SCHEDULING WITH LATEST ARRIVAL CONSOLIDATION IN SERVICE NETWORK DESIGN PROBLEMS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

AYBENİZ YİĞİT

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE DOCTOR OF PHILOSOPHY
IN
INDUSTRIAL ENGINEERING

JULY 2014

Approval of the thesis:

SCHEDULING WITH LATEST ARRIVAL CONSOLIDATION IN SERVICE NETWORK DESIGN PROBLEMS

submitted by **AYBENİZ YİĞİT** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy in Industrial Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen _______________________
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. Murat Köksalan _______________________
Head of Department, **Industrial Engineering**

Assoc. Prof. Dr. Canan Sepil _______________________
Supervisor, **Industrial Engineering Dept., METU**

Examining Committee Members:

Assoc. Prof. Dr. Sinan Gürel _______________________
Industrial Engineering Dept., METU

Assoc. Prof. Dr. Canan Sepil _______________________
Industrial Engineering Dept., METU

Assist. Prof. Dr. Cem İyigün _______________________
Industrial Engineering Dept., METU

Assist. Prof. Dr. Hediye Tüydeş Yaman _______________________
Civil Engineering Dept., METU

Assist. Prof. Dr. Mustafa Alp Ertem _______________________
Industrial Engineering Dept., Çankaya University

Date: 24 July 2014

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: Aybeniz, Yiğit

Signature :

ABSTRACT

SCHEDULING WITH LATEST ARRIVAL CONSOLIDATION IN SERVICE NETWORK DESIGN PROBLEMS

Yiğit, Aybeniz
PhD, Department of Industrial Engineering
Supervisor: Assoc. Prof. Dr. Canan Sepil

July 2014, 323 pages

In this thesis, we consider the service network design problem of a ground-transportation based delivery system, in which routes of demands of commodities between any origin-destination pair are determined. A commodity can be sent from its origin to its destination through direct delivery, however such a routing would not effectively make use of the vehicles used for transportation. To benefit from economies of scale, a networking policy based on consolidation is generally applied in service networks. We consider freight-consolidation, in which different commodities are consolidated to be transported using common vehicles and consolidation operations are performed at stations, some of which are chosen as terminals. Nonsimultaneous arrival of commodities necessitates waiting times at stations/terminals. The latest arrival consolidation in service network design problem is then, a minimax model that considers the delays at terminals and focuses on minimization of the arrival time of the last arrived commodity to its destination. For the solution of the model, we present exact and heuristic solution procedures. We develop a tailored Generalized Benders Decomposition algorithm and to address larger size networks, we develop a Large Neighborhood Search based algorithm. We show the effectiveness of the heuristic solution procedure by performing extensive computational experiments. In the constructed service network, each direct ride between stations is assumed to be performed by the same

vehicle. Extending this assumption to allow multiple vehicles and using event-activity-network representation, we develop a delay management model for service networks that apply latest arrival consolidation.

Keywords: Consolidated multicommodity network design, latest arrival, delay management, Generalized Benders Decomposition, Large Neighborhood Search.

ÖZ

SERVİS AĞI TASARIMI PROBLEMLERİNDE EN GEÇ ULAŞANLARA GÖRE BİRLEŞTİRME İLE ZAMAN ÇİZELGELEMESİ

Yiğit, Aybeniz
Doktora, Endüstri Mühendisliği Bölümü
Tez Yöneticisi : Doç. Dr. Canan Sepil

