds, whereas the alighting time per passenger unit (t_y) has been set to 2 seconds.

Focusing now on the objective function weights, the monetary time parameter θ = 4.86 monetary units/hour and the tradeoff factor β = 0.9, so that we attach more importance to the users' costs.

Finally, the infrastructure and planning budgets are set as follows: $\bar{c}_{net}$ = 4000 monetary units and $\bar{c}_{veh}$

= 800 monetary units, respectively. The cost of a new vehicle acquisition (c_b^a) is 20 currency units. Thus, up to 40 new vehicles can be acquired. No working lines have been considered.

All the aforementioned data has been used as input for the results reported in subsections 2.4, 4.2, 5.2, 6.6 and 7.5. We have not described in detail these networks in the subsections mentioned, since they would be repetitive.

8.1.2 Case study networks

In this section, we present two case study networks: a medium-sized network, which represents the urban area of Seville (a small Spanish city located on the very south of the country); and a large-sized network, which represents the urban area of Santiago de Chile (the capital city of Chile).

8.1.2.1 Seville Network

Seville Network is a medium-sized network consisting of 24 nodes, 264 links and 552 od-demand pairs. Figure 8.3 shows the layout of the network, where for the sake of clarification we have omitted all the data costs associated with the links and the nodes. Instead, we present them in the following pages 202 - 205.

The first page holds the link maintenance costs in a lower-triangular matrix, such that each i-j cell indicates the construction cost of the link (i, j) plus its inverse. They are expressed as multiples of the currency unit. The next page includes the traveling distance costs in a lower-triangular matrix as well. So each $i - j$ cell denotes the distance of the link (i, j) plus its inverse. They are expressed in kilometers. The third page contains the shortest walking time costs for each od-pair in a full matrix. They are expressed in hours. Finally, the fourth page contains the od-demand matrix expressed in thousand trips per hour.

Regarding the node costs, the following vector gives the maintenance costs of the 24 nodes, which can work as stations:

$$C_i^m = \{21.7, 21.7, 24.1, 17.9, 21.7, 17.9, 22.2, 22, 22, 24.3, 17.6, 21.9, 19.5, 22.9, 22, 24.4, 22.2 \ldots \\ \ldots, 22.4, 32.7, 37.4, 33.7, 29.1, 32.8, 31.9\}$$

Its construction costs as well as the link construction costs are 10 times its corresponding maintenance costs. The number of nodes and links which can be constructed are limited to an infrastructure budget of 15000 currency units.

Moving on to the planning data, the link capacities and service planning costs are set to $\bar{f}_{ij} = 12$ and $c_{ij}^f = 20,\ \forall (i, j) \in A_{TP}$, respectively. The 17-vehicle fleet is heterogeneous and its features are as follows. Each vehicle has an average speed of 30 km/h and a whole capacity of 275 passengers. The whole amount of service they carry out cannot exceed the planning horizon of 12 hours. If these 17 vehicles are not enough to satisfy the demand assigned to the public transportation network, some new vehicles can be acquired, but without surpassing the 800 currency units of the planning budget.

The remaining unmentioned input parameters are the same ones used in the test networks. They include the absence of working lines as well as the possibility that every node can work as a station, as shown above in the Figure 8.3.

Traveling distance cost matrix

$$\left(\begin{array}{cccccccccccc}
- & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.44 & - & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.88 & 0.64 & - & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1.02 & 0.98 & 0 & - & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1.02 & 0 & - & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & - & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1.12 & 0 & 0.98 & 0 & - & 0 & 0 & 0 & 0 & 0 \\
0.92 & 0.92 & 0.28 & 0 & 0.74 & 0 & 0.88 & - & 0 & 0 & 0 & 0 \\
0.92 & 0.92 & 0.28 & 0 & 0.74 & 0 & 0.84 & 0.12 & - & 0 & 0 & 0 \\
1.04 & 1.04 & 0.04 & 0 & 0.62 & 0 & 0.88 & 0.12 & 0.12 & - & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1.06 & 0 & 0 & 0 & 0 & - & 0 \\
0 & 0 & 0 & 0 & 0.08 & 0 & 0.34 & 0 & 0 & 1.06 & 0 & - \\
0 & 0 & 0 & 0 & 0.18 & 0 & 1.08 & 0.92 & 0.92 & 0.08 & 0 & 0.9 \\
0 & 1.06 & 0.62 & 0 & 0.88 & 0 & 0.86 & 0.78 & 0.66 & 0.78 & 0 & 0 \\
0 & 0 & 0.76 & 0 & 0 & 0 & 0 & 0.92 & 0.08 & 0.92 & 1.12 & 0 \\
0 & 0.94 & 0.5 & 0 & 0.1 & 0 & 1.1 & 0.66 & 0.54 & 0.66 & 0 & 0 \\
0 & 0 & 0.94 & 0 & 0.12 & 0 & 0.86 & 0.66 & 0.66 & 0.54 & 0 & 0.68 \\
0.94 & 0.5 & 0.14 & 0 & 1.16 & 0 & 0 & 0.42 & 0.42 & 0.54 & 0 & 0 \\
0 & 0 & 0.867 & 0 & 0.653 & 0 & 0.327 & 0.587 & 0.587 & 0.553 & 0 & 0.593 \\
0 & 0 & 0 & 0 & 0 & 0 & 0.708 & 0 & 0 & 0 & 1.08 & 0.888 \\
0 & 0.916 & 0.308 & 0 & 0.744 & 0 & 0.844 & 0.468 & 0.348 & 0.468 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0.795 & 0 & 0 & 0 & 0 & 0.265 & 0 \\
1.02 & 0.585 & 0.885 & 0.735 & 0 & 0 & 0 & 1.17 & 1.17 & 0 & 0 & 0 \\
0 & 0 & 0.72 & 0 & 0.46 & 0 & 0 & 0.56 & 0.68 & 0.56 & 0 & 0
\end{array}\right.$$

$$\left.\begin{array}{cccccccccccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
- & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.98 & - & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0.54 & - & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0.24 & 0.3 & - & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.26 & 0.76 & 0 & 0.92 & - & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0.56 & 0.7 & 0.44 & 1.08 & - & 0 & 0 & 0 & 0 & 0 & 0 \\
0.753 & 0.607 & 1.15 & 0.847 & 0.533 & 1.01 & - & 0 & 0 & 0 & 0 & 0 \\
0 & 0.808 & 1.08 & 0.928 & 0 & 0 & 1.03 & - & 0 & 0 & 0 & 0 \\
0.924 & 0.312 & 0.556 & 0.256 & 0.664 & 0.416 & 0.591 & 1.12 & - & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & - & 0 & 0 \\
0 & 1.15 & 0.615 & 0.905 & 0 & 0.745 & 0 & 0 & 1.16 & 0 & - & 0 \\
0.48 & 0 & 0 & 0 & 0.58 & 0.86 & 1.11 & 0 & 1.03 & 0 & 0 & -
\end{array}\right)$$

Walking time cost Matrix

-	0.22	0.44	0.51	0.83	1.91	0.88	0.46	0.46	0.52	1.38	1.05
0.22	-	0.32	0.49	0.83	1.69	0.88	0.46	0.46	0.52	1.16	1.05
0.44	0.32	-	0.81	0.51	1.47	0.56	0.14	0.14	0.02	0.94	0.73
0.51	0.49	0.81	-	1.32	1.04	1.37	0.95	0.95	1.01	0.87	1.54
0.83	0.83	0.51	1.32	-	1.06	0.49	0.37	0.37	0.31	1.07	0.04
1.91	1.69	1.47	1.04	1.06	-	1.11	1.55	1.49	1.55	0.53	1.02
0.88	0.88	0.56	1.37	0.49	1.11	-	0.44	0.42	0.44	0.07	0.17
0.46	0.46	0.14	0.95	0.37	1.55	0.44	-	0.06	0.06	1.02	0.59
0.46	0.46	0.14	0.95	0.37	1.49	0.42	0.06	-	0.06	0.96	0.59
0.52	0.52	0.02	1.01	0.31	1.55	0.44	0.06	0.06	-	1.02	0.53
1.38	1.16	0.94	0.87	1.07	0.53	0.07	1.02	0.96	1.02	-	0.87
1.05	1.05	0.73	1.54	0.04	1.02	0.17	0.59	0.59	0.53	0.87	-
0.92	0.92	0.06	1.41	0.09	1.65	0.54	0.46	0.46	0.04	1.12	0.45
0.75	0.53	0.31	0.94	0.44	1.16	0.43	0.39	0.33	0.39	0.63	0.06
0.82	0.06	0.38	0.67	0.65	1.09	0.07	0.46	0.04	0.46	0.56	0.87
0.69	0.47	0.25	0.82	0.05	1.22	0.55	0.33	0.27	0.33	0.69	0.72
0.79	0.79	0.47	1.28	0.06	1.54	0.43	0.33	0.33	0.27	1.01	0.34
0.47	0.25	0.07	0.74	0.58	1.44	0.63	0.21	0.21	0.27	0.91	0.08
0.753	0.753	0.433	1.24	0.327	1.27	0.163	0.293	0.293	0.277	0.743	0.297
1.15	0.934	0.714	1.21	0.844	0.756	0.354	0.794	0.734	0.794	0.538	0.444
0.594	0.458	0.154	0.948	0.372	1.32	0.422	0.234	0.174	0.234	0.786	0.592
1.51	1.29	1.07	1	1.2	0.398	0.757	1.15	1.09	1.15	0.133	0.927
0.512	0.292	0.443	0.367	0.953	1.4	1	0.583	0.583	0.642	0.868	1.17
0.68	0.68	0.36	1.17	0.23	1.83	0.72	0.28	0.34	0.28	1.03	0.63

0.92	0.75	0.82	0.69	0.79	0.47	0.753	1.15	0.594	1.51	0.512	0.68
0.92	0.53	0.06	0.47	0.79	0.25	0.753	0.934	0.458	1.29	0.292	0.68
0.06	0.31	0.38	0.25	0.47	0.07	0.433	0.714	0.154	1.07	0.443	0.36
1.41	0.94	0.67	0.82	1.28	0.74	1.24	1.21	0.948	1	0.367	1.17
0.09	0.44	0.65	0.05	0.06	0.58	0.327	0.844	0.372	1.2	0.953	0.23
1.65	1.16	1.09	1.22	1.54	1.44	1.27	0.756	1.32	0.398	1.4	1.83
0.54	0.43	0.07	0.55	0.43	0.63	0.163	0.354	0.422	0.757	1	0.72
0.46	0.39	0.46	0.33	0.33	0.21	0.293	0.794	0.234	1.15	0.583	0.28
0.46	0.33	0.04	0.27	0.33	0.21	0.293	0.734	0.174	1.09	0.583	0.34
0.04	0.39	0.46	0.33	0.27	0.27	0.277	0.794	0.234	1.15	0.642	0.28
1.12	0.63	0.56	0.69	1.01	0.91	0.743	0.538	0.786	0.133	0.868	1.03
0.45	0.06	0.87	0.72	0.34	0.08	0.297	0.444	0.592	0.927	1.17	0.63
-	0.49	0.74	0.59	0.13	0.67	0.377	0.894	0.462	1.25	1.04	0.24
0.49	-	0.27	0.12	0.38	0.28	0.303	0.404	0.156	0.762	0.573	0.67
0.74	0.27	-	0.15	0.61	0.35	0.573	0.538	0.278	0.693	0.308	0.74
0.59	0.12	0.15	-	0.46	0.22	0.423	0.464	0.128	0.823	0.453	0.61
0.13	0.38	0.61	0.46	-	0.54	0.267	0.784	0.332	1.14	0.912	0.29
0.67	0.28	0.35	0.22	0.54	-	0.503	0.684	0.208	1.04	0.373	0.43
0.377	0.303	0.573	0.423	0.267	0.503	-	0.517	0.295	0.876	0.876	0.557
0.894	0.404	0.538	0.464	0.784	0.684	0.517	-	0.56	0.596	0.841	1.07
0.462	0.156	0.278	0.128	0.332	0.208	0.295	0.56	-	0.918	0.581	0.514
1.25	0.762	0.693	0.823	1.14	1.04	0.876	0.596	0.918	-	1	1.43
1.04	0.573	0.308	0.453	0.912	0.373	0.876	0.841	0.581	1	-	0.802
0.24	0.67	0.74	0.61	0.29	0.43	0.557	1.07	0.514	1.43	0.802	-

Links maintenance cost matrix

-	0	0	0	0	0	0	0	0	0	0	0
44	-	0	0	0	0	0	0	0	0	0	0
88	64	-	0	0	0	0	0	0	0	0	0
102	98	0	-	0	0	0	0	0	0	0	0
0	0	102	0	-	0	0	0	0	0	0	0
0	0	0	0	0	-	0	0	0	0	0	0
0	0	112	0	98	0	-	0	0	0	0	0
92	92	28	0	74	0	88	-	0	0	0	0
92	92	28	0	74	0	84	12	-	0	0	0
104	104	40	0	62	0	88	12	12	-	0	0
0	0	0	0	0	106	0	0	0	0	-	0
0	0	0	0	80	0	34	0	0	106	0	-
0	0	0	0	18	0	108	92	92	80	0	90
0	106	62	0	88	0	86	78	66	78	0	0
0	0	76	0	0	0	0	92	80	92	112	0
0	94	50	0	100	0	110	66	54	66	0	0
0	0	94	0	12	0	86	66	66	54	0	68
94	50	14	0	116	0	0	42	42	54	0	0
0	0	86.6	0	65.4	0	32.6	58.6	58.6	55.4	0	59.4
0	0	0	0	0	0	70.8	0	0	0	107.6	88.8
0	91.6	30.8	0	74.4	0	84.4	46.8	34.8	46.8	0	0
0	0	0	0	0	79.6	0	0	0	0	26.6	0
102.6	58.6	88.6	73.6	0	0	0	116.6	116.6	0	0	0
0	0	72	0	46	0	0	56	68	56	0	0

0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
-	0	0	0	0	0	0	0	0	0	0	0
98	-	0	0	0	0	0	0	0	0	0	0
0	54	-	0	0	0	0	0	0	0	0	0
0	24	30	-	0	0	0	0	0	0	0	0
26	76	0	92	-	0	0	0	0	0	0	0
0	56	70	44	108	-	0	0	0	0	0	0
75.4	60.6	114.6	84.6	53.4	100.6	-	0	0	0	0	0
0	80.8	107.6	92.8	0	0	103.4	-	0	0	0	0
92.4	31.2	55.6	25.6	66.4	41.6	59	112	-	0	0	0
0	0	0	0	0	0	0	0	0	-	0	0
0	114.6	61.6	90.6	0	74.6	0	0	116.2	0	-	0
48	0	0	0	58	86	111.4	0	102.8	0	0	-

OD-Demand Matrix

-	23	23	23	23	46	23	23	23	46	46	23	46	23	46	46	46	23	78	78	78	78	78	78
23	-	23	23	23	46	23	23	23	46	46	23	46	23	46	46	46	23	78	78	78	78	78	78
27	27	-	27	27	55	27	27	27	55	55	27	55	27	55	55	55	27	94	94	94	94	94	94
15	15	15	-	15	31	15	15	15	31	31	15	31	15	31	31	31	15	53	53	53	53	53	53
23	23	23	23	-	46	23	23	23	46	46	23	46	23	46	46	46	23	78	78	78	78	78	78
19	19	19	19	19	-	19	19	19	16	16	19	16	19	16	16	16	19	24	24	24	24	24	24
24	24	24	24	24	48	-	24	24	48	48	24	48	24	48	48	48	24	81	81	81	81	81	81
23	23	23	23	23	46	23	-	23	46	46	23	46	23	46	46	46	23	78	78	78	78	78	78
23	23	23	23	23	46	23	23	-	46	46	23	46	23	46	46	46	23	78	78	78	78	78	78
43	43	43	43	43	36	43	43	43	-	36	43	36	43	36	36	36	43	54	54	54	54	54	54
19	19	19	19	19	16	19	19	19	16	-	19	16	19	16	16	16	19	24	24	24	24	24	24
23	23	23	23	23	46	23	23	23	46	46	-	46	23	46	46	46	23	78	78	78	78	78	78
26	26	26	26	26	21	26	26	26	21	26	21	-	21	26	26	26	21	32	32	32	32	32	32
24	24	24	24	24	50	24	24	24	50	50	24	50	-	50	50	50	25	84	84	84	84	84	84
34	34	34	34	34	28	34	34	34	28	28	34	28	34	-	28	28	34	43	43	43	43	43	43
43	43	43	43	43	36	43	43	43	36	36	43	36	43	36	-	36	43	54	54	54	54	54	54
36	36	36	36	36	30	36	36	36	26	30	36	30	36	30	30	-	36	45	45	45	45	45	45
23	23	23	23	23	46	23	23	23	46	46	23	46	23	46	46	46	-	78	78	78	78	78	78
56	56	56	56	56	61	56	56	56	61	61	56	61	56	56	56	56	61	-	55	55	55	55	55
73	73	73	73	73	79	73	73	73	79	79	73	79	73	79	79	79	73	72	-	72	72	72	72
60	60	60	60	60	65	60	60	60	65	65	60	65	60	65	65	65	60	58	58	-	58	58	58
43	43	43	43	43	46	43	43	43	46	46	43	46	43	46	46	46	43	42	42	42	-	42	42
56	56	56	56	56	61	56	56	56	61	61	56	61	56	61	61	61	56	55	55	55	55	-	55
56	56	56	56	56	61	56	56	56	61	61	56	61	56	61	61	61	56	55	55	55	55	55	-

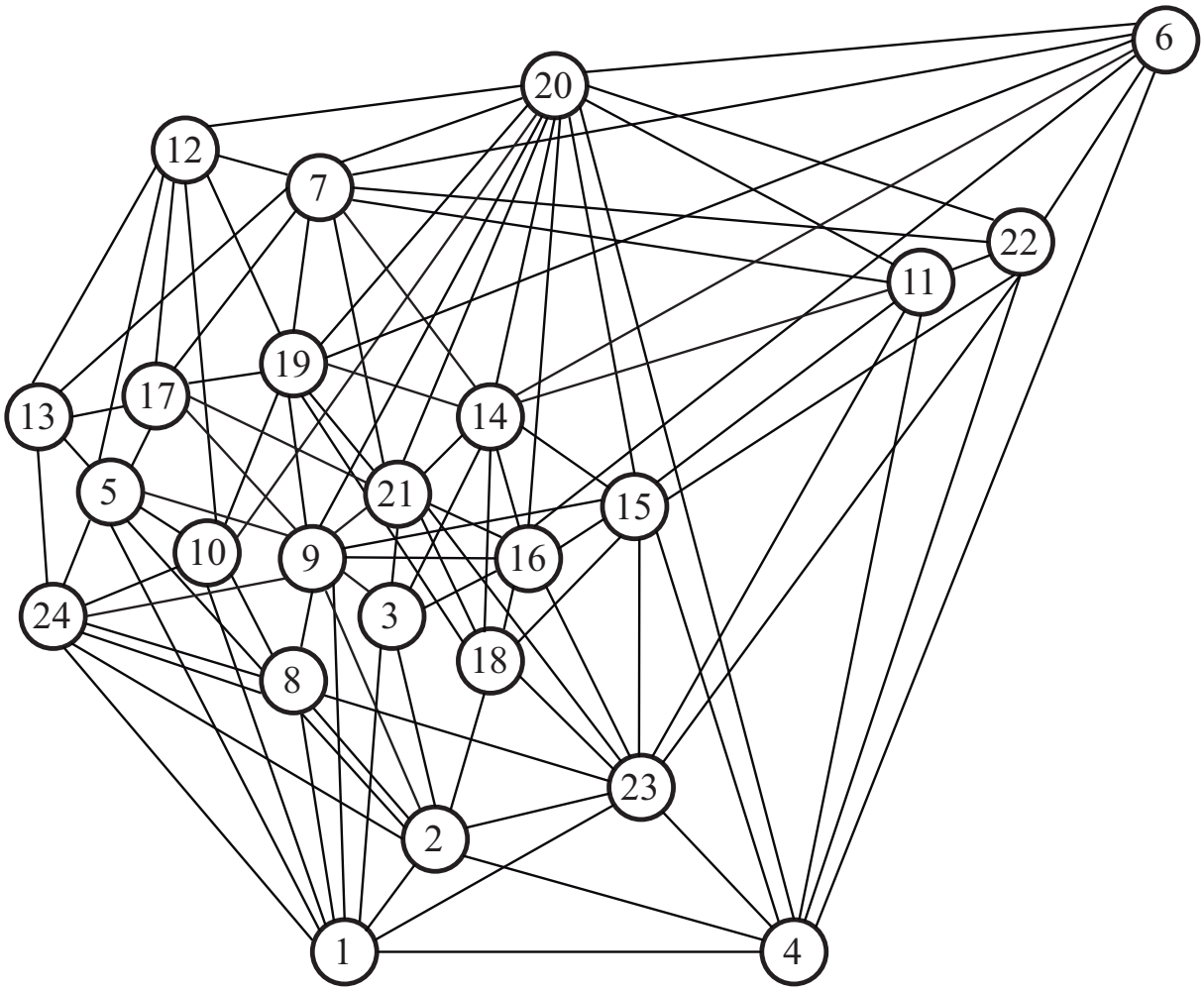


Figure 8.3: Seville Network.

8.1.2.2 Results

This subsection presents the results obtained from the Seville network by means of routing model 3, where a predefined set of candidate line corridors (Λ) are given as inputs to the inelastic demand version of the model. Regarding resources, we have used an R5500 working station with Intel(R) processor Xeon(R) CPU E5645 2.40 GHz and 48 Gbytes of RAM. The model has been coded in AMPL, using a CPLEX v.12.4.0 solver with a time limit of 24 hours.

This Λ set can be generated by taking into account all-to-all routing nodes $i \in N_{TP}^{N}$, as stated in chapter 4. However, this involves generating an extremely large Λ, especially for the corridors with rectilinear topology. For instance, taking only $K_r = 2$ entails $\Lambda_r = 552 \times 2 = 1104$ rectilinear corridors; thus some node reduction needs to be performed. At the top of figure 8.4, we show two histograms related to the demand distribution expressed by o-d pairs and by attraction nodes, respectively. From both graphs, it seems that the demand distribution is quite homogeneous, and we can therefore remove some od-pairs and attraction nodes which are not significant. To do this, we on the boxplots of both distributions at the bottom of the same figure. Moreover, we show the values of its corresponding third quartiles (Q_3 = 56 and Q_3 = 2335 thousand of trips/hour, respectively). These values are used to drop od-pairs from the demand and from the demand attraction node sets, those elements whose associated demand is lower. If we carried out this operation, we yield the reduced amounts of 160 od-demand pairs and 6 demand nodes, respectively.

Finally, we take the associated routing nodes from the two reduced sets. The routing nodes are the extreme candidate nodes for the corridor generation algorithm (CGA). As for the user behavioral rules, parameters ϕ_{max} and Δ_{min} have been set to the values recommended by Zijpp & Catalano [174]. Having performed several runs on the CGA, we found that the best tradeoff between the size and homogeneity of Λ was using $K_r = 3$ and $K_c = 5$, resulting in a total of 493 corridors from which 478 are rectilinear and 15 circular.

The computed Λ set has been used under three different scenarios in which one, two and three new lines, respectively, were under construction. Moreover, each scenario was solved six times (except for the first scenario, which was only solved twice) using the following combinations: the branch & bound of CPLEX without the line splitting algorithm (LSA), the Benders decomposition without LSA, the branch & bound of CPLEX with the two variants of the LSA, and the Benders decomposition with the two variants of the LSA. The next figures show features of the main solution, as well as the performance of each algorithmic combination.

Figure 8.5 contains the optimal network layout solution for the three scenarios where the line segments are thicker and in different colors. The black color is associated with line 1, whereas the blue and red colors represent lines 2 and 3, respectively. Additionally, the common links and nodes have been depicted in green. The graph at the top shows the solution corresponding to scenario 1, whereas the graph in the middle contains the one related to scenario 2. Finally, the graph at the bottom depicts the solution associated with scenario 3. We can see from this that the chosen corridors for the common lines ($L1$ in scenarios 1 and 2, $L1$ and $L2$ in scenarios 2 and 3) are the same and have a rectilinear topology. Regarding the role of the nodes, all those belonging to line 1 are stations in operation, whatever the scenario is; whereas in lines 2 and 3, common nodes 9 and 11 are just passing points.

Moving on to planning features, figure 8.6 analyzes the main line planning features. The graph at the top shows the total line capacities expressed as the number of vehicles per hour, whereas the graphs at the bottom depict their average occupancies. From left to right, we have the line capacity occupancies of scenarios 1, 2 and 3, respectively. According to them, line capacities vary as the number of lines increases, except for line 1 in scenarios 2 and 3, which remains constant. This line also has the highest capacity and in fact doubles the capacity of line 2 in scenario 2. Moreover, the capacities of lines 2 and 3 are quite moderate. As for the capacity utilities, their occupancy levels are high in scenarios 1 and 3 (around 80 %), whereas in scenario 2 they are moderate (around 60%). Furthermore, the capacity utility of line 1 goes down in scenarios 2 and 3 (to around 14%), due to the construction of additional lines. Thus, in scenarios 2 and 3 the demand is more balanced among the lines.

Figure 8.7 describes the demand coverage. It consists of two graphs. The one at the top shows the demand modal splitting between the public transportation network and the pedestrian network. The one at the bottom depicts the splitting between the demand directly served and the one performing some transfers. From them, we can see that demand using the public transportation network is high in all scenarios ($\geq 75\%$). However, the number of transfers decreases as the number of constructed lines increases (scenario 1 - 63.5%, scenario 2 - 57.5%, scenario 3 - 56.1%). Complementary to these figures, the following figure 8.8 shows the histograms of the total demand time distribution. From top to bottom, we have the ones related to scenarios 1, 2 and 3, respectively. Notice that more portion of the demand concentrates on the lowest time intervals as the number of constructed lines increases. However, there is a tiny portion of demand which still remains in high time intervals.

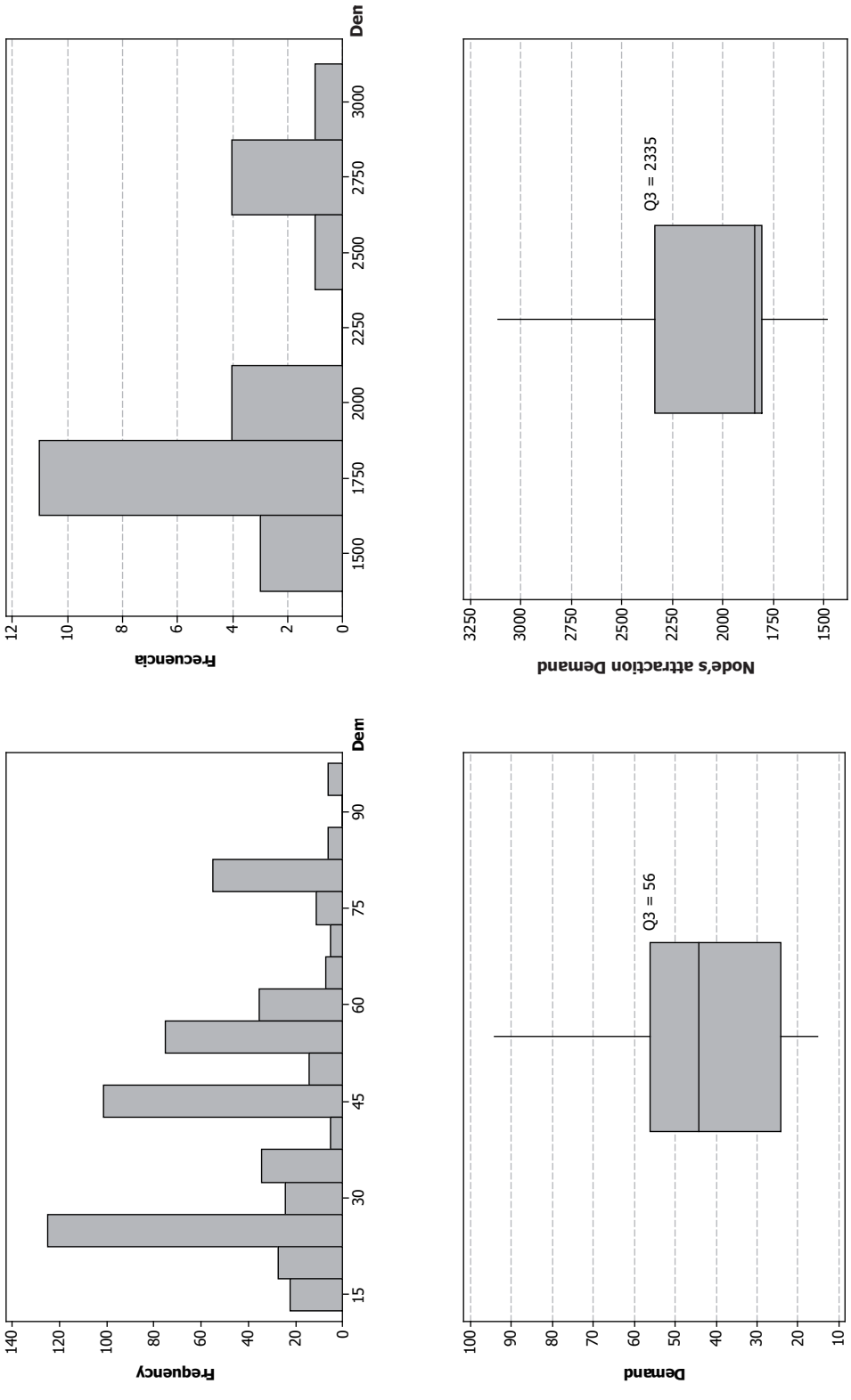


Figure 8.4: Demand profile of the Seville Network. At the top-left, the histogram of the demand distribution. At the bottom-left, the boxplot of the demand. At the top-right, the histogram of the node's attraction demand distribution. At the bottom-right, the boxplot of the node's attraction demand

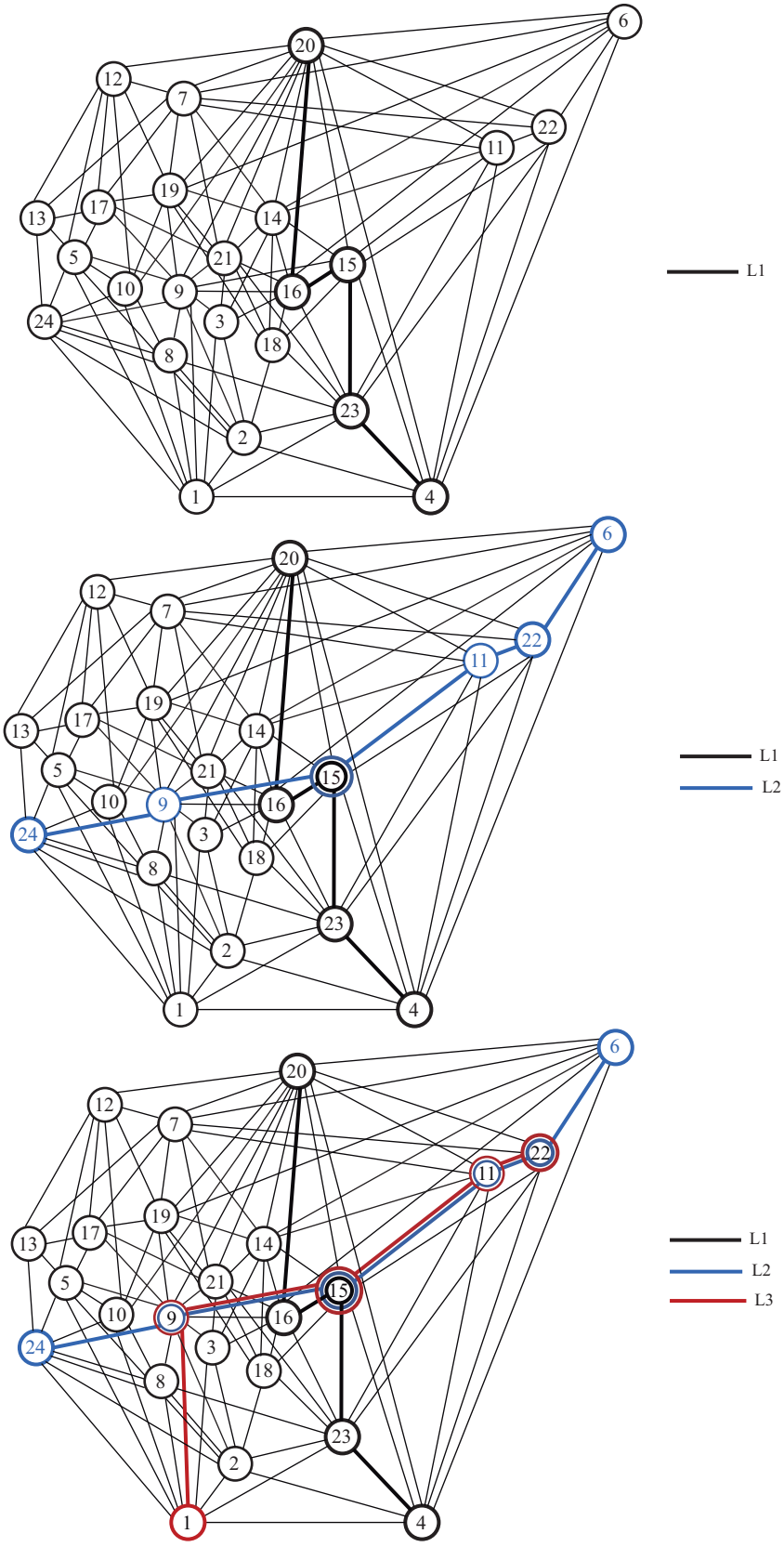


Figure 8.5: Network layout solution for the different scenarios tested on the Seville Network. At the top, the solution for scenario 1 with only one line under construction. In the middle, the solution for scenario two with two lines. At the bottom, the solution for scenario 3 with three lines.

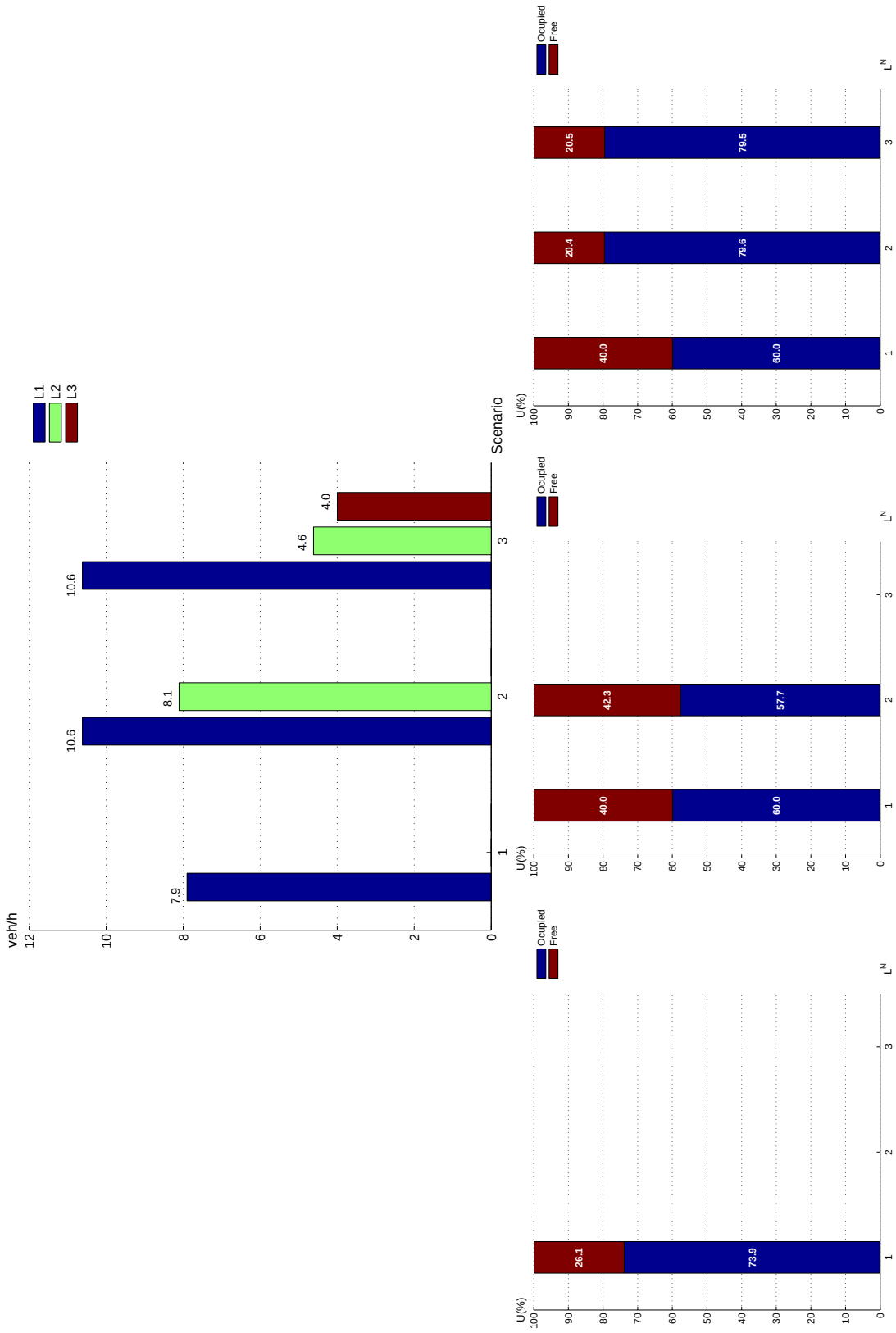


Figure 8.6: Overview of the main planning features for the Seville Network. At the top, line capacities grouped in close bars for each scenario. At the bottom-left, average line occupancies for scenario 1. At the bottom-middle, average line occupancies for scenario 2. At the bottom-right, average line occupancies for scenario 3.

Let's now move on to the general results. Figure 8.9 describes the general results for the tests performed on Seville's network. It consists of six graphs. The two on the left depict the optimal or best global objective function value (2.1) found so far. The two in the middle depict the total amount of CPU time consumed and the two on the right contain the distribution of CPU time among the Benders subproblems for the direct resolution of the model (without line splitting algorithms). The bars on the graphs showing the objective function and CPU times are grouped by solving techniques and sorted by scenario. From left to right, we have the values for scenarios 1, 2 and 3, respectively. Additionally, the graphs at the top are related to all methods where the Branch & Bound of CPLEX was used to solve each $MILP$, whereas the ones at the bottom are associated with those methods which employed the Benders decomposition as the $MILP$ solver. From them, we can see that best results are obtained from the combination technique of using a line splitting algorithm with non-incremental demand plus the CPLEX as an $MILP$ solver. Furthermore, whatever the $MILP$ solver is, the two LSA variants provide a good tradeoff between solution quality and CPU time. From the two LSA variants, it seems that non-incremental is slightly better than incremental since it gives the same objective function value but in less CPU time. Regarding the $MILP$ technique analysis, the Benders decomposition provides the worst CPU times. When it is used with no line splitting in scenarios 2 and 3, it does not convergence. Notice that it reaches the time limit horizon. This bad performance is due mainly to the resolution of the master problem, which consumes almost the total time (around the 97%, according to the graph at the bottom-right).

To conclude, we include figures 8.10 - 8.20, which show the Benders gap and global objective function convergence for the different scenarios and LSA variants. From them, we can see that the algorithm converges to a small gap ($< 1\%$), although quite slowly. The evolution of the global objective function is quite erratic since the first iterations deliver values either near to the optimal or to the best one found, then it moves to the worst values before going back to the initial values towards the end. At this point, the objective function improves faster and reaches the (nearly-) optimal solution. This evolution is typical of the Benders algorithm, although some authors have been working on that drawback. For instance, Saharidis *et al* [157] generate a set of Benders cuts (apart from the traditional ones), such that more master's variables are constrained. Sherali & Lunday [160] define the notion of maximal non-dominated cuts and allow obtaining tighter cuts. This field of research deserves more attention, and will be the subject of further research.

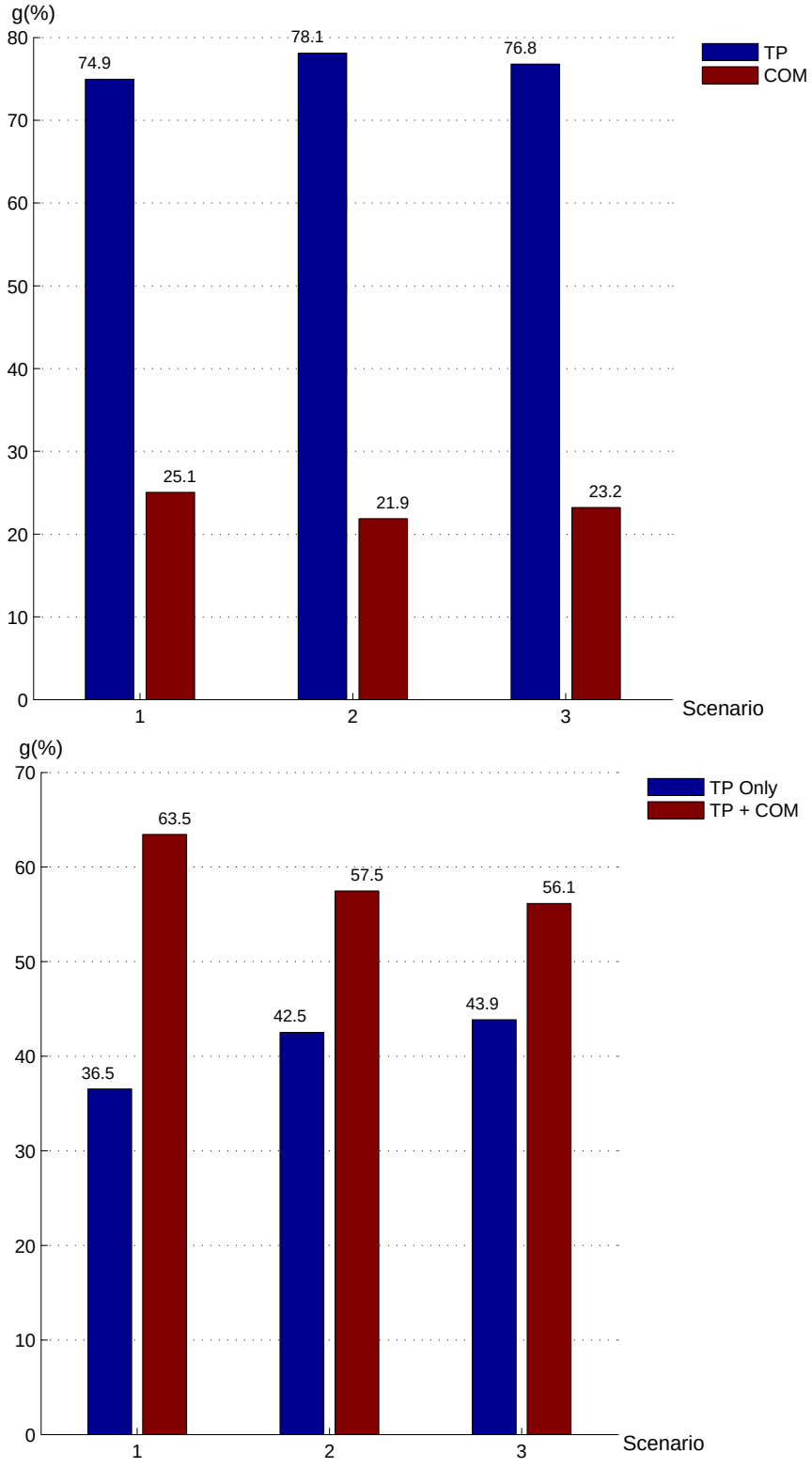


Figure 8.7: Demand Coverage details for the different scenarios tested on the Seville Network. At the top, the modal demand splitting. At the bottom, the quality of service for the demand going through public transportation.

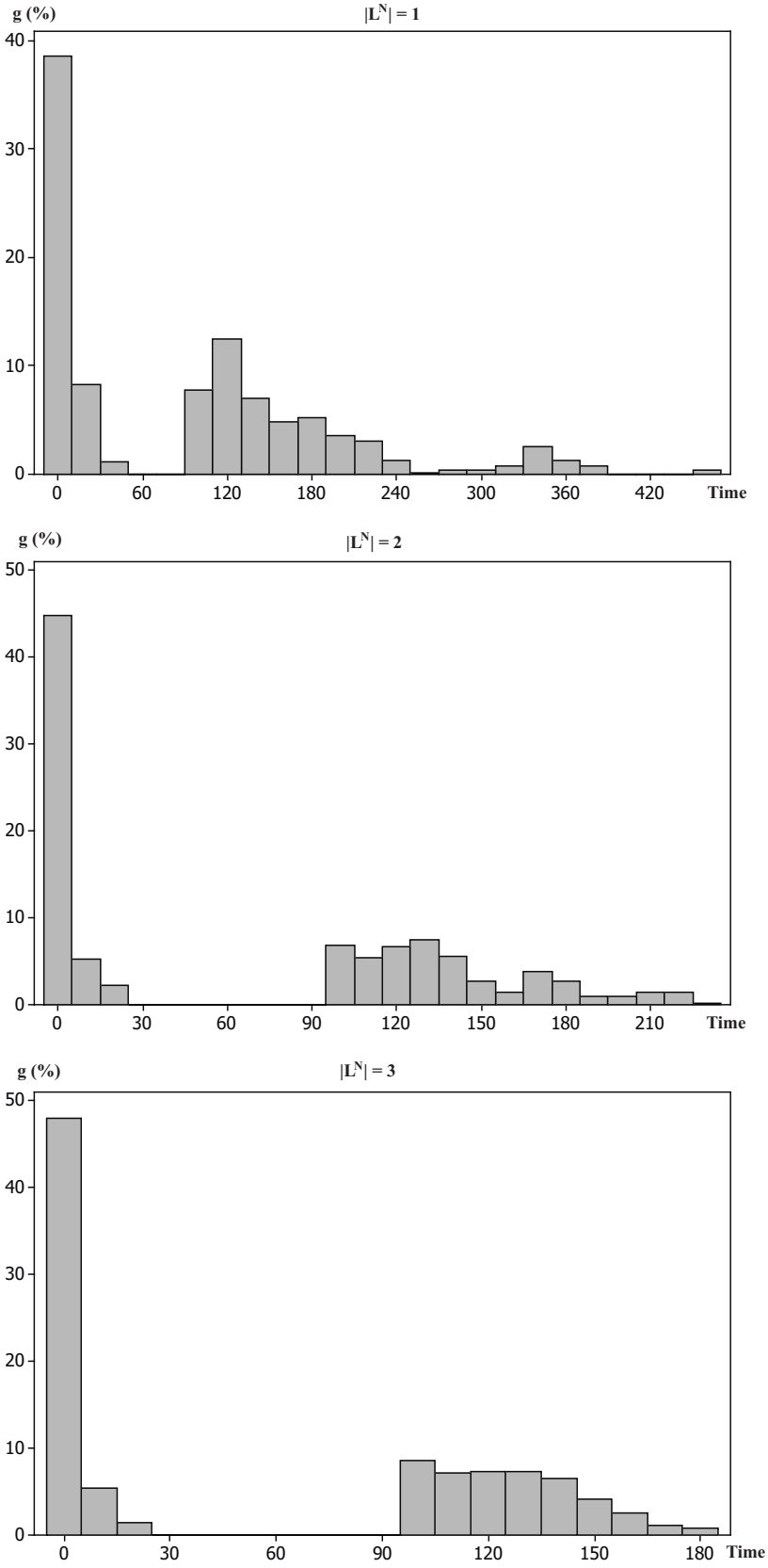


Figure 8.8: Histograms of the demand time distribution for Seville's network. At the top, the demand time distribution for scenario 1. In the middle, the demand time distribution for scenario 2. At the bottom, the demand time distribution for scenario 3.