Temmuz 2014, 323 Sayfa

Bu tezde, kara taşımacılığı temelli bir dağıtım sisteminde bulunan ürün taleplerinin başlangıç noktaları ile varış noktaları arasındaki rotalarının belirlendiği bir servis ağı tasarımı problemi üzerinde çalışılmıştır. Bir ürün başlangıç noktasından varış noktasına direk taşıma yönetimi ile gönderilebilir. Fakat bu şekildeki bir rotalama taşıma araçlarının verimsiz kullanımına neden olmaktadır. Ölçek ekonomisinden faydalanmak için, bir ağ tasarımı yöntemi olan ürünlerin birleştirilmesi politikası servis ağı tasarımı problemlerinde yaygın olarak kullanılmaktadır. Biz de, farklı ürünlerin birleştirilip ortak taşıma araçları ile dağıtıldıkları ve ürün birleştirme işlemlerinin istasyonlarda gerçekleştirildiği bir ürün birleştirme yapısını temel alan bir model oluşturduk. Uyguladığımız birleştirme yönteminde istasyonların bir kısmı terminal olarak seçilmektedir. Ürünlerin istasyonlara/terminallere eş zamanlı ulaşmamaları beklemelere neden olmaktadır. Servis ağı tasarımı problemlerindeki en geç ulaşanlara göre birleştirme, terminallerdeki gecikmelere odaklanan ve varış noktasına en geç ulaşan ürünün ulaşma vaktini eniyileyen bir modeldir. Bu modelin çözümü için geliştirilmiş olan kesin ve sezgisel yöntemler sunulacaktır. Kesin çözüm yöntemi olarak bir Genelleştirilmiş Benders Ayrışım algoritması geliştirilmiştir. Daha büyük servis ağları için de Geniş Komşuluk Esaslı Arama sezgisel yönetimi kullanılarak bir algoritma geliştirilmiştir. Sezgisel çözüm

yöntemimizin etkisi geniş bir kapsamda gerçekleştirdiğimiz hesaplamalı deneyler ile gösterilmiştir. Modellediğimiz servis ağında, istasyonlar arasındaki direk transferlerin aynı taşıma aracı ile gerçekleştirildiği varsayılmıştır. Bu varsayımın kapsamını direk transferlerin birden fazla taşıma aracı ile gerçekleştirilebileceği şekilde genişleterek ve olay-etkinlik-ağı gösterimini kullanarak, en geç ulaşanlara göre birleştirme yönetiminin uygulandığı servis ağları için bir gecikme yönetim modelini de geliştirdik.

Anahtar Kelimeler: Birleştirilmiş çoklu ürün taşıma ağ tasarımı, en geç ulaşan ürün, gecikme yönetimi, Genelleştirilmiş Benders Ayrışım yöntemi, Geniş Komşuluk Esaslı Arama sezgisel yöntemi.

To My Mother İclâl Yiğit

ACKNOWLEDGMENTS

Throughout my Ph.D. studies, many people have supported me through various ways in my academic and personal life. I am grateful to all of them for all their efforts and best wishes.

First of all, I would like to express my gratitude to my thesis advisor Assoc. Prof. Dr. Canan Sepil for her precious guidance, supervision, and advice. I am indebted to her for her support, motivation, and patience. I also would like to express my deepest gratitude to Assoc. Prof. Dr. Sinan Gürel for his valuable guidance and motivation throughout my thesis.

I would like to thank to my thesis committee members Assoc. Prof. Dr. Sinan Gürel and Assist. Prof. Dr. Hediye Tüydeş Yaman for their precious comments and suggestions during my thesis period. I also would like to thank to Assist. Prof. Dr. Cem İyigün and Assist. Prof. Dr. Mustafa Alp Ertem for participating into my jury and providing their valuable comments.

I am grateful to ASELSAN Inc. for institutionally supporting and encouraging my Ph.D. studies.

I would like to express my sincere gratitude to my manager İsmet ATALAR, who encouraged me to start Ph.D. and provide constant motivation and support throughout my Ph.D. studies.

I am grateful to my manager Dr. Hüseyin Yavuz, who has always provided his precious guidance, support, and motivation at all steps of my Ph.D. studies. I would like to express my gratitude to my manager Seyit Yıldırım for all his supports and motivation.

I would like to express my special thanks to my dear colleague İnci Yüksel Ergün and my dear friends Engin Topan, İpek Topan, and Uğur Arıkan for their support and motivation.

I am dedicating this thesis to my mother İclâl Yiğit, who has always been with me from my first step into my education life and trying to do her best to support me. I am grateful to her for all her devotion to me. I also thank to my father Ekrem Yiğit, who encouraged me to start Ph.D., and to my brothers Zafer Yiğit and Tamer Yiğit and to all members of Yiğit family for their support and love.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

ABBREVIATIONS	
BD	Benders Decomposition
CAB	Civil Aeronautics Board
CCB	Covering Cut Bundle
CB	Combinatorial Benders
CP	Consolidated Path
CMND	Fixed Charge Capacitated Multicommodity Network Design
DLA	Delay Management Problem in Service Network Design Problems that Apply Latest Arrival Consolidation
D-SP	The Dual of the Sub Problem in Benders and Generalized Benders Decomposition
EAN	Event Activity Network
GA	Genetic Algorithm
GBD	Generalized Benders Decomposition
GOS	Global Optimum Search
HDP	High Density Pareto
LA-CMNDP	Latest Arrival Consolidated Multicommodity Network Design Problem
LNS	Large Neighborhood Search
LNS-LA	Large Neighborhood Search algorithm for LA-CMNDP
LTL	Less-than-truckload
MDC	Maximum Density Cut
MFS	Maximum Feasible Subsystem
MIP	Mixed Integer Programming
MIS	Minimal Infeasible Subsystems

MINLP	Mixed Integer Nonlinear Programming
NLP	Nonlinear Programming
RMP	Relaxed Master Problem in BD and GBD Procedures
SLApHLP	Single Allocation Latest Arrival p-Hub Location Problem
SND	Service Network Design
SP	The Sub Problem in BD and GBD Procedures
TL	Truckload
TU	Totally Unimodular
UMApHCP	Uncapacitated Multiple Allocation p-Hub Center Problem
VDNS	Variable Depth Neighborhood Search
VLSN	Very Large Scale Neighborhood
VNS	Variable Neighborhood Search

CHAPTER 1

INTRODUCTION

Freight transportation has always been a vital part in the economy of countries and its importance is getting higher due to globalization and increased transport needs. Besides transport expenditures, transport sector involves large scale of investment decisions related to infrastructures and technologies that constitute an important part in national development plans. To be competitive in freight transportation and to maintain sustainability, designing a transport system that provides high quality service with lower costs becomes a critical issue. This purpose necessitates the planning and the efficient use of transport resources including vehicles, facilities, equipments, and crew. The integrated planning of these issues belongs to tactical level and is handled through Service Network Design (SND) problems.

1.1 Motivation

SND problems mainly deal with the optimization of transfers occurring between many node pairs and design a service network together with commodity routes utilizing a multi commodity flow structure. When there are multiple commodities, the resources (vehicles, crew, handling operations) and the infrastructure (capacities of facilities, handling equipments) of the network must be shared among these commodities. When direct deliveries are assumed, each commodity would be sent through the shortest delivery path from the origin to the destination of the commodity. To benefit from economies of scale, usually consolidation is applied. In Figure 1 (a) direct delivery and in Figure 1 (b) consolidation strategies are presented.

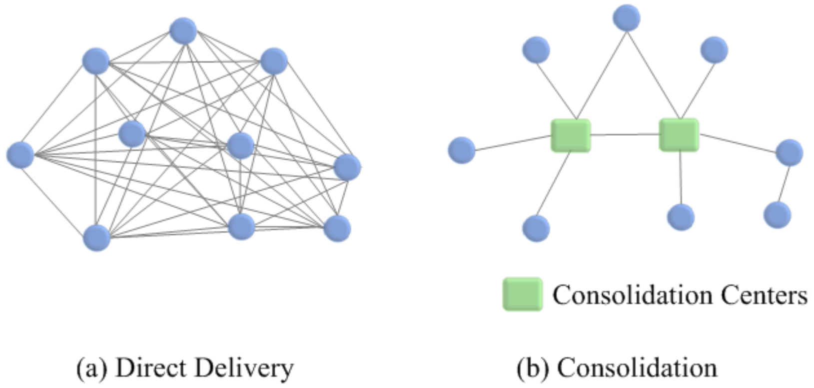


Figure 1. Direct Delivery vs Consolidation

Compared to direct delivery strategy, a commodity may travel a longer route by applying consolidation but large amount of savings in transportation costs and network design costs (terminal location, infrastructure construction and improvement) are incurred. A consolidation structure can be defined by a set of terminals, a set of stations, inter and intra transfer strategies between each set, and the conditions on the elements of a route including the number of intermediate stops and the allowable number of terminals or stations a route can have.