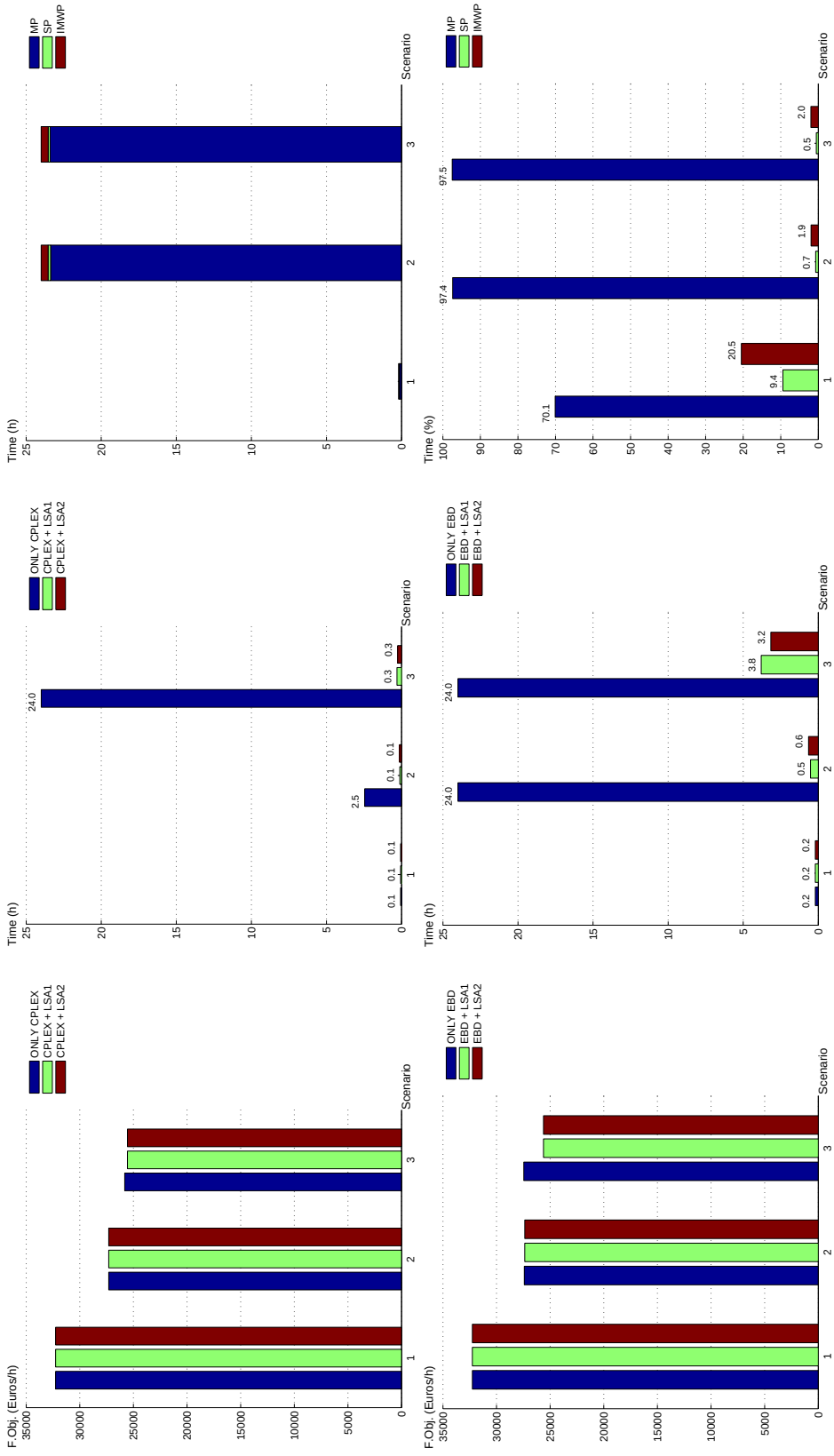


Figure 8.9: General Results on the different scenarios for the Seville Network. At the top-left and top-middle, the best global objective function value and the total CPU time, respectively, for each line splitting technique using Benders decomposition as the MILP solving technique. At the bottom-left and bottom-middle, the optimal global objective function value and the total CPU time, respectively, for each line splitting technique using CPLEX as the MILP solving technique. At the top-right and bottom-right, the detailed CPU times of each Benders subproblem in absolute and relative values, respectively, with no line splitting.

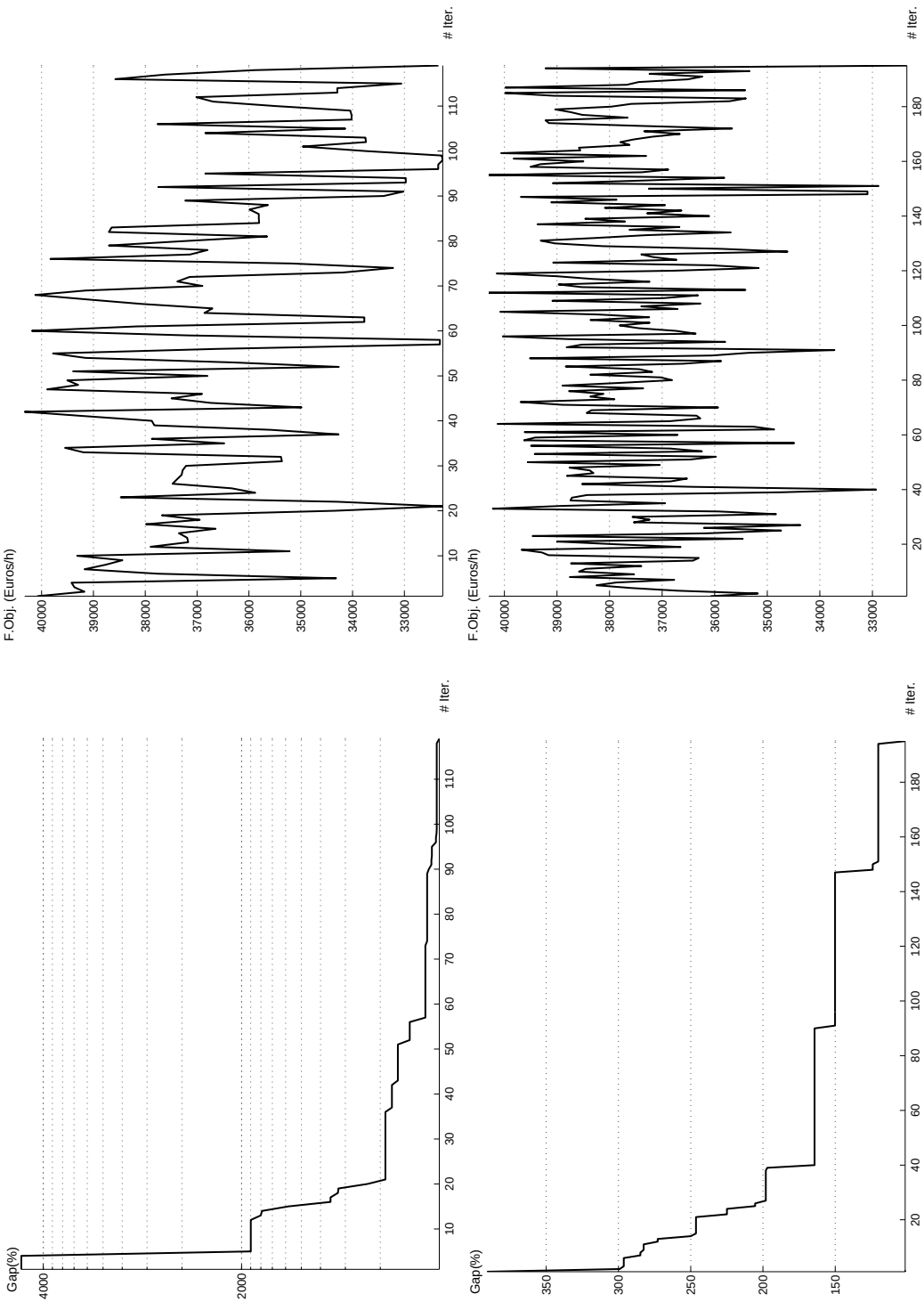


Figure 8.10: Benders convergence details in the first scenario for the Seville Network. At the top-left, gap evolution for the phase 0. At the top-right, global objective function evolution for the phase 0. At the bottom-left, gap evolution for the phase 1. At the bottom-right, global objective function evolution for the phase 1.

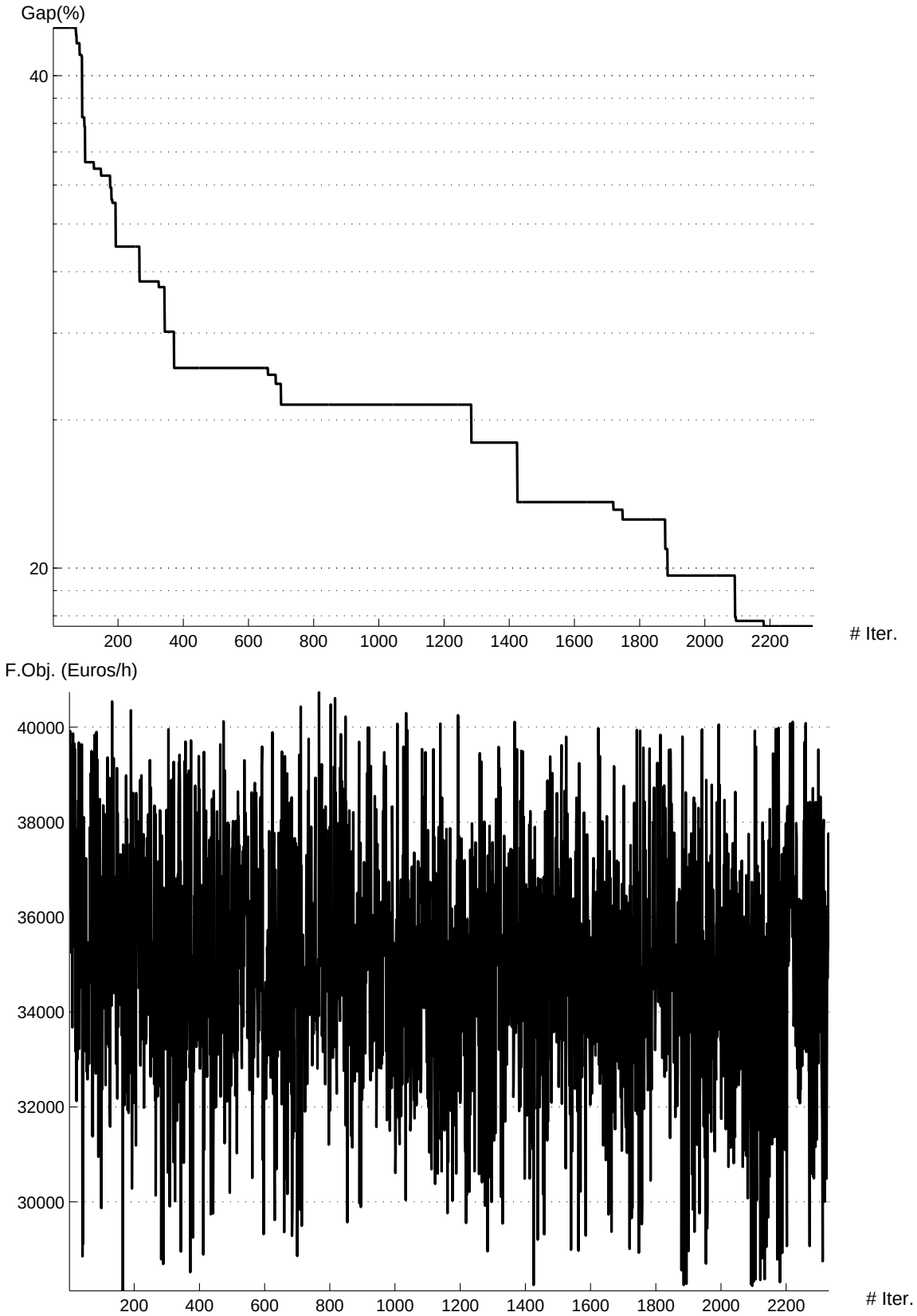


Figure 8.11: Benders convergence details in the second scenario for the Seville Network. At the top, gap evolution for the phase 0. At the bottom, gap evolution for the phase 0.

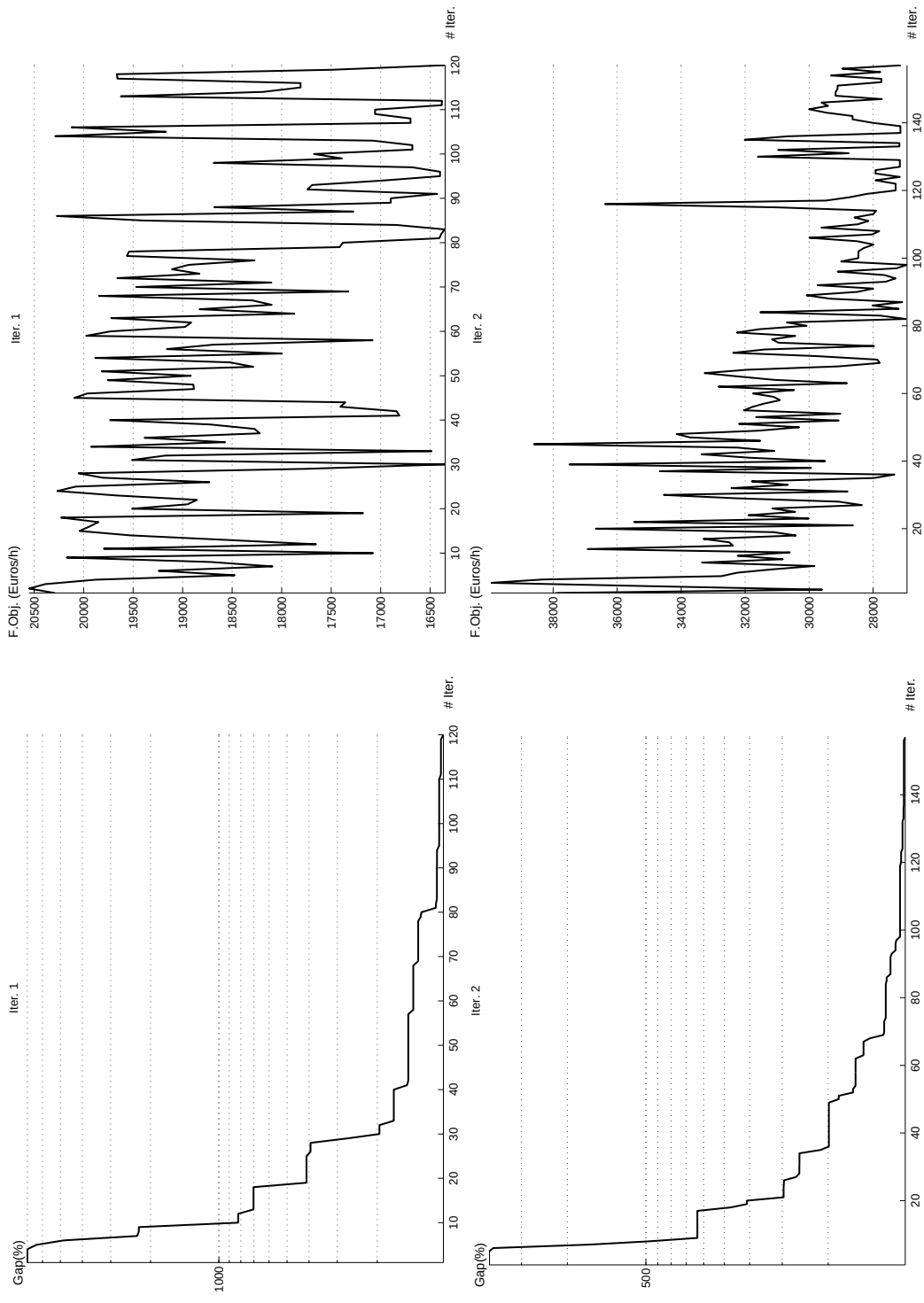


Figure 8.12: Benders phase 0 convergence details under the incremental variant of the LSA in the second scenario for the Seville Network. At the top-left, gap evolution for the first iteration. At the top-right, global objective function evolution for the first iteration. At the bottom-left, gap evolution for the second iteration. At the bottom-right, global objective function evolution for the second iteration.

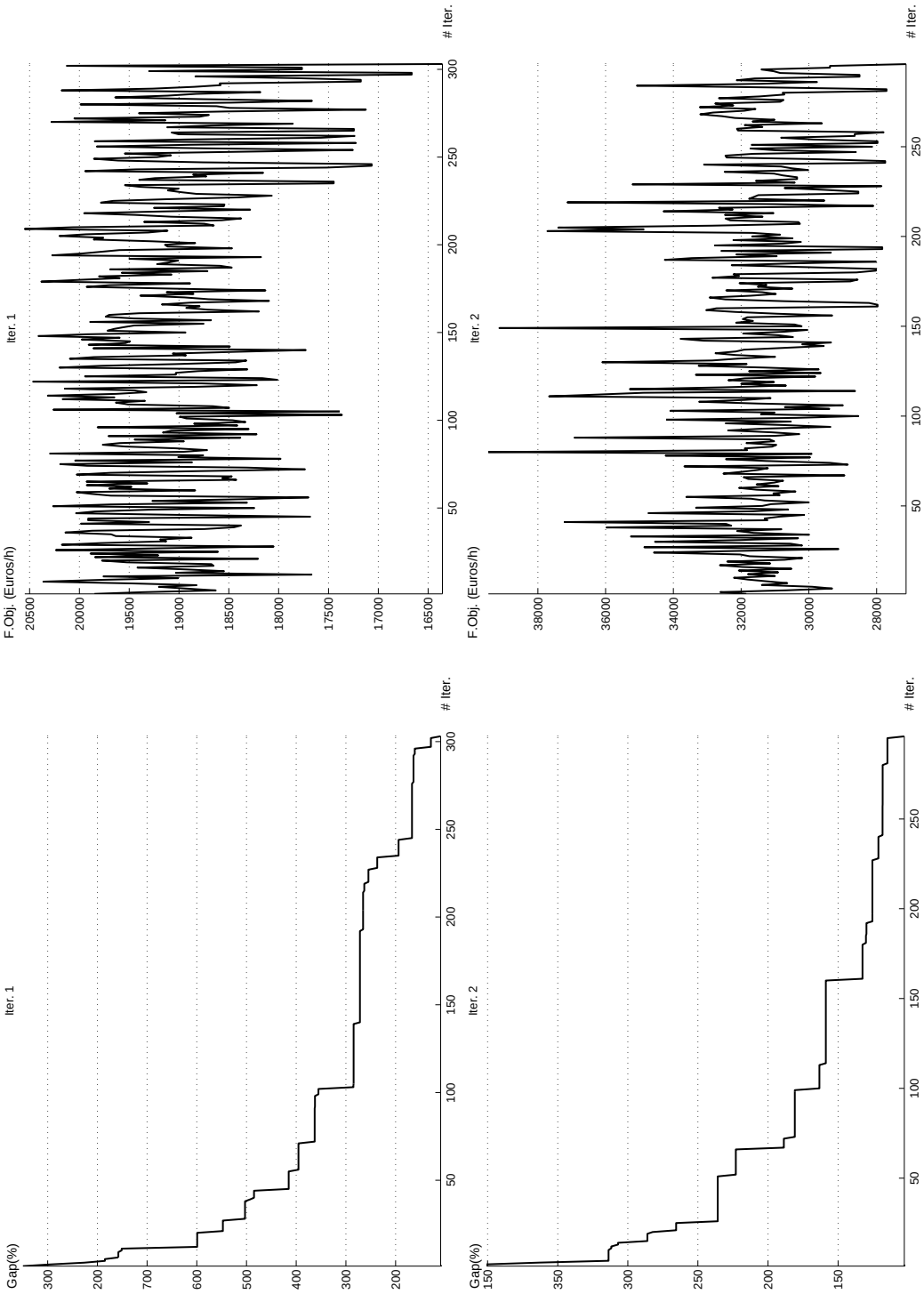


Figure 8.13: Benders phase 1 convergence details under the incremental variant of the LSA in the second scenario for the Seville Network. At the top-left, gap evolution for the first iteration. At the top-right, global objective function evolution for the first iteration. At the bottom-left, gap evolution for the second iteration. At the bottom-right, global objective function evolution for the second iteration.

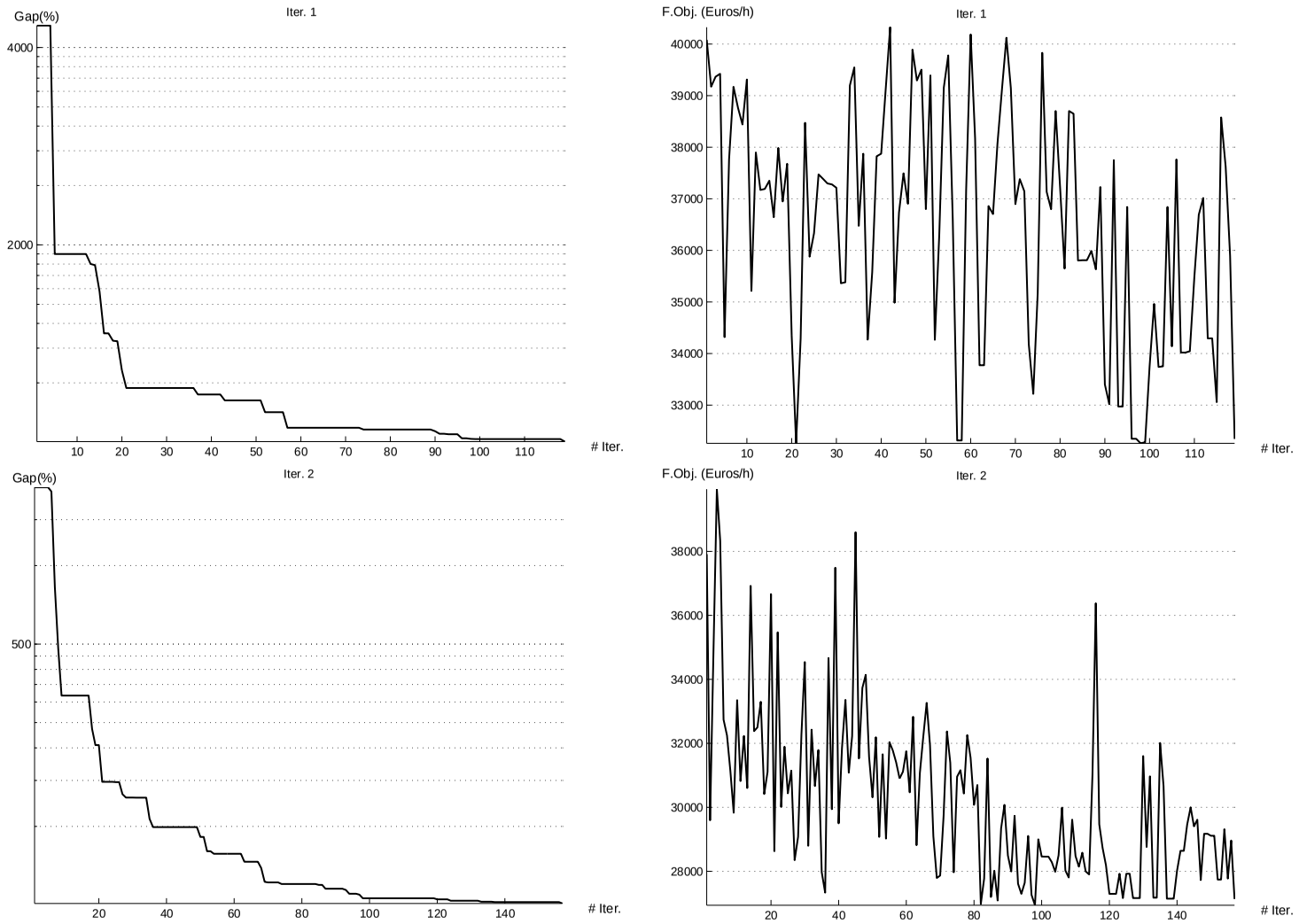


Figure 8.14: Benders phase 0 convergence details under the non-incremental variant of the LSA in the second scenario for the Seville Network. At the top-left, gap evolution for the first iteration. At the top-right, global objective function evolution for the first iteration. At the bottom-left, gap evolution for the second iteration. At the bottom-right, global objective function evolution for the second iteration.

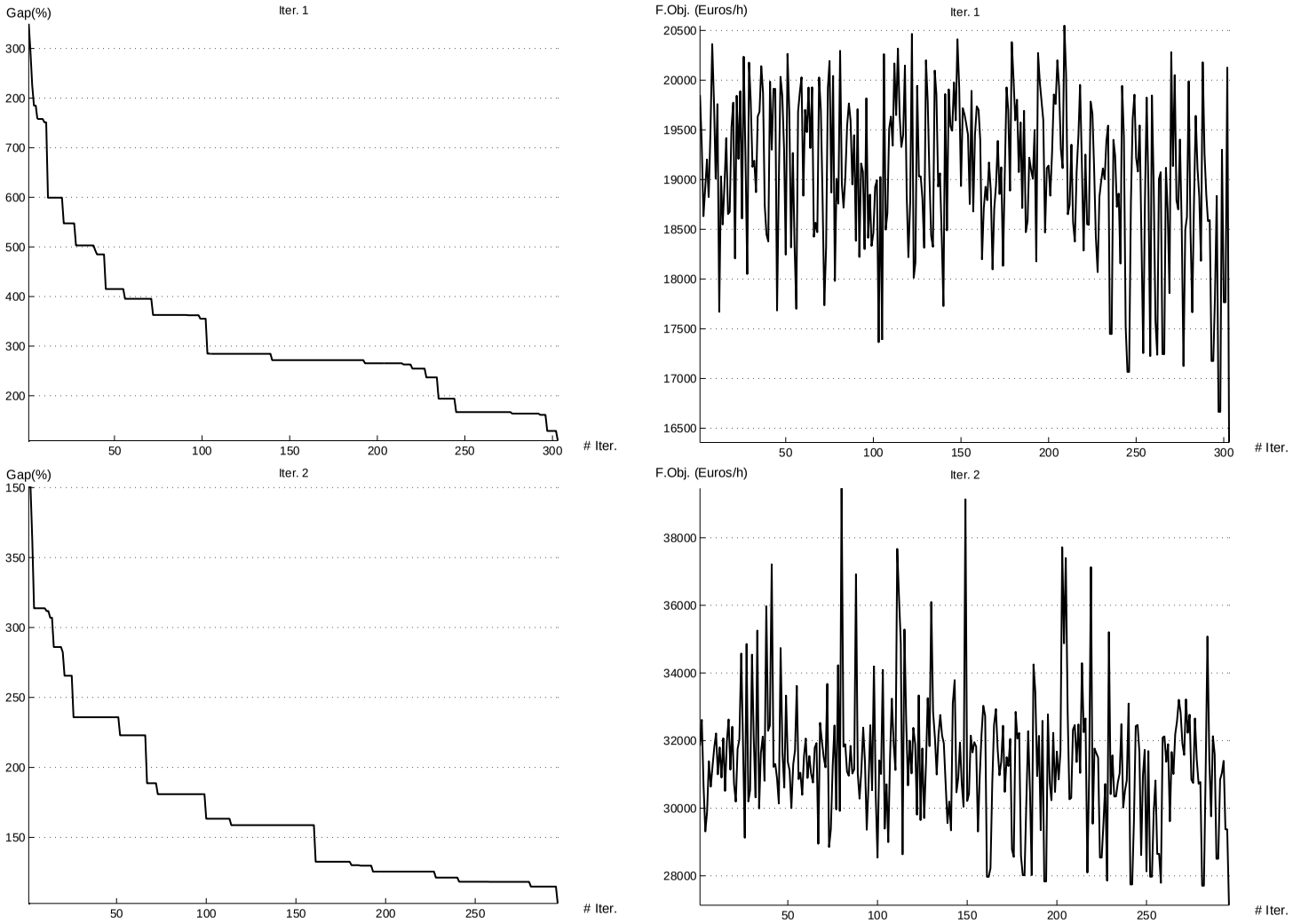


Figure 8.15: Benders phase 1 convergence details under the non-incremental variant of the LSA in the second scenario for the Seville Network. At the top-left, gap evolution for the first iteration. At the top-right, global objective function evolution for the first iteration. At the bottom-left, gap evolution for the second iteration. At the bottom-right, global objective function evolution for the second iteration.

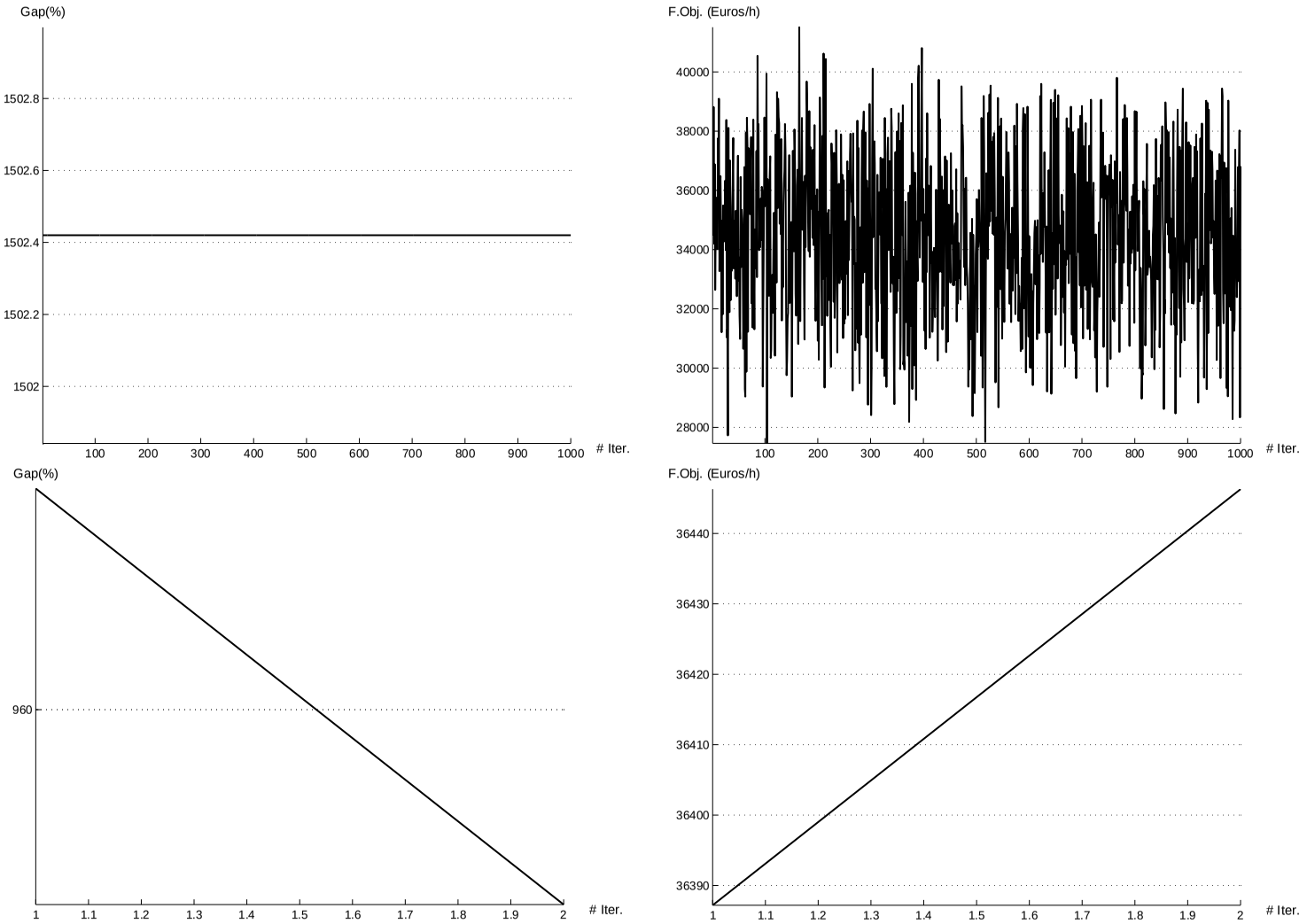


Figure 8.16: Benders convergence details in the third scenario for the Seville Network. At the top-left, gap evolution for the phase 0. At the bottom-left, gap evolution for the phase 0. At the top-right, gap evolution for the phase 0. At the bottom-right, gap evolution for the phase 0.

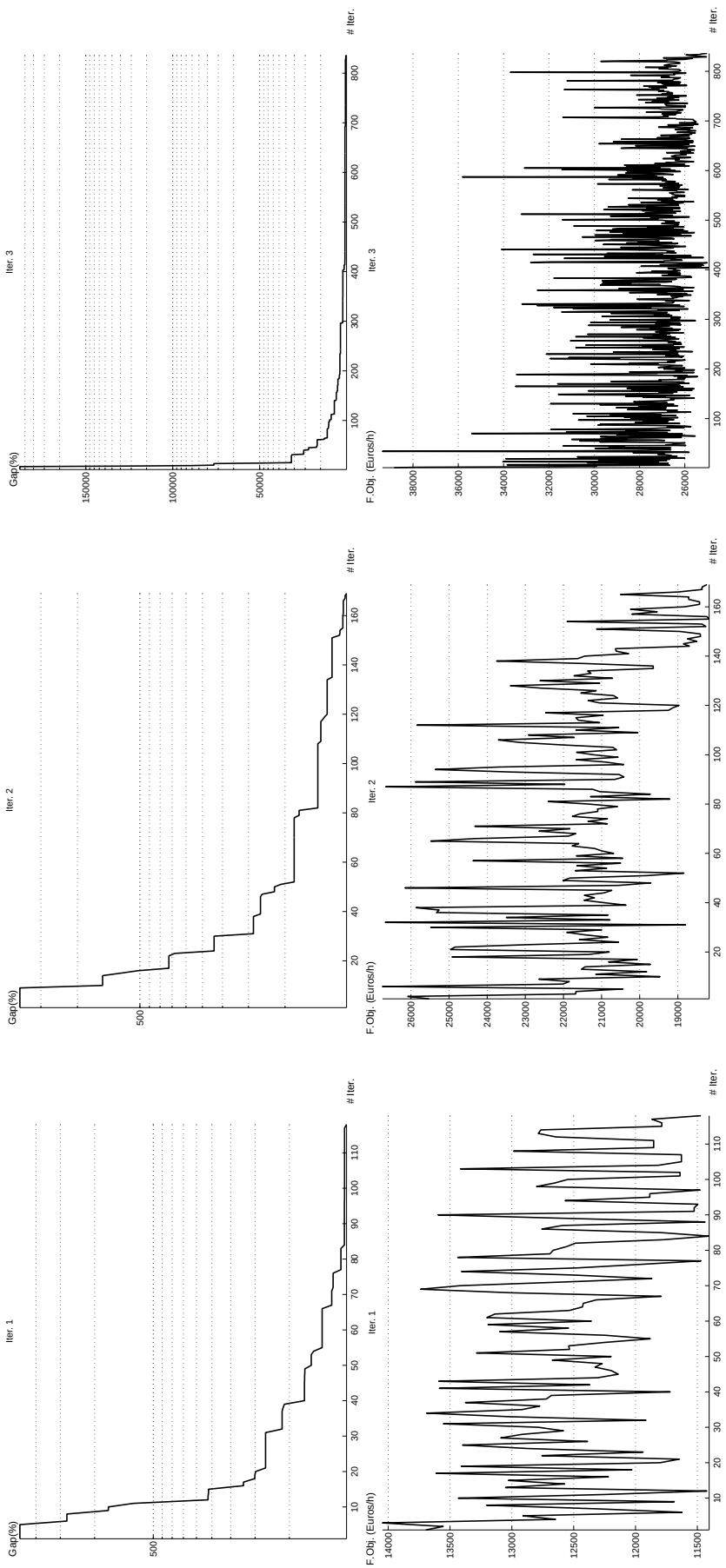


Figure 8.17: Benders phase 0 convergence details under the incremental variant of the LSA in the third scenario for the Seville Network. At the top-left, gap evolution for the first iteration. At the top-right, global objective function evolution for the first iteration. In the middle-left, gap evolution for the second iteration. In the middle-right, global objective function evolution for the second iteration. At the bottom-left, gap evolution for the third iteration. At the bottom-right, global objective function evolution for the third iteration.

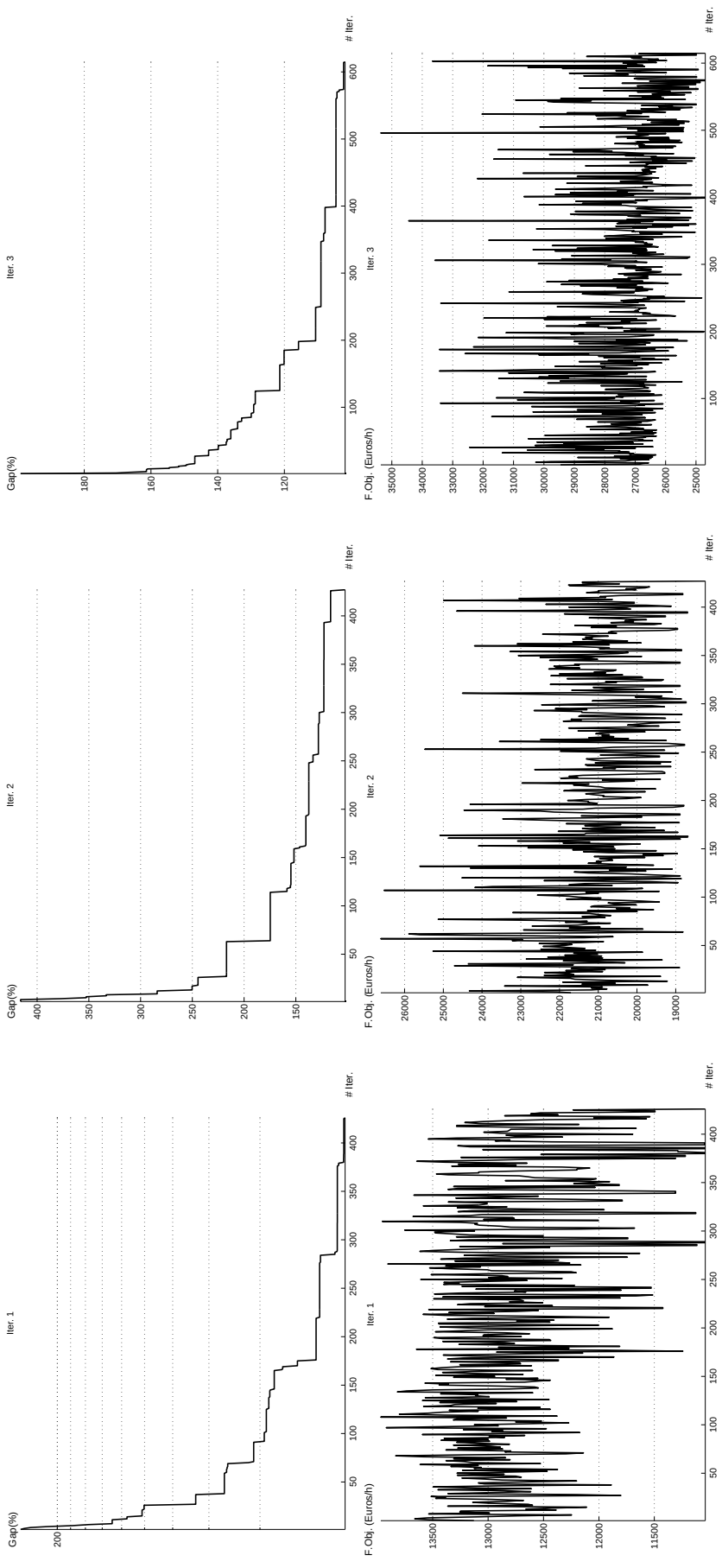


Figure 8.18: Benders phase 1 convergence details under the incremental variant of the LSA in the third scenario for the Seville Network. At the top-left, gap evolution for the first iteration. At the top-right, global objective function evolution for the first iteration. In the middle-left, gap evolution for the second iteration. In the middle-right, global objective function evolution for the second iteration. At the bottom-left, gap evolution for the third iteration. At the bottom-right, global objective function evolution for the third iteration.

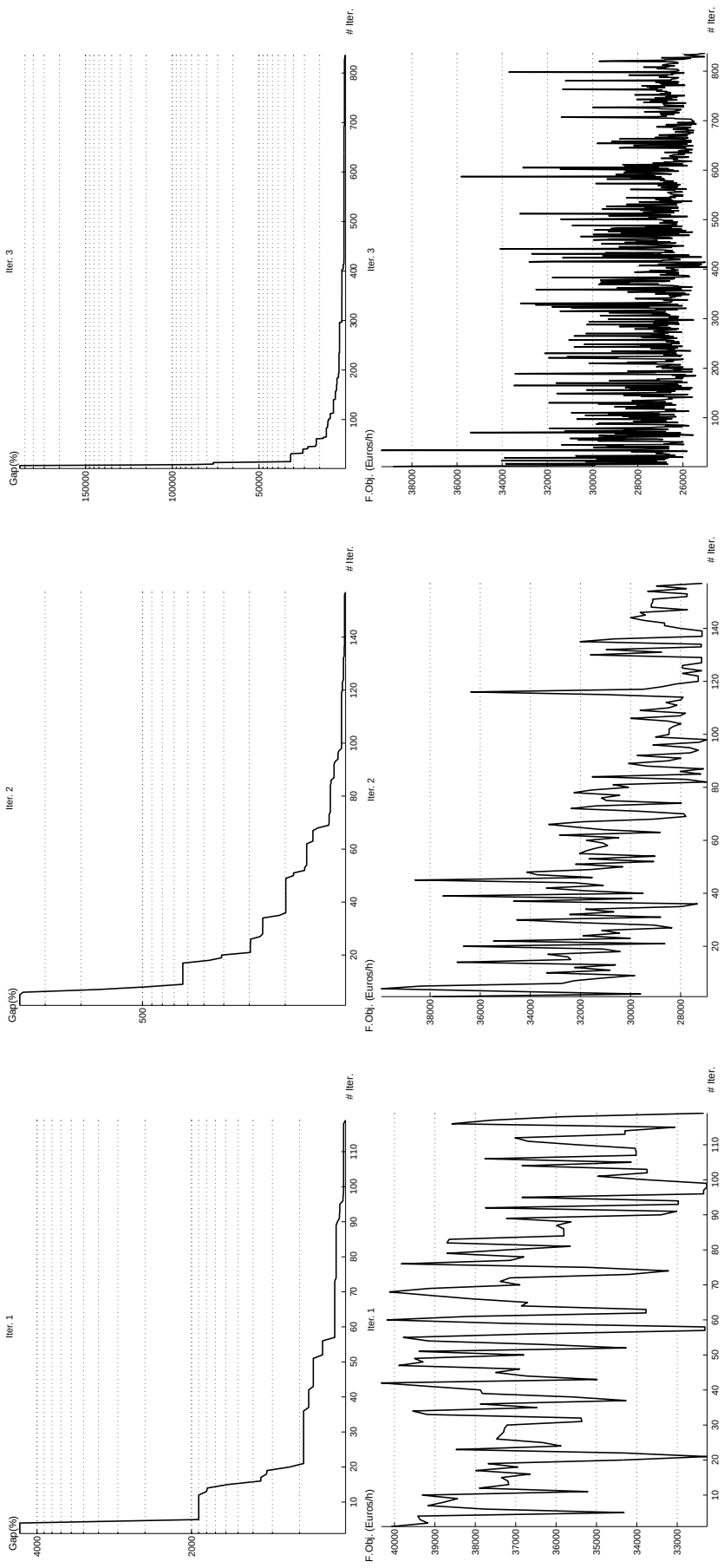


Figure 8.19: Benders phase 0 convergence details under the non-incremental variant of the LSA in the third scenario for the Seville Network. At the top-left, gap evolution for the first iteration. At the top-right, global objective function evolution for the first iteration. In the middle-left, gap evolution for the second iteration. In the middle-right, global objective function evolution for the second iteration. At the bottom-left, gap evolution for the third iteration. At the bottom-right, global objective function evolution for the third iteration.

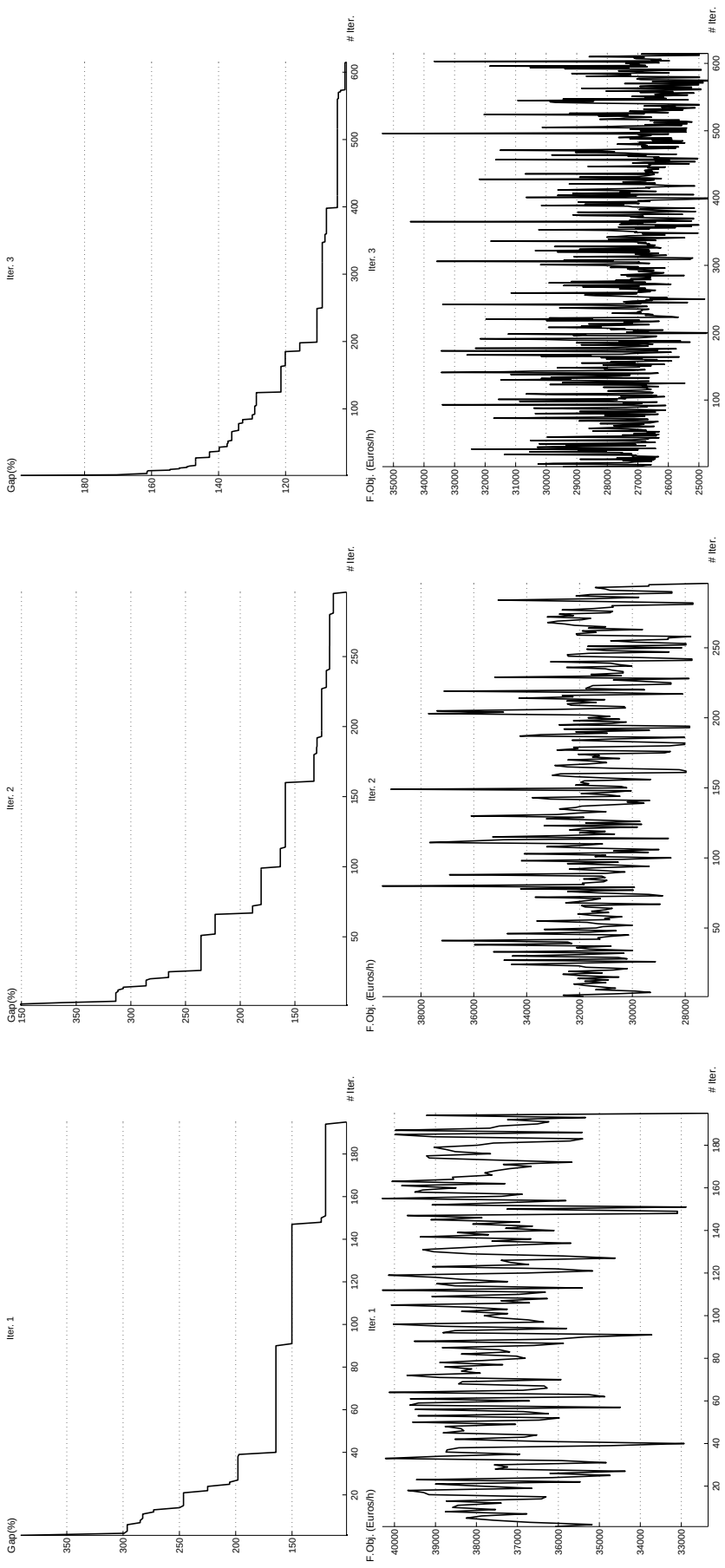


Figure 8.20: Benders phase 1 convergence details under the non-incremental variant of the LSA in the third scenario for the Seville Network. At the top-left, gap evolution for the first iteration. At the top-right, global objective function evolution for the first iteration. In the middle-left, gap evolution for the second iteration. In the middle-right, global objective function evolution for the second iteration. At the bottom-left, gap evolution for the third iteration. At the bottom-right, global objective function evolution for the third iteration.

8.1.2.3 Santiago de Chile's Network

Santiago de Chile's Network is a large-sized network consisting of 146 nodes, 520 links and 21316 od-demand pairs. Figure 8.21 shows the working network, which includes five lines, labeled: L1 (red), L2 (yellow), L4 (dark blue), L4A (light blue) and L5 (green). The working network contains 100 nodes, from which 8 nodes are transfer nodes (nodes where passengers can move from one line to another), and 206 links. The transfer nodes are filled in with white to distinguish them from ordinary nodes (nodes which simply carry out service in a single line). Finally, their corresponding names are written in black color.