The main decisions in SND problems include;

- freight consolidation,
- terminal congestion,
- design of services,
- design of routes,
- service levels,
- empty vehicle repositioning, (Crainic and Laporte (1997), Pedersen (2005)).

Freight consolidation is applied to benefit from economies of scale resulting from transportation of large quantities in a more economic way. In consolidated transportation, same vehicle may be used for the transportation of freight for different customers, whose origins and destinations can also be different. By consolidating freight, the vehicle capacity can be fully utilized, which lowers fixed costs related to vehicles, whereas variable costs increase due to detours and transshipments, Wieberneit (2008).

Terminals are the major consolidation centers where arriving vehicles are unloaded, commodities are consolidated and loaded to departing vehicles. Arrival patterns and the amount of incoming flows determine *terminal congestion*. Terminals have limited capacities (handling equipments, consolidation infrastructure, personnel) for performing the main consolidation operations. Thereby, to prevent terminal congestion and the resulting delays, issues related to consolidation structures, terminal operations, and routing strategies should be focused on.

SND problems design *services* and *routes* for commodities and vehicles. A service can be defined by vehicle types, speed and capacity of the vehicles, and the selection of the transportation mode (ground, rail, air, maritime, intermodal). The vehicle routes are defined by originating and ending terminals, and the intermediate stops. The commodity routes consist of a sequence of stations and the vehicles that perform the transportation between each station, Crainic and Laporte (1997).

All routes have to satisfy *service levels* that are determined by delivery times or time windows. Delivery time requirements state the total allowable time for performing all deliveries within the transportation system. Time windows state the earliest and the latest times for initiating and terminating the services at stations. Considering the priorities of different commodity classes, it is possible to define several types of service levels within the same planning system.

Due to imbalances in commodity flows, differences in supply and demand of vehicles occur at terminals. These differences can be corrected by *repositioning empty vehicles* in such a way to satisfy the demand of the next planning period, Crainic (2000).

Depending on the characteristics of the application areas, different consolidation types are applied in SND problems. Cargo delivery network design studies generally apply a hub-and-spoke consolidation structure and the related studies are in the domain of hub location problems. Hubs are the consolidation and transfer centers. The non-hub nodes are usually called as spokes. Hub location problems locate hubs and determine the allocation of spokes to hubs. The basic consolidation structure applied by hub location problems assumes that in the resulting hub-and-spoke network;

- every hub pair is connected to each other,
- no direct link is allowed between spoke pairs,
- commodities are delivered at least visiting one hub and at most two hubs.

There are several studies relaxing these assumptions allowing more than two hubs (Campbell et al. (2005a,b)), transfers between spokes (Kuby and Gray (1993), Yaman et al. (2007)), and not connecting every hub pair in the constructed hub network, which is defined as incomplete hub network design problem by Alumur (2009). In incomplete hub network design problem of Alumur (2009), the transportation network is assumed to be fully connected but the constructed hub network is allowed to be incomplete.

Hub location problems assume a fully connected transportation network, in which a direct link can be established between each node pair. However, this assumption may not always hold for other transportation systems like railroad planning problems that operate on physical networks in which the connections are determined by railways. This main difference affects the formulation structures of both problems. In hub location models, assignment type variables, in which a node is assigned to another node is used, whereas railroad planning studies use multicommodity flow variables to model the commodity routes.

The freight consolidation in railways is modeled by railroad blocking problem. In railroad transportation, the terminals of railways are called as yards, where incoming flows are reclassified and assembled into outgoing flows. In railroad blocking problem, to reduce the intermediate handlings taking place at yards, commodities are grouped together to form *blocks*. Once a block is formed, it is not reclassified until it reaches its destination. By this way, classification costs and delays are tried to be kept at minimum.

When consolidation is applied in service networks, delays occur at certain nodes due to nonsimultaneous arrivals of commodities. These delays may constitute an important part in total delivery times and should be focused for improving service time and service quality aspects. In railroad blocking studies the effect of waiting times, congestion delays and queues for consolidation operations at yards are studied and generally modeled implicitly such as incorporating into main models related delay costs, Bodin et al. (1980), Crainic et al. (1984), Marin and Salmeron (1996a) or modeling delays as links in the network, Haghani (1989), Zhu (2010).

Classical hub location models dealing with service levels only consider travel times. Kara and Tansel (2001) is the first to consider the transient times spent at hubs by determining the departure time of a vehicle according to the latest arriving cargo that will be loaded to that vehicle. Authors call this problem as latest arrival hub location problem and present a novel model that minimizes the maximum latest arrival time. The concept of latest arrival is incorporated into different hub location problems by using the minimax objective function. To the best of our knowledge, all latest arrival studies on hub location problems assumes single allocation of non-hub nodes to hubs and requires full-cross-traffic assumption for the latest arrival time calculations.

1.2 Scope of the Work

In this thesis, we study the latest arrival concept in freight transportation with the aim of enhancing its application area to broader class of problems in tactical and operational levels. For this purpose, firstly we model the latest arrival concept on incomplete physical networks, in which the nodes are allowed to be assigned to multiple other nodes directly for sending and receiving flow, and the latest arrival time calculations does not necessitate full-cross-traffic assumption. This is the first time in literature that such a formulation is introduced for service network design problems. We name this problem as latest arrival consolidated multicommodity network design problem (LA-CMNDP), which belongs to tactical level.

In LA-CMNDP, consolidation is applied at some points of the transportation network in a similar way to the blocking of railroad planning models. Individual commodities that may share same routes throughout their delivery paths are grouped together. We will refer to these consolidated commodities as blocks as in the railroad literature throughout the thesis. The transportation between each node pair defines a block including many commodities with different origins and destinations. In Figure 2, a sample service network consisting of eight nodes and two commodities is presented, and arcs $\{(1,3),(2,3),(3,6),(6,8)\}$ represent the blocks.

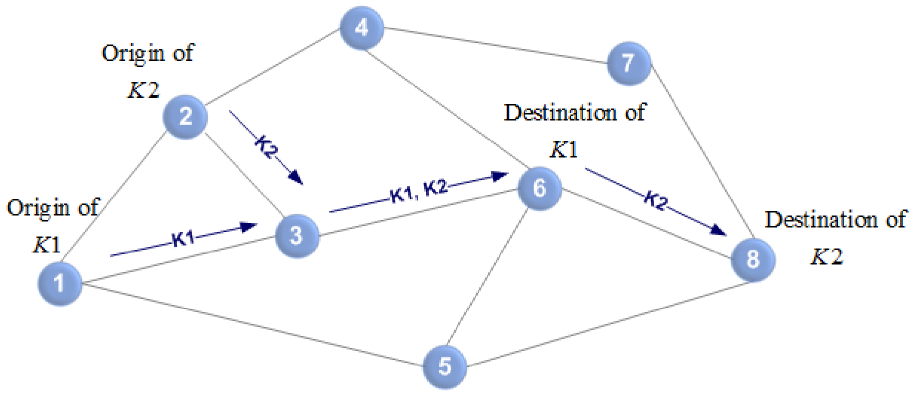


Figure 2. Blocks in a Service Network

Since each arc of the service network defines a block and behaves as a single entity, a block can depart from a node only after all of its commodities arrive to that node. Therefore, the earliest departure time from a node for a block is determined by the latest arriving commodity that is to be placed into that block. This property of the service network relates with the latest arrival concept of Kara and Tansel (2001) in hub location problems. Throughout the thesis, we refer this consolidation structure as the latest arrival consolidation structure.