Figure 8.22 depicts the routing graph, the one from which the new lines can be built. It includes 53 nodes and 318 links, and allows reduction of the number of transfers for those od-pairs with origin and/or destination in some extreme or near-extreme nodes of lines L4, L2, L5 and L4A. Notice that the routing subgraph at the bottom-center allows linking of the L5 terminal node, Plaza de Maip, to the L2 terminal node, La Cisterna, whereas the routing subgraph at the top-left connects the L2 terminal node, Vespucio Norte, to the L1 terminal node, San Pablo. Finally, the routing subgraph at the top-right links the L1 terminal node, San Pablo, to the L2 terminal node, Vespucio Norte. From now on, we will refer to each routing subgraph as the sector or routing sector.

The input data settings have been carried out as follows. Regarding the extended infrastructure parameters, the construction costs of the stretches have been computed by means of equation (33) in the appendix, taking a monetary cost of one length unit of stretch $c_{ij}^{c,1} = 189$ millions of euros, an amortization horizon $H^a = 25$ years, and the stretch length d_{ij}^{TP} according to the values in column 3 of Tables (19) - (21) in the appendix. The value of $c_{ij}^{c,1}$ has been taken indirectly from Ocaa [143]. Since this paper is quite old, we have multiplied the corresponding average value for the case of underground stretches by a factor of 3 to emulate the effect of inflation. Column 6 of tables (19) - (21) in the appendix contains the resulting values.

As for the construction costs of the stations, we have not applied formulas (34) and (35) in the appendix since we cannot determine a priori the absorbed demand set W_i. Instead of this, we have set all costs to an average value of 3.25 million euros, taken indirectly from Ocaa [143]. Again, since this paper is quite old, we have multiplied the corresponding average value of the underground station by a factor of 3 to emulate the effect of inflation. Moreover, we have divided the resulting value of the amortization horizon $H^a = 25$ years.

The maintenance costs of stretches and stations have been taken as 0.1 times their corresponding construction costs. Finally, the infrastructure budget has been set to the sum of all construction costs, thus allowing construction of as many resources as possible.

Now, focusing on planning parameters, the service costs of the stretches have been obtained by means of equations (36) - (38) in the appendix, taking a vehicle's power consumption cost $c_e^{tu} = 0.1555$ euros/h, a nominal power of the vehicle's engine $\eta_e \cdot \bar{P}_e = 1680$ kWh, an average base salary $c_s = 979$ euros/month and a working horizon $H_w = 9$ h/day. These estimated values have been taken from different Chilean web sites. Column 4 of tables (14) - (21) in the appendix contains the resulting values.

As for the stretch capacities, we have employed equation (39) in the appendix to compute them with a safety coefficient $K_s = 0.25$ and an average service time $t_j^s = 1$ min for all the station nodes. The travel times (t_{ij}^{TP}) have been computed as the quotient of the travel distance (d_{ij}^{TP}) and the underground's average speed of 60 km/h. Column 5 of tables (14) - (21) in the appendix contains the resulting capacities.

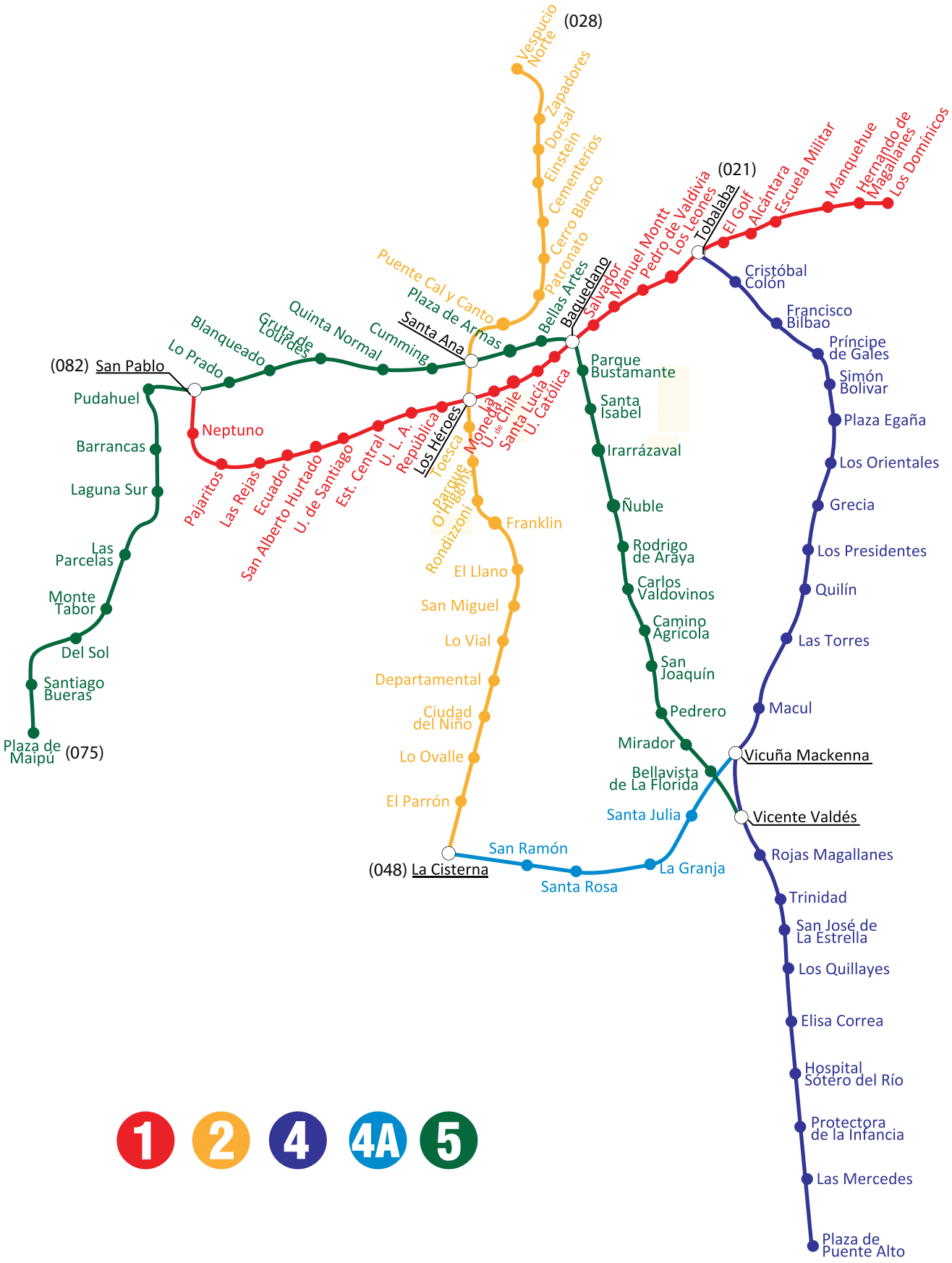


Figure 8.21: Working network of Santiago de Chile underground.

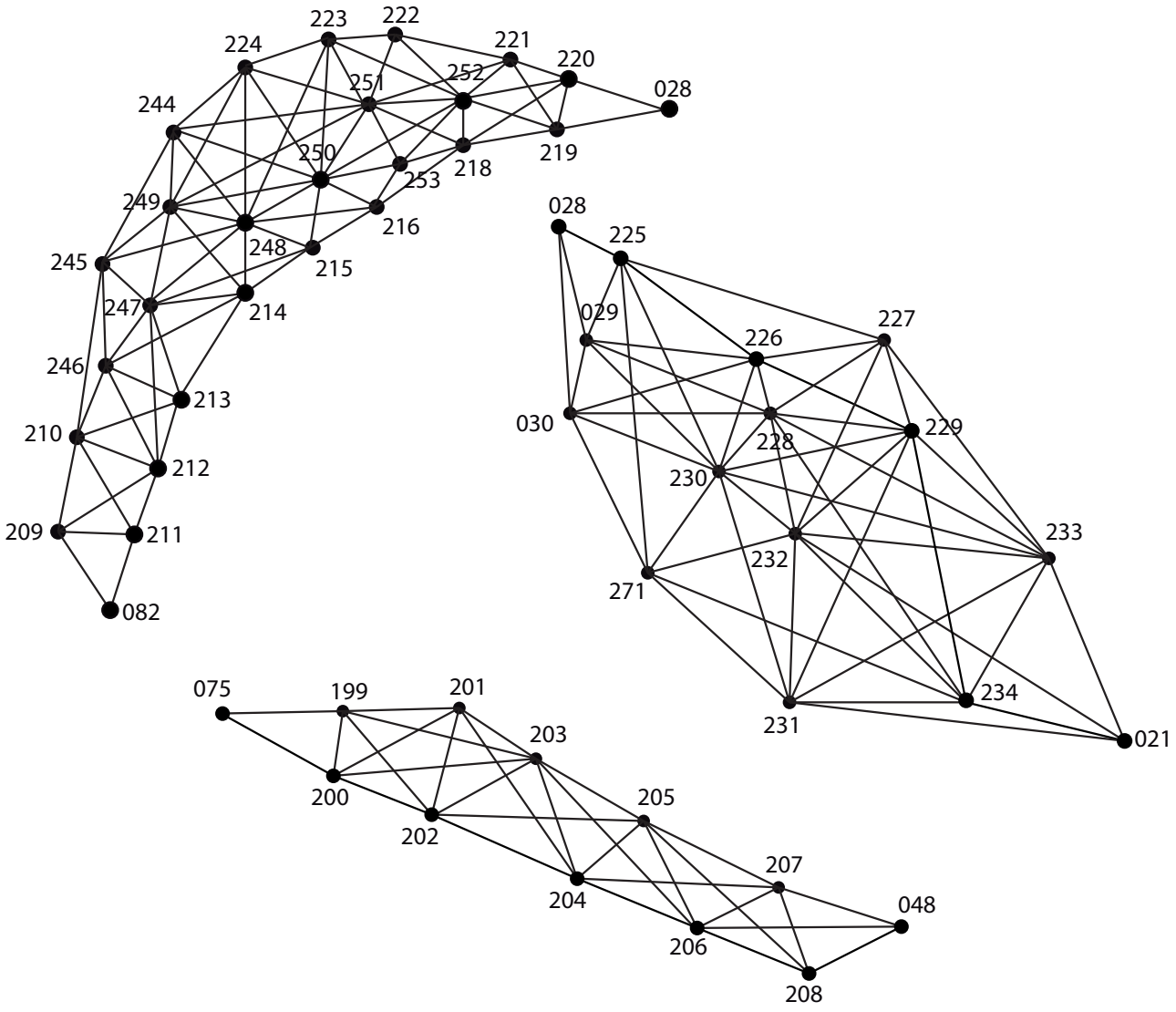


Figure 8.22: Routing Sectors.

Moving on to vehicles features costs, the acquisition cost of new vehicles have been computed according to equation (40) in the appendix, taking an average number of carriages $n_{car} = 6$ and the cost of each one $c_{car} = 0.8015$ millions of euros. The latter is closely related to the one obtained from Ocaa [143]. However, to take into account the inflation effect, we have weighted the corresponding value by 3. The number of acquired vehicles is subject to a planning budget of 47 euros/min. As for the setting cost of vehicles, this cost has been set to zero, since its contribution was meaningless compared to the acquisition cost of new vehicles. Finally, the vehicle capacity (the total number of passengers which can be held) is set according to equation (42) in the appendix, taking an average carriage capacity $q_{car} = 189$ passengers.

Finally, we present the passenger assignment settings. The unit times related to the in-station passenger flow are set at 2/30 minutes for alighting and waiting-in-vehicle movements (t_x and t_y, respectively), whereas 1/30 minutes are spent on boarding (t_a). Regarding walking times, they have been obtained as the quotient of the Manhattan distance (weighted by a correction factor of 24.6) and the passengers' average speed of $v_{com} = 4.5km/h$. To compute the Manhattan distance, the grid coordinates of Figures 8.21 - 8.22 have been used. All these times, as well as in-vehicle travel times, have been weighted by a factor of $\theta = 0.081$ euros/min.

In order to obtain a likely o-d trip matrix for the test network, a trip distribution model (TDM) of the general gravity type has been used. This class of TDM aims to obtain the od-demand trips, such that they verify the following:

$$g_w = A_{p(w)} \cdot B_{q(w)} \cdot f(u_w),\ \forall w \in W \tag{8.1}$$

$$\sum_{q(w)\in D_p} g_w = G_p,\ \forall p \in O \tag{8.2}$$

$$\sum_{p(w)\in O_q} g_w = G_q,\ \forall q \in D \tag{8.3}$$

$$g_w \geq 0 \tag{8.4}$$

Equation (8.1) establishes a relationship between the passenger travel costs, which are provided by means of a deterrence function ($f(u_w)$), and the amount of trips carried out for each given pair of origins $p(w)$ and destinations $q(w)$. $A_{p(w)}$ and $B_{q(w)}$ are the origin and destination balancing coefficients, respectively. The next equations (8.2)-(8.3) link the number of generations and attractions for each origin and destination (G_p and G_q, respectively) to the od-demand trips which they are related to. Finally, equation (8.4) imposes non-negativity on the number of trips per od-pair.

The GCM model is solved by means of a modification of the Furness Algorithm [79], which is implemented in several transit planning software packages such us EMME [95]. This algorithm determines $A_{p(w)}$ and $B_{q(w)}$, taking as inputs G_p, G_q and $f(u_w)$. The computation of $f(u_w)$ is carried out in two steps. Firstly, we compute the approximated travel times for each od-pair u_w as the travel times of the shortest paths throughout the in-vehicle links of the working and routing network without considering capacity. The link travel times have been calculated as the quotient of their travel distances (d_{ij}^{TP}) and a working speed of 30 km/h (half of the vehicles speed, considering that in the model solution this value will approach the average commercial speed of the service lines). Secondly, we choose the most suitable deterrence function ($f(u_w)$) in the histogram of the computed u_w times. Without any further analysis, its parameters have been evaluated by a wild (although reasonable) guess. According to the graph at the top of Figure 8.23, we approximated the deterrence function $f(u_w)$ with the following:

$$f(u_w) = u_w^n \cdot \exp^{-\lambda \cdot u_w} \tag{8.5}$$

which is the gamma function parameterized by n and λ. For urban trip distribution, it is recommended to set λ as the inverse of the average trip, which is 9.29 minutes. n has been set to 2. Having computed coefficients $A_{p(w)}$ and $B_{q(w)}$, we obtain the estimated passenger trips (g_w) by using equation (8.1). The graph at the bottom of Figure 8.23 shows the histogram of the o-d flows. Complementarily, Table 8.1 contains the basic statistics.

The formulas for the input data setting and the tables can be found in the appendix.

	Non Zero Values	Min. value	Max. value	Av. value	Total Trips
G_W	21.460	0.001586 pax/min	1.8799 pax/min	1.0070 pax/min	3194.44 pax/min

Table 8.1: Estimated demand features.

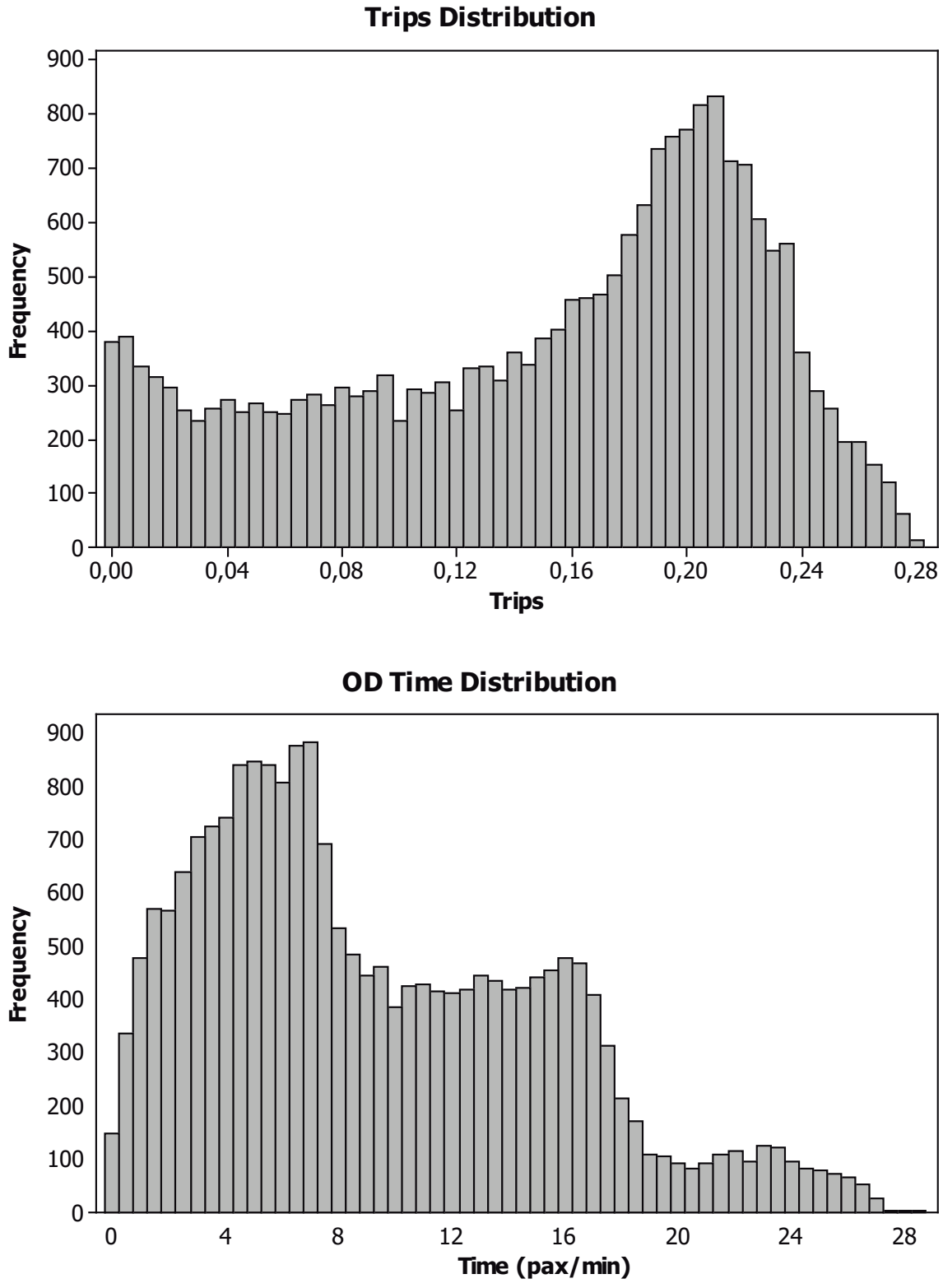


Figure 8.23: Trips Histogram.

8.1.2.4 Results

This section presents the results obtained for extending the Santiago de Chile underground network with different numbers of lines under construction. Their traces have been computed by means of the routing model M3 (see subsection 2.3.3.3 in chapter 2). Regarding computational resources, we have used a R5500 working station with Intel(R) processor Xeon(R) CPU E5645 2.40 GHz and 48 Gbytes of RAM. The model has been coded in AMPL, using the CPLEX v.12.4.0 solver with a time limit of 24 hours.

To determine the pool of corridors (Λ) under consideration, we have employed a variant of the $RCGP$ procedure (see section 4.1.1). It differs from the original in the following. It takes only as inputs a subset of node pairs $i, j \in N_{TP}^{N}$ for computing the K-Shortest rectilinear paths. We have proceeded in that way because the routing topology of the graph is circular (see Figure 8.21), thus it makes no sense to compute all-to-all K-shortest rectilinear and circular paths. The chosen pairs of nodes $i, j \in N_{TP}$ are the extreme nodes of the routing subgraphs depicted in Figure 8.22 and shown in the first column of table 8.2. Furthermore, parameters ϕ_{max} and Δ_{min}, associated with the user behavioral rules, are set to the values recommended by Zijpp & Catalano [174], resulting in a total of 100 corridors distributed among each routing subgraph, as shown in the third column of table 8.2.

Sector	**Description**	Λ^s
1	Tobalaba - Vespucio Norte	36
2	Vespucio Norte - San Pablo	35
3	Plaza de Maip - La Cisterna	26

Table 8.2: Number of corridors generated for each sector of the Santiago de Chile routing graph.

This pool of corridors has been applied to the resolution of three different scenarios set with 1, 2 and 3 lines under construction. Each scenario has a working fleet of 188 vehicles and a planning horizon of 18 hours. The remaining parameters are set as explained in the previous section.

Regarding solving techniques, we have applied to each aforementioned scenario six different algorithms. The first triad uses CPLEX as the $MILP$ under different line splitting algorithms, whereas the other triad applies the Papadakos schema (see table 6.3) under the same line splitting algorithms. The first one consists of the direct resolution of the complete $MILP$, whereas the other two use the LSA algorithm (see chapter 5) under non-incremental and incremental demand load variants, respectively.

In the rest of the section, we analyze the best solution found by the aforementioned algorithms. Any other solution discussed will be specified. Moreover, we will present the time performance for all six algorithms, with special attention to those involving the Benders decomposition. Finally, we will compare all of them, showing that the Benders decomposition under the LSA is the best option in terms of the ratio of solution quality and solving time.

Figures 8.24 - 8.25 depict the network extension solution for scenario 3, where three new lines labeled L6, L7 and L8 have been constructed. In scenario 1, only L6 is constructed, whereas in scenario 2, L6 and L7 are constructed. Notice that lines L6, L7 and L8 correspond to routing sectors 2, 3 and 1, respectively, according to column 1 of table 8.2. Moreover, the corridors assigned to them are the same; therefore we have drawn only on figure 8.25 for their routing in accordance with the scenario 3 solution, where all line nodes perform service.

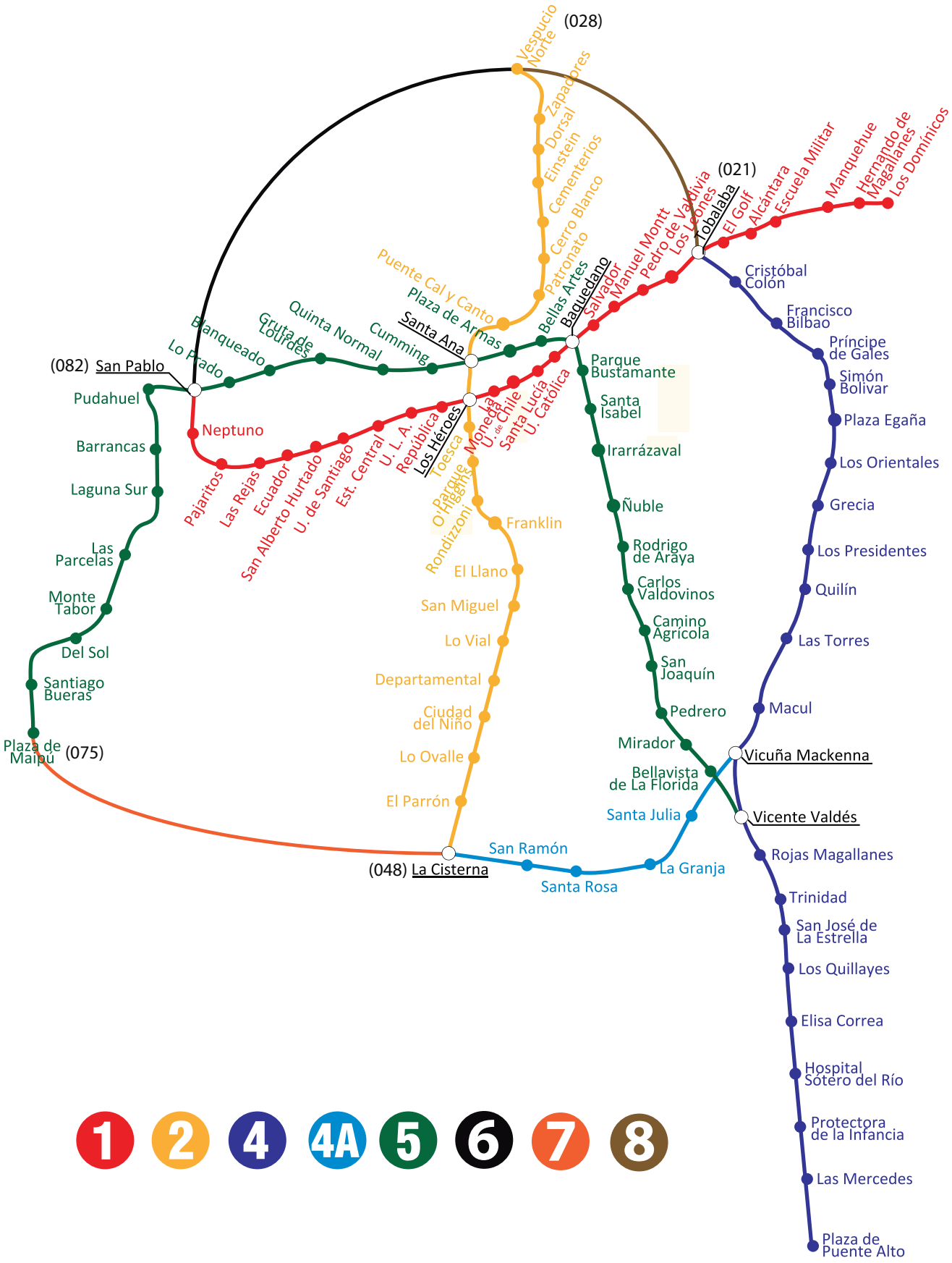


Figure 8.24: Proposed extension of the Santiago de Chile underground network.

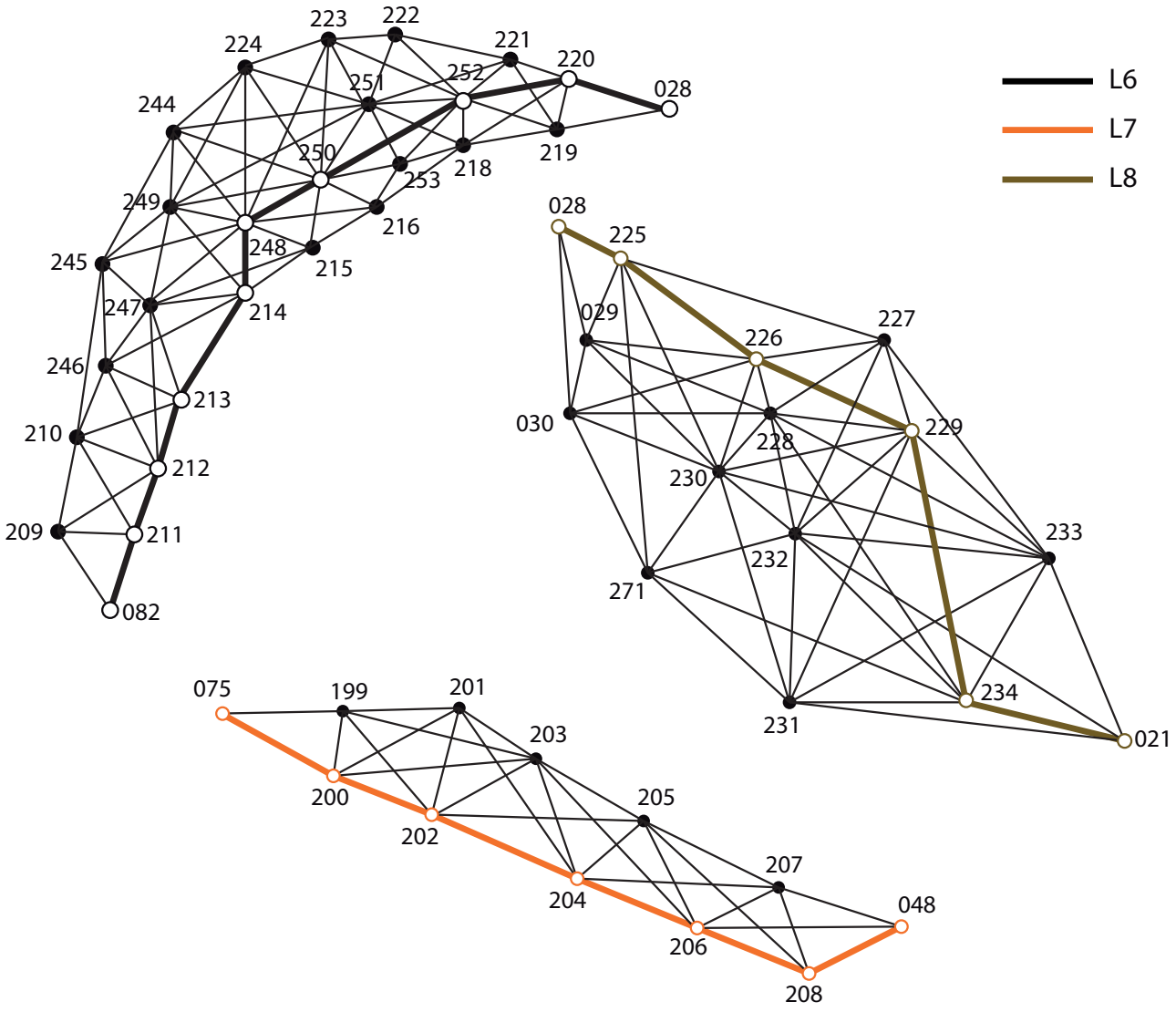


Figure 8.25: Details of the proposed extension layout of the Santiago de Chile underground.

The next Figure 8.26 contains four bar graphs which describe the main planning features of the solution. The one at the top depicts the line capacities (q^l) expressed in pax./min for all scenarios. Each one is contained in bar clusters and they are sorted from left to right, with the left corresponding to the first scenario and the right to the third scenario. The three graphs below show the average line occupancies ($O(q^l)$). The two in the middle correspond to scenarios 1 and 2 and the one at the bottom to scenario 3. In a general view, we can see that line capacities related to working lines remain constant in all scenarios except for line L4A, which has a sudden increase in the second scenario. As for new lines, L6 also remains constant in all scenarios, whereas line L7 decreases by almost 100 units in scenario 3. This decrease is due to the emergence of line $L8$, which attracts part of the demand going through line L7.

Focusing now on average line occupancies, we can split them into three categories according to their congestion levels: moderate (70 - 80 %), high (81 - 90 %) and very high (91 - 100 %). Observe that lines L1, L4A, L6 and L7 fall in the first category, lines L4 and L8 in the second, and Lines L2 and L5 in the third.

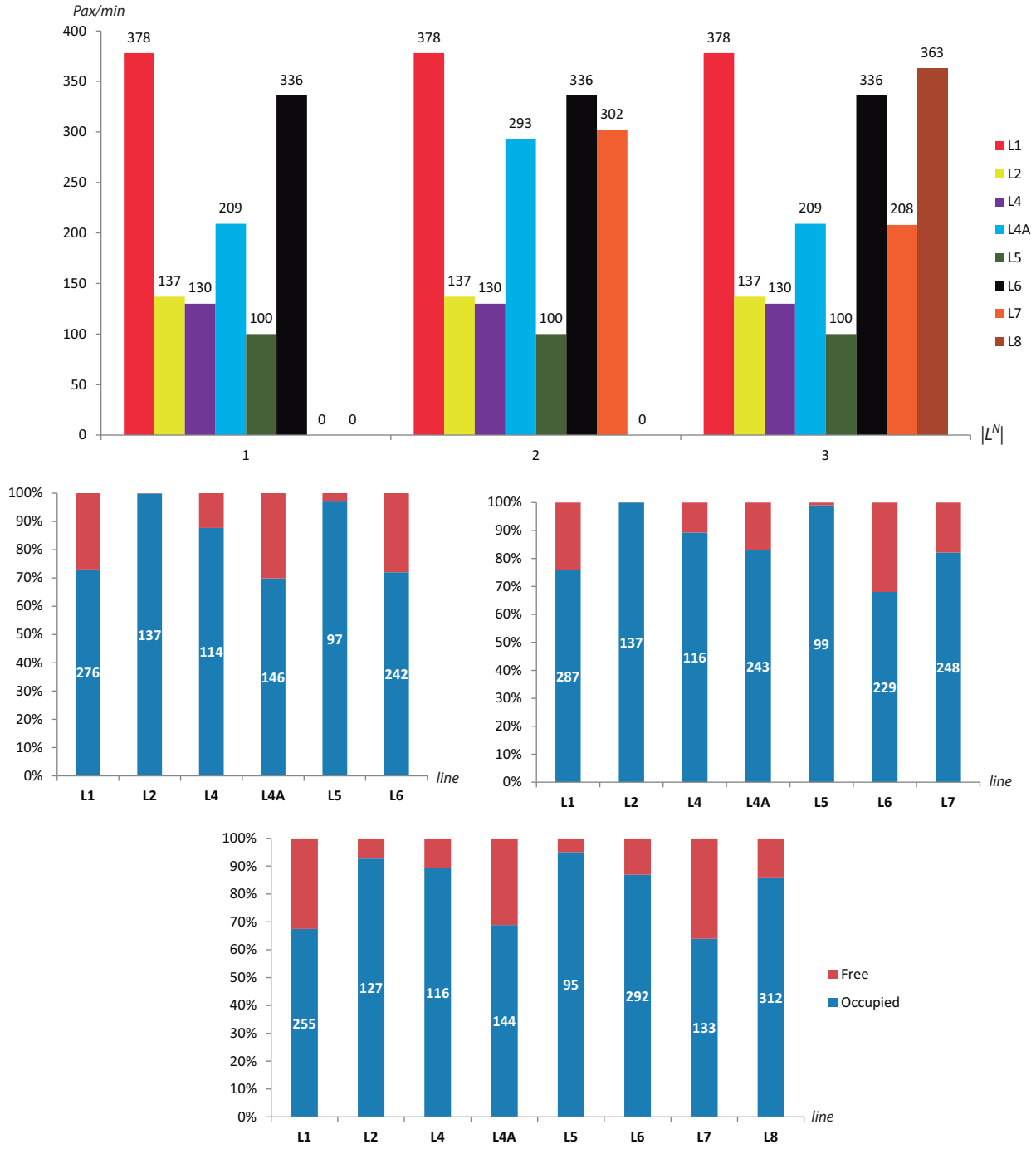


Figure 8.26: Overview of the main planning features of the solution. At the top, line capacities grouped in close bars for each scenario. In the middle, average line occupancies for scenarios 1 and 2. At the bottom, average line occupancies for scenario 3.

Moving on to demand issues, figure 8.27 analyzes the demand coverage. The graph at the top side, shows the demand balance between the public transportation and walking modes, whereas the graph at the bottom indicates only the demand going by public transportation. TP1 and TP2 stand for the demand in public transportation with and without walking transfers, respectively. From these graphs, we can see that the level of demand covered by the public transportation mode is quite high for all scenarios and that it increases a slight 4% from scenario 1 to scenario 3. However, the percentage of demand with no walking during transfers improves significantly in scenario 3. Complementary to these figures, the following figure 8.28 shows the

histograms of the o-d times expressed in seconds. From top to bottom, we have the ones related to scenarios 1, 2 and 3, respectively. Notice that a greater portion of the demand concentrates in the lowest time intervals as the number of constructed lines increases. However, there is a significant portion of demand that fails to remain at low time intervals.

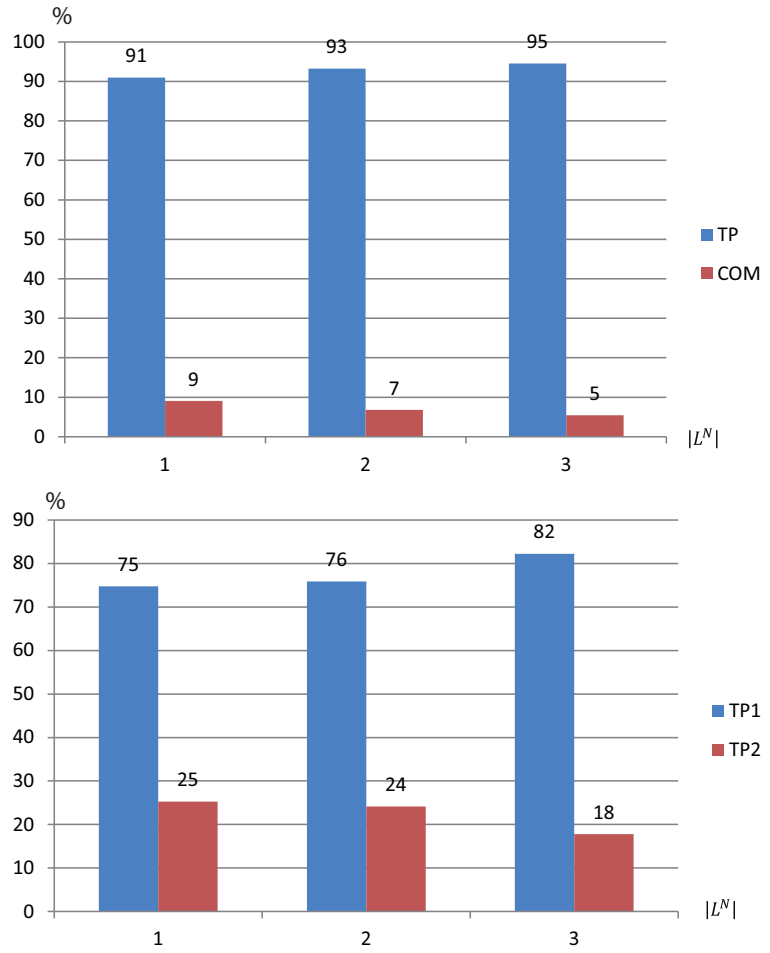


Figure 8.27: Demand Coverage.

Regarding computational time requirements, figure 8.29 contains two graphs related to the performance of the Benders decomposition without line splitting. The one at the top shows the total amount of time as well as the contribution of each Benders problem, whereas the one at the bottom depicts the percentage of each problem. From these graphs, we observe two different situations. One in which the total solving time is relatively low (1.5 hours, approximately), and the master problem (MP) contributes very little to time when compared to the subproblem (SP) and the independent Magnanti & Wong problem ($IMWP$) (see scenario 1). In contrast, another situation arises when the total solving time reaches the time limit (24 hours) and the MP contributes the most (see scenarios 2 and 3). Consequently, as the number of lines under construction increases, the MP becomes more difficult to solve, and thus the total computational time of the algorithm increases exponentially.

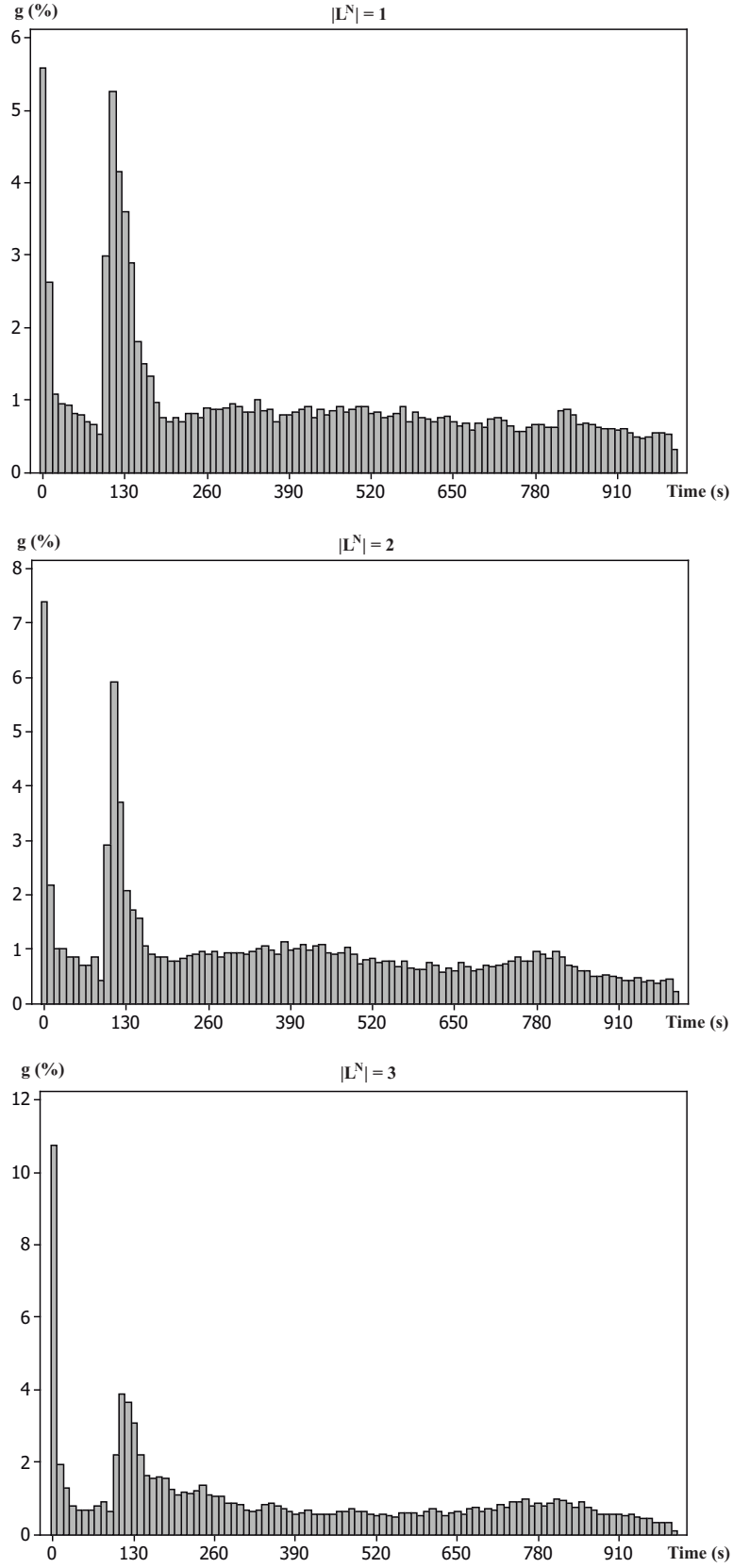


Figure 8.28: Histograms of the o-d times for the Santiago de Chile network expressed in seconds. At the top, the demand time distribution for scenario 1. In the middle, the demand time distribution for scenario 2. At the bottom, the demand time distribution for scenario 3.

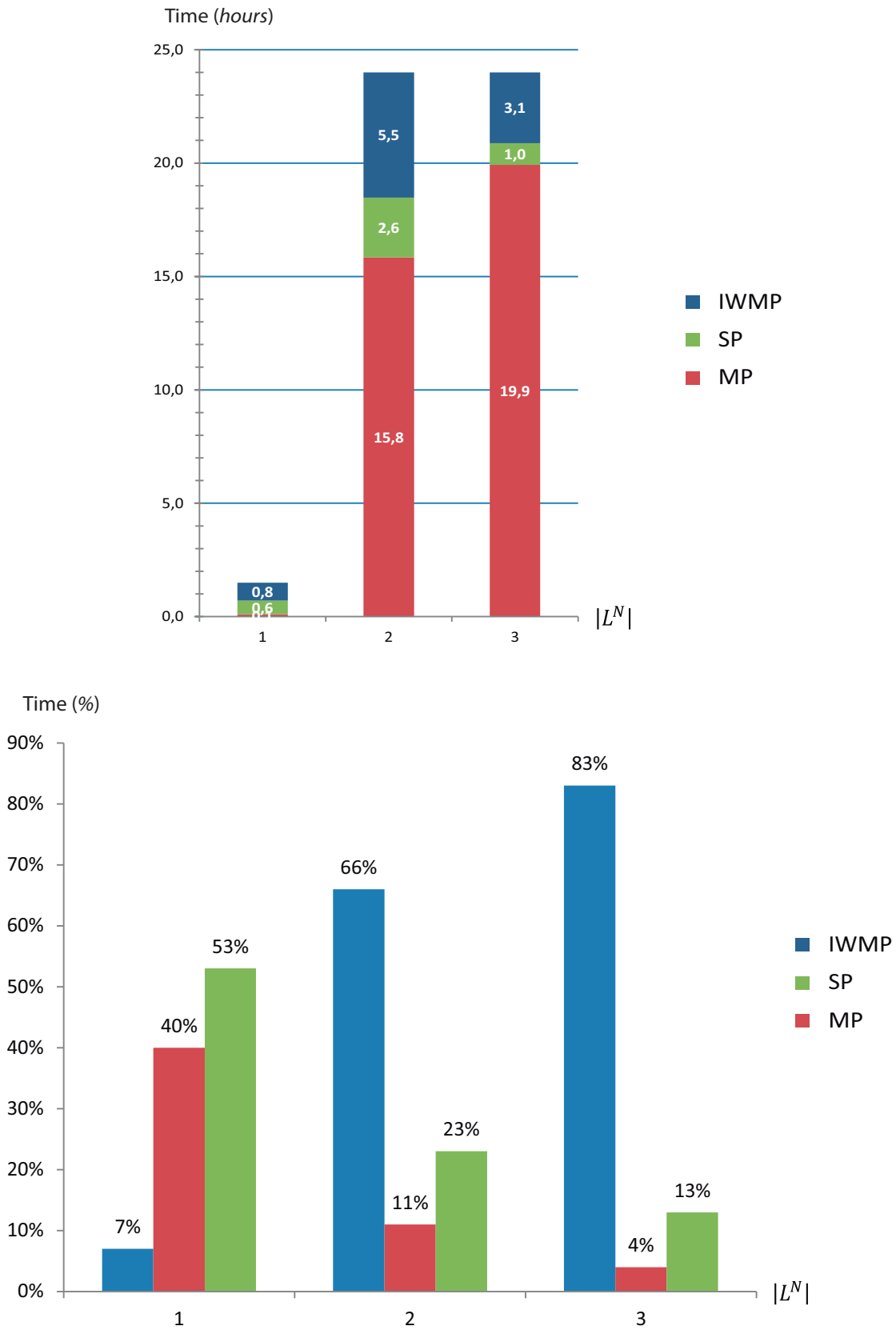


Figure 8.29: Fraction of solving time requirements for each step of the EBD algorithm.

The next Figures 8.30 - 8.31 show a comparison of all algorithms. The first figure contains two graphs. The left one depicts the (near-) optimal value of the global objective function (2.1) for each scenario and algorithm, where the Branch & Bound of CPLEX has been used as the MILP solving technique. On the right, we can see the same graph, but for the case where the Benders decomposition has been employed as the MILP solving technique. Regarding the second figure, the left graph shows the CPU time consumed by the same triad as the one in the left graph of the first figure. Analogously, the right graph depicts the CPU times of the same triad in the right graph of the first figure. From both figures, we can clearly see that the Papadakos schema under the LSA variants ($PBD+LSA1$ and $PBD+LSA2$) outperforms the rest in every scenario. Furthermore, the non-incremental variant of the line splitting algorithm ($LSA1$) is preferable to the incremental version ($LSA2$), since it gives the lowest CPU times while maintaining a good solution quality in all scenarios. Excluding the direct resolution of the complete MILP using CPLEX in scenario 2, the solution obtained by all algorithms is very similar, although the solving time in algorithms $PBD + LSA1$ and $PBD + LSA2$ is much better. It ranges only from 1.5h to 6.4 hours (worst case). Compared to the nearly worst solving times (i.e., those associated with algorithms using also the LSA), they take from 3 to 5 times longer. The worst CPU times arise in the direct resolution of the complete $MILP$, whatever the MILP solving technique is used. However, the PBD gives a better objective function value within the time horizon of 24h.

To conclude the results section, we also show the gap and the global objective function evolutions for all scenarios. Figure 8.32 depicts these evolutions for the first scenario, whereas figures 8.33 - 8.35 and 8.36 - 8.38 depict those related to the second and third scenarios, respectively. From the gap evolution graphs, we observe that the gap converges to a value of less than 1%, similarly to all scenarios. Moreover, it has a plateau in the first iterations. Regarding the objective function, its evolution is completely different. It also converges, but in a zigzag fashion. This is due to the fact that in the standard Benders decomposition, the global objective function does not necessarily decrease when the gap does. However, as pointed out in the end of Seville's results subsection, this drawback can be overcome by adapting some recently developed Benders convergence enhancements (see Saharidis *et al* [157] and Sherali & Lunday [160]). This field of research deserves more attention and will be the subject of further research.