LA-CMNDP has a minimax objective function that minimizes the maximum latest arrival time to destinations. Compared to minimum cost models that focus on system wide transportation costs, minimax models aim to establish service guarantees for each customer. As shipments get smaller like the case in motor carriers emphasized by Campbell (2005), the freight become more specialized and the delivery times become more critical. Therefore, models focusing on service quality aspects are preferred. Another application area of minimax models is emergency planning problems, in which the main objective is to cover all nodes of a network in the minimum possible time.

After a service network is designed by LA-CMNDP, the next planning problem belongs to operational level and it consists of determining the detailed vehicle schedules/timetables and the assignment of commodity paths to vehicle routes. From the outputs of LA-CMNDP, the minimum number of vehicles to perform all deliveries can be determined. Due to latest arrival consolidation structure, at some nodes of the service network the departing vehicles have to wait for the late arrivals. These waiting times cause delays in arrival times to destinations of all commodities within the transportation system. In case of a late arriving commodity, there are two options for the departing vehicles either to wait or to depart. The depart option reduces the delays of commodities but can increase the number of vehicles required to perform all deliveries. On the other hand, the wait option does not increase the vehicle numbers and does not reduce the delays.

A similar problem that determines the wait-depart decisions of vehicles is studied in public transportation literature with the name of delay management problem. In case of a delay in the arrival time of a vehicle, the connecting vehicles either wait or depart. If the connecting vehicle waits for the delayed vehicle, then the passengers within the connecting vehicle and the passengers that will get on this vehicle later will all be delayed. On the other hand, if the connecting vehicle departs without waiting, then the passengers within the delayed vehicle will miss their connection and have to wait for the next departing vehicle. The planning of these wait-depart decisions of all vehicles within a public transportation system and the required updates in the timetable of the vehicles are modeled as delay management problem by Schöbel (2001). The main delay management problem variants are studied comprehensively in Schöbel (2006).

We present a model for the delay management problem arising in service network design problems that apply latest arrival consolidation (DLA). Our model determines the wait-depart decisions of all vehicles, the detailed schedules of the vehicles, and the assignment of commodity paths to vehicle routes by minimizing the total delay of all commodities at all stations. We use event-activity-network representation of project networks to model DLA.

1.3 Outline of the Thesis

In Chapter 2, we present the main network optimization problems in freight transportation with a taxonomy of models within the decision hierarchy: strategic, tactical, and operational planning levels and we state the conceptual framework of this thesis within freight transportation.

For the sake of completeness, all literature reviews are presented in the same chapter. Literature reviews regarding (i) the studies in freight transportation that are relevant to LA-CMNDP, (ii) the exact solution procedures we apply to LA-CMNDP, and (iii) the main delay management problems are presented in Chapter 3.

In Chapter 4, we formally describe LA-CMNDP stating the main properties and present a mathematical model along with the derivation of arrival and departure times. We develop exact and heuristic solution procedures for LA-CMNDP. In the exact solution procedures, we apply Generalize Benders Decomposition (GBD) and propose eight GBD algorithms. In Chapter 5, we present our GBD algorithms with their main properties and report their computational performances. In the heuristic solution procedure, we develop a Large Neighborhood Search (LNS) algorithm that outperforms all GBD algorithms and the exact solution program CPLEX. The developed LNS algorithm solves larger size networks in reasonable computational times. In Chapter 6, we provide the details of our heuristic algorithm and show its effectiveness by reporting extensive computational experiments.

In Chapter 7, we present the delay management model for service network design problems that apply latest arrival consolidation, and present a discussion on possible solution procedures.

Chapter 8 concludes the thesis summarizing our main findings/results and stating the contributions of this thesis and possible future study issues.

CHAPTER 2

NETWORK OPTIMIZATION PROBLEMS IN FREIGHT TRANSPORTATION

Network optimization problems are widely studied in freight transportation to model the planning problems arising in each decision level: strategic, tactical, and operational. Strategic planning problems involve large capital investments to meet the long-term requirements of the transport system and they are generally defined as logistics system design. Tactical planning problems deal with the effective use of resources in an integrated manner and determine the main operating policies. Service Network Design problems establish the main problem class in tactical level. Operational planning problems focus on daily activities and involve constructing detailed plans.