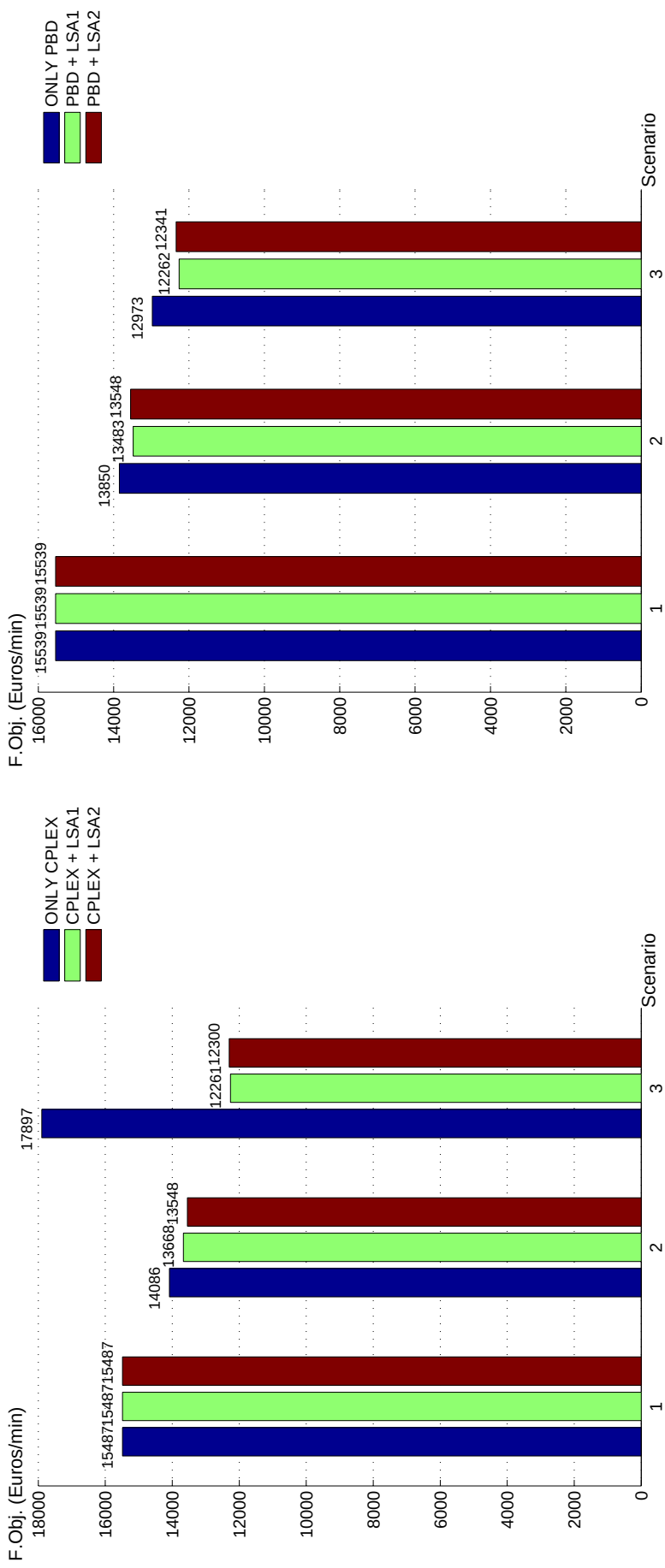


Figure 8.30: (Near-) Optimal Global objective function values of the different scenarios for the Santiago de Chile Network. On the left, the values related to the B & B of CPLEX under the different line splitting methods. On the right, the values associated with the Benders decomposition under different line splitting methods.

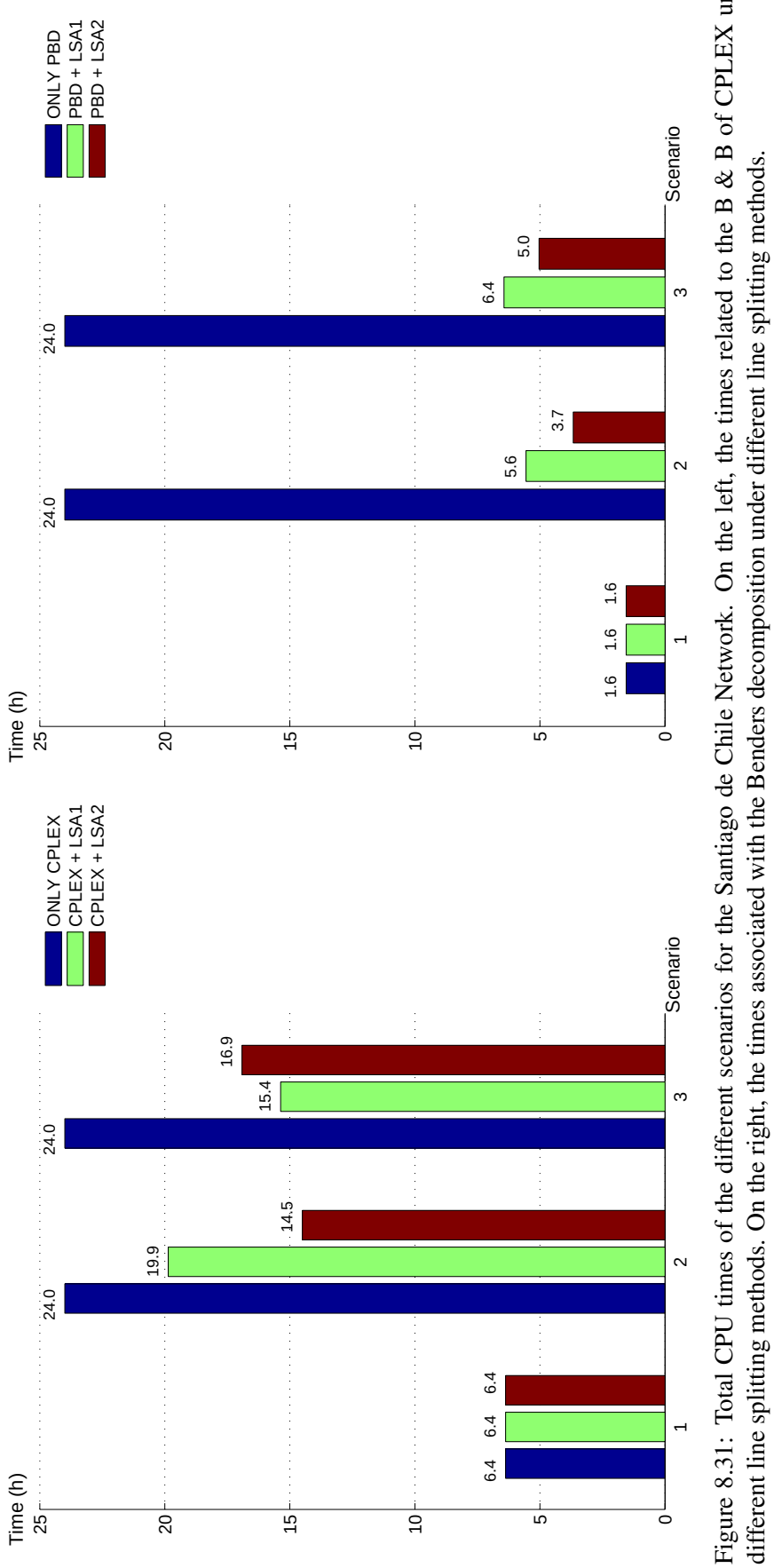


Figure 8.31: Total CPU times of the different scenarios for the Santiago de Chile Network. On the left, the times related to the B & B of CPLEX under different line splitting methods. On the right, the times associated with the Benders decomposition under different line splitting methods.

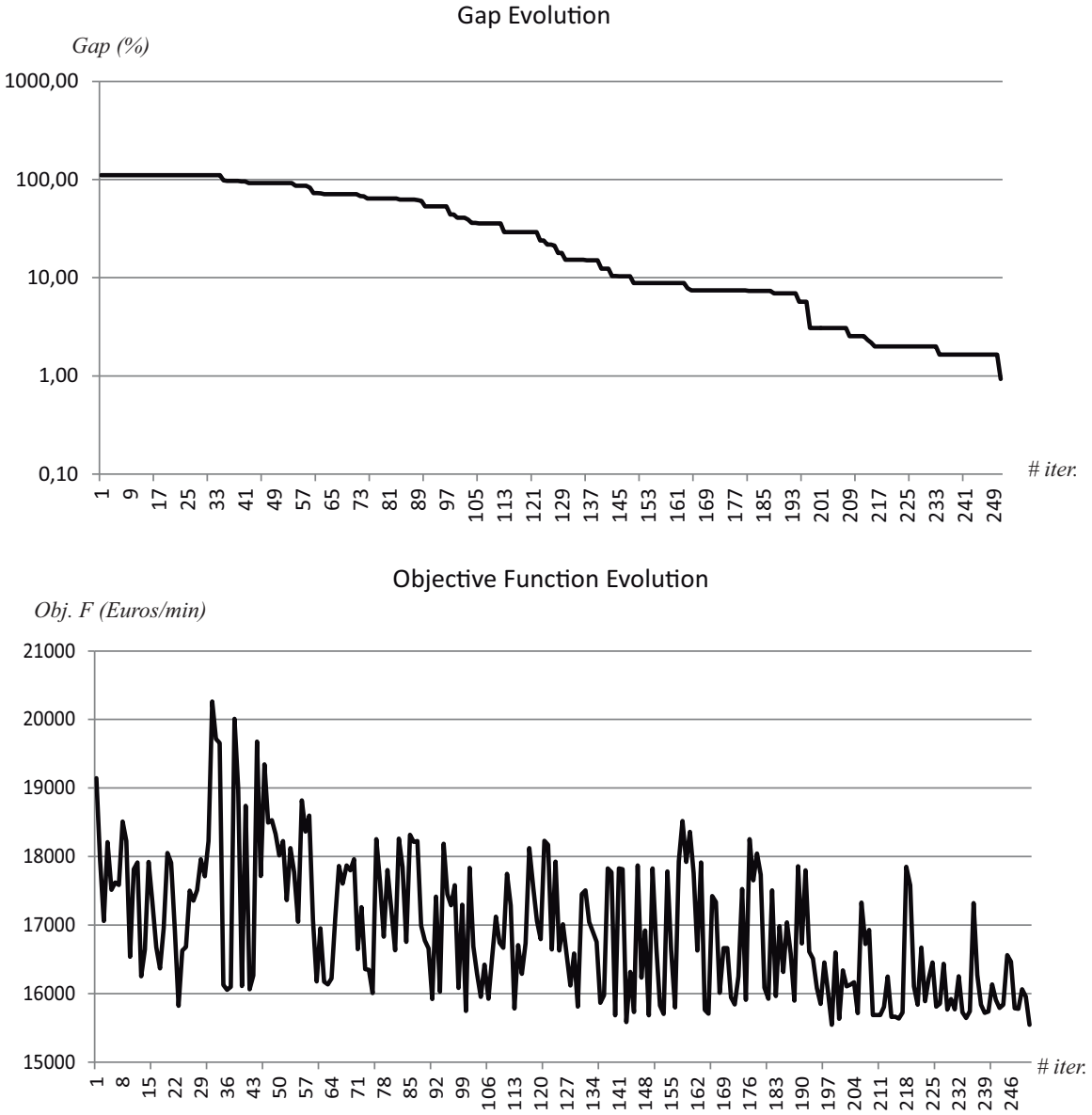


Figure 8.32: Gap and Objective function evolution for the first scenario under the Benders decomposition without line splitting algorithm.

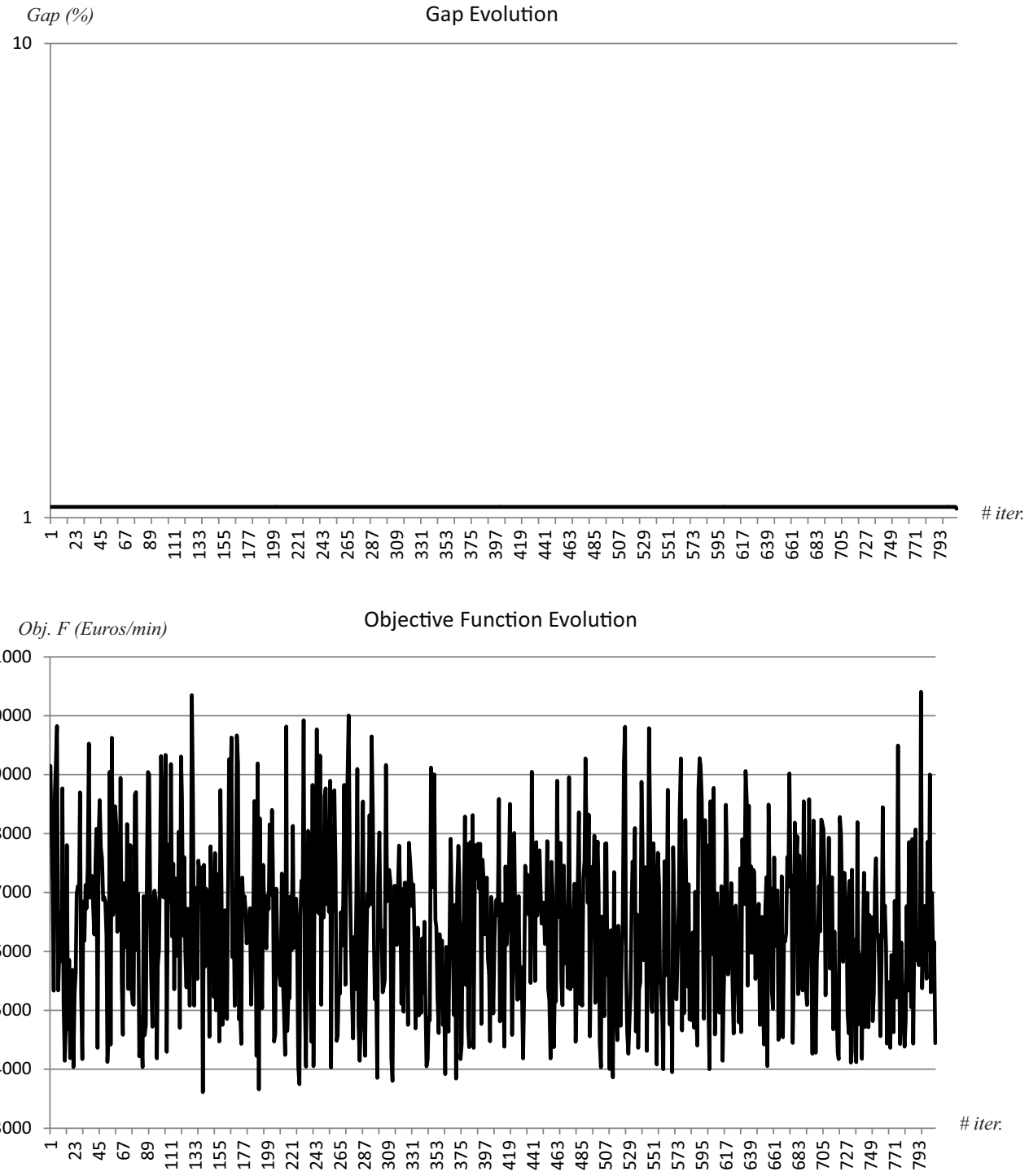


Figure 8.33: Gap and Objective function evolution for the second scenario under the Benders decomposition without line splitting algorithm.

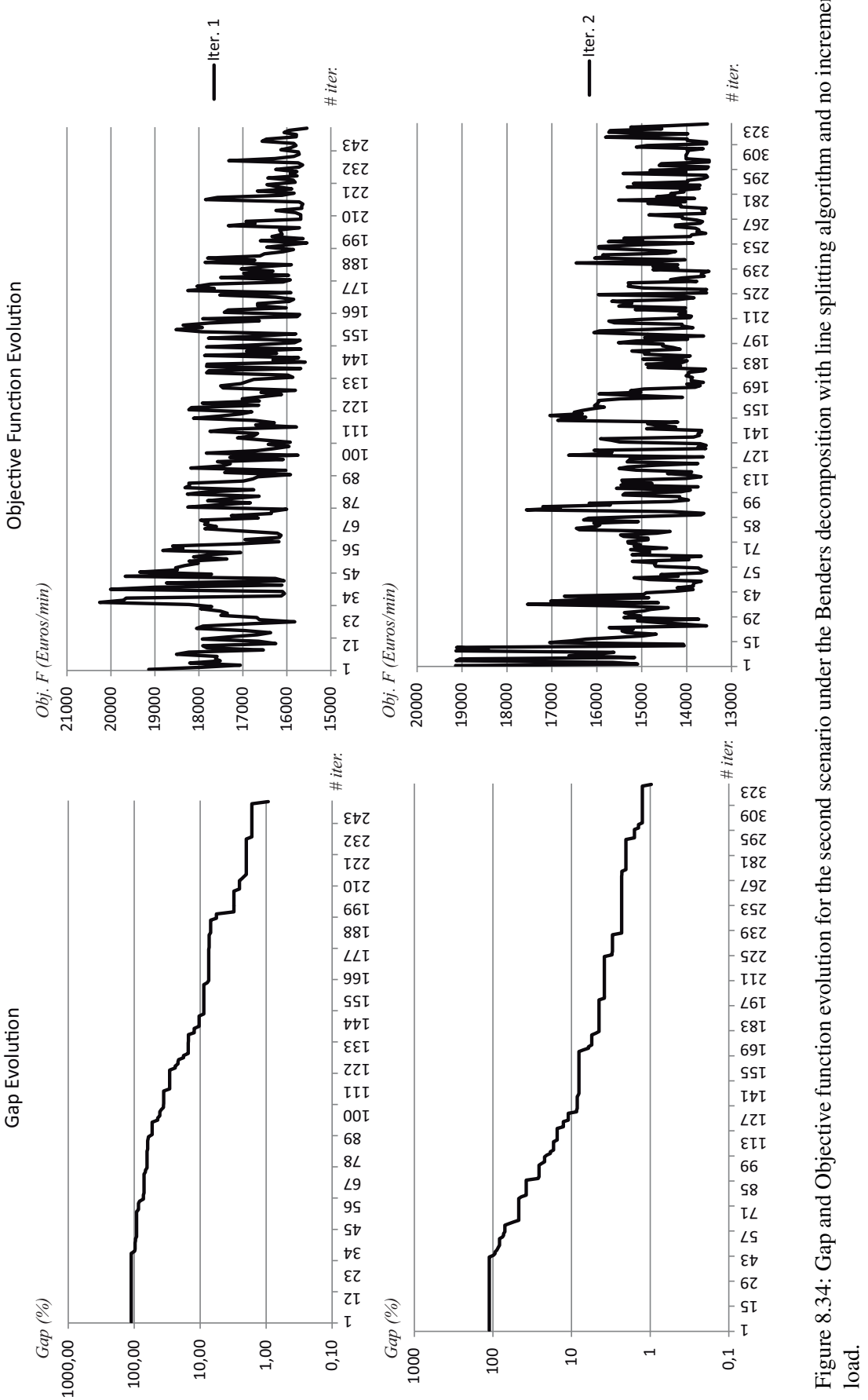


Figure 8.34: Gap and Objective function evolution for the second scenario under the Benders decomposition with line splitting algorithm and no incremental load.

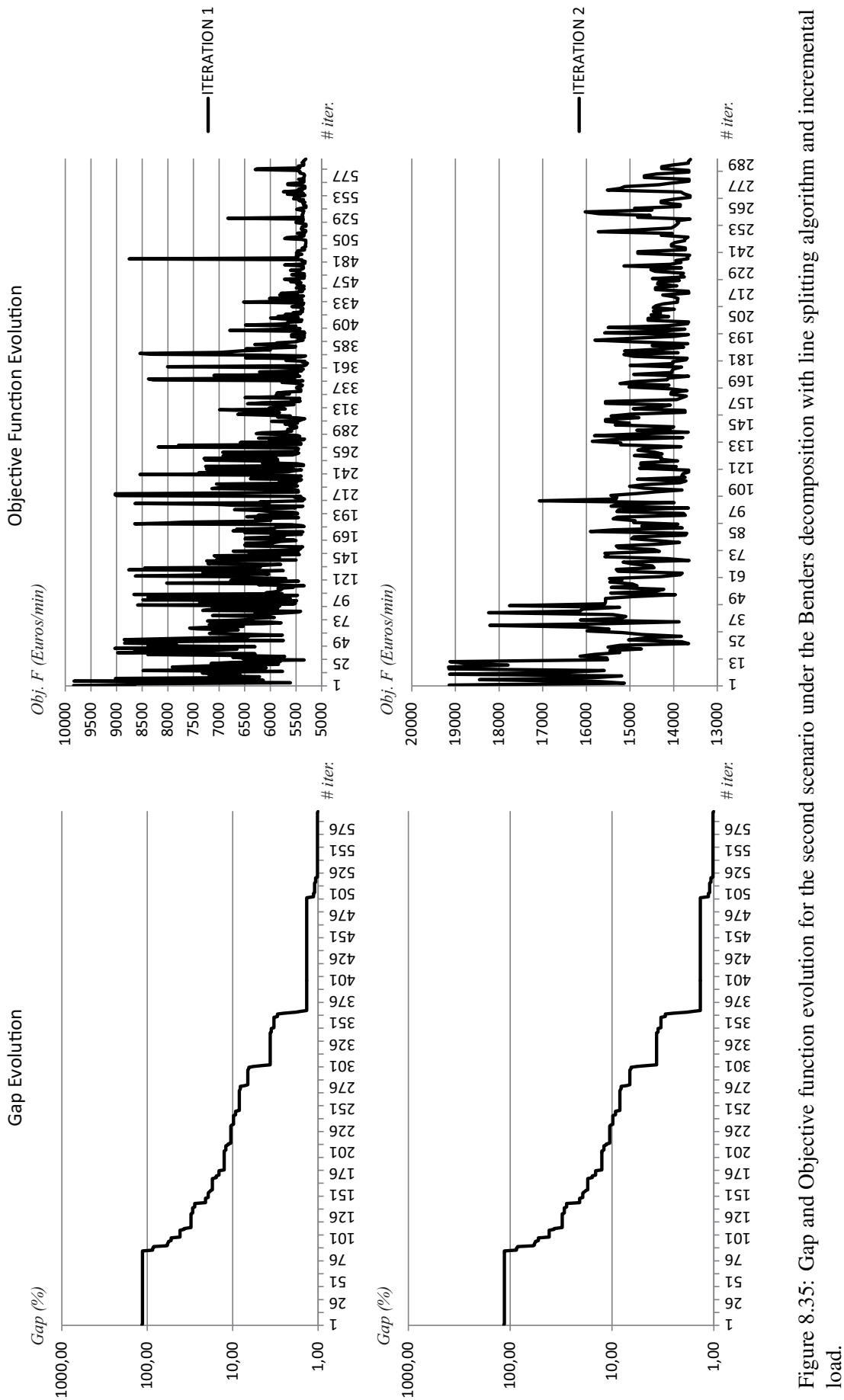


Figure 8.35: Gap and Objective function evolution for the second scenario under the Benders decomposition with line splitting algorithm and incremental load.

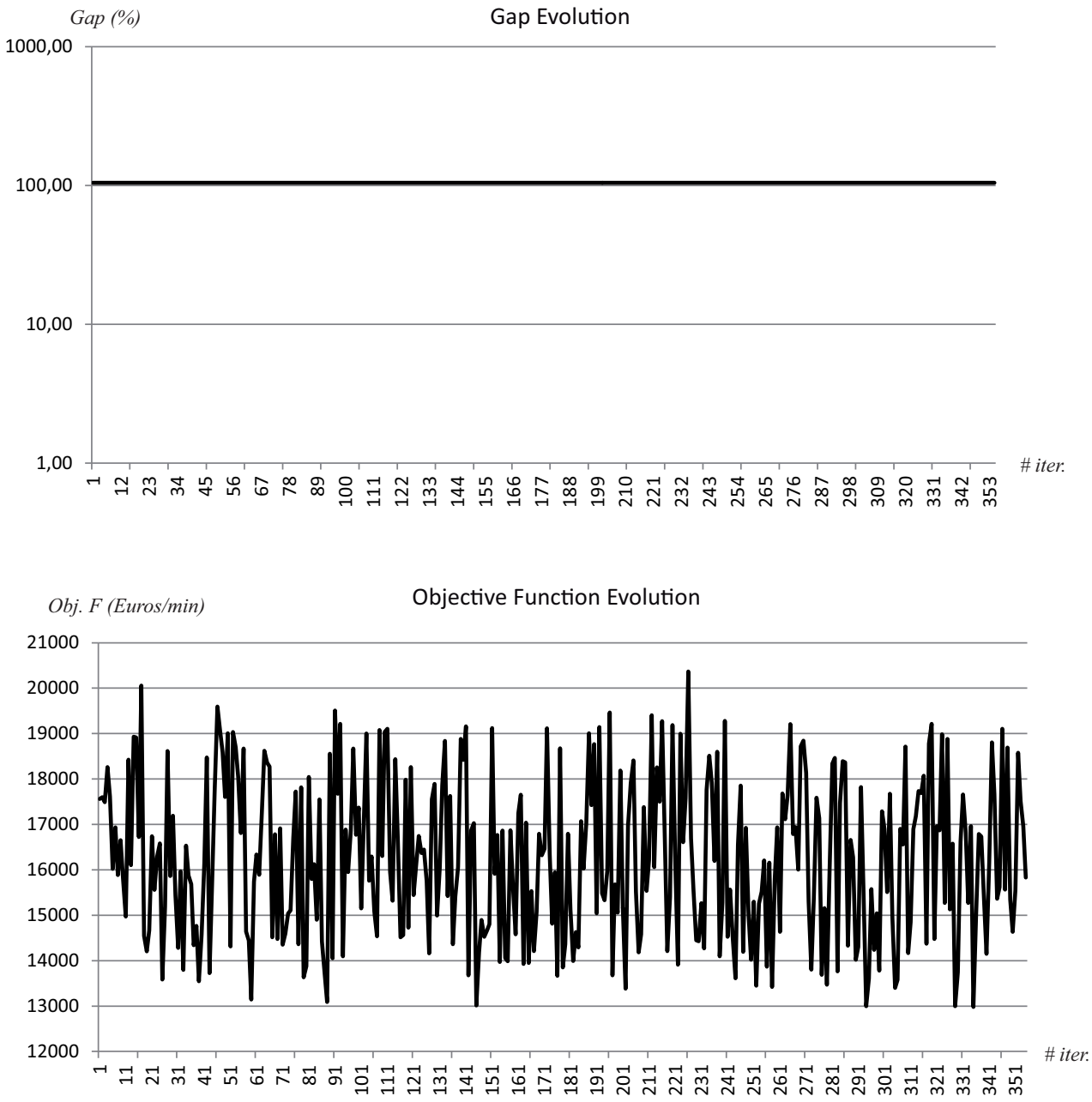


Figure 8.36: Gap and Objective function evolution for the third scenario under the Benders decomposition without line splitting algorithm.

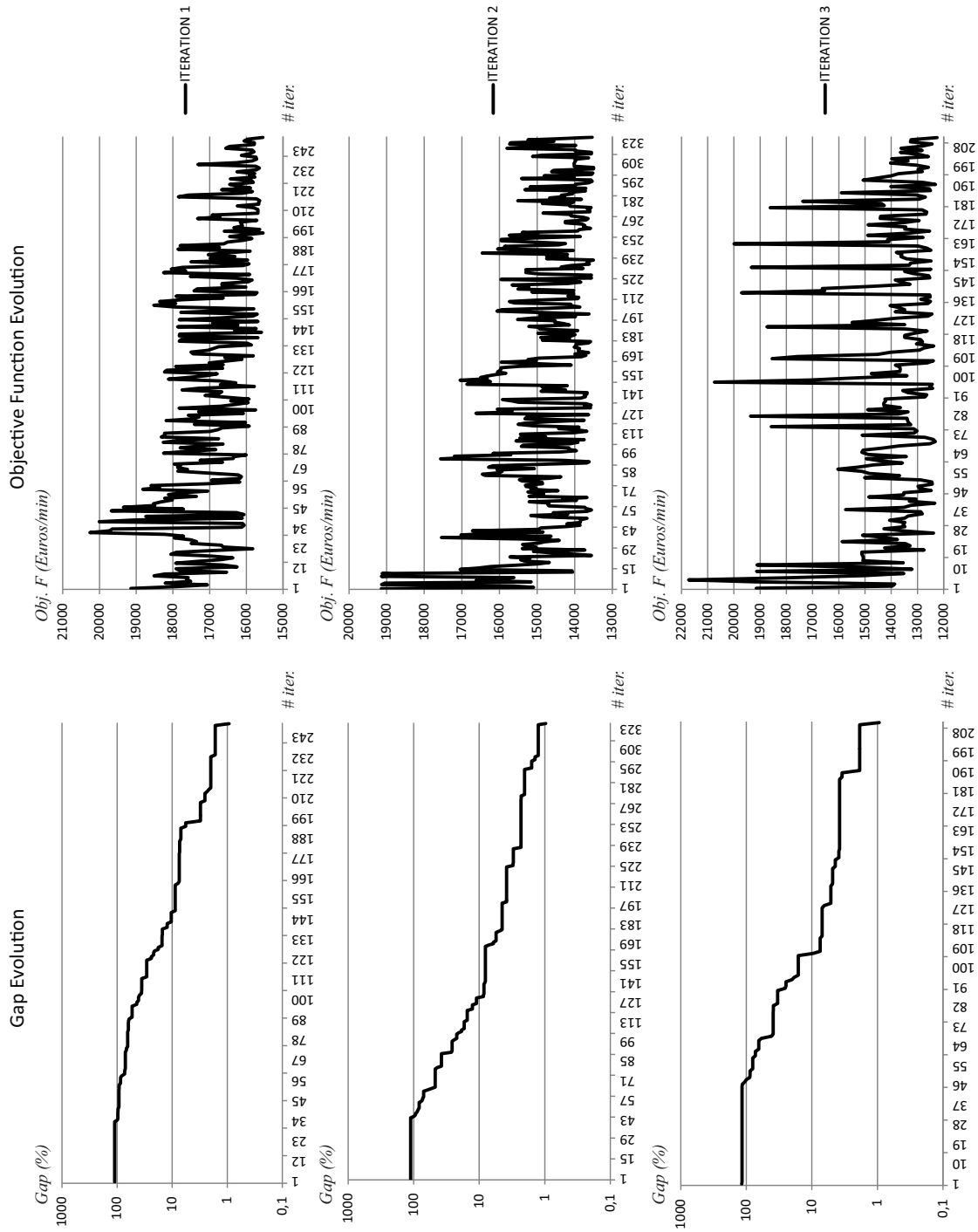


Figure 8.37: Gap and Objective function evolution for the third scenario under the Benders decomposition with line splitting algorithm and no incremental load.

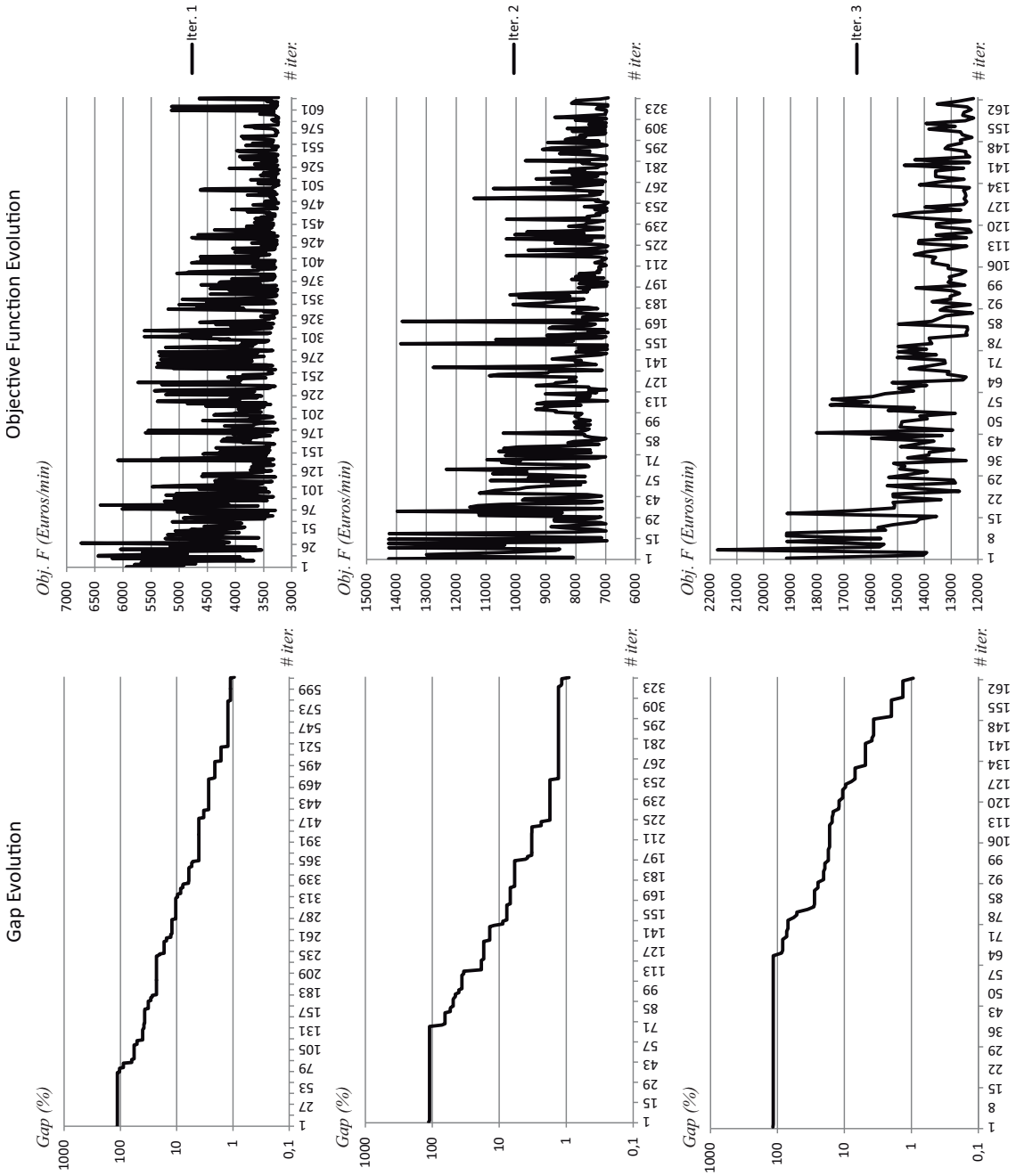


Figure 8.38: Gap and Objective function evolution for the third scenario under the Benders decomposition with line splitting algorithm and incremental load.

8.1.3 Study of the Effect of Express Service Design

This section is devoted to study how the inclusion of express service design affects the performance of the inelastic demand version of the model (IDM), as well as its solving time. To do that, we repeated all the experiments carried out in the two real study cases, i.e., the Seville network and the Santiago de Chile network, disallowing the model generating express services. This can be achieved by doing the following changes in the mathematical programming problem:

- Replacing variable y_i^l variable with y_i^l wherever y_i^l appears.
- Dropping off network design constraint (2.5).
- Adding the constraint $\sum_{c \in \Lambda} \delta_c^l \leq 1, \ \ \forall l \in L^N$ so that a corridor cannot be assigned to more than one line.
- Dropping off the passenger flow balance constraints (2.53) - (2.55) and expression $\Psi_{ij}^{p,l}$ from (2.51) - (2.52).

Figures (8.39) - (8.44) show the optimal global objective function (GOF) and the total CPU time ($TCPU$) spent on solving the inelastic demand version of the model, allowing and disallowing express service design (ESD). On the graphs, the continuous line represents the GOF and $TCPU$ values related to the IDM allowing ESD, whereas the discontinuous line shows the GOF and $TCPU$ values associated with the IDM disallowing ESD. In the remaining of the section, the IDM disallowing ESD will be referred to as the IDM with local service design (LSD), as written down in the legend box of the pictures.

Let us analyse, first, the results corresponding to the Seville Network. From Figures (8.39)-(8.41), we can see that the GOF coincides in scenarios where only one line can be constructed, no matter the solving methodology is. However, in the other scenarios the GOF is always higher in the IDM with LSD, although not much. The average relative gap is around 3%. As for the $TCPU$, it is almost always lower in the IDM with LSD and the time savings are higher as the number of lines under construction increases. However, there are two exception cases, which are included in the right side graphs of Figure 8.39, where some or both of these two observations do not hold. They correspond to the resolution of the instances where two and three lines are under construction, respectively, and where the Benders Decomposition without using any line splitting procedure is applied. The reason of this change in the performance is that the required time to solve these instances to optimality is higher than the time limit (24 hours), and thus, the actual total CPU times as well as the optimal objective functions are unknown.

Regarding the results of the Santiago de Chile Network, the picture is quite similar. One difference is that the GOF is almost identical in all experiments. Consequently, the lines representing the IDM with ESD and LSD overlap. The other difference is that the reduction in $TCPU$ of the IDM with LSD, with respect to the IDM with ESD, remains constant in the majority of the experiments. However, there are some exception cases, which are included in Figure 8.42, where both differences do not hold. The main reason is the same as the one stated in the analysis of the Seville results.

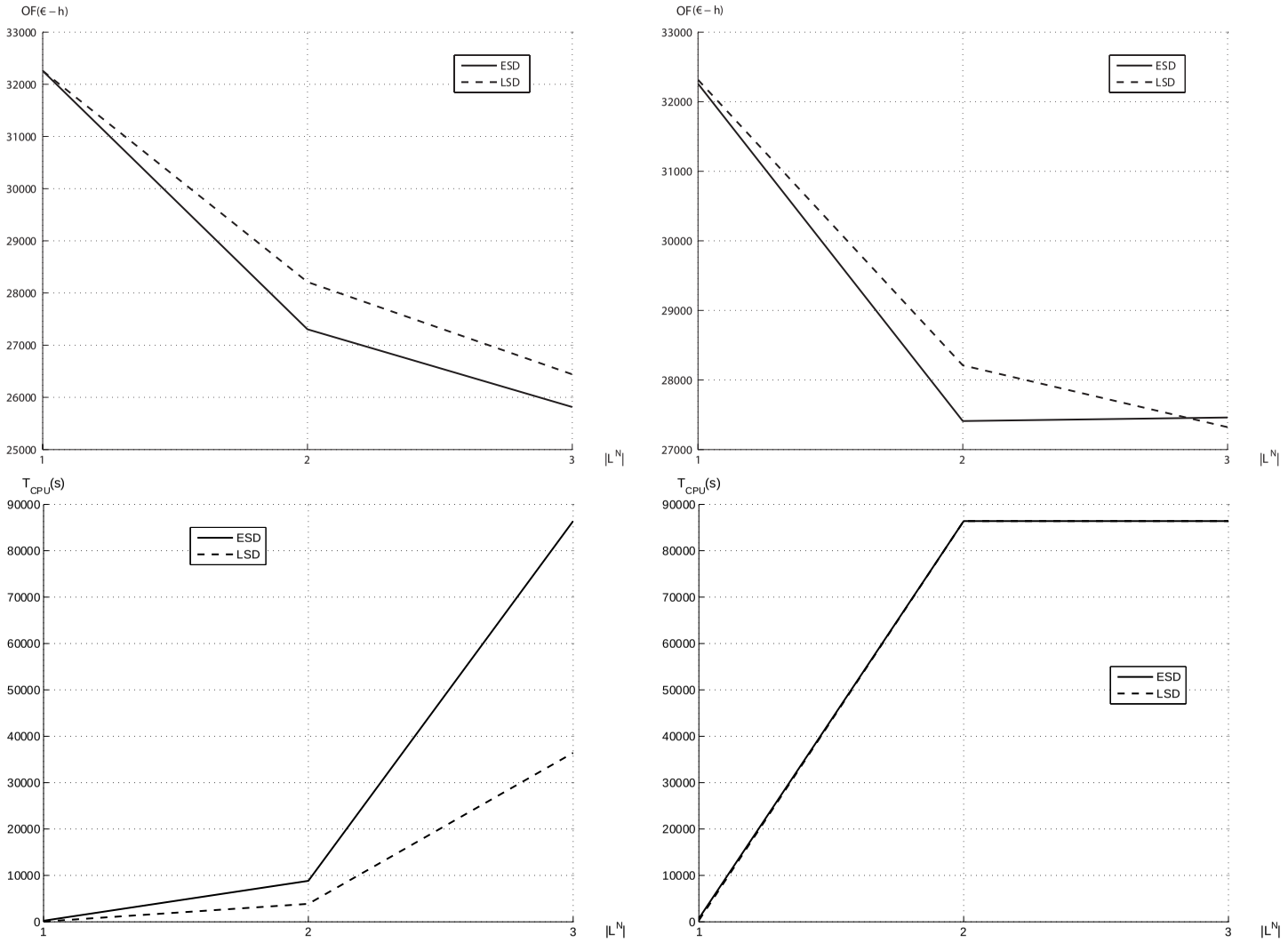


Figure 8.39: Comparison between express and local service design on the Seville Network without using line splitting techniques. At the top-left, optimal objective function value reported by CPLEX. At the bottom-left, total CPU time spent by CPLEX on solving the MILP. At the top-right, optimal objective function value reported by the Benders Decomposition. At the bottom-right, total CPU time spent by Benders Decomposition on solving the MILP

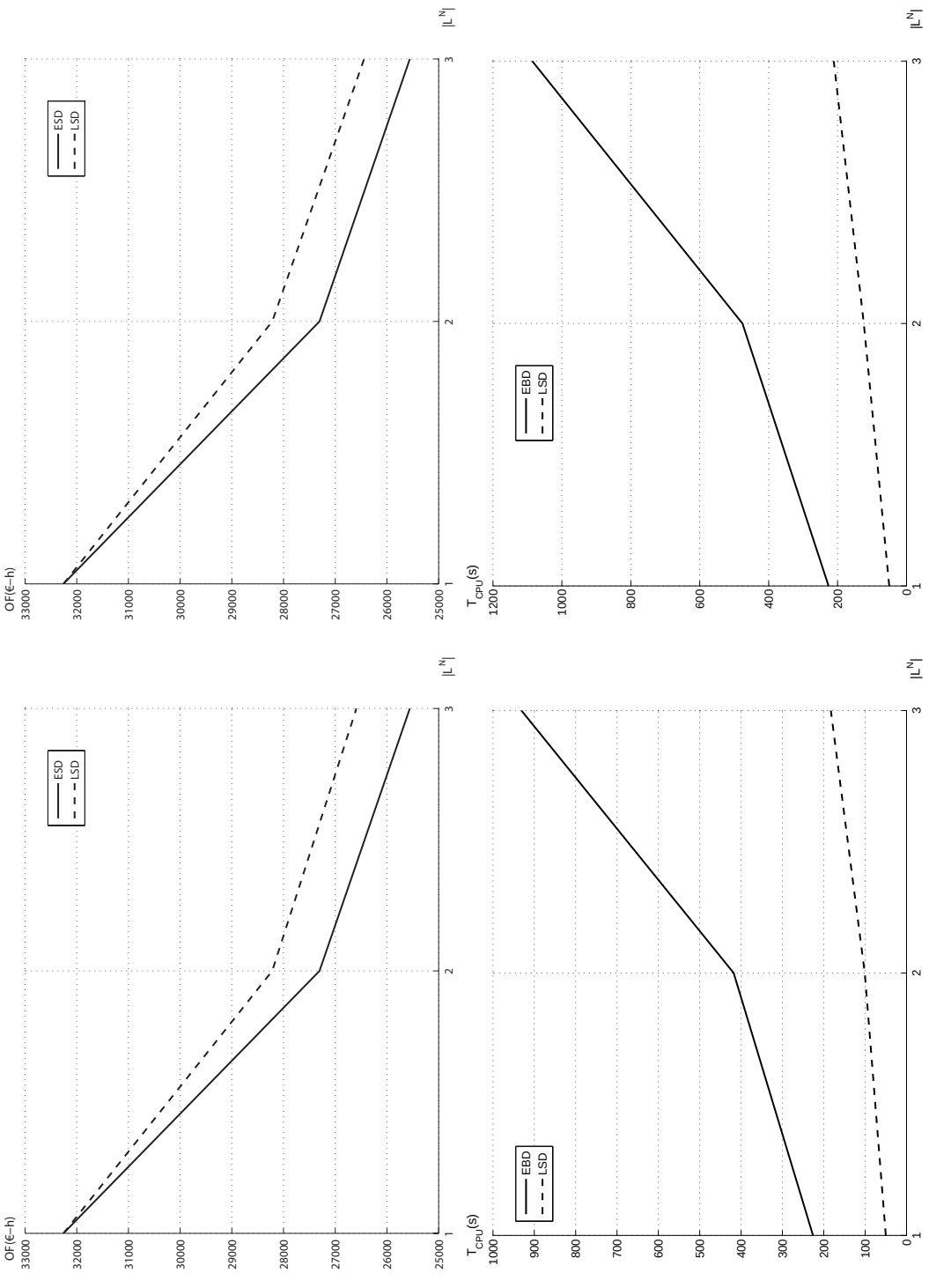


Figure 8.40: Comparison between express and local service design on the Seville Network for each line splitting technique using CPLEX as the MILP solving technique. At the top-left, optimal objective function value for the line splitting algorithm with non-incremental load. At the bottom-left, total CPU time for the line splitting algorithm with non-incremental load. At the top-right, optimal objective function value for the line splitting algorithm with incremental load. At the bottom-right, total CPU time for the line splitting algorithm with incremental load

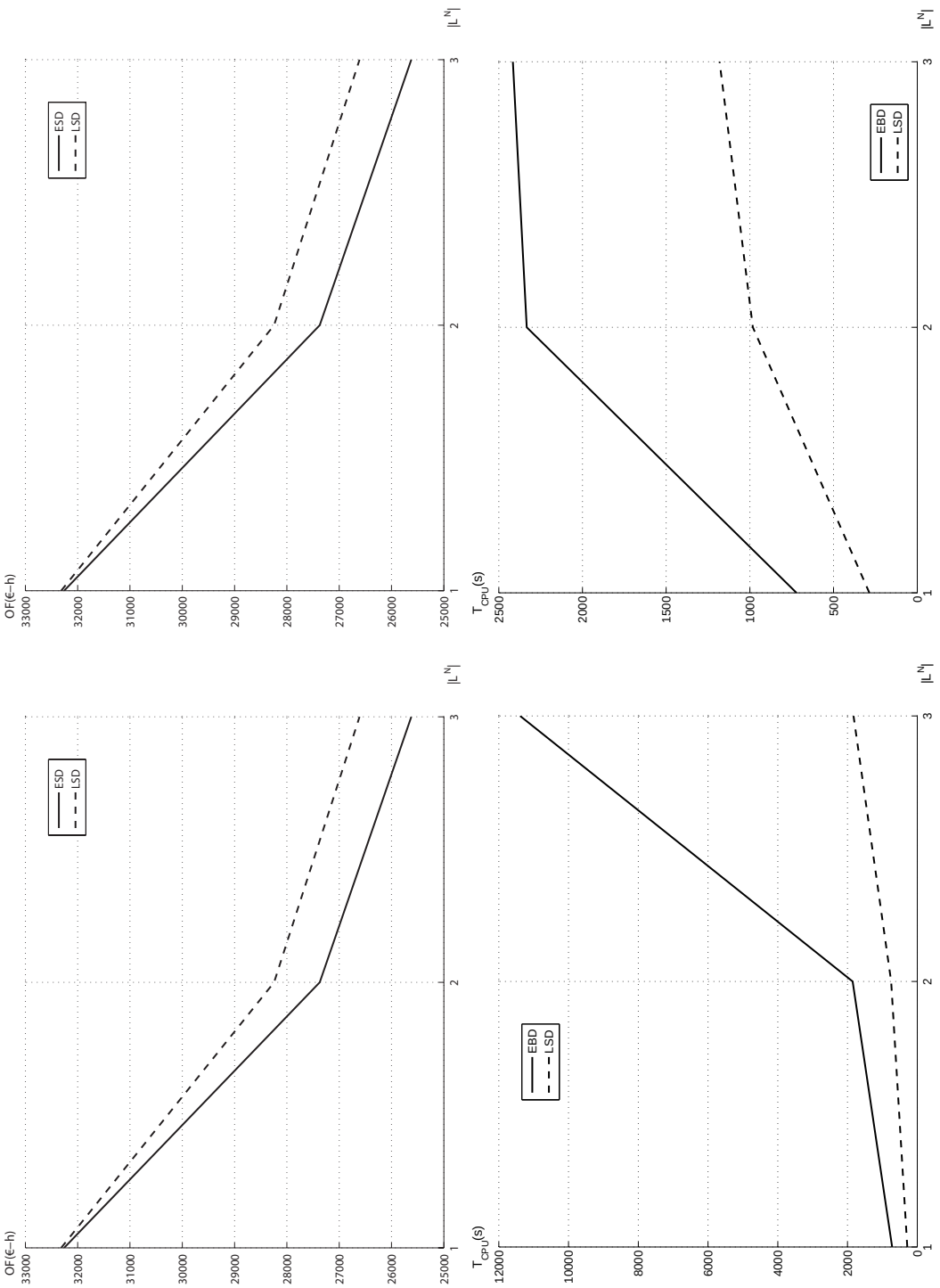


Figure 8.41: Comparison between express and local service design on the Seville Network for each line splitting technique using the Benders Decomposition as the MILP solving technique. At the top-left, optimal objective function value for the line splitting algorithm with non-incremental load. At the bottom-left, total CPU time for the line splitting algorithm with non-incremental load. At the top-right, optimal objective function value for the line splitting algorithm with incremental load. At the bottom-right, total CPU time for the line splitting algorithm with incremental load

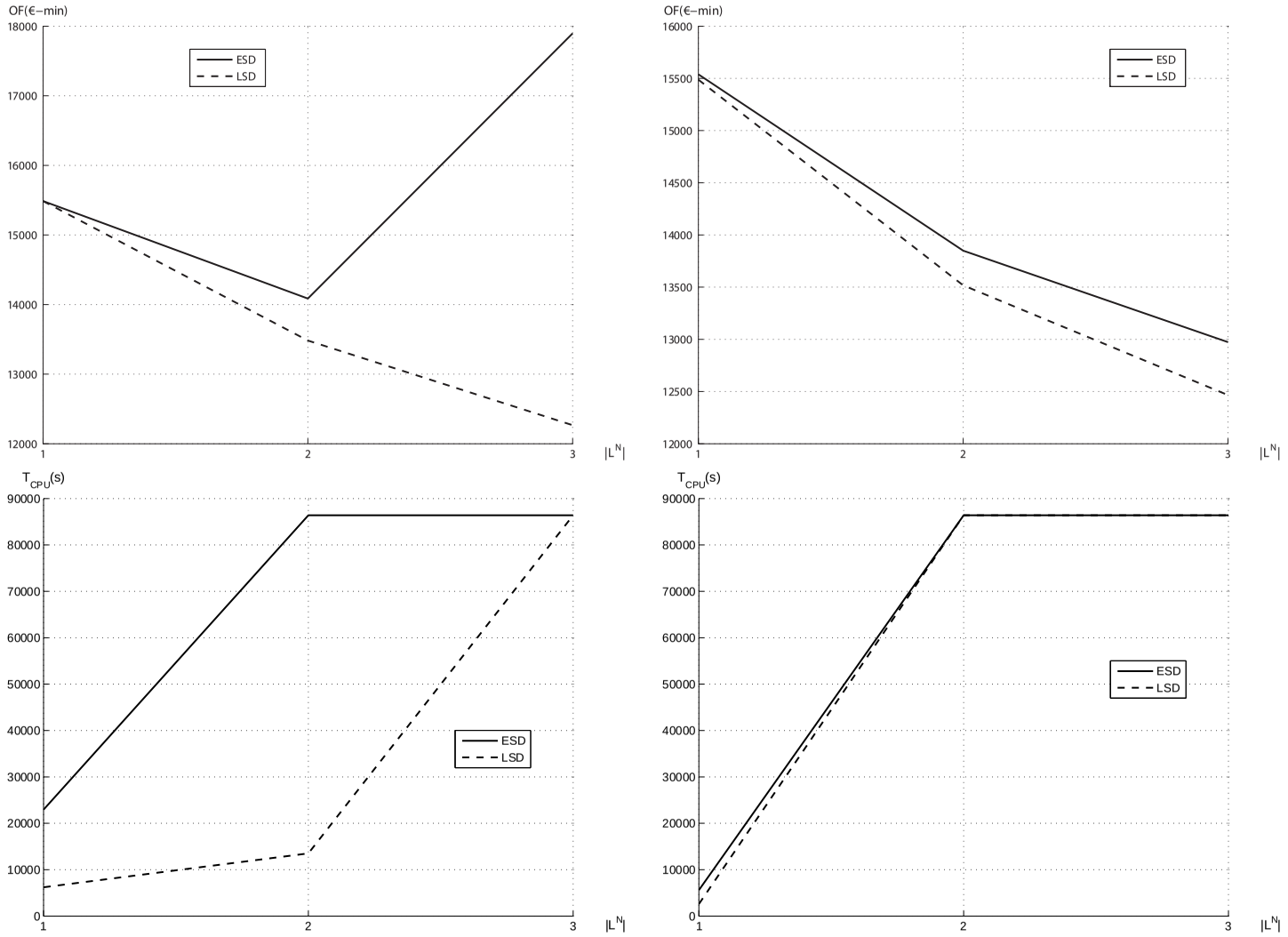


Figure 8.42: Comparison between express and local service design on the Santiago de Chile Network without using line splitting techniques. At the top-left, optimal objective function value reported by CPLEX. At the bottom-left, total CPU time spent by CPLEX on solving the MILP. At the top-right, optimal objective function value reported by the Benders Decomposition. At the bottom-right, total CPU time spent by Benders Decomposition on solving the MILP

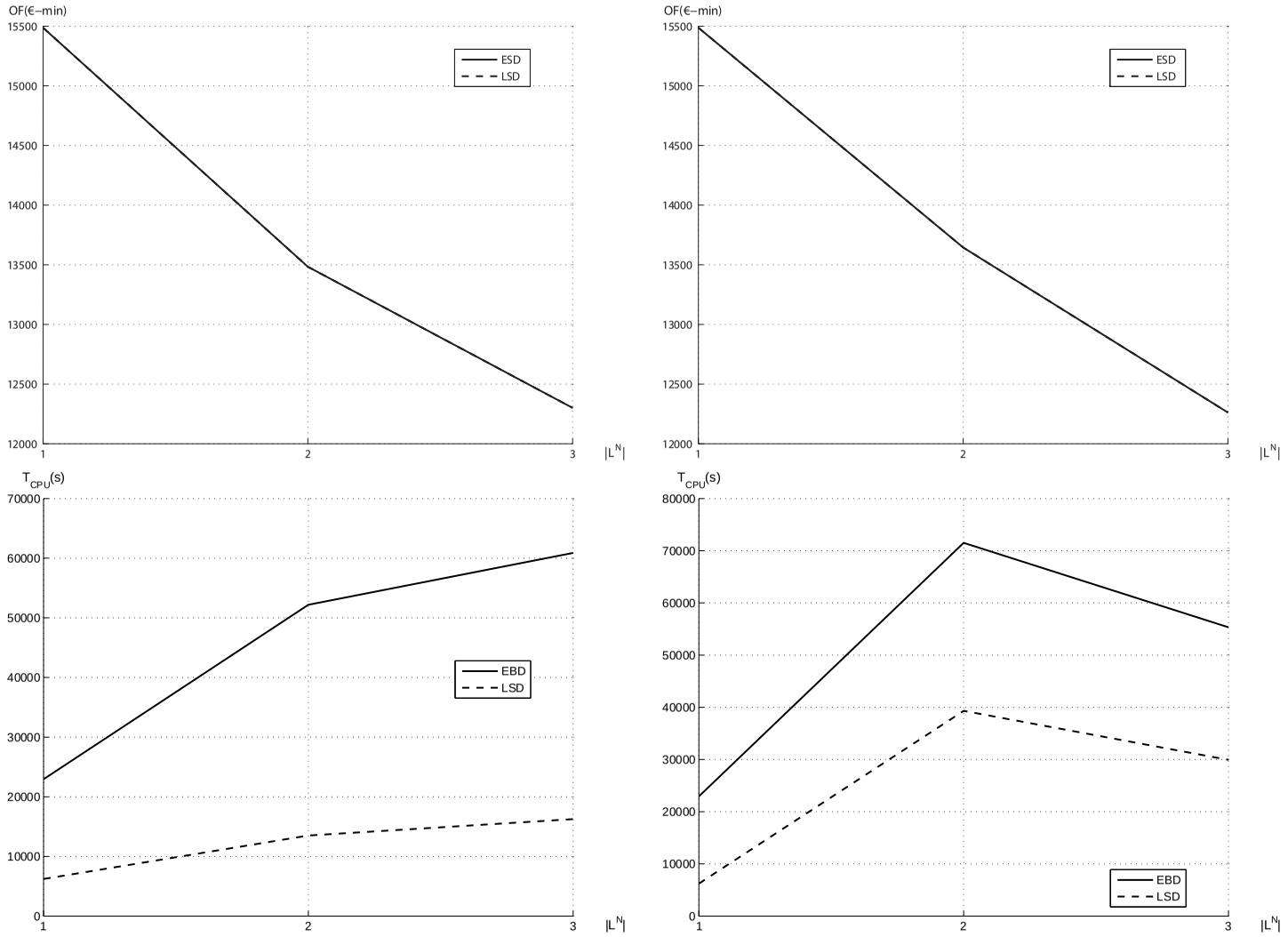


Figure 8.43: Comparison between express and local service design on the Santiago de Chile Network for each line splitting technique using CPLEX as the MILP solving technique. At the top-left, optimal objective function value for the line splitting algorithm with non-incremental load. At the bottom-left, total CPU time for the line splitting algorithm with non-incremental load. At the top-right, optimal objective function value for the line splitting algorithm with incremental load. At the bottom-right, total CPU time for the line splitting algorithm with incremental load

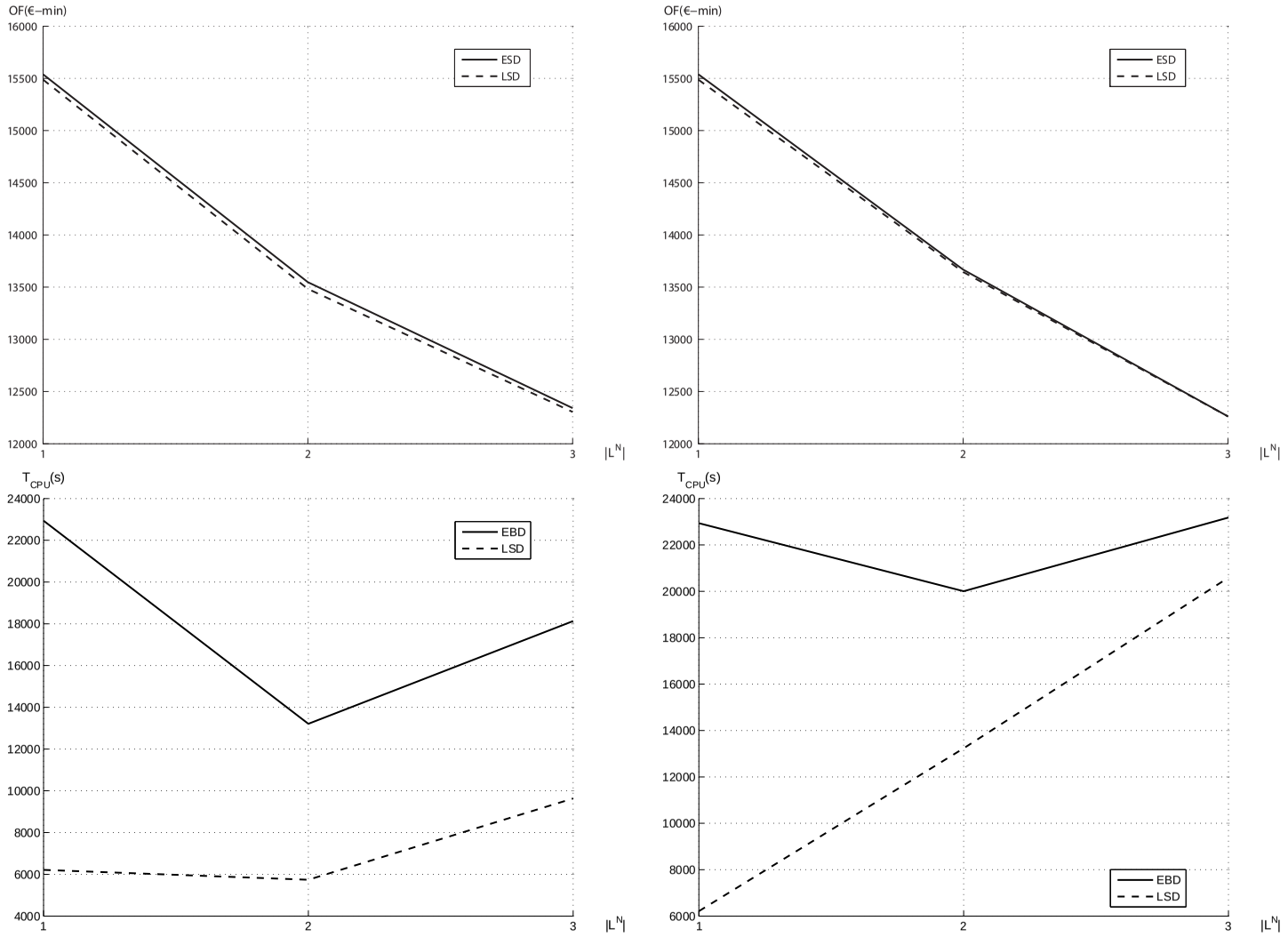


Figure 8.44: Comparison between express and local service design on the Santiago de Chile Network for each line splitting technique using the Benders Decomposition as the MILP solving technique. At the top-left, optimal objective function value for the line splitting algorithm with non-incremental load. At the bottom-left, total CPU time for the line splitting algorithm with non-incremental load. At the top-right, optimal objective function value for the line splitting algorithm with incremental load. At the bottom-right, total CPU time for the line splitting algorithm with incremental load

8.1.4 Conclusions

As shown in the results subsections of the study cases, the use of Benders Decomposition depends on the size of the problem in the inelastic demand version of the model. For moderate-sized networks, it is more suitable to employ the Branch & Bound of CPLEX, whereas for larger networks, Benders Decomposition gives (near-) optimal solutions with much less computational time and within a gap of less than 1%.

The inelastic demand version of the model can be solved to (near-) optimality within a reasonable amount of time by carrying out just a few approximation methods. For instance, in cases where more than one line is under construction, it is more appropriate to employ the line splitting algorithm under the non-incremental variant, since it provides solutions no worse than 1% of relative gap at one less order of magnitude of CPU time.

Focusing now on the Benders scheme, the use of the Papadakos schema (shown in Table 6.3 of chapter 6) is more appropriate for cases where the number of corridors is low (let's say around 100), whereas our specialized Benders scheme (shown in Table 6.6 of chapter 6) gives the best results for a larger number of corridors.

Finally, the solution provided by the model seems quite reasonable in terms of line layout and planning configurations. Also, the types of scenarios which can be emulated are varied and may include some working lines. Therefore, the model can be used as an aid for operators to make important and timely decisions, especially in relation to railway and underground networks.

Chapter 9

Summary and Conclusions

This chapter presents a summary of the reported research, highlighting the relevant results and proposing some lines for further research. Its structure is as follows. Firstly, an outline of the relevant work done in this research is explained; then the general conclusions are made, based mainly on the experimentation chapter results; and finally, some lines for further research are pointed out. These lines are not intended to embrace all the possibilities, and thus further proposals will be welcome.

9.1 Summary

We have presented a network design and line frequency setting model which allows us to construct a rapid transit network. On the one hand, the network design uses a set of candidate stations to determine the extension of the current set of working lines without exceeding the available network infrastructure budget. On the other hand, the line frequency setting assigns vehicles and services to the constructed lines, taking into account link and vehicle capacity restrictions and adjusting to the vehicle fleet size and planning horizon requirements.

A mix-integer linear mathematical programming problem ($MILP$), presented in chapter 2, implements these features. The routing part is stated by means of three different formulations. The first two are exact formulations, which come from state-of-the-art works related to vehicle routing problems, whereas the last one consists of an explicit enumeration of the line's routing.

The solving approach is based on a quasi-exact methodology, which simultaneously determines the two intertwined problems (the network design and the line frequency setting). The first part of the work accounts for some reasonable simplifications we have made in order to solve real-sized networks. The first of these simplifications is the use of the third routing formulation, which gives as inputs to the optimization problem a reduced set of candidate sequences of chained line segments (corridors). However, it does not provide the service nodes (stops), which are determined in the optimization phase. The second and last one is the development of a heuristic to solve more than one line. The method carries out a line splitting in which a sequential series of small instances with the same mathematical structure as the $MILP$ are solved.

The sequence of chained line segments is computed by means of a K-shortest path algorithm, which includes some constraint satisfaction. The main building blocks consist of a rectilinear corridor generator and a circular corridor generator. Both generators are based on different adaptations of the Yen algorithm [189] to cope with the type of topology and some constraint verification. The last mentioned feature gives a novelty to the approach since related works [68], [30], [174] do not take into account constraint satisfaction within the generation process and/or they focus only on directed graphs (last reference).

The line splitting algorithm is an ad hoc heuristic that takes advantage of one of the model's features that is already in operation: the option of working with a subnetwork. Therefore, we can determine each line separately and add it to the model in subsequent resolutions as a fictitious working line. The last resolution provides the global solution.

In collaboration with these two techniques, we have also proposed a decomposition method, presented in chapter 6, which is based on the Benders Decomposition (BD) [14] and some ad hoc techniques, as well as the enhancements made by Magnanti & Wong [125], Papadakos [147], and McDaniel & Devine [136]. These additional methods have significantly improved the BD convergence and reduced the number of integral master problem resolutions, these being two key aspects that have proven to be time consuming. The enhanced version of the BD has been used to solve each small $MILP$ to (near-) optimality.

As a final contribution, a preliminary mathematical programming problem has been introduced that takes into account public transport demand elasticities that originate with the modal choice between private car or public transport. Users face this choice when changes in the public transportation system are made, such as the inclusion of new lines and/or the involved frequency changes in lines already in operation. This model has been formulated as a mixed-integer linear bilevel programming problem whose upper level function is conceptually the same as that of the inelastic case developed in chapter 2. The lower level, however, combines a modal choice model with a passenger assignment model, which is solved by a specific technique suggested in [40]. This technique exploits the fact that integer variables appear only in the upper level and, in this case, an adaptation of the Benders decomposition algorithm can be devised in order to solve this specific class of mixed-integer bilevel programming problems.

9.2 Conclusions

As shown in the computational tests of chapter 8, the inelastic demand version of the model can be solved to near-optimality within a reasonable amount of time while doing the aforementioned simplifications. For instance, in cases where more than one line is under construction, it is more appropriate to employ the line splitting algorithm under the non-incremental variant, since it provides solutions no worse than 1% of relative gap within one less order of magnitude in CPU time.

Regarding the $MILP$ solving technique, the use of Benders Decomposition depends on the problem size. For moderate-sized networks, it is more suitable to employ the Branch & Bound of CPLEX, whereas for larger networks, Benders Decomposition gives the (near-) optimal solutions with much less computational time, within a gap of less than 1%.

Focusing now on the Benders decomposition scheme, the use of the Papadakos schema (see Table 6.3 of chapter 6) is more appropriate for cases where the number of corridors is low (let's say around 100); whereas for a larger number of corridors, our specialized Benders scheme (see Table 6.6 of chapter 6) gives the best results.

Finally, in terms of line layout and planning configurations, the solution provided by the model seems quite reasonable (see figures 8.5 - 8.7 in the Seville results section and figures 8.24 - 8.27 in the Santiago de Chile results section) and the type of scenarios which can be emulated are varied. They may also include some working lines. Therefore, the model can be used as an aid to operators who need to make important decisions within a reasonable time horizon, especially those related to railway and underground networks.

As for the elastic demand version, the model has been tested only on small-sized networks because it is in early development. However, it seems to provide reasonable solutions. According to the results subsection

7.5, we have seen that many of the od-demand pairs, which perform too many transfers and/or long walking distances in the inelastic mode, prefer to take the private mode in this elastic version. The changes are not very significant in the public transportation network layout, but they are in the number of planning resources used (vehicles and number of services). Thus, it allows savings in the planning budget while keeping good levels of service quality.

9.3 Directions for further research

This work opens several lines for further research. Roughly speaking, they can be split into two groups:

a) Model extensions

b) Algorithmic improvements.

Group a) also implies development of new methodologies, which could consider the following issues:

a1) The inclusion of passenger waiting times at stops.

a2) The consideration of variable dwell times at stops.

a3) The consideration of stop capacity.

a4) The consideration of passenger comfort.

a5) The extension of the network flow model to deal with bus systems.

a6) Explicit modeling of the entropy function in the model with elastic demand.

In our view, the first issue should be tackled first. It entails discarding the current Benders decomposition scheme and moving on to the generalized Benders decomposition (GBD) of Geoffrion [83], which is capable of solving non-linear convex problems. According to a preliminary study on the work of de Camargo & de Miranda Jr. [52], all the methods embedded into the Benders scheme (i.e., the enhancements of [125], [147] and [136], as well as the ad hoc techniques applied to both inelastic and elastic demand versions) can be easily incorporated into the GBD. As a first attempt, passenger waiting times under non-congested scenarios will need to be dealt with. That means that passengers can board the first vehicle halting at the stop. In other words, we suppose that vehicles have enough capacity to hold passengers, whatever the demand level is. In this situation the passenger assignment model of Spiess & Florian [167] seems appropriate, and incorporating it into the GBD entails solving two extra problems, apart from the ones stated in this research. Both consist of resolving two shortest path problems which have no link capacities. Thus, through the use of an efficient algorithm like Dijkstra [58], they can be solved independently for each group of commodities with the same origin or destination (depending on the formulation choice). The difference lies in the fixed layout network being considered, and its link costs. One is related to the solution of the current master problem, whereas the other is associated with the current corepoint, which is an interior point of the master problem's feasible solution.

The second issue is also quite important and needs to be incorporated into the GBD as well. We have also carried out a preliminary study and it does not entail extending the GBD proposed for the first point. Moreover, the two extra problems, mentioned above, maintain the same mathematical structure. However, more extra variables and constraints are added to all of the Benders subproblems.

The third issue plays an important role as well, since the stop platform must be well-dimensioned under allowable density levels to hold the maximum number of passengers (which occurs during peak hours).

This dimensional aspect entails consideration of a tradeoff between passenger waiting times and the cost per square meter of the platform area. Conceptually speaking, it is not easy to model, and thus we do not have any preliminary study.

The fourth point has no high priority but can be interesting to implement as well. We are thinking of adapting the approach of Cadarso & Marín [17] to our model. The authors use the notion of excess in-vehicle demand, where the word "excess" accounts for the number of standees under a given density level. In that work, two density levels are used for semi-congested and congested scenarios, respectively, and each excess of demand is penalized in the objective function by means of two different weights.

The fifth point does not entail any change in solving the scheme. However, it requires some effort in order to avoid an excessive increase in the number of new decision variables and constraints in the new passenger assignment model. Roughly speaking, we will need to consider the following:

- Several candidate service nodes associated with every origin and destination demand node.
- Distinction of the way in which the demand is served in the routing model.

The first issue entails adding some walking links between the centroid node (origin or destination demand node) and the nearest alighting/boarding nodes. Consequently, the corridor generation algorithm needs to take into account all these combinations, which means that more corridors will be generated. On the other hand, the second point will require a reformulation of all the variables and constraints related to the role of the node. As a first attempt, we are thinking of using two binary variables, one for each way of circulation. Each variable will denote whether or not vehicles halt at the node for the given way of circulation.

Finally, the sixth issue deals directly with the entropy function. Currently, a piecewise linear function approximates the entropy function by means of lines which are active at certain intervals of the original function. This approach adds too many constraints to subproblems $SP1$, $SP2$, $IWMP1$ and $IWMP2$, and thus it entails a significant increase in solving time. Moreover, an approximation error, whose magnitude depends on the number of computed lines, is also produced. However, dealing with an explicit representation of the entropy function entails formulation of the abovementioned subproblems as non-linear continuous subproblems, due to the entropy term in the objective function. This non-linearity issue can be overcome by means of the GBD as well, since the entropy function is convex.

Regarding algorithmic improvements, we have in mind the following points:

b1) Benders convergence improvements.

b2) Efficient resolution of the reduced integral master problem in phase 0.

b3) Development of a preprocessing technique in order to reduce the number of Benders cuts used in the Master Problem with elastic demand.

b4) Implementation of the whole methodology in a low level language environment.

The first point can be achieved by implementing two methods which can also be combined with the GBD to continuously enhance convergence of the decomposition. One of them is based on the work of Saharidis *et al* [157], who generate a constraint bundle for each Benders iteration. It includes the traditional Benders cuts, plus some additional cuts which aim to constrain more the master variables. The other method relies on the idea of Sherali & Lunday [160], who define the notion of maximal non-dominated cuts. This idea allows obtaining tighter cuts by computing the corepoint. This is situated in the most interior of the politope region, whose vertices are defined by the master problem solutions computed so far. Both approaches

require computation of extra Benders subproblems, and thus we do not know beforehand how much the whole computational time will decrease. However, these techniques appear promising, since the authors report good results for several instances of the fixed charge network flow problem.

The second point seems to be a good direction for successful improvement. Currently, the reduced integral master problem obtains a rounded (feasible) solution from the relaxed master problem, but it takes a lot of time for medium- and large-sized networks. To address this point, we are thinking of first analyzing which are the key factors involved in choosing the best line corridor from those which are active in each line under construction. Then, this knowledge will be used to develop a procedure for approximating them without any optimization.

The third point is mandatory for solving medium- to large-sized networks with the mixed-integer linear bilevel programming problem for elastic demand. Currently, it is only capable of solving small-sized networks to (near-) optimality, because of the large amount of time needed to solve the master problem to optimality. This increment in time stems from the presence of the δ variables, which ensure that the master problem is bounded. A good preprocessing routine that reduces the number of Bender's cuts may significantly reduce its solving time. This would also reduce the number of variables that the master problem must face.

The last point seems to be a very promising improvement, since the Benders decomposition performance decreases significantly when using a high level language like *AMPL*. To mention some of the causes: each time the master problem is solved, only a new active constraint is added, thus it would be desirable to re-optimize it not only by taking into account the last solution, but also by using the calculated branch & bound tree. However, this is not possible since *AMPL* tells *CPLEX* to erase the current branch & bound tree so that a new one is created from scratch. Another drawback is that there is no control of memory allocation. Because of this, no temporary files are created to save information needed for *AMPL* and *CPLEX* to communicate with each other; and this slows down the resolution of each Benders problem.

Bibliography

[1] Abdulaal, M., LeBlanc, L.J. *Methods for combining modal split and equilibrium assignment models*. Transp. Sci. 13: 292 - 314, 1979.

[2] Agrawai, J., Mathew, T.V. *Transit route design using parallel genetic algorithm*. J. Comput. Civ. Eng. 18(3): 248 - 256, 2004.

[3] Ahuja, R.K., Magnanti, T.L., Orlin, J.B. *Network Flows: Theory, Algorithms and Applications*. Prentice Hall, 1993.

[4] Androutsopoulos, K. N., Zografos, K. G. *Solving the k-shortest path problem with time windows in a time varying network*. Operations Research Letters 36: 692-695, 2008.

[5] Baaj, M.H., Mahmassani, H.S. *TRUST: A Lisp Program for the Analysis of Transit Route Configuration*, Transportation Research Record 1283, Transportation Research Board, Washington, D.C., 125-135, 1990.

[6] Baaj, M.H., Mahmassani, H.S. *An AI-Based Approach for Transit Route System Planning and Design*, Journal of Advanced Transportation 25(2): 187-210, 1991.

[7] Baaj, M.H., Mahmassani, H.S. *Hybrid Route Generation Heuristic Algorithm for the Design of Transit Networks*, Transportation Research Part C 3(1): 31-50, 1995.

[8] Bard, J.F., Moore, J.T. *A branch and bound algorithm for the bilevel programming problem*. SIAM J. Sci. Stat. Comp. 11: 281292, 1990.

[9] Barnhart, C., *Airline Schedule Optimization*. In: Belobaba, P., Odoni, A: and Barnhart, C. editor(s). The Global Airline Industry, Wiley. ISBN: 978-0-470-74077-4, 2009.

[10] Barra, A., Carvalho, L., Teypaz, N., Cung, V.-D., Balassiano, R. *Solving the Transit Network Design problem with Constraint Programming*. Proceedings of the 11th World Conference in Transport Research, 2007.

[11] Bazaraa, S., Bazaraa, S.M.S., Shetty, C.M., Sherali, H.D. *Non-Linear Programming. Theory and Algorithms*. John Wiley and Sons (WIE), May 1993.

[12] Bhat, C. *Discrete Choice Modeling*. Class notes, UT-Austin, 2003.

[13] Bellman, R., Kabala, R. *On kth best policies*. Journal of SIAM 8: 582 - 585, 1960.

[14] Benders, J. *Partitioning procedures for solving mixed-variables programming problems*. Numerische Mathematik 4(3): 238-252, 1962.

[15] Borndörfer, R., Grotschel, M., Pfetsch, M. E. *A Column-Generation Approach to Line Planning in Public Transport*. Transportation Science 41(1): 123 - 132, 2007.

[16] Bruno, G., Ghiani, G., Improta, G. *A multi-modal approach to the location of a rapid transit line*. European Journal of Operational Research 104: 321 - 332, 1998.

[17] Cadarso, L., Marín, A. Robust rolling stock in rapid transit networks. Computers & Operations Research 38: 1131-1142, 2011.

[18] Cadarso, L., Marín, A., *Integration of Timetable Planning and Rolling Stock in Rapid Transit Networks*. Ann Oper Res 99(1): 113 - 135, 2012.

[19] Cadarso, L., Marín, A. *Robust passenger oriented timetable and fleet assignment integration in airline planning*. J Air Transp Manag 26: 44-49, 2013.

[20] Cadarso, L., Marín, A., and Maróti, G. *Recovery of Disruptions in Rapid Transit Networks*, Transport Res E-Log 53: 15-33, 2013a.

[21] Cadarso, L., Vaze, V., Barnhart, B., Marín, A. *Integrated Airline Scheduling: Considering Competition Effects and the Entry of the High Speed Rail*, Submitted to *Transp Sci*, 2013b.

[22] Carrese, S., Gori, S. *An urban bus network design procedure*. M. Patriksson and M. Labb (eds.). Transportation Planning 64: 177 - 195, 2004.

[23] Ceder, A. *Bus frequency determination using passenger count data*. Transportation Research 5/6(18):439-453, 1984.

[24] Ceder, A. *Optimal design of transit short-turn trips*. Transportation Research record 1221:8-22, 1990.

[25] Ceder, R.B., Wilson, N.H. *Bus Network Design*, Transportation Research Part B, Vol. 20, No. 4, P 331-344, 1986.

[26] Ceder, A. *A procedure to adjust transit trip departure times through minimizing the maximum headway*. Computers and Operations Research Journal 18(5):417-431, 1991.

[27] Ceder, A., Tal, O. *Timetable synchronization for buses. Computer-Aided Transit Scheduling*. Proceedings of the Seventh International Workshop on Computer-Aided Scheduling of Public Transport, Cambridge, MA, USA, 245-258, 1997.

[28] Ceder, A., Israeli, Y. *User and operator perspectives in transit network design*. Transportation Research Record 1623: 37, 1998.

[29] Cepeda, M., Cominetti, R., Florian, M. *A frequency-based assignment model for congested transit networks with strict capacity constraints: characterization and computation of equilibria*, Transportation Research Part B 40: 437-459, 2006.

[30] Cipriani, E., Gori, S., Petrelli, M. *A bus network design procedure with elastic demand for large urban areas*. Public Transport 4(1): 57-76, 2012.

[31] Cipriani, E., Fusco, G., Gori, S., Petrelli, M. *A procedure for the solution of the urban bus network design problem with elastic demand*. Advanced OR and AI Methods in Transportation: Proc., 10th Meeting of the EURO Working Group on Transportation, Publishing House of Poznan Univ. of Technology, Poland, 681 - 685, 2005.

[32] Cipriani, E, Gori, S, Petrelli, M. *Transit network design: A procedure and an application to a large urban area*. Transportation Research Part C 20(1): 3 - 14, 2012.

[33] Codina, E., Marín, A., López, F. *Diseño de líneas auxiliares de autobuses para situaciones de ruptura de los servicios de cercanías*. CIT2010. Madrid 2010.

[34] Codina, E., Marín, A., Ibaez, L.C., López, F. *Asignación de frecuencias a lneas auxiliares de transporte público en situaciones de elevada demanda*. MORE 2010. ISBN-13:978-84-693-4034-9

[35] Codina, E., Marín, A., López, F. *Assignment of services in bus lines under congestion*. EURO XXIV. Lisbon, 11-14th July 2010.

[36] Codina, E., Marín, A., López F. *Design of public transportation lines in congested scenarios*. CNU-MOpen, Centre de Recerca Matemtica. Bellaterra, October 2010.

[37] Codina, E., Marín, A., López, F. *A frequency setting model for auxiliary bus line on disrupted Rapid Transit networks*. IFORS. Melbourne 2011.

[38] Codina, E., Marín, A., López, F. *Modeling and optimization of frequencies on congested bus lines*. ACMOS, Saint Malo & Mont Saint Michel, 104 - 109, April 2012. ISBN: 978-1-61804-080-0.

[39] Codina, E., Marín, A., López, F. *A model for setting services on auxiliary bus lines under congestion*. TOP 21 (1): 48-83. Ed. Springer Verlag. 2013 doi: 10.1007/s11750-012-0250-z.

[40] Codina E., (2013) Personal communication.

[41] Colson, B., Marcotte, P., Savard, G. *An overview of bilevel optimization*. Annals of Operations Research 153: 235256, 2007.

[42] Cominetti, R., Correa, J. *Common-Lines and Passenger Assignment in Congested Transit Networks*, Transportation Science 35(3):250-267, 2001.

[43] Constantin, I., Florian, M. *Optimizing frequencies in a transit network : a non-linear bi-level programming approach.* Int. Trans. Opl. Res. Vol. 2 p 149-164, 1995.

[44] Cordeau, J.F., Soumis, F., Desrosiers, J. *A Benders Decomposition Approach for the Locomotive and Car Assignment Problem*. Transportation Science 34: 133 - 149, 2000.

[45] Cortés, C. E., Jara-Díaz, S., Tirachini, A. *Integrating short turning and deadheading in the optimization of transit services*. Transportation Research Part A 45: 419 - 434, 2011.

[46] Chakroborty, P. *Genetic algorithms for optimal urban transit network design*. Comput. Aided Civ. Infrastruct. Eng. 18(3): 184 - 200, 2003.

[47] Chen ,X., Hellinga, B., Chang, C., Fu, L.*Optimization of Headways for Bus Rapid Transit System with Stop-Skipping Control*. TRB 2012 Annual Meeting.

[48] Chien, S., Yang, Z., Hou, E. *Genetic algorithm approach for transit route planning and design*. J. Transp. Eng. 127(3): 200207, 2001.

[49] Chiraphadhanakul, V., Barnhart, C. *Incremental Bus Service Design: Combining Limited-Stop and Local Bus Services*. Public Transport 5 (1-2): 53-78, 2013.

[50] Chriqui, C., Robillard, P. *Common Bus Lines*. Transportation Science 9(1): 115-121, (1975).

[51] Daduna, J. R., Paixao, J.M.P. *Vehicle scheduling for public mass transit and overview (Daduna J.R., Branco I. and Paixao J.M.P., Eds)*. Computer-Aided Transit Scheduling. Springer-Verlag, Berlin, 1995.

[52] de Camargo, R.S., de Miranda Jr., G.*Addressing congestion on single allocation hub-and-spoke networks*. Pesquisa Operacional 32(3): 465-496, 2012. ISSN 1678-5142.

[53] de Cea, J., Fernández, E. *Transit Assignment for Congested Public Transport Systems: An Equilibrium Model*. Transportation Science 27(2): 133-147, 1993.

[54] DeNegre, S.T., Ralphs ,T.K. *A branch and cut algorithm for integer bilevel linear programs. Operations research and cyber-infrastructure*. Operations Research/Computer Science Interfaces Series 47(2), Part 1, 65 - 78, 2009.

[55] Desaulniers, G., Hickman, M. D. *Public Transit, chapter 2*. Number 14 in Handbooks in Operations Research and Management Science, 69-127. Holland North, 2007.

[56] Desrosiers, J., Dumas, Y., Solomon, M., Soumis, F. *Time constrained routing and scheduling. (M.O. Ball, T.L. Magnanti, C.L. Monma, G.L. Nemhauser, Eds)*. Network Routing Volume 8 of Handbooks in Operations Research and Management Science, Elsevier Science B.V. 35-39 , 1995.

[57] Dial, R.B. *Transit Pathfinder Algorithm*. Highway Research Record 205: 67-85, 1967.

[58] Dijkstra, E.W. *A Note on Two Problems in Connection with Graphs*. Numeriche Mathematik, Vol. 1, P 269-271, 1959.

[59] Dua, V., Pistikopoulos, E.N. *An algorithm for the solution of multiparametric mixed integer linear programming problems*. Ann. Oper. Res. 99: 123 - 139, 2000.

[60] Dubois, D., Bell, G., and Llibre, M. *A Set of Methods in Transportation Network Synthesis and Analysis*, Journal of Operations Research Society, Vol. 30: 797-808, 1979.

[61] Eberlein, X.-J., Wilson, N. H. M., Barnhart, C., Bernstein D.*The real-time deadheading problem in transit operations control*. Transportation Research Board 32(2): 77-100, 1998.

[62] Ercolano, J. *Limited-stop bus operations: An evaluation*. Transportation Research Record 994: 24-29, 1984.

[63] Erlenkotter, D. *A dual-based procedure for uncapacitated facility location*. Operation Research 26: 992 - 1009, 1978.

[64] Escudero, L.F., Muñoz S. *An approach for solving a modification of the extended rapid transit network design problem*, Top 17: 320-334, 2009.

[65] Eppstein, D. *Finding the K shortest paths*. In: Proc. 35th IEEE Symp. Foundations of Computer Science (FOCS94): 154 - 165, 1994.

[66] Eppstein, D. *Finding the K shortest paths*. SIAM Journal Computing 28 (2): 652 - 673, 1998.

[67] Faísca, N. P., Dua, V., Rustem, B., Saraiva, P. M., Pistikopoulos, E. N. *Parametric global optimisation for bilevel programming*. J Glob Optim 38: 609 - 623, 2007.

[68] Fan, W., Machemehl, R.B. *Optimal Transit Route Network Design Problem: Algorithms, Implementations and Numerical Results*. Research SWUTC/04/167244-1. Southwest Region University Transportation Center for Transportation Research University of Texas at Austin. May 2004.

[69] Fan, W., Machemehl, R.B. *Optimal Transit Route Network Design Problem with Variable Transit Demand: Genetic Algorithm Approach*. Journal of Transportation Engineering, 132(1): 40 - 51, 2006.

[70] Fan, W., Machemehl, R.B. *Tabu Search Strategies for the Public Transportation Network Optimizations with Variable Transit Demand*. Computer-Aided Civil and Infrastructure Engineering 23: 502 - 520, 2008.

[71] Fernández, E., de Cea, J., Norambuena, M.I. *Una metodología para el diseño topológico de sistemas de transporte público urbano de pasajeros*. In: Proceedings of the XI Congreso Chileno de Ingeniería de Transporte, Santiago de Chile, November 2003.

[72] Fernández, J., de Cea, J., Malbran, H.R. *Demand responsive urban public transport system design: Methodology and application*. Transportation Reserach Part A 42: 951-972, 2008.

[73] Ferrari, P. *A model of urban transport management*. Transportation Research Part B 33: 43 - 61, 1999.

[74] Florian, M. G., Guerin, G., Bushel, G. *The Engine Scheduling Problem in a Railway Network*. Informs Journal 14: 121-138, 1976.

[75] Floyd, R.W. *Algorithm 97: Shortest Path*. Comm. ACM 5(6): 345, 1962.

[76] Fortuny-Amat, J. &, McCarl, B. *A representation and economic interpretation of a two-level programming problem*. J Oper Res Soc 32: 783 - 792, 1981.

[77] Freyss, M., Giesen, R., Muñoz, J.C. *Continuous Approximation for Skip-Stop Operation in Rail Transit*. Procedia - Social and Behavioral Sciences 80: 186 - 210, 2013.

[78] Fu, L., Liu, Q, Calamai, P. *A Real-Time Optimization Model for Dynamic Scheduling of Transit Operations*. 82nd Annual Meeting of the Transportation Research Board, Washington, DC, 2003.

[79] Furness, K.P. *Time function iteration, Traffic Engineering and Control*. Traffic Engineering and Control 7: 458-460, 1965.

[80] Furth, P. G. *Zonal Route Design for Transit Corridors*. Transportation Science 20 (1): 1-12, 1986.

[81] Gabriel, S.A., Shim, Y., Conejo, A.J., de la Torre, S., García-Bertrand, R. *A Benders decomposition method for discretely constrained mathematical programs with equilibrium constraints*. Journal of the Operational Research Society 61: 1404 - 1419, 2010.

[82] Gallo, M., Montella, B., Di Acierno, L. *The transit network design problem with elastic demand and internalisation of external costs: An application to rail frequency optimisation*. Transportation Research Part C 19: 1276 - 1305, 2011.

[83] Geoffrion, A. M. *Generalized Benders Decomposition*. Journal of Optimization Theory and Applications 10(4): 237-260, 1972.

[84] Geoffrion, A. M., Graves, G. *Multicommodity Distribution System Design by Benders Decomposition*. Management Science 5: 822 - 844, 1974.

[85] Goossens, J.W., van Hoesel, S., Kroon, L. *On solving multi-type railway line planning problems*. European Journal of Operational Research 168: 403 - 424, 2006.

[86] Grossmann I.E. & Floudas C.A. *Active constraint strategy for flexibility analysis in chemical processes*. Comput. Chem. Eng. 11: 675 - 693, 1987.

[87] Grover, F. *Improved Linear Integer Programming Formulations of Nonlinear Integer Problems*. Management Science 22(4): 455-460, 1975.

[88] Hadjiconstantinou, E. & Christofides, N. *An efficient implementation of an algorithm for finding k shortest simple paths*. Networks 34: 88 - 101, 1999.

[89] Han, J., Lee, S., Jonghyung, S., Kim, J. *Meta-heuristic algorithms for a transit route design*. Advanced OR and AI Methods in Transportation, 2005.

[90] Hasselström, D. *Public transportation planning: A mathematical approach.*, PhD dissertation, Univ. of Gothenburg, Gothenburg, Sweden, 1981.

[91] Haupt, T., Näokel, K., Reiter, U. *Schedule based dynamic transit assignment. In Schedule-based modeling of transportaion networks.* N.H.M. Wilson, A. Nuzzolo Ed. Springer, 2009.

[92] Hoffman, W., Pavley, R. *A method for the solution of the nth best path problem*. Journal of the Association for Computing Machinery (ACM) 6: 506 - 514, 1959.

[93] Hu, J., Shi, X., Song, J., Xu, Y. *Optimal design for urban mass transit network based on evolutionary algorithm*. Lecture notes in computer science 3611, L. Wang, K. Chen, and Y. S. Ong, eds., Springer, Berlin-Heidelberg, 2005.

[94] Ignizio, J.P. *Solving Large-Scale Problem: A venture into a New Dimension*. Operation Research 3: 217 - 225, 1980.

[95] INRO Consultants, EMME Users Manual. Software Release 9.1, 2010.

[96] Israeli, Y., Ceder, A. *Designing transit routes at the network level*. In: Proceedings of the First Vehicle Navigation and Information Systems Conference. IEEE Vehicular Technology Society: 310 - 316, 1989.

[97] Israeli, Y. *Transit route and scheduling design at the network level*. Doctoral dissertation, Technion Israel Institute of Technology, 1992.

[98] Israeli, Y., Ceder, A. *Transit route design using scheduling and multiobjective programming techniques*. In: Daduna, J.R., Branco, I., Piaxão, J. (Eds.), Computer-Aided Transit Scheduling. Lecture Notes in Economics andMathematical Systems 430: 56 - 75, 1995.

[99] Jin, W., Chen, S., Jiang, H. *Finding the K shortest paths in a time-schedule network with constraints on arcs*. Computers & Operations Research 40: 2975 - 2982, 2013.

[100] Jordan, W.C., Turnquist, M.A. *Zone scheduling of bus routes to improve service reliability*. Transportation Science 13(3): 242 - 268, 1979.

[101] Katoh, N., Ibaraki, T., Mine H. *An efficient algorithm for k shortest simple paths*. Networks 12: 411 - 427, 1982.

[102] Kurauchi, F., Bell, G.H., Schmcker, J.D. *Capacity Constrained Transit Assignment with Common Lines*, Journal of Mathematical Modelling and Algorithms 2:309-327, 2003.

[103] Lam W.H.K., Gao Z.Y., Chan K.S. and Yang H. *A Stochastic User Equilibrium Assignment Model for Congested Transit Networks*. Transportation Research - Part B 33: 351-368, 1999.

[104] Lam W.H.K., Zhou J. and Sheng Z.H. *A Capacity Restraint Transit Assignment with Elastic Line Frequency*. Transportation Research - Part B 36: 919-938, 2002.

[105] Lampkin W., Saalmans P.D. *The Design of Routes, Service Frequencies, and Schedules for a Municipal Bus Undertaking: A Case Study*. Operational Research Quarterly 18(4): 375-397, 1967.

[106] Laporte G., Mesa, J. A., Ortega, F.A. *Locating stations on rapid transit lines*. Computers and Operations Research 29: 741-759, 2002.

[107] Laporte, G., Mesa, J.A., Ortega, F.A. *Maximizing trip coverage in the location of a single rapid transit alignment*. Annals of Operations Research 136: 49-63, 2005.

[108] Laporte G., Marín, A., Mesa, J.A., Ortega, F.A. *An integrated methodology for the rapid transit network design problem*. In: Garaets F, Kroon L, Schöebel A., Wagner D., Zarioliagis C. (eds). Algorithmic methods for railway optimization. Lectures notes in computer science 4359: 187 - 199, 2006.

[109] Laporte, G., Mesa, J.A., Perea, F. *A game theoretic framework for the robust railway transit network design problem*. Transportation Research Part B 44(4): 447 - 459, 2010.

[110] Larraín, H. *Asignación en un corredor de transporte público, considerando congestión en los tiempos de viaje*. Master thesis report, Santiago de Chile, 2006.

[111] Larraín, H., Muñoz, J.C., Giesen, R. *How to design express services on a Bus Transit Network*. Webseminar on BRT Center for excellence, Santiago de Chile, September 23, 2013.

[112] Last, A., Leak, S.E. *Transept: A Bus Model*. Traffic Engineering and Control 18(1): 14-20, 1976.

[113] Lawler, E.L. *Combinatorial Optimization: Networks and Matroids*. Holt, Rinehart & Winston, New York, 1976.

[114] LeBlanc, L.J. *Transit System Network Design*. Transp. Res. B. 22B(5): 383-390, 1988.

[115] le Clercq, F. *A Public Transport Assignment Method*. Traffic Engineering and Control 14(2): 91-96, 1972.

[116] Lee, Y-J., Vuchic, V.R. *Transit Network Design with Variable Demand*. J. Transp. Eng. 131: 1-10, 2005.

[117] Leiva, C., Muñoz, J.C., Giesen, R., Larrain, H. *Design of limited-stop services for an urban bus corridor with capacity constraints*.Transportation Research Part B 44(10): 1186-1201, 2010.

[118] Li,Y., Rousseau, J. M., Gendreau, M. *Real-Time Scheduling on A Transit Bus Route: A 0-1 Stochastic Programming Model*. Proceedings of the Thirty-Third Annual Meeting, Transportation Research Forum: 157-166, 1991.

[119] Liu, Z., Yan, Y., Qu, X., Zhang, Y. *Bus stop-skipping scheme with random travel time*. Transportation Research Part C 35: 46 - 56, 2013.