In this chapter, we present the main problems in each decision level of freight transportation in Section 2.1. We present a taxonomy of the network optimization problems in freight transportation in Section 2.2. Considering the characteristics of different transportation modes (ground, rail, air, maritime, and intermodal), we provide the most prominent problems of each decision level according to each transportation mode in Section 2.3. In Section 2.4, we present the conceptual framework of this thesis in freight transportation stating our main modeling concerns. In Section 2.5, we present the relevance of this thesis to other optimization problems.

2.1 Planning Levels

Designing and managing a transportation system involves many interrelated decision problems that have to be modeled within a hierarchical structure by considering the input-output relationships between each hierarchy. The decisions that require high level of management and that affect the operations of the transport system for a long period of time belong to strategic level. The strategic level problems deal with the design and the development of a transport system. The facility location decisions involving where to locate facilities, at what capacity, in which physical layout; the acquisition of major resources such as vehicles, fleet, equipments; the installation of main infrastructures like rail lines, highways; and the decisions related to long term planning of all these entities such as capacity expansion and abandonment issues are in the domain of strategic level problems.

Tactical level models determine the main operating policies of the transport system by focusing on the effective use of all resources and providing high quality service to customers. The major class of problems in tactical level are Service Network Design problems, in which the decisions regarding the service routes, frequency, and schedules; the capacity, type, and speed of the vehicles; the type of the transportation modes are given. In addition to designing the service network, to determine the vehicle routes and to reposition the empty vehicles after they perform the scheduled deliveries are also belong to tactical level.

The detailed planning of the transport system focusing on the efficient use of resources to perform the daily operations is modeled at operational level. The allocation of crew to vehicles, constructing detailed timetables of vehicles, loading plans, the detailed planning of vehicle movements such as dispatching plans, and all planning activities to ensure the continuity of the system such as maintenance plans are in the domain of operational level problems.

2.2 Taxonomy of Network Optimization Models in Freight Transportation

The planning problems ranging from strategic level to operational level are generally modeled as network flow/design problems. Considering the main network optimization models in freight transportation, we develop a taxonomy basically considering the classification structures of Magnanti and Wong (1984) and Crainic and Laporte (1997). We present our taxonomy in Figure 3.