[120] Löbel, A. *Solving large scale multiple-depot vehicle scheduling problems. (N. H. M. Wilson, ed)*.Computer-Aided Transit Scheduling. Notes in Economics and Mathematical Systems 471, Springer-Verlag 192-220, Berlin, 1999.

[121] Lohatepanont, M., Barnhart, C., *Airline Schedule Planning: Integrated Models and Algorithms for Schedule Design and Fleet Assignment*. Transportation Science 38(1): 19-32, 2004.

[122] López, F., Codina, E., Marín, A. *A model for the network planning and design of suburban railway systems*. CASPT12. Santiago de Chile, July 2012.

[123] López, F., Codina, E., Marín, A. *Integrating network design and line planning in rapid transit systems*. Submitted to Public Transpor Journal on December 2012. Ed. Springer Verlag. PUTR-D-12-00051.

[124] López, F., Codina, E., Marín, A, Cominetti, R. *Enhanced Benders Decomposition applied to a model of network topology design and frequency setting for rapid transit systems*. XXVI European Conference on Operational Research (EURO). Rome, Italy, July 2013.

[125] Magnanti, T., Wong, R. *Accelerating Benders decomposition: Algorithmic enhancement and model selection criteria*. Operations Research 29(3): 464-484, 1981.

[126] Magnanti, T.L., Wong, R.T. *Network Design and Transportation Planning: Models and Algorithms*. Transportation Science 18: 1 - 55, 1984.

[127] Mandl, C. E. *Evaluation and Optimization of Urban Public Transportation Networks*. European Journal of Operational Reserach 5: 396 - 404, 1979.

[128] Marín, A. *An extension to rapid transit network design problem*. Top 15: 231 - 241, 2007.

[129] Marín, A., Jaramillo, P. *Urban rapid transit network capacity expansion*. European Journal of Operational Research 191: 45 - 60, 2008.

[130] Marín, A., García-Rodenas, R., *Location of infrastructure in urban railway networks*. Computers & Operations Research 36: 1461 - 1477, 2009.

[131] Marín, A., Jaramillo, P. *Urban rapid transit network design: accelerated Benders decomposition*. Annals of Operational Research 169: 35 - 53, 2009.

[132] Marín, A., Mesa, J.A., Perea, F., *Integrating Robust Railway Network Design and Line Planning under Failures*. In: Ahuja, R.K. *et al*. Eds.: Robust and Online Large-Scale Optimization. Lecture Notes in Comput. Sci. 5868: 273-292, 2009.

[133] Marwah, B. R., Farokh, S., Umrigar, S., Patnaik, S. B. *Optimal design of bus routes and frequencies for Ahmedabad*. Transportation Research Record 994: 41 - 47, 1984.

[134] Mauttone, A., Urquhart, M. *A route set construction algorithm for the transit network design problem*. Computers & Operations Research, 36(8): 2440 - 2449, 2009.

[135] Mauttone, A. *Models and Algorithms for the optimal design of bus routes in public transportation systems*. Thesis Report, Montevideo, March 2011.

[136] Mc Daniel, D., Devine, M. *A modified bender's partitioning algorithm for mixed integer programming*. Management Science 24(3): 312-319, 1977.

[137] McFadden, D. *Conditional logit analysis of qualitative choice behavior*. In: Zaremmbka, P. (Ed.), Frontiers in Econometrics. Academic Press, New York, 1973.

[138] Mercier, A., Cordeau, J.F., Soumis, F. *A computational study of Benders decomposition for the integrated aircraft routing and crew scheduling problem*. Computers & Operations Research 32: 1451 - 1476, 2005.

[139] Mesquita, M., Paixao, J.M.P. *Exact algorithms for the multi-depot vehicle scheduling problem based on multicommodity network flow type formulations. (N. H. M. Wilson, ed)*. Computer-Aided Transit Scheduling. Springer-Verlag 221-243, Berlin 1999.

[140] Ngamchai, S., Lovell, D. J. *Optimal Time Transfer in Bus Transit Route Network Design Using a Genetic Algorithm*. J. Transp. Eng. 129: 510 - 521, 2003.

[141] Nguyen, S., Pallottino, S. *Equilibrium Traffic Assignment for Large Scale Transit Networks*. European Journal of Operational Research 37: 176-186, 1988.

[142] Noriega, Y. M., Florian, M. *L'optimisation des frequences d'un rseau de transport en commun moyennement congestion.* INFOR vol 41, no 2. 129-153, 2003.

[143] Ocaña, R. V. O. *Subway costs in developed countries: An essay on the analysis and the comparison of subway systems in Caracas, Hong Kong, Mexico, Santiago and Sao Paulo*. Economa XIV 4, 89-120, 1989.

[144] Odoni, A.R., Rousseau, J.M., Wilson, N.H.M. *Models in Urban and Air Transportation. In: S.M. Pollock, M.H. Rothkopf, and A. Barnett (eds.)*. Operations Research and the Public Sector, Handbooks in Operations Research and Management Science 6, North-Holland, Amsterdam, 107-150, 1994.

[145] Oppenheim, N. *Urban Travel Demand Modeling: From Individual Choices to General Equilibrium. Chapter 4 - Combined Travel Demand Modeling under uncongested conditions*. Wiley & Sons, Incorporated, January 1995.

[146] Pacheco, J., Alvarez, A., Casado, S., Gonzalez-Velarde, J. L. *A tabu search approach to an urban transport problem in northern Spain*. Computers & Operations Research 36: 967 - 979, 2009.

[147] Papadakos, N. *Practical enhancements to the Magnanti-Wong method*. Operational Research Letters 36: 444 - 449, 2008.

[148] Papadakos, N. *Integrated airline scheduling*. Computers & Operations Research 36: 176 - 195, 2009.

[149] Papola, N., Filippi, F., Gentile, G., Meschini, L. *Schedule based transit assignment: new dynamic equilibrium model with vehicle capacity constraints. In Schedule-based modeling of transportation networks.* N.H.M. Wilson, A. Nuzzolo Ed. Springer, 2009.

[150] Pattnaik, S. B., Mohan, S., Tom, V. M. *Urban bus transit route network design using genetic algorithm*. J. Transp. Eng. 124: 368 - 375, 1998.

[151] Petersen Clifford, C. *A Note on Transforming the Product of Variables to Linear Form in Linear Programs*. Working Paper, Purdue University, 1971.

[152] Pita, J., Barnhart, C., Antunes, A. P. *Integrated Flight Scheduling and Fleet Assignment Under Airport Congestion*. Transportation Science, Articles in Advance 1-16, 2012. DOI: 10.1287/trsc.1120.0442.

[153] Quak, C.B. *Bus line planning. A passenger-oriented approach of the construction of a global line network and an efficient timetable*. Delft University, Netherlands, 2003.

[154] Rea, J.C., *Designing Urban Transit Systems: An Approach to the Rout-Technology Selection Problem*. PB 204881, University of Washington, Seattle, W A, 1971.

[155] Richardson, R. *An Optimization Approach to Routing Aircraft*. Transportation Science 10: 52-71, 1976.

[156] Saharidis, G. K., Ierapetritou, M. G. *Resolution method for mixed integer bi-level linear problems based on decomposition technique*. Journal of Global Optimization 44(1): 29 - 51, 2009.

[157] Saharidis, G. K. D., Minoux, M., Ierapetritou, M. G. *Accelerating Benders method using covering cut bundle generation*. Intl. Trans. in Op. Res. 17: 221 - 237, 2010.

[158] Saharidis, G.K.D., Conejo, A.J., Kozanidis, G.*Exact Solution Methodologies for Linear and (Mixed) Integer Bilevel Programming*. E.-G. Talbi (Ed.): Metaheuristics for Bi-level Optimization, SCI 482: 221 - 245, 2013.

[159] Schnabel, W., Löhse, D.*Grundlagen der Strasssen-Verkehrstechnik und der Verkehrsplanung, Band 2, neue bearb*. Aufl.1997. Verlag fr Bauwesen Gmbh, Berlin, 1997.

[160] Sherali, H. D., Lunday, B. J. *On generating maximal nondominated Benders cuts*. Ann Oper Res 210(1): 57-72. Springer Verlag, 2011.

[161] Shi, C., Lu, J., Zhang, G. *An extended KuhnTucker approach for linear bi-level programming*. Appl. Math. Comput. 162: 51 - 63, 2005.

[162] Shier, D.R. *Iterative methods for determining the K shortest paths in a network*. Networks 6: 205 - 229, 1976.

[163] Shier, D.R. *On algorithms for finding the K-shortest paths in a network*. Networks 9: 195 - 214, 1979.

[164] Silman, L.A., Barzily, Z., Passy, U. *Planning the Route System for Urban Buses*. Computers and Operations Research 1: 210-211, 1974.

[165] Soehodho, S., Koshi, M. *Design of Public Transit Network in Urban Area with Elastic Demand*. Journal of Advanced Transportation 33(3): 335-369, 1999.

[166] Spiess, H. *On Optimal Route Choice Strategies in Transit Networks*. Centre de Recherche sur les Transports, 285, Universit de Montral, 1984.

[167] Spiess, H., Florian, M. *Optimal Strategies: A New Assignment Model for Transit Networks*. Transportation Research - Part B 23(2): 83-102, 1989.

[168] Suh, W., Chon, K.-S., Rhee, S. M. *Effect of Skip-Stop Policy on A Korean Subway System*. Transportation Research Record 1793: 33-39, 2002.

[169] Sun, A., Hickman, M. *The Real-Time Stop-Skipping Problem*. Journal of Intelligent Transportation Systems 9 (2): 91-109, 2005.

[170] Sun, C., Wei, Z., Yuanqing, W.*Scheduling Combination and Headway Optimization of Bus Rapid Transit*. Journal of transportation systems engineering and information technology 8(5): 61-67, 2008.

[171] Szeto, W.Y., Wub, Y. *A simultaneous bus route design and frequency setting problem for Tin Shui Wai, Hong Kong*. European Journal of Operational Research 209: 141 - 155, 2011.

[172] Tom, V. M., Mohan, S. *Transit Route Network Design Using Frequency Coded Genetic Algorithm*. J. Transp. Eng. 129: 186 - 195, 2003.

[173] Ulusoy, Y., Chien, S., Wei, C. *Optimal All-Stop, Short-Turn, and Express Transit Services Under Heterogeneous Demand*. Transportation Research Record: Journal of the Transportation Research Board 2197(1):8-18, DOI 10.3141/2197-02, 2010.

[174] van der Zijpp, N.J., Catalano, S.F. *Path enumeration by finding the constrained K-shortest paths*. Transportation Research Part B 39: 545 - 563, 2005.

[175] van Nes, R., Hamerslag, R., Immer, B.H. *The Design of Public Transport Networks*, Transportation Research Record, No. 1202, Transportation Research Board, Washington, D.C., 1202: 74-83, 1988.

[176] van Oudheusden, D. L., Ranjithan, S., Singh, K. N. *The design of bus route systemsAn interactive location allocation approach*. Transportation 14(3): 253 - 270, 1987.

[177] von Stackelberg, H. *The theory of the Market Economy*. Oxford University Press, Oxford, UK, 1952.

[178] Vuchic, V. R. *Skip-Stop Operation as a Method for Transit Speed Increase*. Traffic Quarterly 27: 307-327, 1973.

[179] Walker, C., S. J., and Ryan, D. *Simultaneous disruption recovery of a train timetable and crew roster in real time*. Comput. Oper. Res 32(8): 2077 - 2094, 2005.

[180] Wan, Q. K., Hong, K. Lo. *A Mixed Integer Formulation for Multiple-Route Transit Network Design*. Journal of Mathematical Modelling and Algorithms 2: 299 - 308, 2003.

[181] Warshall, S. *A theorem on boolean matrices*. J. ACM 9(1): 11-12, 1962.

[182] Wen, U.P., Yang, Y.H. *Algorithms for solving the mixed integer two level linear programming problem*. Comput. Op. Res. 17: 133 - 142, 1990.

[183] Williams, H.C.W.L. *On the formulation of travel demand models and economic evaluation measures of user benefit*. Environment and Planning A 9(3): 285 - 344, 1977.

[184] Wilson, A.G. *A statistical theory of spatial distribution models*. Transportation Research 1: 253 - 269, 1967.

[185] Wilson, N., Macchi, R., Fellows, R., Deckoff, A. *Improving Service on the MBTA Green Line Through Better Operations Control*. Transportation Research Record 1361: 296-304, 1992.

[186] Wu, J.H., Florian, M., *A Simplicial Decomposition Method for the Transit Equilibrium Assignment Problem*. Annals of Operations Research 44: 245-260, 1993.

[187] Wu, J.H., Florian M. and Marcotte P. *Transit Equilibrium Assignment: A Model and Solution Algorithms*. Transportation Science 28(3): 193-203, 1994.

[188] Xu, P., Wang, L. *An Exact Algorithm for the Bilevel Mixed Integer Linear Programming Problem under Three Simplifying Assumptions*. To appear in Computers & Operations Research.

[189] Yen, J.Y. *Finding the K Shortest Loopless Paths in a Network*. Management Science 17(11): 712-716, 1971.

[190] Yoo, G-S., Kim, D-K, Chon, K.S. *Frequency Design in Urban Transit Networks with Variable Demand: Model and Algorithm*. KSCE Journal of Civil Engineering 14(3): 403-411, 2010.

[191] Zhao, F. *Optimization of Transit Network to minimize transfers*. Report BD-015-02. Research Center Florida Department of Transportation, 2003.

[192] Zhao, F. *Large-Scale Transit Network Optimization by Minimizing User Cost and Transfers*. Journal of Public Transportation 9: 2, 2006 .

[193] Zhao, F., Zeng, X. *Optimization of transit route network, vehicle headways and timetables for large-scale transit networks.* European Journal of Operational Research, 186: 841-855, 2008.

Appendix

This appendix provides complementary information for some of the research report chapters.

A quick note about linearizing the product of one binary and one continuous variable

This section presents a summary of section 2 of Glover's paper [87]. In that section, the author shows an improved technique for linearizing the product of two decision variables where one of them is binary and the other one continuous. That linearization technique is an improvement on Petersen's work [151].

The problem is stated as follows. Given the two decision variables $x \in \{0, 1\}$ and $w \in R^+$ with the following relationship:

$$x = 0 \rightarrow \ L_0 \leq w \leq U_0 \tag{1}$$

$$x = 1 \rightarrow \ L_1 \leq w \leq U_1 \tag{2}$$

where L_0, U_0, L_1, U_1, are not constant and whose values depend on other decision variables and model constraints. Nonetheless, its respective upper and lower bounds $\overline{U}_0$, $\overline{U}_1$, $\overline{L}_0$, $\overline{L}_1$, $\underline{U}_0$, $\underline{U}_1$, $\underline{L}_0$, $\underline{L}_1$ are known. Therefore:

$$\underline{L}_0 \ \leq \ L_0 \leq \overline{L}_0 \tag{3}$$

$$\underline{U}_0 \ \leq \ U_0 \leq \overline{U}_0 \tag{4}$$

$$\underline{L}_1 \ \leq \ L_1 \leq \overline{L}_1 \tag{5}$$

$$\underline{U}_1 \ \leq \ U_1 \leq \overline{U}_1 \tag{6}$$

Grover proves that it is possible to obtain a set of linear constraints which satisfies the logical conditions stated in (1)-(2) and the bounds shown in (3) - (6) . These constraints are set by applying the following equations:

$$U_0 + (\overline{U}_1 - \underline{U}_0) \cdot x \geq w \geq L_0 + (\underline{L}_1 - \overline{L}_0) \cdot x \tag{7}$$

$$U_1 + (\overline{U}_0 - \underline{U}_1) \cdot (1 - x) \geq w \geq L_1 + (\underline{L}_0 - \overline{L}_1) \cdot (1 - x) \tag{8}$$

A particular interesting case concerning our research arises when $L_0 = U_0 = \underline{L}_0 = \overline{L}_0 = \underline{U}_0 = \overline{U}_0 = \underline{U}_1 = \underline{L}_1 = 0$ and $L_1 = U_1$. In that situation, the set of constraints presented is simplified as follows:

$$\overline{LU}_1 \cdot x \geq w \geq 0 \tag{9}$$

$$LU_1 \geq w \geq LU_1 - \overline{LU}_1 \cdot (1 - x) \tag{10}$$

where LU_1 stands for the mathematical expression that decision variable w will take when $x = 1$ and constraint (9) defines the lower and upper bounds of w, $\overline{LU}_1$ and $\underline{LU}_1$, respectively.

Building network flow

This section explains how the routing graph G_{TP} must be manipulated in order to obtain the network flow G^f_{TP}, on which the passenger flow balance constraints (2.50)-(2.56) are based.

The G^r_{TP} consists of public transportation nodes and stretches (N_{TP} and A_{TP} sets, respectively). The former is split into the nodes in which some service is performed (N^S_{TP}) and into the ones which are simply passing points (N^P_{TP}). All of the mentioned sets are compound sets since they are defined for each line $l \in L$, except set N^P_{TP} which is only defined for $l \in L^E$.

Knowing beforehand the contents of these sets, the G^f_{TP} can be obtained as follows. Firstly, the incoming and outgoing passenger flow nodes (N^{S+}_{TP} and N^{S-}_{TP}, respectively) are computed in two steps according to equations (11)-(12) and (14)-(15) where Γ^l_i stands for the set of nodes adjacent to i. Moreover, if there is any $i \in N^P_{TP}$, we also compute the in-outgoing flow nodes (N^{P+-}_{TP}) in two steps by means of equations (13) and (16).

$$N^{S+}_{TP}(i,l) = \left\{ij+,\ \forall j \in \Gamma^l_i\right\},\quad \forall l \in L,\ i \in N^S_{TP}(l) \tag{11}$$

$$N^{S-}_{TP}(i,l) = \left\{ij-,\ \forall j \in \Gamma^l_i\right\},\quad \forall l \in L,\ i \in N^S_{TP}(l) \tag{12}$$

$$N^{P+-}_{TP}(i,l) = \{kij,\ \forall (i,k),(k,j) \in A^r_{TP}(l) \mid i \neq j\},\ \forall l \in L^E,\ k \in N^P_{TP}(l) \tag{13}$$

$$N^{S+}_{TP}(l) = \bigcup_{i \in N^S_{TP}(l)} N^{S+}_{TP}(i,l),\quad \forall l \in L \tag{14}$$

$$N^{S-}_{TP}(l) = \bigcup_{i \in N^S_{TP}(l)} N^{S-}_{TP}(i,l),\quad \forall l \in L \tag{15}$$

$$N^{P+-}_{TP}(l) = \bigcup_{i \in N^P_{TP}(l)} N^{P+-}_{TP}(i,l),\quad \forall l \in L^E \tag{16}$$

Having computed the network flow nodes, we proceed to obtain the links. Firstly, we compute the boarding and alighting links (A_a and A_y sets, respectively) by means of equations (17) - (18). Secondly, we obtain the waiting in-vehicle links (A_x) by using these computed links and applying equation (19).

$$A_a(l) = \left\{(i,j),\ \forall i \in N^S_{TP}(l),\ j \in N^{S-}_{TP}(i,l)\right\},\quad \forall l \in L \tag{17}$$

$$A_y(l) = \left\{(j,i),\ \forall i \in N^S_{TP}(l),\ j \in N^{S+}_{TP}(i,l)\right\},\quad \forall l \in L \tag{18}$$

$$A_x(l) = \left\{(m,n),\ \forall i \in N^S_{TP}(l),\ m \in N^{S-}_{TP}(i,l),\ n \in N^{S+}_{TP}(i,l) \mid i,m,n \in \Pi^l_x\right\},\quad \forall l \in L \tag{19}$$

Π^l_x is defined in (20) and denotes the set of the threesome i, m, n, which is valid.

$$\Pi_x^l = \{(m,i) \in A_y(l),\ (i,n) \in A_a(l),\ \text{ord}(m) \neq \text{ord}(n)\} \tag{20}$$

The set of in-station exchange flows (A_{xya}) is then computed by means of equation (21) as the union of each exchange flow link type.

$$A_{xya}(l) = A_y(l) \cup A_a(l) \cup A_x(l), \quad \forall l \in L \tag{21}$$

Finally, the in-vehicle links A_{inv} are computed according to equations (22) - (26).

$$A_{inv}(l) = A_{inv}^1(l) \cup A_{inv}^2(l) \cup A_{inv}^3(l) \cup A_{inv}^4(l), \quad \forall l \in L \tag{22}$$

Each in-vehicle link subset (A_{inv}^i) denotes a partition of A_{inv} in which the extreme nodes of the links correspond to one of these situations: nodes i, j are service nodes (A_{inv}^1); nodes i, j are passing points (A_{inv}^2); node i is a service node; node j is a passing point (A_{inv}^3); node j is a service node; and finally, node i is a passing point (A_{inv}^4).

$$A_{inv}^1(l) = \left\{(ij-, ji+),\ \forall (i,j) \in A_{TP}^r(l) \mid i,j \in N_{TP}^S(l)\right\}, \quad \forall l \in L \tag{23}$$

$$A_{inv}^2(l) = \left\{(imj, jin),\ \forall (i,j) \in A_{TP}^r(l),\ m \in \Gamma_i^{l+},\ n \in \Gamma_i^{l-} \mid i,j \in N_{TP}^P(l)\right\}, \quad \forall l \in L^E \tag{24}$$

$$A_{inv}^3(l) = \left\{(ij-, jin),\ \forall (i,j) \in A_{TP}^r(l),\ n \in \Gamma_i^{l-} \mid i \in N_{TP}^S(l),\ j \in N_{TP}^P(l)\right\}, \quad \forall l \in L^E \tag{25}$$

$$A_{inv}^4(l) = \left\{(imj, ji+),\ \forall (i,j) \in A_{TP}^r(l),\ m \in \Gamma_i^{l+} \mid i \in N_{TP}^P(l),\ j \in N_{TP}^S(l)\right\}, \quad \forall l \in L^E \tag{26}$$

Γ_i^{l-} and Γ_i^{l+} denote the set of nodes which emerges from and are incident to i, respectively.

The network flow size, i.e, the total number of nodes and links in G_{TP}^f, are also given by equations (27) - (28). Moreover, their breakdown by role and line type can be obtained by applying the formulas shown in Tables 1 and 2.

Type	L^N	L_R^E	L_C^E
in-vehicle	$\lvert A_{TP}^N \rvert$	$\sum_{l \in L^E} \lvert A_{TP}(l) \rvert$	
waiting	$2 \cdot \sum_{i \in N_{TP}^N} \left(\frac{\lvert \Gamma_i \rvert}{2}\right)$	$2 \cdot \sum_{l \in L^E} \left(\lvert N_{TP}^{E,S}(l) \rvert - 2\right)$	$2 \cdot \sum_{l \in L^E} \lvert N_{TP}^{E,S}(l) \rvert$
not waiting		-	
alighting boarding	$\sum_{i \in N_{TP}^N} \lvert \Gamma_i \rvert$	$2 \cdot \sum_{l \in L^E} \left(\lvert N_{TP}^{E,S}(l) \rvert - 1\right)$	$2 \cdot \sum_{l \in L^E} \lvert N_{TP}^{E,S}(l) \rvert$

Table 1: Number of links according to their type

$$|A| = 2 \cdot \left[2 \cdot \sum_{i \in N_{TP}^N} \left(\frac{|\Gamma_i|}{2}\right) + |\Gamma_i|\right] + |A_{TP}^N| + |A_{TP}(l)| + 4 \cdot \left[\sum_{l \in L^E} 1.5 \cdot |N_{TP}^{E,S}(l)| - 2 \cdot |L_R^E|\right] \tag{27}$$

Type	L^N	L^E_R	L^E_C
Ground	$\lvert N^N_{TP}\rvert$	$\sum_{l\in L^E} \lvert N^{E,S}_{TP}(l)\rvert$	
Incoming Outgoing	$\sum_{i\in N^N_{TP}} \lvert\delta_i\rvert$	$2\cdot\sum_{l\in L^E}\Big(\lvert N^{E,S}_{TP}(l)\rvert - 1\Big)$	$2\cdot\sum_{l\in L^E} \lvert N^{E,S}_{TP}(l)\rvert$
In-Outgoing	-	$2\cdot\sum_{l\in L^E} \lvert N^{E,P}_{TP}(l)\rvert$	

Table 2: Number of nodes according to their type

$$|N| = |N^N_{TP}| + \sum_{i\in N^N_{TP}} |\delta_i| + 4\left[\sum_{l\in L^E}\Big(|N^{E,S}_{TP}(l)| + 0.5\cdot|N^{E,P}_{TP}(l)|\Big) - |L^E_R|\right] \tag{28}$$

Column labels L^R_E and L^C_E stand for the set of working lines whose topology is rectilinear and circular, respectively. Additionally, parameter $|\Gamma_i|$ denotes the number of nodes adjacent to i.

Illustrative Example 1

The extended network flow increases substantially the size of the model. However, it is required in order to deal with different service patterns: express services, where vehicles halt at a subset of nodes contained in the line; or local services, in which vehicles halt at every node. In this appendix, we will give an illustrative example which shows the drawbacks of neglecting the in-station movements from the network flow model.

Let us consider the two o-d demand pairs depicted in figure 1, where the pair $(1, 9)$ holds 10000 passengers and the pair $(4, 6)$ contains 100. Notice also that, links in thicker line have two costs denoting the in-vehicle traveling times and walking times, respectively; whereas the ones in thinner line are related to walking times, only. For the sake of simplicity but without loss of generality, let us also neglect infrastructure costs, exploitation costs, resource capacities, and suppose that the time horizon is wide enough so that no line cycle is constraint. Under these assumptions, we allow the model to construct up to two new lines in order to satisfy the o-d demand pairs $(1, 9)$ and $(4, 6)$.

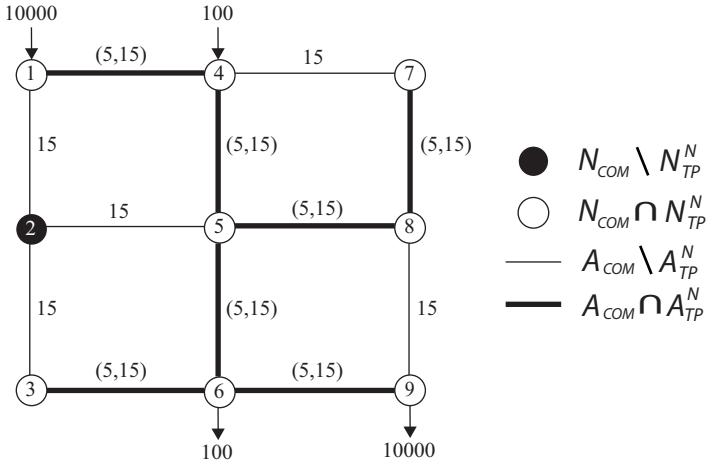


Figure 1: A routing network example.

If we neglect by now the in-station movements, the network flow can be directly obtained from the routing graph of figure 1, where the thicker links are considered as directed in-vehicle links. Notice that, the shortest path of o-d pair $(1, 9)$ is $\mathcal{K}_{(1,9)} = 1-4-5-6-9$ and the one of o-d pair $(4, 6)$ is $\mathcal{K}_{(4,6)} = 4-5-6$. Since $\mathcal{K}_{(4,6)} \subseteq \mathcal{K}_{(1,9)}$, the optimal solution is to construct a single line $L1 : 1-4-5-6-9$ where assigned vehicles perform service in all nodes excepting node 5. Thus, the users objective function takes the following value:

$$z_{pax}^{1} = (5 + 5 + 5 + 5) \times 10000 + (5 + 5) \times 100 = 201000 \tag{29}$$

where the term θ, which weights the value of time, is taken as the unity for the sake of simplicity.

Now let us consider the same example but incorporating the in-station movements as depicted in figure 2.2 in chapter 2, where boarding and in-vehicle waiting unit times (t_a and t_x, respectively) are set to 9 seconds and alighting unit time (t_y) takes the value of 2 seconds. If we include these times in the aforementioned solution, the value of the users objective function ($z_{pax}^{1'}$) will be incremented as follows:

$$\begin{aligned} z_{pax}^{1'} = (20 + \mathbf{0.15} + \mathbf{0.33} + \mathbf{0.15} + \mathbf{0.15}) \times 10000 + \\ (10 + \mathbf{0.15} + \mathbf{0.33}) \times 100 = 201000 + \mathbf{7848} = 208848 \end{aligned} \tag{30}$$

where the highlighted numbers denote the times associated with the in-station movements. This value does not longer correspond to the optimal one since now the optimal network design solution is to construct two

express lines: $L1: 1-4-5-6-9$, where vehicles only halt at nodes 1 and 9, and $L2: 4-5-6$, in which vehicles only performed service in nodes 4 and 6. Its corresponding optimal users objective function (z^2_{pax}) takes the following value:

$$z^2_{pax} = (20 + \mathbf{0.15} + \mathbf{0.33}) \times 10000 + (10 + \mathbf{0.15} + \mathbf{0.33}) \times 100 = \\ = 201000 + \mathbf{4848} = 205848 \tag{31}$$

which is lower than the value of $z^{1'}_{pax} = 208848$. Therefore, we have demonstrated the usefulness of including in-station movements.

Illustrative Example 2

This appendix is devoted to presenting an application example of the circular shortest path algorithm and the Yen's k-shortest path algorithm for circular corridors, explained in chapter 4. To do this, we use the 9-node network depicted in Figure 8.2. For the sake of clarification, we provide here the data needed by the algorithms. Firstly, the used travel times and link construction cost matrices are as follows:

$$T = \begin{pmatrix} - & 0.375 & 0.525 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 \\ 0.375 & - & 0.450 & 0.825 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 \\ 0.525 & 0.450 & - & 0.825 & 0.375 & 0.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.825 & 0.825 & - & 0.600 & 0.525 & 0.675 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.375 & 0.600 & - & 0.375 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 0.525 & 0.375 & - & 0.900 & 0.675 & 0.975 \\ 0.000 & 0.000 & 0.000 & 0.675 & 0.000 & 0.900 & - & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.675 & 0.000 & - & 0.000 \\ 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.975 & 0.000 & 0.000 & - \end{pmatrix}$$

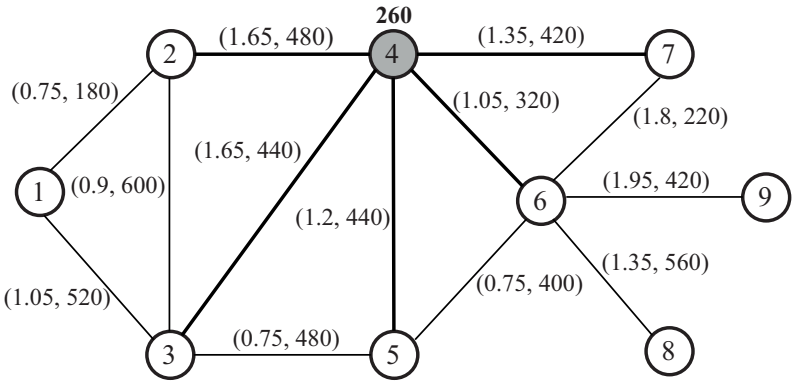
$$C_a = \begin{pmatrix} - & 90 & 260 & 0 & 0 & 0 & 0 & 0 & 0 \\ 90 & - & 300 & 240 & 0 & 0 & 0 & 0 & 0 \\ 260 & 300 & - & 220 & 240 & 0 & 0 & 0 & 0 \\ 0 & 240 & 220 & - & 220 & 160 & 210 & 0 & 0 \\ 0 & 0 & 240 & 220 & - & 200 & 0 & 0 & 0 \\ 0 & 0 & 0 & 160 & 200 & - & 110 & 280 & 210 \\ 0 & 0 & 0 & 210 & 0 & 110 & - & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 280 & 0 & - & 0 \\ 0 & 0 & 0 & 0 & 0 & 210 & 0 & 0 & - \end{pmatrix}$$

where travel times in T are given in minutes, and construction costs in C_a are multiples of a certain currency unit. Moving on, the node construction cost vector is set to $C_n = [280, 320, 270, 260, 300, 420, 440, 330, 250]$. These construction costs are also expressed as multiples of a given currency unit. Finally, the infrastructure budgets are both established to $\bar{c}_{net} = 8000$ and the planning horizons to $\bar{h} = 3$ hours.

Application of the Circular shortest Path Algorithm

The circular shortest path algorithm ($CSPA$) is applied to the 9-node network (mentioned at the beginning of this section) by using a modified version of the in-vehicle travel time matrix ($T^{'}$), where each cell contains the value of t_{ij} plus its inverse cost t_{ji}.

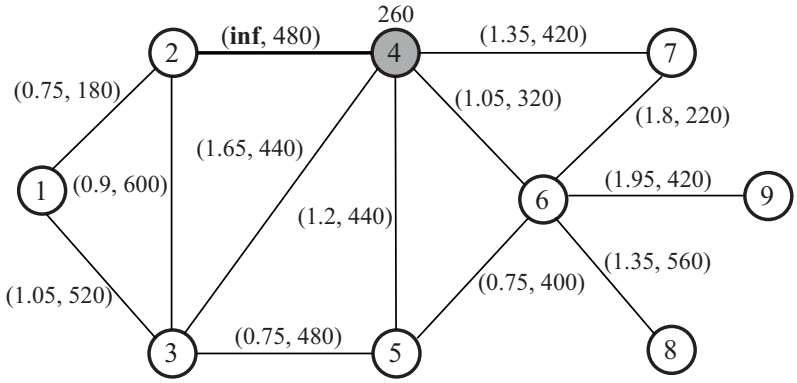
This example is based on finding the shortest circular corridor starting and ending at node 4. As depicted in the first figure, the $CSPA$ begins by finding the neighborhood of this node. Put another way, it picks up all the nodes adjacent to 4 and saves them in list δ_4.



Initialization phase

$N_s = \{4\}$
$\delta_4 = \{2, 3, 5, 6, 7\}$
$S = \{\}$
$R = \{\}$
$Q = \{\}$

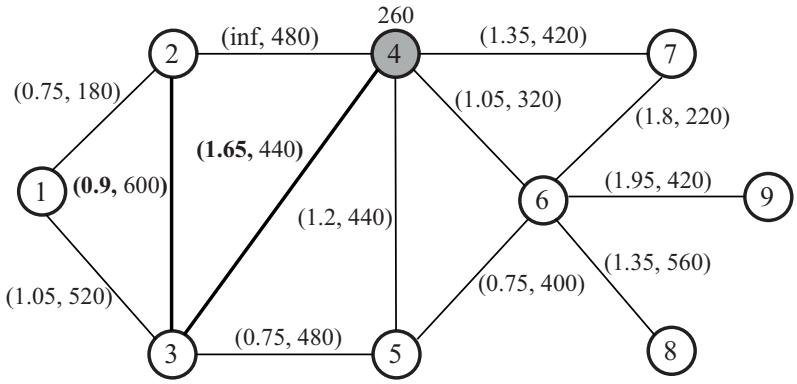
Having obtained δ_4, the iterative part of the $CSPA$ starts. As depicted in the following figure, the first step consists of picking up the first node in δ_4, which is node 2, and disabling its associated links $(2, 4),\ (2, 4)$, which are adjacent to node 4.



Step 1.1

$N_s = \{4\}$
$\delta_4 = \{\mathbf{2}, 3, 5, 6, 7\}$
$S = \{\}$
$R = \{\}$
$Q = \{\}$

The second step involves finding the shortest path from node 4 to node 2. This operation is carried out through an adaptation of Dijkstra's algorithm (its pseudo-code is written down in table 4.4). Having obtained the shortest path, it stores the temporary path S and verifies if it satisfies the first user behavior rule, using $\phi^c_{max} = 1.5$. To show this evaluation, we include a table containing a list of links which are adjacent to node 4 and candidates for the shortest path from 4 to the processing node of δ_4. However, we focus only on the active links, i.e., the links contained in S. In that case, they are $(2, 4)$ and $(3, 4)$. Since link $(3, 4)$ does not meet the rule, the path S is rejected and the $CSPA$ jumps to evaluating the next node in δ_4. All these operations are included in what we call step (Iteration).2. See The following figure.

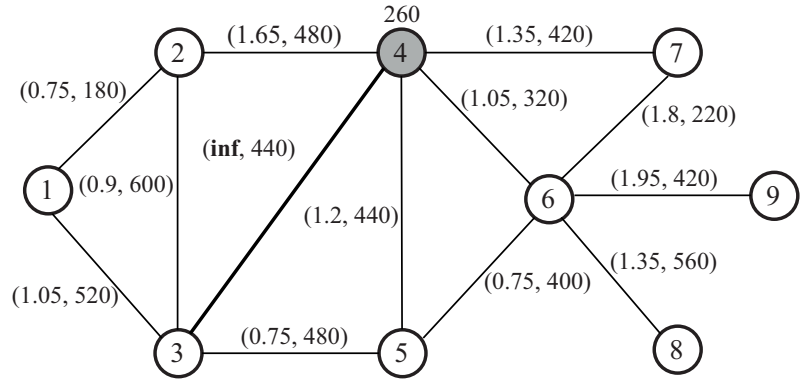


Step 1.2

$N_s = \{4\}$
$\delta_4 = \{\mathbf{2}, 3, 5, 6, 7\}$
$S = \{\mathbf{2,3,4}\}$
$R = \{\}$
$Q = \{\}$

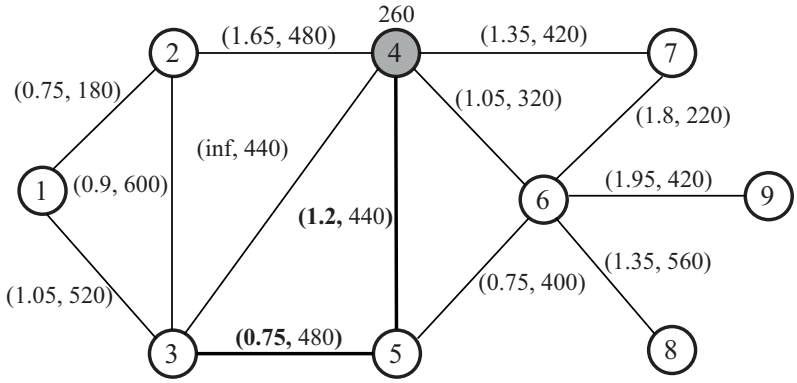
	(2,4)	(3,4)	(4,5)	(4,6)	(4,7)
$A : t(S_{i-4})$	**2.55**	**1.65**	-	-	-
$B : \phi^c_{max} t(P^1_{i-4})$	**2.475**	**2.475**	1.8	1.575	2.025
$A \leq B$?	**No**	**Yes**	-	-	-

The aforementioned steps 1 and 2 are repeated for node 3. So, the adjacent links $(3, 4)$ and $(4, 3)$ are disabled and the shortest path from node 3 to 4 is computed and stored in S (See the next two figures). However, this time, S is not rejected since all the path links $(3, 4)$ and $(4, 5)$, which are also adjacent to node 4, fulfill the first user behavior rule. Consequently, the third step is carried out (see next figure). In that step, a temporary shortest circular path R linking node 4 to the path in S is created, and the infrastructure budgetary and time horizon constraints are evaluated. To show these operations, we include two tables below the first user behavior rule evaluation table. The one on the left is related to satisfying the time horizon, whereas the one at the center is associated with verifying the infrastructure budgetary. The latter is computed by taking into account the construction costs of the links (depicted above each link in the figures) and the construction costs of the starting/terminal node of path R. Since both requirements are met, path R becomes a candidate circular path for node 4 and it is therefore transferred to list Q (see next figure).



Step 2.1

$N_s = \{4\}$
$\delta_4 = \{2, \mathbf{3}, 5, 6, 7\}$
$S = \{\}$
$R = \{\}$
$Q = \{\}$

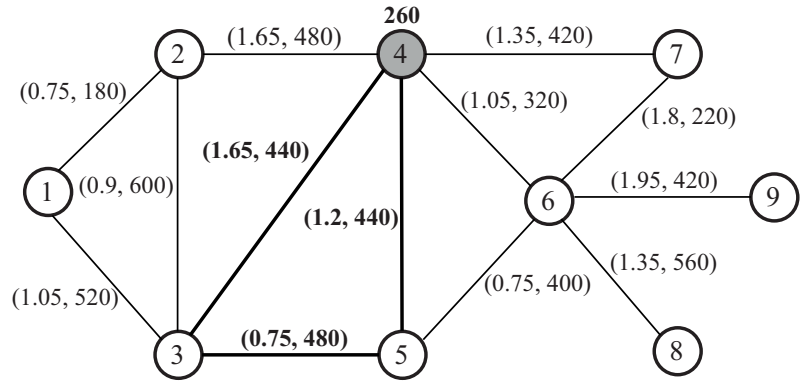


Step 2.2

$N_s = \{4\}$
$\delta_4 = \{2, \mathbf{3}, 5, 6, 7\}$
$S = \{\mathbf{3,5,4}\}$ $R = \{\}$
$Q = \{\}$

	(2,4)	(3,4)	(4,5)	(4,6)	(4,7)
$A : t(S_{i-4})$	-	**1.95**	**1.2**	-	-
$B : \phi^c_{max} t(P^1_{i-4})$	2.475	**2.475**	**1.8**	1.575	2.025
$A \leq B$?	-	**Yes**	**Yes**	-	-

The following figures show the evaluations of the remaining nodes in δ_4. Since all of them pertain to one of the chain of steps already explained, and the figures are self-contained, we will skip an explicit explanation. Having evaluated all the nodes in δ_4, list Q contains only two candidate circular paths (See last figure). Between them, the $CSPA$ chooses the second path: $\{4, 5, 6, 4\}$, since it has the lowest travel time cost.

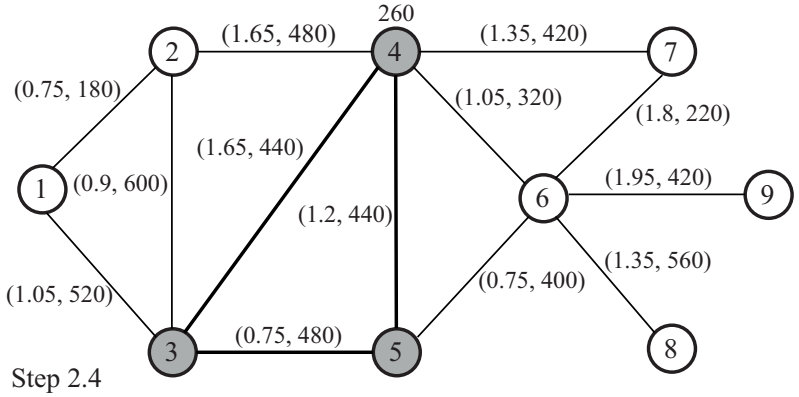


Step 2.3

$N_s = \{4\}$
$\delta_4 = \{2, \mathbf{3}, 5, 6, 7\}$
$S = \{3, 5, 4\}$
$R = \{\mathbf{4,3,5,4}\}$
$Q = \{\}$

$t(R)$	**3.6**
$\hat{h}$	**180**
$t(R) \leq \hat{h}$?	**Yes**

$C(R)$	**1430**
$\bar{c}_{net}$	**8000**
$C(R) \leq \bar{c}_{net}$?	**Yes**

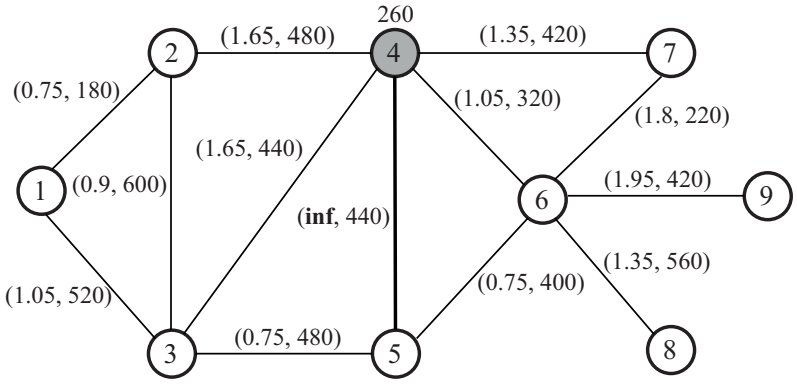


Step 2.4

Step 2.4

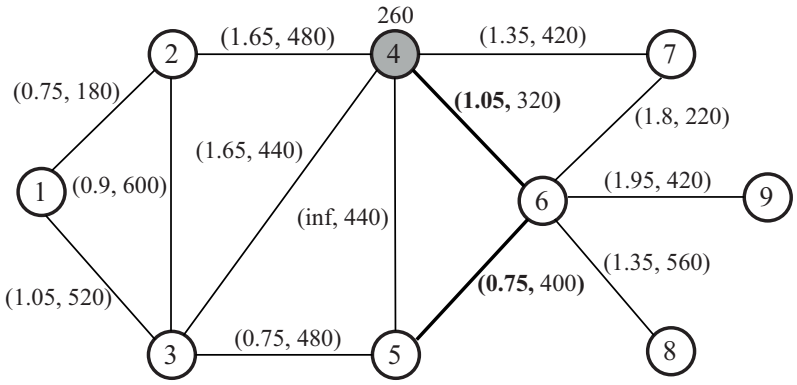
$N_s = \{4\}$
$\delta_4 = \{2, \mathbf{3}, 5, 6, 7\}$
$S = \{3, 5, 4\}$
$R = \{\}$
$Q = \{\{\mathbf{4,3,5,4}\}, \mathbf{3.6}\}$

	(2,4)	(3,4)	(4,5)	(4,6)	(4,7)
$A : t(S_{i-4})$	-	1.95	1.2	-	-
$B : \phi^c_{max} t(P^1_{i-4})$	2.475	2.475	1.8	1.575	2.025
$A \leq B$?	-	Yes	Yes	-	-



Step 3.1

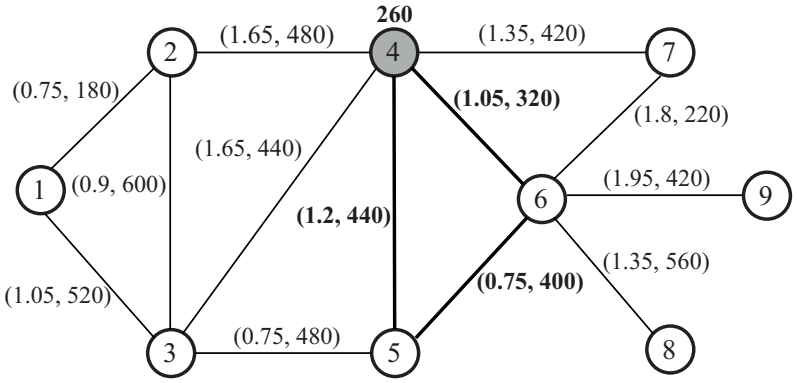
$N_s = \{4\}$
$\delta_4 = \{2, 3, \mathbf{5}, 6, 7\}$
$S = \{\}$
$R = \{\}$
$Q = \{\{4, 3, 5, 4\}, 3.6\}$



Step 3.2

N_s = { 4 }
$\delta_4 = \{2, 3, \mathbf{5}, 6, 7\}$
$S = \{\mathbf{5,6,4}\}$
$R = \{\}$
$Q = \{\{4, 3, 5, 4\}, 3.6\}$

	(2,4)	(3,4)	(4,5)	(4,6)	(4,7)
$A : t(S_{i-4})$	-	-	**1.8**	**1.05**	-
$B : \phi^c_{max} t(P^1_{i-4})$	2.475	2.475	**1.8**	**1.575**	2.025
$A \leq B$?	-	-	**Yes**	**Yes**	-

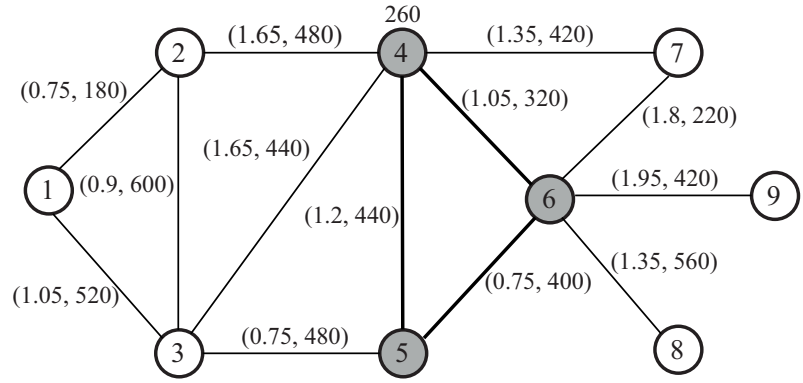


Step 3.3

N_s = { 4 }
$\delta_4 = \{2, 3, \mathbf{5}, 6, 7\}$
$S = \{\}$, $R = \{\mathbf{4,5,6,4}\}$
$Q = \{\{4, 3, 5, 4\}, 3.6\}$

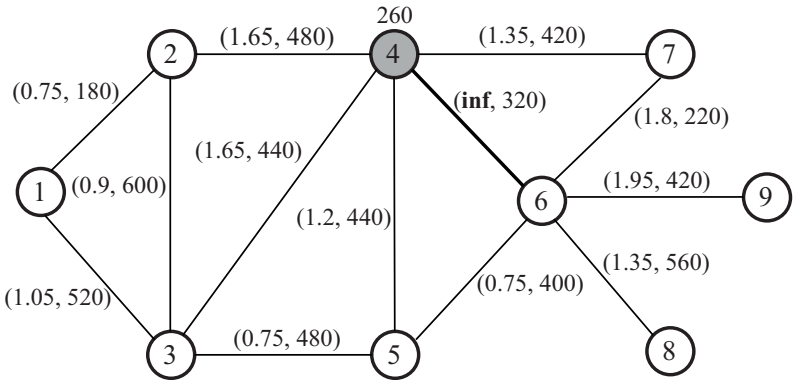
$t(R)$	**3**
$\hat{h}$	**180**
$t(R) \leq \hat{h}$?	**Yes**

$C(R)$	**1430**
$\bar{c}_{net}$	**8000**
$C(R) \leq \bar{c}_{net}$?	**Yes**



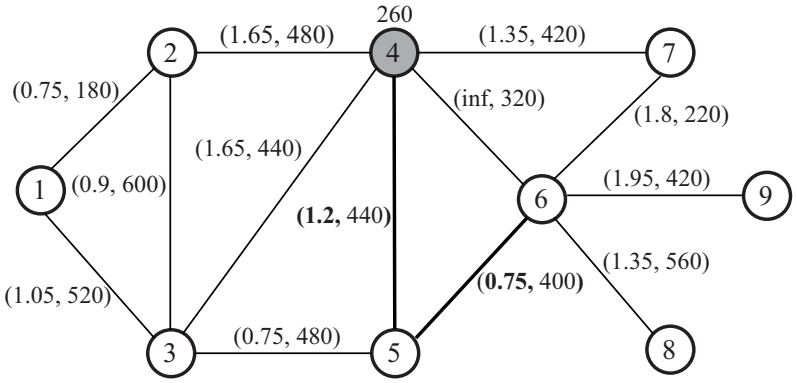
Step 3.4

$N_s = \{4\}$
$\delta_4 = \{2, 3, \mathbf{5}, 6, 7\}$
$S = \{\}$
$R = \{\}$
$Q = \{\{4, 3, 5, 4\}, 3.6$
$\{\mathbf{4,5,6,4}\}, \mathbf{3}\}$



Step 4.1

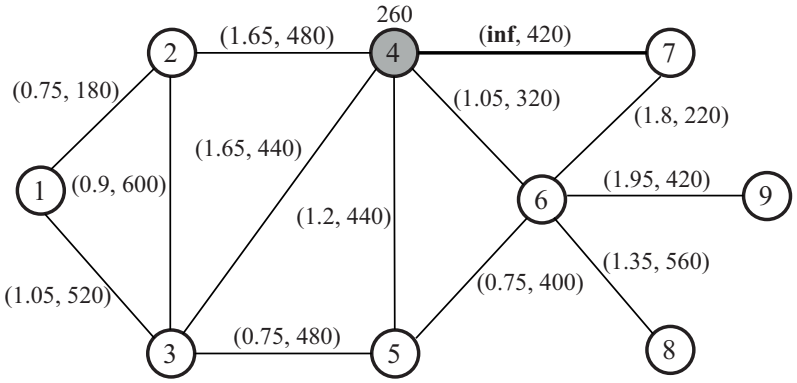
$N_s = \{4\}$
$\delta_4 = \{2, 3, 5, \mathbf{6}, 7\}$
$S = \{\}$
$R = \{\}$
$Q = \{\{4, 3, 5, 4\}, 3.6$
$\{4, 5, 6, 4\}, 3\}$



Step 4.2

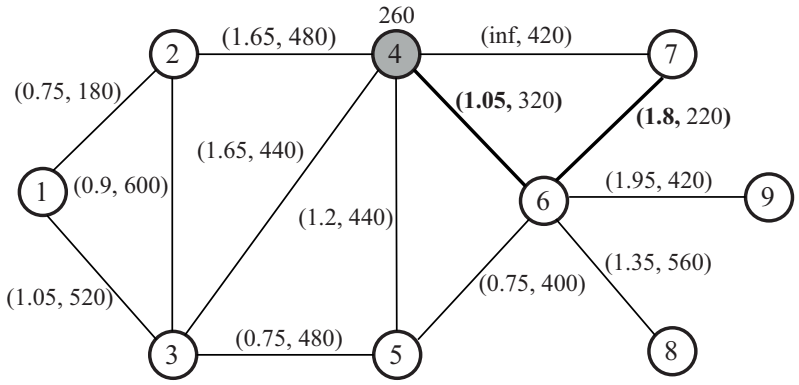
$N_s = \{4\}$
$\delta_4 = \{2, 3, 5, \mathbf{6}, 7\}$
$S = \{\mathbf{6,5,4}\}$
$R = \{\}$
$Q = \{\{4, 3, 5, 4\}, 3.6$
$\{4, 5, 6, 4\}, 3\}$

	(2,4)	(3,4)	(4,5)	(4,6)	(4,7)
$A : t(S_{i-4})$	-	-	-	**1.95**	**1.2**
$B : \phi^c_{max} t(P^1_{i-4})$	2.475	2.475	1.8	**1.575**	**2.025**
$A \leq B$?	-	-	-	**No**	**Yes**



Step 5.1

$N_s = \{4\}$
$\delta_4 = \{2, 3, 5, 6, \mathbf{7}\}$
$S = \{\}$
$R = \{\}$
$Q = \{\{4, 3, 5, 4\}, 3.6$
$\{4, 5, 6, 4\}, 3\}$



Step 5.2

N_s = { 4 }
$\delta_4 = \{2, 3, 5, 6, \mathbf{7}\}$
$S = \{\mathbf{7,6,4}\}$
$R = \{\}$
$Q = \{\{4, 3, 5, 4\}, 3.6$
$\{4, 5, 6, 4\}, 3\}$

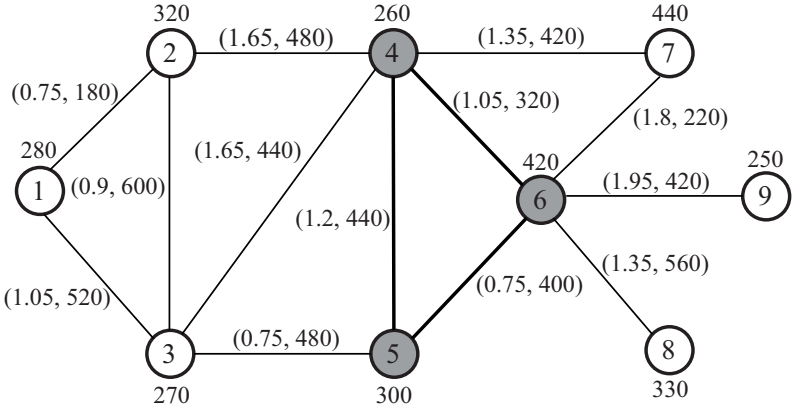
	(2,4)	(3,4)	(4,5)	(4,6)	(4,7)
$A : t(S_{i-4})$	-	-	-	**1.05**	**2.85**
$B : \phi^c_{max} t(P^1_{i-4})$	2.475	2.475	1.8	**1.575**	**2.025**
$A \leq B$?	-	-	-	**Yes**	**No**

Application of Yen's Algorithm for circular corridors

In line with the above illustrative example, we proceed to show how we apply Yen's k-shortest path for circular corridors ($YKSP4CC$) to the 9-node network by using a modified version of the in-vehicle travel time matrix ($T^{'}$), where each cell contains the value of t_{ij} plus its inverse cost t_{ji}. Notice that we have skipped the practical presentation of Yen's algorithm for rectilinear corridors since the circular version is more general and, thus, it includes all the operations performed in the former.

Before continuing with this section, we encourage the reader to carefully study the preceding example, which introduce an illustrative application of the circular shortest path algorithm ($CSPA$). This algorithm is used during the execution of the $YKSP4CC$ and, for the sake of simplicity, we will not provide the details of its application here.

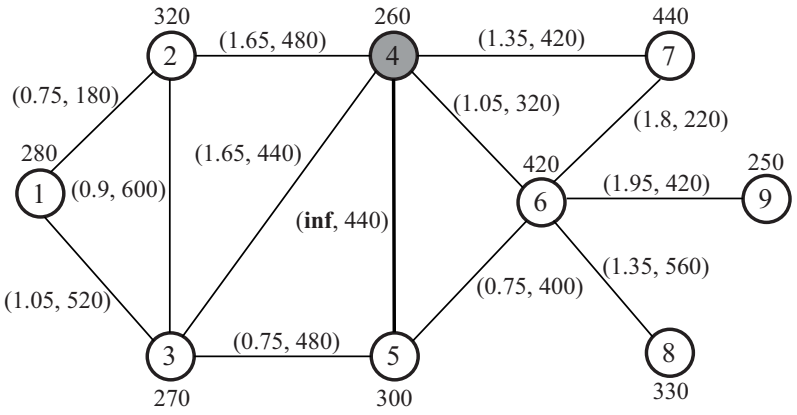
This example is based on finding $K_c = 2$ shortest circular corridors for node 4. The next figure shows the initialization phase of the $YKSP4CC$ algorithm. It basically consists of the setup of lists Q and B. Remember that the former stands for the set of candidate paths to take part in any k^{th} shortest path, whereas the latter is related to all the paths obtained up to the current iteration. Thus, list B is initialized to the shortest circular path of node 4 (P^1).



Initialization phase

$P^1 = \{\mathbf{4,5,6,4}\}$
$B = \{\{\mathbf{4,5,6,4}\}\}$
$i = -$
$S = \{\}$
$R = \{\}$
$Q = \{\}$

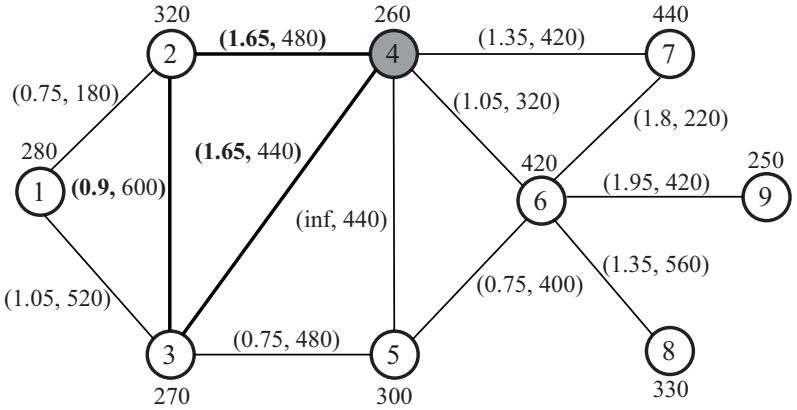
Now, the iterative part of the algorithm starts. As depicted in the following diagram, the algorithm picks up the first node in P^1 as the initial node where the detour under construction must start ($i = 4$). Furthermore, in order to select a different subpath from P^1, link $(4, 5)$ is disabled.



Step 1.1

$P^1 = \{4, 5, 6, 4\}$
$B = \{\{4, 5, 6, 4\}\}$
$i =$ **4**
$S = \{\}$
$R = \{\}$
$Q = \{\}$

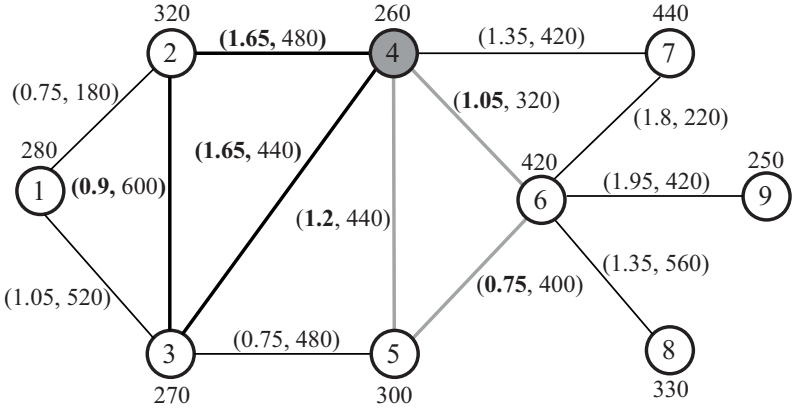
The following step, depicted in the figure below, consists of finding a circular detour not going through link $(4, 5)$ (since it is disabled). Consequently, a call to the aforementioned $CSPA$ algorithm is carried out. Its resolution gives us the circular subpath $\{4, 2, 3, 4\}$ with a total time cost of 4.2 minutes.



Step 1.2

$P^1 = \{4, 5, 6, 4\}$
$B = \{\{4, 5, 6, 4\}\}$
$i = 4$
$S = \{$**4,2,3,4**$\}$
$R = \{\}$
$Q = \{\}$

In the next step, the algorithm checks if this subpath verifies the second user behavior rule (see the immediately following figure) by comparing its total time cost with a portion ($\Delta^c_{min} = 0.5$) of the total time cost of P^1. Since this verification holds, the $YKSP4CC$ proceeds to the next step.



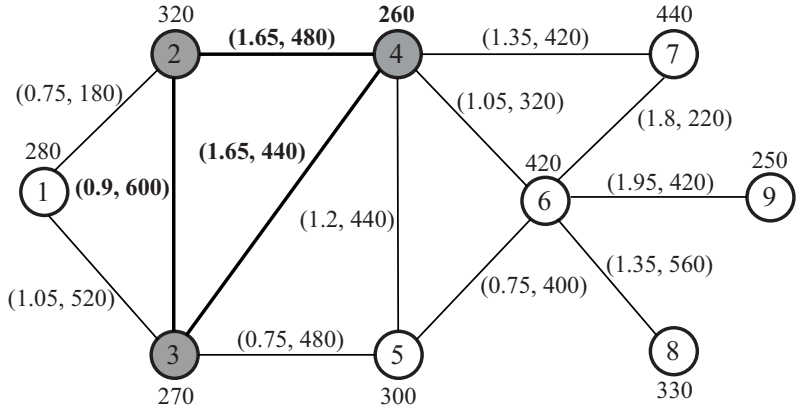
Step 1.3

$P^1 = \{4, 5, 6, 4\}$
$B = \{\{4, 5, 6, 4\}\}$
$i = 4$
$S = \{4, 2, 3, 4\}$
$R = \{\}$
$Q = \{\}$

$t(S)$	4.2
$\Delta^c_{min} \cdot t(P^1)$	1.5
$t(R) \geq \Delta^c_{min} \cdot t(P^1)$?	Yes

At this time, the algorithm creates a new temporary circular path (R), from which it validates the first behavioral rule. This operation is carried out by comparing the time costs of a subset of node pairs $(r, s) \in R$

who hold node $r \in S$ and node $s \in P^1 - S$ to a multiple ($\Phi^c_{max} = 1.6$) of their shortest path. Because node 4 (the first one) provides no pairs meeting these requirements, the $YKSP4CC$ jumps directly to the next step, in which the infrastructure budgetary and time horizon constraints are verified. Since both constraints hold, R is transferred to lists Q and B and the $YKSP4CC$ finishes the first iteration (see the figure below).



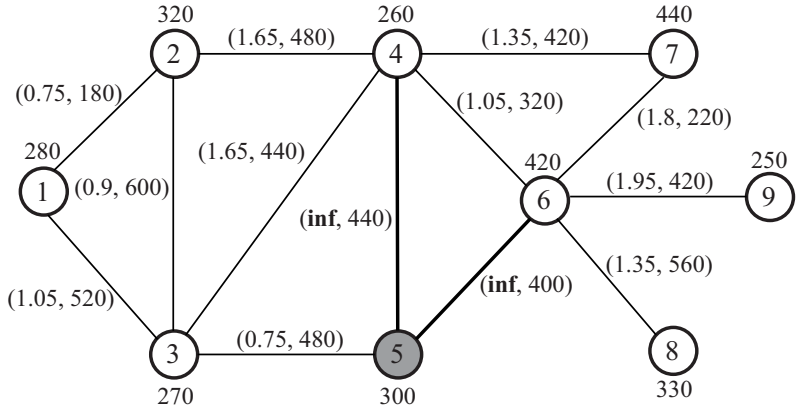
Step 1.5

$P^1 = \{4, 5, 6, 4\}$
$B = \{\{4, 5, 6, 4\},\ \{\mathbf{4,2,3,4}\}\}$
$i = 4$
$S = \{4, 2, 3, 4\}$
$R = \{\mathbf{4,2,3,4}\}$
$Q = \{\{\mathbf{4,2,3,4}\}, \mathbf{4.2}\}$

$t(R)$	4.2
$\hat{h}$	180
$t(R) \leq \hat{h}$?	Yes

$C(R)$	1780
$\bar{c}_{net}$	8000
$C(R) \leq \bar{c}_{net}$?	Yes

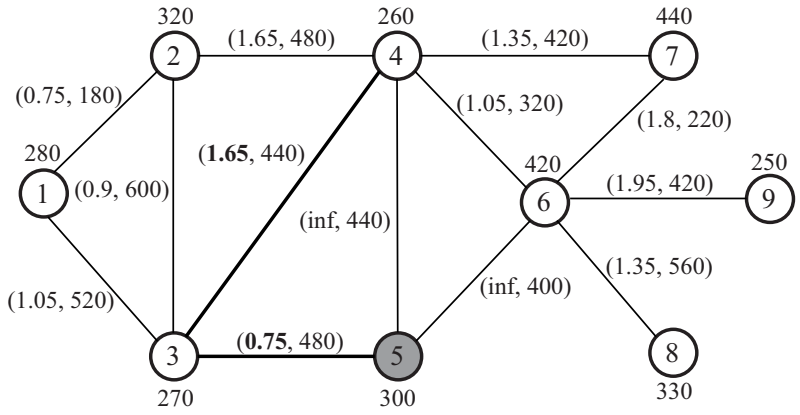
The second iteration begins by picking up the second node in P^1 ($i = 5$). This time, link $(5, 6)$, which connects the "common" part of the new path ($\{4, 5\}$) to the new detour S to be found, is disabled so that it prevents the same path P^1 from being chosen. Moreover, link $(4, 5)$ is also removed. Otherwise, since it links both nodes, 4 and 5, it might possibly be included in the new S (see the figure below).



Step 2.1

$P^1 = \{4, 5, 6, 4\}$
$B = \{\{4, 5, 6, 4\},\ \{4, 2, 3, 4\}\}$
$i = \mathbf{5}$
$S = \{\}$
$R = \{\}$
$Q = \{\{4, 2, 3, 4\}, 4.2\}$

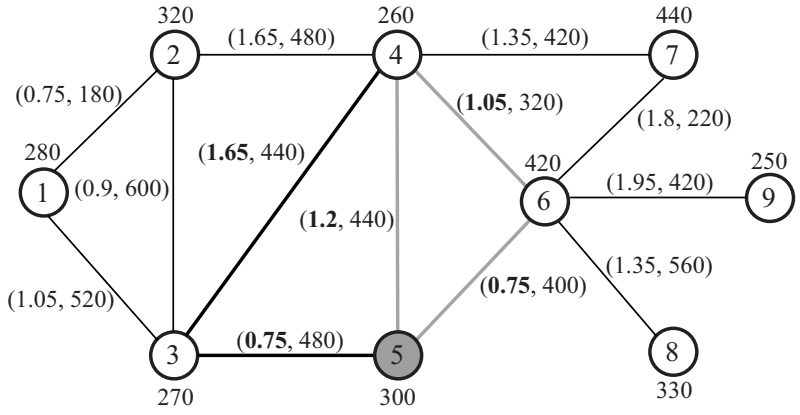
The next step consists of using the adaptation of Dijsktra's algorithm (the DSP procedure), since S must be rectilinear this time. The DSP gives $S = \{5, 3, 4\}$, which has a total time cost of 2.4 (see the figure below).



Step 2.2

$P^1 = \{4, 5, 6, 4\}$
$B = \{\{4, 5, 6, 4\},\ \{4, 2, 3, 4\}\}$
$i = 5$
$S = \{\mathbf{5,3,4}\}$
$R = \{\}$
$Q = \{\{4, 2, 3, 4\}, 4.2\}$

Now the algorithm verifies the second user behavior rule by comparing $t(S)$ to $\Delta^c_{min} \cdot t(P^1)$. Since this validation is correct, the $YKSP4CC$ creates a new temporary shortest circular path $R = \{4, 5, 3, 4\}$ (see the figure below).

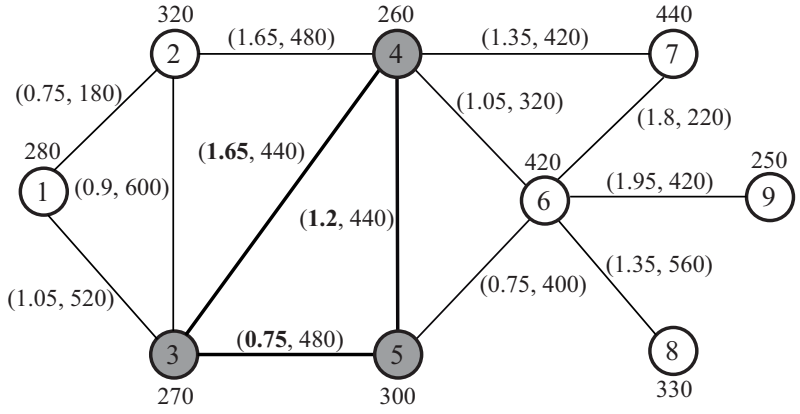


Step 2.3

$P^1 = \{4, 5, 6, 4\}$
$B = \{\{4, 5, 6, 4\},\ \{4, 2, 3, 4\}\}$
$i = 5$
$S = \{5, 3, 4\}$
$R = \{\mathbf{4,5,3,4}\}$
$Q = \{\{4, 2, 3, 4\}, 4.2\}$

$t(S)$	2.4
$\Delta^c_{min} \cdot t(P^1)$	1.5
$t(R) \geq \Delta^c_{min} \cdot t(P^1)$?	Yes

R is used to validate the first user behavior rule as described in the penultimate step of the first iteration. Since this rule is also confirmed, the $YKSP4CC$ goes to the final verification (see the figure below).

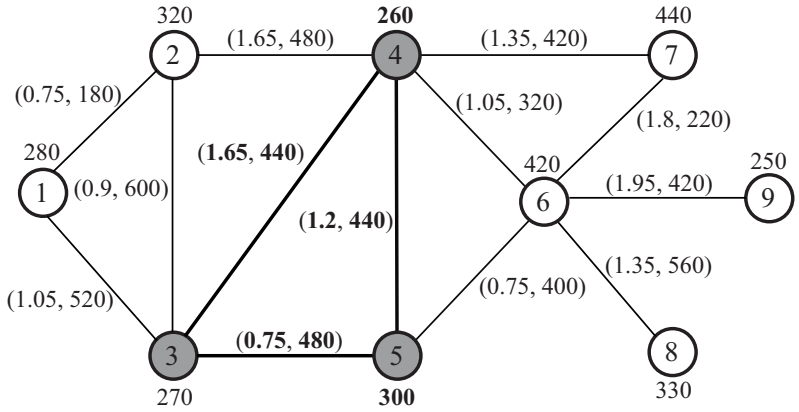


Step 2.4

$P^1 = \{4, 5, 6, 4\}$
$B = \{\{4, 5, 6, 4\},\ \{4, 2, 3, 4\}\}$
$i = 5$
$S = \{5, 3, 4\}$
$R = \{4, 5, 3, 4\}$
$Q = \{\{4, 2, 3, 4\}, 4.2\}$

	(3,4)	(4,5)
$A : t(S_{i-j})$	1.65	1.2
$B : \phi^c_{max} t(P^1_{i-j})$	1.92	2.64
$A \leq B$?	Yes	Yes

The construction cost of the path ($C(R)$) now includes the construction cost of the intermediate node 5, from which detour S emanates. Despite this extra cost, $C(R)$ is still feasible and its total time cost ($t(R)$) does not exceed the planning horizon $\hat{h}$. Consequently, R is transferred to lists Q and B (see the figure below).

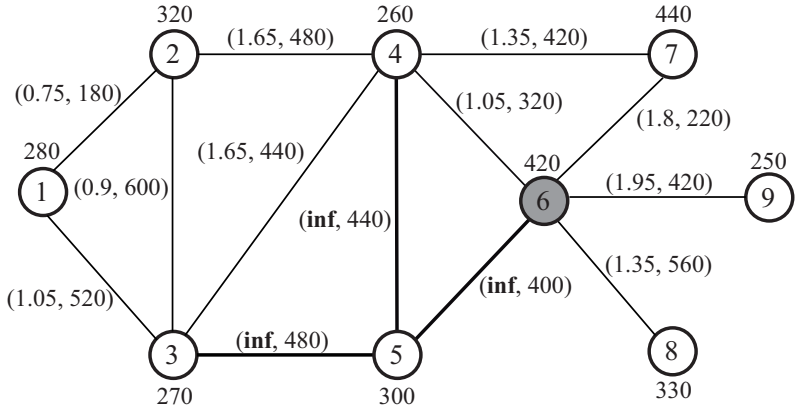


Step 2.5

$P^1 = \{4, 5, 6, 4\}$
$B = \{\{4, 5, 6, 4\},\ \{4, 2, 3, 4\}$
$\{\mathbf{4,5,3,4}\}\}$
$i = 5$
$S = \{5, 3, 4\}$
$R = \{4, 5, 3, 4\}$
$Q = \{\{4, 2, 3, 4\}, 4.2$
$\{\mathbf{4,5,3,4}\ \}, \mathbf{3.6}\}$

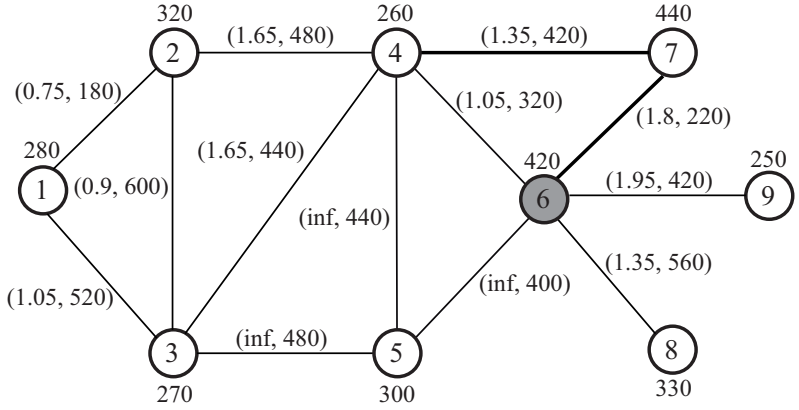
$t(R)$	3.6	$C(R)$	1920
$\hat{h}$	180	$\bar{c}_{net}$	8000
$t(R) \leq \hat{h}$?	Yes	$C(R) \leq \bar{c}_{net}$?	Yes

The remaining iterative steps are skipped since they are quite repetitive. The reader can follow them throughout the remaining figures of the section, with the exception of the last figure, which shows the choice of the second shortest circular path (P^2). Since the third and last iteration do not provide a feasible circular path, the $YKSP4CC$ selects the second path in Q (which has the least time cost) and transfers it to P^2.



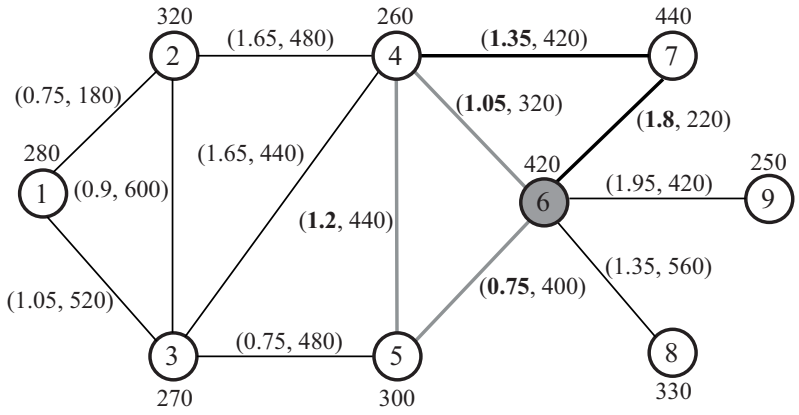
Step 3.1

$P^1 = \{4, 5, 6, 4\}$
$B = \{\{4, 5, 6, 4\},\ \{4, 2, 3, 4\}$
$\{4, 5, 3, 4\}\}$
$i = \mathbf{6}$
$S = \{\}$
$R = \{\}$
$Q = \{\{4, 2, 3, 4\}, 4.2$
$\{4, 5, 3, 4\}, 3.6\}$



Step 3.2

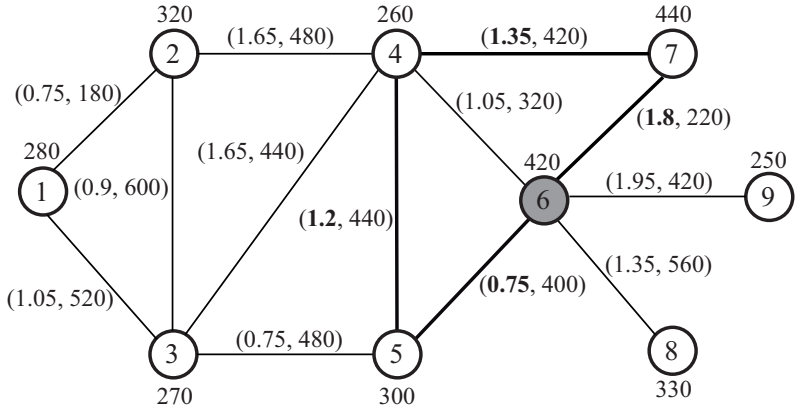
$P^1 = \{4, 5, 6, 4\}$
$B = \{\{4, 5, 6, 4\},\ \{4, 2, 3, 4\}$
$\{4, 5, 3, 4\}\}$
$i = 6$
$S = \{\mathbf{6,7,4}\}$
$R = \{\}$
$Q = \{\{4, 2, 3, 4\}, 4.2$
$\{4, 5, 3, 4\}, 3.6\}$



Step 3.3

$P^1 = \{4, 5, 6, 4\}$
$B = \{\{4, 5, 6, 4\},\ \{4, 2, 3, 4\}$
$\{4, 5, 3, 4\}\}$
$i = 6$
$S = \{6, 7, 4\}$
$R = \{\}$
$Q = \{\{4, 2, 3, 4\}, 4.2$
$\{4, 5, 3, 4\}, 3.6\}$

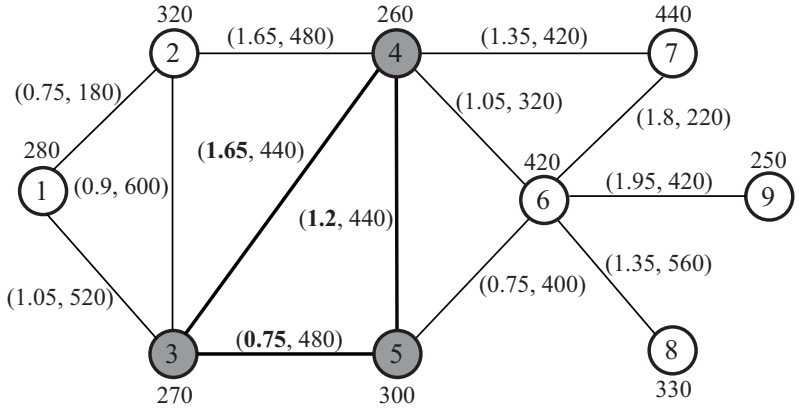
$t(S)$	3.15
$\Delta^c_{min} \cdot t(P^1)$	1.5
$t(R) \geq \Delta^c_{min} \cdot t(P^1)$?	Yes



Step 3.4

$P^1 = \{4, 5, 6, 4\}$
$B = \{\{4, 5, 6, 4\},\ \{4, 2, 3, 4\}$
$\{4, 5, 3, 4\}\}$
$i = 6$
$S = \{6, 7, 4\}$
$R = \{\mathbf{4,5,6,7,4}\}$
$Q = \{\{4, 2, 3, 4\}, 4.2$
$\{4, 5, 3, 4\}, 3.6\}$

	(4,6)	(4,7)	(5,6)	(5,7)
$A : t(S_{i-j})$	1.95	3.75	0.75	1.35
$B : \phi^c_{max} t(P^1_{i-j})$	1.68	2.16	1.2	4.08
$A \leq B$?	No	No	Yes	Yes



Step 4

$P^1 = \{4, 5, 6, 4\}$
$P^2 = \{\mathbf{4,5,3,4}\}$
$B = \{\{4, 5, 6, 4\},\ \{4, 2, 3, 4\}$
$\{4, 5, 3, 4\}\}$
$i = -$
$S = \{\}$
$R = \{\}$
$Q = \{\{4, 2, 3, 4\}, 4.2$
$\{\mathbf{4,5,3,4}\}, \mathbf{3.6}\}$

Illustrative example 3

This appendix aims to exemplify an application of the line splitting algorithm, presented in chapter 5. For the sake of simplification and without losing generality, we have chosen the non-incremental variant. The network under study is depicted in the left-center position of the figures below and consists of 6 nodes and 9 links. Next to the nodes, we have written their construction costs. Next to the links, we have included a two-component vector, which lists from left to right the link construction cost and the total travel time for going through the link in both directions. Above the network, we have depicted a table which denotes the constructed lines and their main features, i.e., their visited nodes and line cycles. On its right side and from top to bottom, we can see the following tables. The first one shows the handling of infrastructure and planning resources, i.e., it indicates the nodes and stretches (from sets N_{TP}^{N} and A_{TP}^{N}) which have or have not been constructed, as well as the number of assigned and acquired vehicles (parameters B and Δb). The next table shows the layout setup of the currently built line, i.e., the nodes and stretches assigned as well as the role of the nodes (sets N_{TP}^{S}, N_{TP}^{P} and A_{TP}). The last Table denotes the budget statuses, i.e., the utilization and availability of infrastructure and planning budgets (parameters $\hat{c}_{net}$ and $\hat{c}_{veh}$).

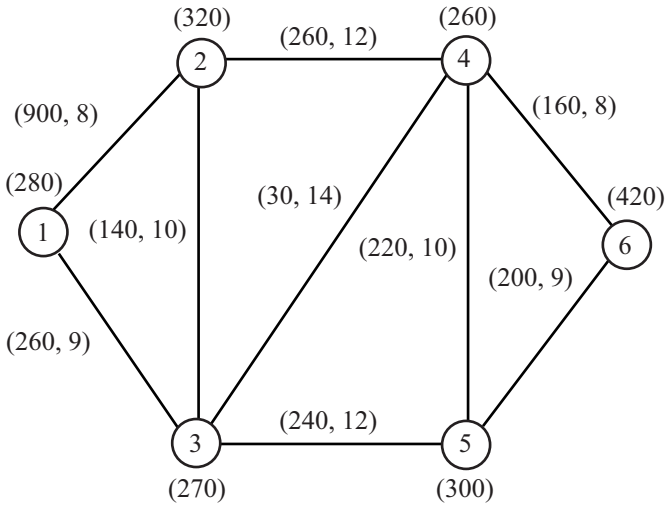
The input parameters not included in the figures are as follows. The number of maximum allowable lines to be constructed is set to $L_{max} = 3$, so the heuristic will carry out three iterations at most. The planning horizon is up to $H = 300$ time units (u.t.) and each link has a capacity of 90 vehicles per H. Additionally, each acquired vehicle has a cost of 500 currency units (c.u.) and can hold up to 100 passengers. This capacity is the same for the available vehicles.

The first figure shows the initial scenario, where no working line exists and the first candidate line to constructed faces three o-d demand pairs depicted below the network. Having solved the first RTNPD instance (see next figure), the o-d demand pair $(1, 6)$ is assigned to the shortest path linking the origin node 1 to the destination node 6, since it is much higher than the other two o-d demand pairs. As a result, the infrastructure budget goes down 2020 c.u. but the vehicle's acquisition budget remains exactly the same, since the number of available vehicles, $B = 10$, is sufficient for strictly satisfying all the demand and entirely filling their capacity. Observe also that the capacity of the links used is not strictly exceeded. The third figure depicts the data update before solving the second RTNPD instance. Consequently, the constructed nodes $\{1, 2\}$ and stretches $\{(1, 2), (2, 4), (4, 6)\}$ are labeled as used resources and the infrastructure budget is decreased to 5980 c.u.s, according to the construction costs of these resources. Finally, these construction costs are set to zero. Notice that, the planning budget as well as the number of available vehicles are not updated, since they can be reallocated to other lines in the next RTNPD resolutions.

The next figure shows the resolution of the second RTNPD instance, where the remaining o-d pairs are assigned to a new line in which the number of vehicles and demand assigned to the previously determined line remain the same. As a result, the remaining infrastructure budget decreases 1270 c.u. and the planning budget falls to 0 c.u., because 10 new vehicles are assigned to satisfy the whole passenger demand. Observe again that the capacity of the links used is not strictly exceeded. Finally, the last figure shows the data update before solving the third RTNPD instance. Consequently, the new constructed nodes $\{3, 5\}$ and stretches $\{(1, 3), (3, 5), (5, 6)\}$ are labeled as used resources and the infrastructure budget is set to 4710 c.u., according to the construction costs of these resources. Finally, these construction costs are set to zero. Again, the vehicle acquisition budget and the number of available vehicles are not updated, since they can be reallocated to other lines in the next RTNPD resolution.

The details of the third iteration of the heuristic are not shown, since no new line is constructed. Consequently, no changes are made to the current line configuration.

l	N^l	c^l
-	-	-



Resources

	Unused	Used
N_{TP}^N	1, 2, 3, 4, 5, 6	-
A_{TP}^N	(1,2), (1,3), (2,3) (2,4), (3,4), (3,5) (4,5), (4,6), (5,6)	-
B	10	0
Δb	10	0

New line layout

N_{TP}^S	-
N_{TP}^P	-
A_{TP}	-

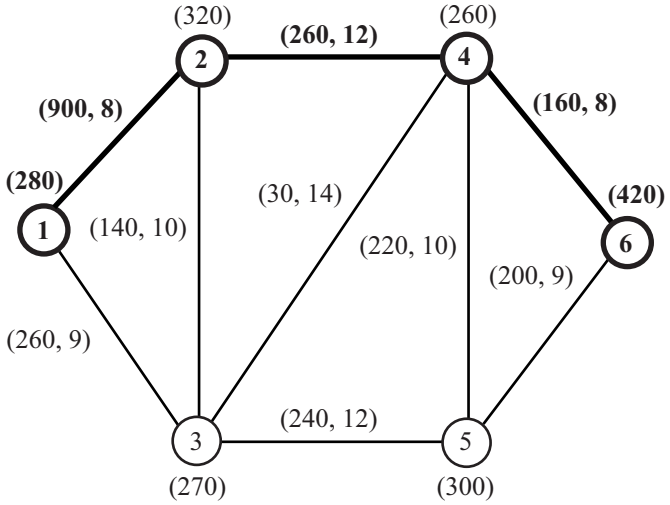
Demand

w	g_w	l_w
$(1, 6)$	9000	-
$(1, 5)$	4500	-
$(3, 6)$	4500	-

Budgets

	Available	Used
$\bar{c}_{net}$	8000 c.u.	0 c.u.
$\bar{c}_{veh}$	5000 c.u.	0 c.u.

l	N^l	c^l
1	**1,2,4,6**	**28 m.u.**



Resources

	Unused	Used
N_{TP}^N	2,3,4,5	**1,6**
A_{TP}^N	(1,3), (2,3), (3,4) (3,5), (4,5), (5,6)	**(1,2), (2,4)** **(4,6)**
B	0	10
Δb	10	0

New line layout

N_{TP}^S	1, 6
N_{TP}^P	2, 4
A_{TP}	(1,2), (2,4), (4,6)

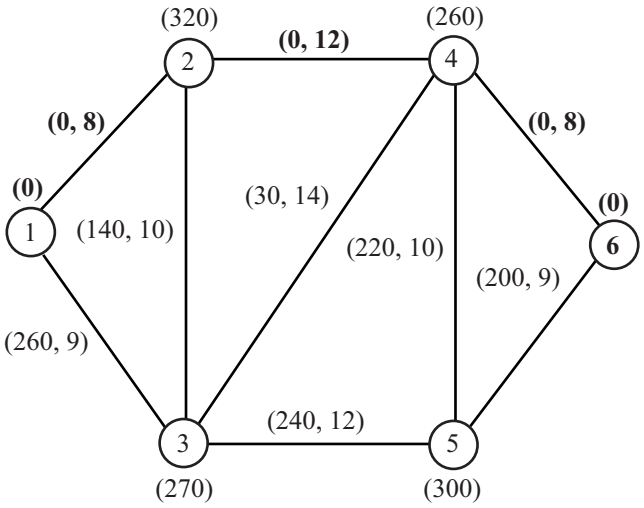
Demand

w	g_w	l_w
(1,6)	**9000**	**1**
$(1, 5)$	4500	-
$(3, 6)$	4500	-

Budgets

	Available	Used
$\bar{c}_{net}$	8000 c.u.	**2020 c.u.**
$\bar{c}_{veh}$	5000 c.u.	0 c.u.

l	N^l	c^l
1	1,2,4,6	28 m.u.



Resources

	Unused	Used
N_{TP}^N	2,3,4,5	1,6
A_{TP}^N	(1,3), (2,3), (3,4) (3,5), (4,5), (5,6)	(1,2), (2,4) (4,6)
B	10	0
Δb	10	0

New line layout

N_{TP}^S	-
N_{TP}^P	-
A_{TP}	-

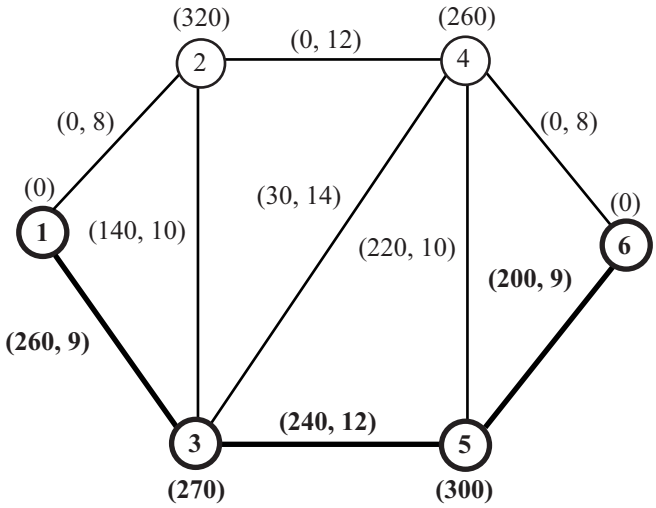
Demand

w	g_w	l_w
$(1,6)$	9000	1
$(1,5)$	4500	-
$(3,6)$	4500	-

Budgets

	Available	Used
$\bar{c}_{net}$	**5980 c.u.**	0 c.u.
$\bar{c}_{veh}$	5000 c.u.	0 c.u.

l	N^l	c^l
1	1,2,4,6	28 m.u.
2	**1,3,5,6**	**30 m.u.**



Resources

	Unused	Used
N_{TP}^N	2,4	1,6,**3,5**
A_{TP}^N	(2,3), (3,4) (4,5)	(1,2),(2,4), (4,6) **(1,3),(3,5),(5,6)**
B	0	10
Δb	1	9

New line layout

N_{TP}^S	1,3,5,6
N_{TP}^P	-
A_{TP}	(1,3), (3,5), (5,6)

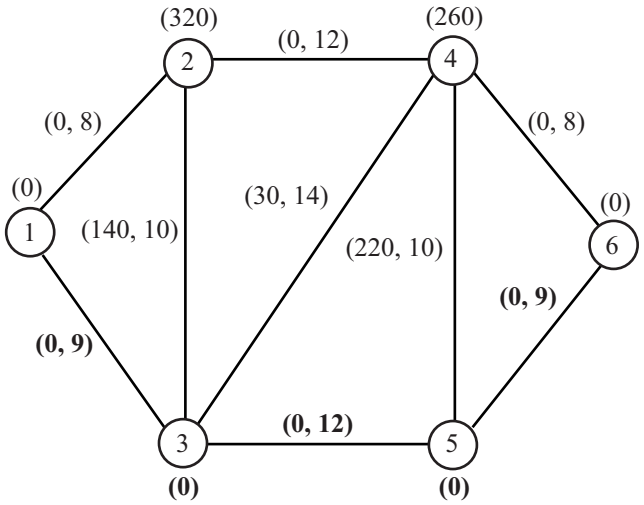
Demand

w	g_w	l_w
$(1,6)$	9000	1
(1,5)	**4500**	**2**
(3,6)	**4500**	**2**

Budgets

	Available	Used
$\bar{c}_{net}$	5980 c.u.	**1270 c.u.**
$\bar{c}_{veh}$	5000 c.u.	4500 c.u.

l	N^l	c^l
1	1,2,4,6	28 m.u.
2	1,3,5,6	30 m.u.



Resources

	Unused	Used
N_{TP}^N	2,4	1,6,3,5
A_{TP}^N	(2,3), (3,4)	(1,2),(2,4),(4,6)
	(4,5)	(1,3),(3,5),(5,6)
B	10	0
Δb	10	0

New line layout

N_{TP}^S	-
N_{TP}^P	-
A_{TP}	-

Demand

w	g_w	l_w
$(1, 6)$	9000	1
$(1, 5)$	4500	2
$(3, 6)$	4500	2

Budgets

	Available	Used
$\bar{c}_{net}$	**4710**	0
$\bar{c}_{veh}$	5000	0

Obtaining a feasible point in the master problem set Y^c

This appendix is devoted to a comprehensive explanation of the procedure **BuiltLines** presented earlier in section 6.4.3, whose pseudo-code is shown in table 3. It contains a set of subroutines that allow us to either carry out the line routing or sequentially select the line corridors (depending on the parameter RoutingModel), as well as their frequency setting.

procedure BuiltLines (**in** RoutingModel, ConstCriterion, FreqCriterion, $\bar{A}$, **out** N, A, f)
  if $|L^N| > 0$ **then**
    if RoutingModel is M1 or M2 **then**
      Get $|L^N|$ 1-stretch lines by calling **SetIniLines** (ConstCriterion, $\hat{c}_{net}$, $N, A, \delta, \hat{c}_{net}$)
      Extend the $|L^N|$ 1-stretch lines by calling **ExtendLines** (ConstCriterion, $\bar{A}$, N, A)
    else
      Select $|L^N|$ corridors by calling **ChoseCorridor** (ConstCriterion, $\bar{A}$, Λ, N, A)
    end if
  end if
  Assign frequency to the extended lines by calling **SetFrequency** (FreqCriterion, N, A, f)
  return N, A, f
end BuiltLines

Table 3: Pseudo-code of the procedure BuiltLines.

Firstly, let's focus on the case that parameter RoutingModel is set to M1 or M2. It entails that the line routing will be carried out by means of a greedy constructive heuristic implemented by subroutines **SetIniLines** and **ExtendLines**. The former determines an initial unique stretch for all the lines under construction, whereas the latter extends these one-stretch lines up to $\hat{A} - 1$ stretches. Both subroutines are called on, provided that there are some lines under construction available ($|L^N| > 0$). Otherwise, only the frequency assignment for working lines is carried out. If one or more lines under construction are available, parameter ConstCriterion establishes the way subroutines **SetIniLines** and **ExtendLines** work, so that different line segments are obtained. ConstCriterion can take two criteria: minimization of infrastructure costs or maximization of infrastructure costs. We refer to infrastructure costs as those costs related to the construction and maintenance of stretches and stations.

The subroutine **SetIniLines** is shown in table 4. The choice of a single stretch depends on the mentioned ConstCriterion parameter and its infrastructure cost c_{ij}, which includes the construction and maintenance costs of the stretch (i, j). In that subroutine, as well as in **ExtendLines**, we have imposed that every node attached to a new line will work as service node. Thus, we also include the construction and maintenance costs of its extreme node-stations. Having selected the most suitable stretch (u, v) per new line, we set its extreme nodes as the temporary extremes of the new line ($\delta^l = \{u, v\}$). Finally, we drop the selected stretch (u, v) from the list of eligible ones ($\tilde{A}$) and update the infrastructure budget according to all infrastructure costs.

The subroutine **ExtendLines** is shown in table 5. Part of its inputs includes the ConstCriterion param-

procedure SetIniLines (**in** ConstCriterion, **in** $\hat{c}_{net}$, **out** $N, A, \delta, \hat{c}_{net}$)

$\tilde{A} \leftarrow \left\{(i,j) \in A_{TP}^{N}\right\}$

$c_{ij} \leftarrow c_i^c + c_i^m + c_j^c + c_j^m + c_{ij}^c + c_{ij}^m, \quad \forall (i,j) \in \tilde{A}$

for each $l \in L^N$ **do**

if ConstCriterion is Minimization **then**

$(u,v) \leftarrow$ Select (i,j) with the minimum c_{ij} in $\tilde{A}$

else

$(u,v) \leftarrow$ Select (i,j) with the maximum c_{ij} in $\tilde{A}$

end if

$N^l \leftarrow \{u,v\}$

$A^l \leftarrow \{(u,v)\}$

$\delta^l \leftarrow \{u,v\}$

$\hat{c}_{net} \leftarrow \hat{c}_{net} - (c_u^c + c_v^c + c_{uv}^c)$

$\tilde{A} \leftarrow \tilde{A} - \{(u,v)\}$

end for

return $N, A, \delta, \hat{c}_{net}$

end SetIniLines

Table 4: Pseudo-code of the procedure SetIniLines.

eter, which establishes a criterion for selecting stretches and an upper bound on the maximum number of stretches that a line can hold ($\bar{A}$). This means that, after having finished that subroutine, each line can hold up to $\bar{A}$ stretches, provided that we have enough infrastructure budget and that the line cycle (c^l) does not exceed the planning horizon H. The extension of each line includes an additional inherent criterion which consists of the following. Whatever the construction criterion is, the new stretches are appended to the line at its temporally extreme nodes $u, v \in \delta^l$.

The inclusion of a new stretch is carried out in three steps. Firstly, the eligible stretches for a given line l ($(i,j) \in \tilde{A}^l$) are computed according to the three requirements contained in R_{ij}. The first one is related to connectivity issues and establishes that all eligible stretches must share, at least, one extreme node with the terminal nodes of the line, which are held in set δ^l. Moreover, the cardinality of δ^l must be equal to two; otherwise, there is no terminal node in the line. In other words, the line is circular and, thus, it cannot be extended any longer.

The second requirement assures that the increment of the line cycle, due to the inclusion of the new link, does not exceed the planning horizon. To compute this increment, we take into account not only the in-vehicle time but also the node service time (t_s) and layover time (t^l), so long as some of the extreme nodes of the eligible link do not correspond to the terminal nodes of the line ($|\delta^l \cup \{i,j\}| - |\delta^l|$).

procedure ExtendLines(**in** ConstCriterion, $\bar{A}$, N, A, $\Delta\hat{c}_{net}$, H **out** N, A)

for each $i \in [2..\bar{A}]$ **do**

for each $l \in L^N$ **do**

$$c^l \leftarrow \sum_{(u,v)\in A^l} (t_{uv} + t_{vu}) + 2 \cdot t_s \cdot |N^l|$$

$$\tilde{A}^l \leftarrow \left\{(i,j) \in \tilde{A} \text{ satisfying } R_{ij}\right\}$$

$$R_{ij} \leftarrow \begin{cases} i \in \delta^l \text{ or } j \in \delta^l \text{ and } \delta^l = 2 \\ c^l + t_{ij} + t_{ji} + 2 \cdot \left[(t_s + t_l) \cdot \left(|\delta^l \cup \{i,j\}| - |\delta^l|\right)\right] \leq H \\ c_{ij}^{c,l} \leq \Delta\hat{c}_{net} \end{cases}$$

if $\tilde{A}^l \neq \emptyset$ **then**

$(u, v, c^c_{uv}) \leftarrow$ Select (i,j) with the set criterion on c^l_{ij} in $\tilde{A}^l$

by calling **BestSuitableLink**(ConstCriterion, $\tilde{A}^l$)

$\Delta\hat{c}_{net} \leftarrow \Delta\hat{c}_{net} - c^c_{uv}$

$A^l \leftarrow A^l \cup \{(u,v)\}$

$\tilde{A} \leftarrow \tilde{A} - \{(u,v)\}$

$\delta^l \leftarrow \delta^l \cup \{u,v\} - \delta^l \cap \{u,v\}$

end if

end for

end for

return N, A

end ExtendLines

Table 5: Pseudo-code of the procedure ExtendLines.

The last requirement establishes that the infrastructure cost of the stretch in line l ($c_{ij}^{c,l}$) does not surpass the remaining infrastructure budget ($\Delta\hat{c}_{net}$). This cost is computed by means of the following equation:

$$c_{ij}^{c,l} = \begin{cases} c^c_{ij} + c^c_j & \text{If } i \in \delta^l \\ c^c_{ij} + c^c_i & \text{Otherwise} \end{cases} \tag{32}$$

Observe that its computation depends on the extreme node of the stretch which is already connected to the line.

The second step consists of finding the best stretch to be included in the line, according to the ConstCriterion parameter and the c^l_{ij} measure. This operation is performed by calling on the procedure **BestSuitableLink**, shown in table 6. Firstly, it computes for a given $(i,j) \in \tilde{A}^l$ the c^l_{ij} measure by verifying which extreme node is connected to the line, and then it compares the c^l_{ij} cost to the one belonging to the temporary best suitable stretch ($\tilde{c}_{ij}^{c,l}$). If the criterion is met, it updates the best suitable link data. These two operations are repeated for every $(i,j) \in \tilde{A}^l$.

procedure BestSuitableLink(**in** ConstCriterion, **in** $\tilde{A}^l$, **in** $\hat{c}_{net}, H$, **out** $u, v, \tilde{c}^l_{ij}, \tilde{c}^{c,l}_{ij}$)

$$\tilde{c}^l_{ij} \leftarrow \begin{cases} \infty & \text{If ConstCriterion is Minimization} \\ 0 & \text{Otherwise} \end{cases}$$

for each $(i,j) \in \tilde{A}^l$ **do**

$$c^l_{ij} \leftarrow \begin{cases} c^{c,l}_{ij} + c^m_{ij} + c^c_j + c^m_j & \text{If } i \in \delta^l \\ c^{c,l}_{ij} + c^m_{ij} + c^c_i + c^m_i & \text{Otherwise} \end{cases}$$

if ConstCriterion is Minimization **and** $c^l_{ij} < \tilde{c}^l_{ij}$ **or** ConstCriterion is Maximization **and** $c^l_{ij} > \tilde{c}^l_{ij}$ **then**

$\tilde{c}^l_{ij} \leftarrow c^l_{ij}$

$\tilde{c}^{c,l}_{ij} \leftarrow c^{c,l}_{ij}$

$u \leftarrow i$

$v \leftarrow j$

end if

end for

return $u, v, \tilde{c}^{c,l}_{ij}$

end BestSuitableLink

Table 6: Pseudo-code of the procedure BestSuitableLink.

The last step of subroutine **ExtendLines** consists of a parameter update related to the extension of the line. It includes the setting of its new extreme nodes and its grade, as well as the computation of the remaining infrastructure budget. Additionally, we record the link and node identifiers in sets A^l and N^l, respectively.

Before explaining the assignment, let's focus on the case that the parameter RoutingModel is set to M3. That means the line segments are going to be selected from the pool Λ. This is done by means of subroutine **ChoseCorridor,** shown in table 7. This subroutine chooses a set of $|L^N|$ corridors according to their γ^c costs and the established criterion in the ConstCriterion parameter. As stated above, this parameter can be set to the minimization or maximization of infrastructure costs. In the case of minimization, the $|L^N|$ corridors with at most $\bar{A}$ links and with minimum γ_c costs will be selected. In contrast, if it is maximization, then the $|L^N|$ corridors with at most $\bar{A}$ links, and with maximum γ_c costs will be chosen. The γ_c cost takes into account the construction and maintenance costs of all stretches and nodes used by the corridor, which are stated in the column vector c of matrices D and E, respectively.

Notice that, the initial $\bar{A}$ parameter can be modified by the procedure in the case of no $\bar{A}$ link corridors. In these situations, it looks for the nearest $\bar{A}$ link corridors towards minus infinity.

After choosing the appropriate corridors or carrying out the routing of the $|L^N|$ lines, we proceed to assign them frequency by means of the subroutine **SetFrequency,** shown in table 8. As inputs, it needs the topology of the line (captured in sets A^l and N^l) and a criterion to establish how vehicles and frequencies are assigned to them (captured in the FreqCriterion parameter). The available criteria are maximum or minimum line frequency assignments.

procedure ChoseCorridor (**in** ConstCriterion, $\bar{A}, \Lambda$ **out** N, A)

$$\Lambda^{\bar{A}} \leftarrow \left\{ c \in \Lambda : \sum_{(i,j) \in A_{TP}^N:\, i<j} D_{ij}^c = \bar{A} \right\}$$

while $\Lambda^{\bar{A}} = \emptyset$ **do**

$\bar{A} \leftarrow \bar{A} - 1$

$$\Lambda^{\bar{A}} \leftarrow \left\{ c \in \Lambda : \sum_{(i,j) \in A_{TP}^N:\, i<j} D_{ij}^c = \bar{A} \right\}$$

end while

$$\gamma_c \leftarrow \sum_{(i,j) \in A_{TP}^N:\, i<j} D_{ij}^c \cdot \left(c_{ij}^c + c_{ij}^m \right) + \sum_{i \in N_{TP}^N} E_i^c \cdot (c_i^c + c_i^m), \quad \forall c \in \Lambda^{\bar{A}}$$

$\Lambda^{\bar{A}} \leftarrow$ **sort** $\Lambda^{\bar{A}}$ with the ConstCriterion on $\gamma^{\bar{A}}$

For each $l \in L^N$ **do**

$c \leftarrow$ **Select** the first element in $\Lambda^{\bar{A}}$

$\Lambda^{\bar{A}} \leftarrow \Lambda^{\bar{A}} - \{c\}$

$$A^l \leftarrow \left\{ (i,j) \in A_{TP}^N : i < j,\ D_{ij}^c = 1 \right\}$$

$$N^l \leftarrow \left\{ i \in N_{TP}^N : E_i^c = 1 \right\}$$

end for

return N, A

end ChoseCorridor

Table 7: Pseudo-code of the procedure ChoseCorridor.

For the maximization case, we proceed as follows. Firstly, we compute the number of lines that use each link $(i,j) \in A_{TP}$ ($|L_{ij}|$). Secondly, we determine the maximum number of vehicles that can be assigned to each line ($\bar{B}$), assuming homogenous assignment. This measure takes into account the available planning budget, so we first find out the maximum number of vehicles which can be acquired ($\bar{\Delta} f_c$). Having done these two calculations, we go into the for-loop where we carry out the frequency assignment. To do that, we first get the maximum line capacity (f_0^l) according to the capacity of the stretches ($\bar{f}_{ij}$), and the number of lines that use them ($|L_{ij}|$). Secondly, we compute an upper bound of the line frequency (f_1^l) by means of the line cycle (c^l) and the number of assigned vehicles ($\bar{B}$). Finally, we select from these two bounds the one that has the least value.

Regarding the minimization case, we simply assign the inverse of the planning horizon, since we consider that only one service per line will be carried out. Notice that the number of services per line $z^l = H \cdot f^l$, so $f^l = \dfrac{z^l}{H}$.

procedure SetFrequency (**in** FreqCriterion, **in** $N,\ A,$ **out** f)

if FreqCriterion is Maximization **then**

$L_{ij} \leftarrow \{l \in L : (r,s) \in A^l\}, \quad \forall (i,j) \in A_{TP}$

$\bar{\Delta} f_c \leftarrow \frac{\bar{c}_{veh}}{c_b^a}$

$\tilde{B} = \left\lfloor \frac{f_c + \Delta f_c}{|L|} \right\rfloor$

for each $l \in L$ **do**

$f_0^l \leftarrow \min \left\{ \frac{\bar{f}_{ij}}{|L_{ij}|}, \quad \forall (i,j) \in A^l \right\}$

$c^l \leftarrow \sum_{(i,j) \in A^l} (t_{ij} + t_{ji}) + 2 \cdot \left[t_s \cdot |N^l| + t_l \cdot \left(|N^l| - |A^l| \right) \right]$

$f_1^l \leftarrow \frac{\tilde{B}}{c^l}$

$f^l \leftarrow \min\{f_0^l,\ f_1^l\}$

end for

else

$f^l \leftarrow \frac{1}{H}$

end if

return f

end SetFrequency

Table 8: Pseudo-code of the procedure SetFrequency.

For the sake of clarification, we include the following table, which lists the specific parameters and sets that are used by the aforementioned procedures:

$\bar{A}$ Maximum number of allowed stretches per line.

$\tilde{A}$ Set of stretches which are not still used by any line at the current line's extension iteration.

$\tilde{A}^l$ Set of candidate stretches to be used by line l at the current line's extension iteration.

A^l Set of stretches assigned to line l at the current line's extension iteration.

$\tilde{B}$ Number of vehicles to be assigned to each line.

c_{ij} Total cost of the required network infrastructure to link node i directly to node j.

c_{ij}^c Construction cost of the required network infrastructure to link node i directly to node j.

c_{ij}^l Total cost of the required extended network infrastructure to link node i directly to node j at line l.

$c_{ij}^{c,l}$ Construction cost of the required extended network infrastructure to link node i directly to node j at line l.

c^l Line cycle.

$ConstCriterion$ Parameter denoting the criterion to be used to construct the line segments.

D_{ij}^c A 0-1 valued cell in a column of matrix D denoting whether or not stretch (i, j) is held in corridor c.

E_i^c A 0-1 valued cell in a column of matrix E denoting whether or not node i is held in corridor c.

f^l Frequency of line l.

f_0^l Maximum allowable frequency assignment to line l according to link capacities.

f_1^l Maximum allowable frequency assignment to line l according to vehicle availability.

$FreqCriterion$ Parameter denoting the criterion to be used to assign frequency to the lines.

L_{ij} Number of lines which share link (i, j).

N^l Set of nodes assigned to line l at the current line's extension iteration.

$RoutingModel$ Parameter denoting the routing model in use.

t_s Node Service time.

t^l Layover time of line l.

δ^l Terminal nodes of line l.

Λ Set of candidate corridors to be assigned to any of the lines under construction.

$\Lambda^{\bar{A}}$ Set of candidate corridors with $\bar{A}$ stretches.

$\bar{\Delta} f_c$ Maximum number of vehicles which can be acquired with the current planning budget.

$\Delta \hat{c}_{net}$ Remaining infrastructure budget.

Guidelines for the input data setting

This appendix aims to provide basic guidelines for setting some input model data. They are split into two groups: the infrastructure input data and the planning input data. The former comprises the setting of the construction and maintenance costs of the infrastructure resources, whereas the latter consists of determining the service costs as well as the main features of the planning resources.

Infrastructure input data

The infrastructure input data includes the construction and maintenance costs of the infrastructure resources, i.e., the stretches and station nodes which are candidates to be used by some new constructed line.

The setting of the construction costs of the stretches (c_{ij}^c) depends mainly on three elements: the monetary cost of one unit length of stretch ($c_{ij}^{c,1}$), the amortization horizon (H_a), and the total length of the stretch (d_{ij}^{TP}). Having determined the three parameters, one needs to apply the following formula:

$$c_{ij}^c = d_{ij}^{TP} \cdot \frac{c_{ij}^{c,1}}{H_a} \tag{33}$$

Construction costs related to stations (c_i^c) depends on three elements: station capacity (q_i), the maximum allowable passenger density (ρ_{pax}), and the cost of one square meter of platform ($c_i^{c,1}$). The q_i is expressed in amounts of passengers, and an estimated average value can be obtained from the following formula:

$$q_i = H \cdot \frac{\sum_{w \in W_i} g^w}{\bar{t}_w} \tag{34}$$

where W_i stands for the subset of od-demand pairs which have origins or destinations in station i, $\bar{t}_w$ is the average passenger waiting time and H is the planning horizon. Having obtained q_i, the following final formula determines the station cost:

$$c_i^c = q_i \cdot \rho_{pax} \cdot c_i^{c,1} \tag{35}$$

Finally, the maintenance costs of stretches and stations are obtained simply by multiplying the construction costs by 0.1. We take this approximation since maintenance costs depend on several parameters which are difficult to estimate.

Planning input data

The planning input data comprises the service costs as well as the main features of the planning resources. We refer to planning resources as the stretches and public transport vehicles which go through them.

Service costs related to stretches (c_{ij}^f) include mainly the power consumption costs (c_b^e) of public transportation vehicles (PTV), as well as the cost of private vehicles (c_d). They are expressed as monetary units/time units, so that they are weighted by the average traveling time through the link (i, j) to compute c_{ij}^f as follows:

$$c_{ij}^f = t_{ij}^{TP} \cdot (c_b^e + c_d) \tag{36}$$

Cost c_b^e can be determined by means of the energy cost per time unit (c_e^{tu}) and the nominal power of the PTV engine ($\eta_e \cdot \bar{P}_e$) as follows:

$$c_b^e = c_e^{tu} \cdot \eta_e \cdot \bar{P}_e \tag{37}$$

Finally, c_d cost is computed taking into account the average base salary (c_s) and total number of working time units per month (H_w), as follows:

$$c_d = \frac{c_s}{H_w} \tag{38}$$

Regarding stretch features, capacity (q_{ij}) has been set by means of the following formula:

$$q_{ij} = \frac{1}{t_{ij}^s(1 + K_s)} \tag{39}$$

where parameter t_{ij}^s stands for the minimum time between two consecutive arrivals at node j departing from node i, and K_s denotes a safety coefficient. This time includes the in-vehicle travel time through the stretch

(t_{ij}^{TP}) as well as the average service time at the destination node (t_j^s)

Vehicle service costs are split into two categories: acquisition and setting costs (c_b^a and c_b^s, respectively). The former takes into account the iff for those vehicles which are not available, i.e., those vehicles which are not in the current working fleet.

c_b^a cost is computed by means of the following formula:

$$c_b^a = \frac{n_{car} \cdot c_{car}}{H_a} \tag{40}$$

where parameters n_{car}, c_{car} and H_a stand for the average number of carriages contained in the PTV, the acquisition cost of a carriage and the amortization horizon, respectively.

c_b^s consists of the cost to move a PTV from its depot location to the starting node of the line where it is assigned. So, its value can only be obtained if we know a priori the location of the PTV depot (d), the initial station of the given line (s), and the shortest or candidate path through which the vehicle will move (P_s^l), as follows:

$$c_b^s = \sum_{(i,j) \in P_s^l} c_{ij}^{TP} \tag{41}$$

where parameter c_{ij}^{TP} denotes the total cost in monetary units of going through stretch (i, j). P_s^l contains the path from d to s in both directions, since the PTV must move back to the depot when the working day finishes.

Finally, the PTV features, i.e., the capacity (q), have been set according to the following formula:

$$q = n_{car} \cdot q_{car} \tag{42}$$

where q_{car} stands for the average carriage capacity. It includes the number of seated and standing passengers.

Basic input data for the Santiago de Chile Network

This appendix is devoted to providing some input data for the Santiago de Chile underground network. This data is needed for carrying out the experiments shown in section 8.1.2.4 of chapter 8. Tables 9 - 13 give the relationship between each node identifier and its label, so that the reader can find out which links are referred to by the following Tables 14 - 18. The last tables 19 - 21 refer to the data related to the links of the routing graph shown in Figure 8.22.

Id	Label	Id	Label
001	San Pablo	015	Universidad Catlica
002	Neptuno	016	Baquedano
003	Pajaritos	017	Salvador
004	Las Rejas	018	Manuel Montt
005	Ecuador	019	Pedro de Valdivia
006	San Alberto Hurtado	020	Los Leones
007	Universidad de Santiago	021	Tobalaba
008	Estacin Central	022	El Golf
009	Unin Latinoamericana	023	Alcntara
010	Repblica	024	Escuela Militar
011	Los Hroes	025	Manquehue
012	La Moneda	026	Hernando de Magallanes
013	Universidad de Chile	027	Los Dominicos
014	Santa Lucía		

Table 9: Relationship between the node identifiers and the label names of the L1 stops.

Id	Label	Id	Label
028	Vespucio Norte	038	Parque O'Higgins
029	Zapadores	039	Rondizzoni
030	Dorsal	040	Franklin
031	Einstein	041	El Llano
032	Cementerios	042	San Miguel
033	Cerro Blanco	043	Lo Vial
034	Patronato	044	Departamental
035	Puente Cal y Canto	045	Ciudad del Nio
036	Santa Ana	046	Lo Ovalle
011	Los Hroes	047	El Parrn
037	Toesca	048	La Cisterna

Table 10: Relationship between the node identifiers and the label names of the L2 stops.

Id	Label	Id	Label
021	Tobalaba	060	Vicua Mackenna
049	Cristbal Coln	061	Vicente Valds
050	Francisco Bilbao	062	Rojas Magallanes
051	Príncipe de Gales	063	Trinidad
052	Simn Bolívar	064	San Jos de la Estrella
053	Plaza Egaa	065	Los Quillayes
054	Los Orientales	066	Elisa Correa
055	Grecia	067	Hospital Stero del Rio
056	Los Presidentes	068	Protectora de la Infancia
057	Quilín	069	Las Mercedes
058	Las Torres	070	Plaza de Puente Alto
059	Macul		

Table 11: Relationship between the node identifiers and the label names of the L4 stops.

Id	Label	Id	Label
060	Vicua Mackenna	073	Santa Rosa
071	Santa Julia	074	San Ramn
072	La Granja	048	La Cisterna

Table 12: Relationship between the node identifiers and the label names of the L4A stops.

Id	Label	Id	Label
075	Plaza de Maip	088	Plaza de Armas
076	Santiago Bueras	089	Bellas Artes
077	Del Sol	016	Baquedano
078	Monte Tabor	090	Parque Bustamante
079	Las Parcelas	091	Santa Isabel
080	Laguna Sur	092	Irarrzabal
081	Barrancas	093	uble
082	Pudahuel	094	Rodrigo de Araya
001	San Pablo	095	Carlos Valdovinos
083	Lo Prado	096	Camino Agricola
084	Blanquedano	097	San Joaquín
085	Gruta de Lourdes	098	Pedrero
086	Quinta Normal	099	Mirador
087	Cumming	100	Bellavista de la Florida
036	Santa Ana	061	Vicente Valds

Table 13: Relationship between the node identifiers and the label names of the L5 stops.

i-node	j-node	d_{ij}^{TP} (km)	c_{ij}^{f} (e $\times$ min/veh)	q_{ij} (veh/min)
001	002	0.7	4.31571	0.470588
002	003	1	6.1653	0.4
003	004	0.9	5.54877	0.421053
004	005	0.7	4.31571	0.470588
005	006	0.7	4.31571	0.470588
006	007	0.6	3.69918	0.5
007	008	0.7	4.31571	0.470588
008	009	0.5	3.08265	0.533333
009	010	0.7	4.31571	0.470588
010	011	0.5	3.08265	0.533333
011	012	0.5	3.08265	0.533333
012	013	0.5	3.08265	0.533333
013	014	0.5	3.08265	0.533333
014	015	0.6	3.69918	0.5
015	016	0.6	3.69918	0.5
016	017	0.9	5.54877	0.421053
017	018	0.7	4.31571	0.470588
018	019	0.7	4.31571	0.470588
019	020	0.6	3.69918	0.5
020	021	0.7	4.31571	0.470588
021	022	0.6	3.69918	0.5
022	023	0.6	3.69918	0.5
023	024	0.6	3.69918	0.5
024	025	1.4	8.63142	0.333333
025	026	1.3	8.01489	0.347826
026	027	1	6.1653	0.4

Table 14: Stretch features for working line L1.

i-node	**j-node**	d_{ij}^{TP} **(km)**	c_{ij}^{f} **(e** $\times$ **min/veh)**	q_{ij} **(veh/min)**
028	029	1.4	8.63142	0.333333
029	030	0.7	4.31571	0.470588
030	031	1	6.1653	0.4
031	032	0.9	5.54877	0.421053
032	033	2.8	17.2628	0.210526
033	034	2.3	14.1802	0.242424
034	035	0.7	4.31571	0.470588
035	036	1.4	8.63142	0.333333
036	011	0.9	5.54877	0.421053
011	037	0.7	4.31571	0.470588
037	038	2.6	16.0298	0.222222
038	039	0.9	5.54877	0.421053
039	040	1.1	6.78183	0.380952
040	041	2.9	17.8794	0.205128
041	042	4.3	26.5108	0.150943
042	043	5.6	34.5257	0.121212
043	044	4	24.6612	0.16
044	045	2.5	15.4133	0.228571
045	046	0.8	4.93224	0.444444
046	047	1.1	6.78183	0.380952
047	048	1.4	8.63142	0.333333

Table 15: Stretch features for working line L2.

i-node	j-node	d_{ij}^{TP} (km)	c_{ij}^{f} (e $\times$ min/veh)	q_{ij} (veh/min)
021	049	1.3	8.01489	0.347826
049	050	0.8	4.93224	0.444444
050	051	1.4	8.63142	0.333333
051	052	2.3	14.1802	0.242424
052	053	0.8	4.93224	0.444444
053	054	1.1	6.78183	0.380952
054	055	3.1	19.1124	0.195122
055	056	5.1	31.443	0.131148
056	057	6	36.9918	0.114286
057	058	3.8	23.4281	0.166667
058	059	1.2	7.39836	0.363636
059	060	1.3	8.01489	0.347826
060	061	0.7	4.31571	0.470588
061	062	1.2	7.39836	0.363636
062	063	3.6	22.1951	0.173913
063	064	4.4	27.1273	0.148148
064	065	2.7	16.6463	0.216216
065	066	0.9	5.54877	0.421053
066	067	0.8	4.93224	0.444444
067	068	2.7	16.6463	0.216216
068	069	1.9	11.7141	0.275862
069	070	0.9	5.54877	0.421053

Table 16: Stretch features for working line L4.

i-node	j-node	d_{ij}^{TP} (km)	c_{ij}^{f} (e $\times$ min/veh)	q_{ij} (veh/min)
060	071	1.5	9.24795	0.32
071	072	1.6	9.86448	0.307692
072	073	1.7	10.481	0.296296
073	074	0.9	5.54877	0.421053
074	048	2	12.3306	0.266667

Table 17: Stretch features for working line L4A.

i-node	j-node	d_{ij}^{TP} (km)	c_{ij}^{f} (e $\times$ min/veh)	q_{ij} (veh/min)
075	076	1.6	9.86448	0.307692
076	077	0.9	5.54877	0.421053
077	078	1.2	7.39836	0.363636
078	079	0.9	5.54877	0.421053
079	080	1.6	9.86448	0.307692
080	081	1.1	6.78183	0.380952
081	082	1.2	7.39836	0.363636
082	001	1.7	10.481	0.296296
001	083	0.6	3.69918	0.5
083	084	0.9	5.54877	0.421053
084	085	1.5	9.24795	0.32
085	086	1.1	6.78183	0.380952
086	087	1.1	6.78183	0.380952
087	036	0.9	5.54877	0.421053
036	088	0.8	4.93224	0.444444
088	089	0.6	3.69918	0.5
089	016	1.2	7.39836	0.363636
016	090	0.5	3.08265	0.533333
090	091	1.6	9.86448	0.307692
091	092	0.7	4.31571	0.470588
092	093	1.5	9.24795	0.32
093	094	4.2	25.8943	0.153846
094	095	6.4	39.4579	0.108108
095	096	8.1	49.9389	0.087912
096	097	6.4	39.4579	0.108108
097	098	4.6	28.3604	0.142857
098	099	2.7	16.6463	0.216216
099	100	0.9	5.54877	0.421053
100	061	0.7	4.31571	0.470588

Table 18: Stretch features for working line L5.

i-node	j-node	d_{ij}^{TP} (km)	c_{ij}^{f} (e-min/veh)	q_{ij} (veh/min)	c_{ij}^{c} (euros)
021	233	1.2	7.39836	2.75	23.0137
021	234	1	6.1653	2.5	19.1781
028	225	0.7	4.31571	2.125	13.4247
028	029	1.4	8.63142	3	26.8493
029	030	0.7	4.31571	2.125	13.4247
029	225	0.9	5.54877	2.375	17.2603
029	226	1.5	9.24795	3.125	28.7671
029	230	1.4	8.63142	3	26.8493
030	226	1.6	9.86448	3.25	30.6849
030	230	1.2	7.39836	2.75	23.0137
225	226	1.5	9.24795	3.125	28.7671
226	227	0.9	5.54877	2.375	17.2603
226	228	0.8	4.93224	2.25	15.3425
226	230	1.1	6.78183	2.625	21.0959
227	229	0.9	5.54877	2.375	17.2603
228	229	1.4	8.63142	3	26.8493
228	230	0.6	3.69918	2	11.5069
228	232	1	6.1653	2.5	19.1781
229	232	1	6.1653	2.5	19.1781
229	233	1.2	7.39836	2.75	23.0137
229	234	1.6	9.86448	3.25	30.6849
230	231	1.4	8.63142	3	26.8493
230	232	0.8	4.93224	2.25	15.3425
231	232	1	6.1653	2.5	19.1781
231	234	1	6.1653	2.5	19.1781
232	234	1.7	10.481	3.375	32.6027
233	234	1.2	7.39836	2.75	23.0137

Table 19: Stretch features for routing sector Tobalaba - Vespucio Norte.

i-node	j-node	d_{ij}^{TP} (km)	c_{ij}^{f} (e-min/veh)	q_{ij} (veh/min)	c_{ij}^{c} (euros)
028	216	1.8	11.0975	3.5	34.5205
028	218	1.5	9.24795	3.125	28.7671
082	209	1.1	6.78183	2.625	21.0959
082	210	0.9	5.54877	2.375	17.2603
209	210	0.9	5.54877	2.375	17.2603
209	211	1.1	6.78183	2.625	21.0959
210	212	1.7	10.481	3.375	32.6027
211	212	1.3	8.01489	2.875	24.9315
211	223	1.9	11.7141	3.625	36.4384
212	213	1.6	9.86448	3.25	30.6849
212	223	1.4	8.63142	3	26.8493
213	214	1.1	6.78183	2.625	21.0959
213	223	1.2	7.39836	2.75	23.0137
214	215	2.1	12.9471	3.875	40.274
214	221	2.5	15.4133	4.375	47.9452
214	222	2	12.3306	3.75	38.3562
214	223	2.9	17.8794	4.875	55.6164
214	224	1.6	9.86448	3.25	30.6849
215	216	1.3	8.01489	2.875	24.9315
215	218	2	12.3306	3.75	38.3562
215	219	1.8	11.0975	3.5	34.5205
215	224	1.5	9.24795	3.125	28.7671
216	218	1	6.1653	2.5	19.1781
218	219	0.9	5.54877	2.375	17.2603
219	220	1.8	11.0975	3.5	34.5205
219	224	2.4	14.7967	4.25	46.0274
220	221	2.4	14.7967	4.25	46.0274
220	224	1.5	9.24795	3.125	28.7671
221	222	2.7	16.6463	4.625	51.7808
222	223	1.6	9.86448	3.25	30.6849
222	224	3.2	19.729	5.25	61.3699

Table 20: Stretch features for routing sector Vespucio Norte - San Pablo.

i-node	j-node	d_{ij}^{TP} (km)	c_{ij}^{f} (e-min/veh)	q_{ij} (veh/min)	c_{ij}^{c} (euros)
048	207	1.5	9.24795	3.125	28.7671
048	208	1.3	8.01489	2.875	24.9315
075	199	1.6	9.86448	3.25	30.6849
075	200	1.8	11.0975	3.5	34.5205
199	200	1	6.1653	2.5	19.1781
199	201	1.6	9.86448	3.25	30.6849
200	201	1.9	11.7141	3.625	36.4384
200	202	1.4	8.63142	3	26.8493
201	202	1.6	9.86448	3.25	30.6849
201	203	1.3	8.01489	2.875	24.9315
202	203	1.6	9.86448	3.25	30.6849
202	204	2	12.3306	3.75	38.3562
203	204	1.7	10.481	3.375	32.6027
203	205	1.6	9.86448	3.25	30.6849
204	205	1.1	6.78183	2.625	21.0959
204	206	1.5	9.24795	3.125	28.7671
205	206	1.5	9.24795	3.125	28.7671
205	207	1.8	11.0975	3.5	34.5205
206	207	1.2	7.39836	2.75	23.0137
206	208	1.4	8.63142	3	26.8493
207	208	1.1	6.78183	2.625	21.0959

Table 21: Stretch features for routing sector Plaza de Maip - La Cisterna.

Objective function

This section shows the breakdown of the optimal objective function of the inelastic demand version of the model for the experiment performed on the Seville network where only two lines can be constructed. On the list, the column Term shows the part of the objective to which the remaining columns refer. The next column, Factor denotes the value which weighs the term. Column Cost reports the value of the term without weighting. Column Weighted Cost shows the value of the term with weighting. Finally, the column Portion denotes the portion of the objective function which represents this term.

Term	Factor	Cost(/h)	Weighted Cost	Portion(%)
Operators	0.10	12955.01	1295.50	4.74 %
Frequencies	0.10	3319.41	331.94	1.22 %
Vehicles Setting	0.10	400.00	40.00	0.15 %
Vehicles Acq.	0.10	0.00	0.00	0.00 %
Stations Const.	0.10	2134.00	213.40	0.78 %
Stations Maint.	0.10	235.40	23.54	0.09 %
Stretches Const.	0.10	6242.00	624.20	2.29 %
Stretches Maint.	0.10	624.20	62.42	0.23 %
Passengers	0.90	28899.16	26009.24	95.26 %
Time	0.90	28899.16	26009.24	95.26 %
In-Vehicle	0.90	1278.71	1150.84	4.21 %
Walking	0.90	18328.45	16495.60	60.41 %
Boarding	0.90	4861.95	4375.75	16.03 %
Alighting	0.90	2430.97	2187.87	8.01 %
Waiting	0.90	1999.08	1799.17	6.59 %
Total		41854.17	27304.74	

Demand Utility tuning

Demand modal splitting is based on the logit probabilistic functions (7.3) - (7.4), which are incorporated into a mathematical program, as shown in chapter 7. However, they need to be calibrated, i.e., logit parameters β_w^{PRI}, θ_w, γ_w^{PRI} and β_w^{TP} need to be fixed. One approach is to carry out a survey on passenger preferences and infer the parameter values from the collected data. However, this process entails spending a great amount of time and money. Moreover, the obtained results would be valid only for a specific region or, at most, for the country's inhabitants, since passenger preferences are affected by the country's culture.

Instead of this approach, we employ the following. We develop a likely logit probalistic modal function, which behaves in a similar manner as if the parameter values were inferred from a survey. It is based on the type of model results we want to obtain when elastic demand is applied. We consider the following principle: the higher the travel time throughout the public transportation network, the less the demand there is from people willing to use it.

The travel time throughout the public transportation network for a given od-pair is not known in advance, since it is part of the model results. However, we can infer the lower and upper bounds in a straightforward manner. The lower bound corresponds to the determination of the shortest path throughout the extended network depicted in figure 2.2. Its mathematical form is as follows:

$$\min_{v} \sum_{w \in W} g_w \cdot \sum_{(i,j) \in A_{TP} \setminus A_x} t_{ij}^{TP} \cdot \sum_{l \in L_{ij}} v_{ij}^{w,l} \tag{43}$$

$s.t.:$

$$\sum_{l \in L_i} \left(\sum_{j \in l_{a(i)}} v_{ij}^{w,l} - \sum_{j \in l_{y(i)}} v_{ji}^{w,l} \right) = t_i^w, \quad \forall i \in N,\ w \in W \tag{44}$$

$$v_{inv^+(i)}^{w,l} = v_{y(i)}^{w,l} + \sum_{j \in A^l_{x^+(i)}} v_{ij}^{w,l}, \quad \forall l \in L,\ i \in N_{TP}^{S+}(l),\ w \in W \tag{45}$$

$$v_{inv^-(i)}^{w,l} = v_{a(i)}^{w,l} + \sum_{j \in A^l_{x^-(i)}} v_{ji}^{w,l}, \quad \forall l \in L,\ i \in N_{TP}^{S-}(l),\ w \in W \tag{46}$$

Fortunately, this problem does not need to be solved by a linear optimization solver. Instead, we can employ Diskstra's algorithm [58] or the Floyd & Warshall procedure [58], [181] if we compute all the shortest paths at the same time. The latter obtains lower bounds faster if we consider a complete OD-demand matrix.

Regarding the upper bound. It can be obtained through solving a largest path problem throughout the extended network. However, this value could be too far from reality. Furthermore, the resolution of this problem is time consuming. Instead, we employ a practical rule. We consider that a given OD-pair is not willing to use the public transportation network if its whole travel time exceeds K times the shortest one.

Having determined the lower and upper bounds, we associate to them its probabilities (the demand portion which will be assigned to the public transportation mode). As depicted in figure 2, the lower travel time bound t_{min}^w is related to the upper probability bound P_u. Its value is set to 0.95, thus we encourage the model to assign 95% of the demand to the public transportation mode when the travel time of the OD-demand pair tends toward the minimum. In contrast, when it approaches the maximum travel time, the demand portion

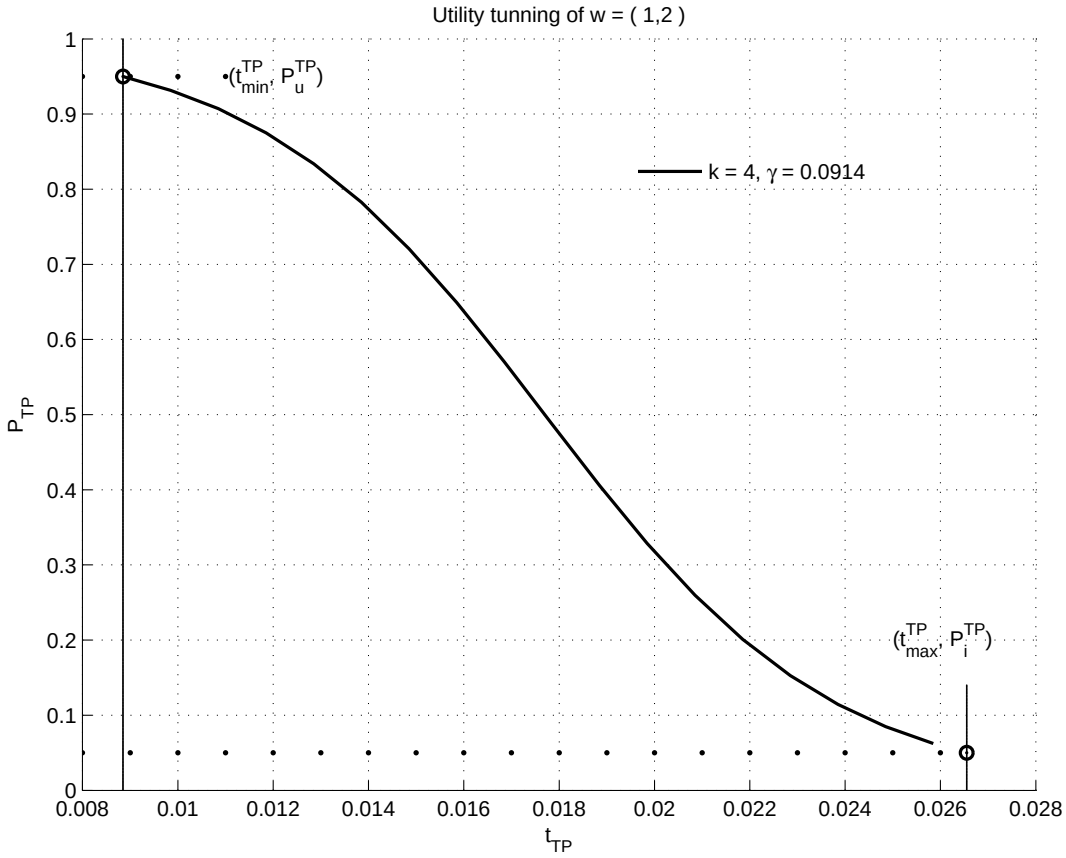


Figure 2: Logit probability function for a given o-d demand pair w.

goes rapidly down to 0. Its lower probability bound P_i is set to 0.05.

Now, we use the points (t^w_{min}, P_u) and (t^w_{max}, P_i) in collaboration with the logit probabilistic functions (7.3) - (7.4) to establish the following relationships:

$$U^{PRI}_w - U^{TP}_w(t^w_{min}) = \ln\left(\frac{1-P_u}{P_u}\right) \tag{47}$$

$$U^{PRI}_w - U^{TP}_w(t^w_{max}) = \ln\left(\frac{1-P_i}{P_i}\right) \tag{48}$$

where U^{PRI}_w and U^{TP}_w are the utility modal functions (7.1) - (7.2). By combining (47) with (48) and rearranging terms, we obtain the following close expression which gives the value of θ_w:

$$\theta_w = \frac{1}{t^w_{max} - t^w_{min}} \cdot \ln\left[\frac{(1-P_i)\cdot P_u}{(1-P_u)\cdot P_i}\right] \tag{49}$$

Now using the value of θ_w, we can determine a close expression with respect to a weighted sum of the utility parameters β^{PRI}_w, β^{TP}_w and γ^{PRI}_w by means of relationship (47) or (48). Consequently, we have two degrees of freedom which can be overcome by applying common sense rules. For instance, we can add a new equation relating the values of β^{PRI}_w and β^{TP}_w as follows:

$$\beta_w^{PRI} = K^w \cdot \beta_w^{TP} \tag{50}$$

where input parameter $K^w \geq 1$ establishes that the public transportation fare is K^w times cheaper than the parking cost at origin $p(w)$ and/or destination $q(w)$. To break the remaining degree of freedom, we look for the fuel cost per time unit, γ^{PRI}, supposing that the features of the type of vehicle used for any OD demand pair are quite similar in terms of fuel consumption. Then, to obtain its corresponding cost for a specific OD-demand pair, we simply weight that unitary cost γ^{PRI} by the average private distance traveled by the OD-demand pair d_w^{PRI}, which is given as an input parameter.

To sum up, we first compute θ_w by means of (49) and $\gamma_w^{PRI} = \gamma^{PRI} \cdot d_w^{PRI}$ and then β_w^{TP} and β_w^{PRI} by means of (50) and (47) or (48). If we decide on first computing β_w^{TP} with the point equation (47) and θ_w expression (49), we obtain the following equation:

$$\beta_w^{TP} = \frac{1}{(1 - K^w)} \cdot \ln \left[\frac{(1 - P_u)^{\sigma+1} \cdot P_i^{\sigma}}{(1 - P_i)^{\sigma} \cdot P_i^{\sigma+1}} \right] \tag{51}$$

where σ stands for the following expression:

$$\sigma = \frac{t_{min}^w - t_w^{PRI}}{t_{max}^w - t_{min}^w} \tag{52}$$

Entropy discretization

Parameters a_r^w and b_r^w stand for the coefficients of the piecewise linear first order approximations of the entropy modal functions (7.32) - (7.33). Their calibration depends on the method used to fit them and the maximum allowable error. There are many fitting-methods in the literature. For instance, the first order approximation of Taylor's polinomium. However, this type of method uses a constant step to establish the intervals where the line equations are active. It usually entails construction of a high number of line equations and, thus, it yields to a large number of constraints (7.49) - (7.50).

Instead of using this method, we employ a numerical method which works with a variable step. The big picture is shown in table 23. It begins by computing a unique line equation linking the extreme values of a given OD-demand pair g_{min}^w and g_{max}^w and then looking for the demand value g_r, where there is the maximum approximation error. This step is carried out by calling on the subroutine **FindMaxError,** shown in table 22. Observe that the maximum error is computed as the relative difference between the value of the exact entropy function $f(g_i^w)$ and its approximation function $\tilde{f}(g_i^w)$.

procedure FindMaxError (**in** $a^w,\ b^w,\ g^w,$ **out** $\epsilon,\ r,\ g_r$)

$\epsilon \leftarrow 0$

For each $i \in [1..|g^w| - 1]$ **do**

$\tilde{f}(g_i^w) \leftarrow a_{r(i)}^w + b_{r(i)}^w \cdot g_i^w$

$f(g_i^w) \leftarrow g_i^w \cdot (\ln g_i^w - 1)$

$$\tilde{\epsilon} \leftarrow \frac{|\tilde{f}(g_i^w) - f(g_i^w)|}{f(g_i^w)}$$

If $\epsilon < \tilde{\epsilon}$ **then**

$r \leftarrow$ Interval where g_r^w is held in

$g_r \leftarrow g_r^w$

$\epsilon \leftarrow \tilde{\epsilon}$

end if

end for

return $\epsilon,\ r,\ g_r$

end FindMaxError

Table 22: Pseudo-code of the procedure FindMaxError.

Having computed the maximum error, the line coefficients a_r^w, b_r^w and a_{r+1}^w, b_{r+1}^w related to the interval r, where the error is located, are updated according to the discrete demand points $\tilde{g}^w$, which are assigned to these intervals. Demand point g_r is associated with the maximum error.

These steps are repeated as long as the maximum error $\tilde{\epsilon}$ exceeds the pre-specified error ϵ. Having finished the **Fitting** procedure, the whole set of line coefficients a^w and b^w are returned, as well as intervals n^w, to which they are related.

procedure Fitting (**in** $\epsilon,\ g^w_{max},\ g^w_{min},$ **out** $a^w,\ b^w,\ n^w$)

$$a^w_1 \leftarrow \frac{g^w_{min} \cdot f(g^w_{max}) - g^w_{max} \cdot f(g^w_{min})}{g^w_{min} - g^w_{max}}$$

$$b^w_1 \leftarrow \frac{f(g^w_{max}) - f(g^w_{min})}{g^w_{max} - g^w_{min}}$$

$$g^w \leftarrow \{g^w_{min} \,..\, g^w_{max}\}$$

$$\tilde{g}^w \leftarrow \{g^w_{min},\ g^w_{max}\}$$

$$n^w \leftarrow \{1\}$$

do

$(\tilde{\epsilon}, r, g_r) \leftarrow$ Get the maximum approximation error and its interval by calling **FindMaxError(** $a^w,\ b^w,\ g^w,\ n^w,\ \tilde{\epsilon},\ r,\ g_r$**)**

For each $i \in [|n^w|..r+1]$ **do**

$$a^w_{i+1} \leftarrow a^w_i$$

$$b^w_{i+1} \leftarrow b^w_i$$

end for

$$a^w_r \leftarrow \frac{\tilde{g}^w_r \cdot f(g_r) - g_r \cdot f(\tilde{g}^w_r)}{\tilde{g}^w_r - g_r}$$

$$b^w_r \leftarrow \frac{f(g_r) - f(\tilde{g}^w_r)}{g_r - \tilde{g}^w_r}$$

$$a^w_{r+1} \leftarrow \frac{g_r \cdot f(\tilde{g}^w_{r+1}) - \tilde{g}^w_{r+1} \cdot f(g_r)}{g_r - \tilde{g}^w_{r+1}}$$

$$b^w_{r+1} \leftarrow \frac{f(\tilde{g}^w_{r+1}) - f(g_r)}{\tilde{g}^w_{r+1} - g_r}$$

$$\tilde{g}^w \leftarrow \tilde{g}^w_{1-r} \cup \{g_r\} \cup \tilde{g}^w_{r+1-|n^w|+1}$$

$$n^w \leftarrow n^w \cup \{|n^w| + 1\}$$

while $(\tilde{\epsilon} < \epsilon)$

return $a^w,\ b^w,\ n^w$

end Fitting

Table 23: Pseudo-code of the procedure Fitting.