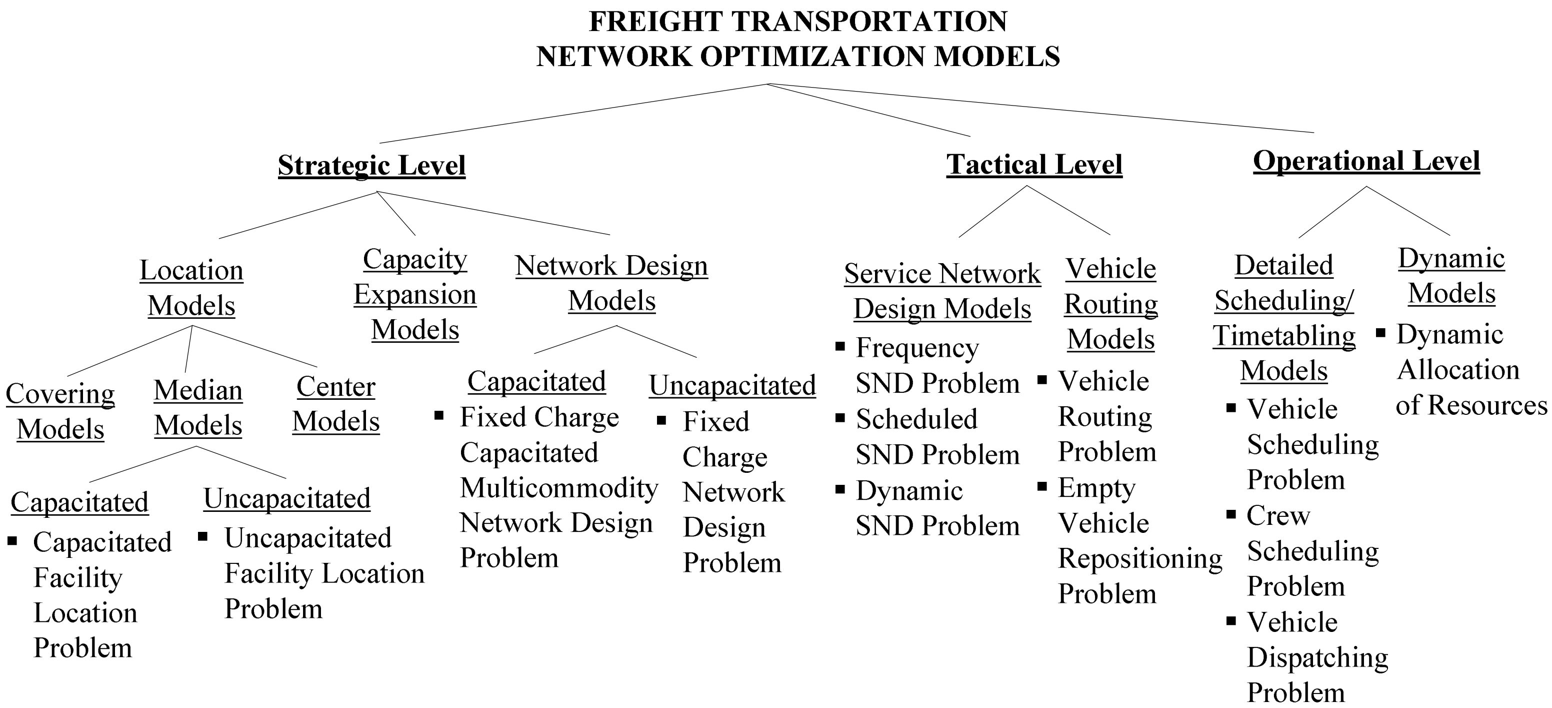


Figure 3. Taxonomy of Network Optimization Models in Freight Transportation

Location models establish the major problem class in strategic planning level, due to related high investment costs and its long-term impact on decision problems in all planning levels. The locations of facilities state the major limitations on the operations of the transportation system. For this reason, the location decisions are given considering the transfers between node pairs, distribution and variability of demand patterns throughout the transportation network. Past term data as well as forecasts regarding projections to future periods are generally used in these models.

Location models can be categorized into three classes, Crainic and Laporte (1997):

(i) *Covering Models:* The main objective is to locate the facilities so as to cover all other nodes of the network without exceeding predetermined travel times/distances.

(ii) *Median Models:* A predetermined number of facilities are located to minimize total weighted travel distances.

(iii) *Center Models:* A predetermined number of facilities are located to minimize the maximum distance between a facility and a node or to the maximum travel time between the node pairs.

Hub location models establish an important body of knowledge in location literature and have direct applications regarding models stated in (i)-(iii).

Network design models cover a wide range of problems including a class of problems (i.e. minimum spanning tree, shortest path, travelling salesman) that establish the basic building blocks of large-scale network optimization models. We take into our taxonomy the network design models that can be applied with more general purposes to design a freight transportation network, i.e. fixed charge network design models, in which links are established between selected node pairs to perform the transfers. Since the capacitated version establishes the major problem class in multicommodity network design models, we provide the formulation and the literature review of related applications in detail in Section 3.1.

The development and the maintenance of the facilities, the transportation network, and the infrastructure are also in the domain of strategic level problems. By analyzing the increase or decrease trends in transportation requirements with respect to seasonal periods and/or customer regions the capacity expansion or abandonment decisions of major facilities and the psychical network have to be modeled. For more information about capacity expansion models, the survey paper presented by Luss (1982) can be referenced. Luss (1982) states the basic decisions in capacity expansion models as the size, time, and the location of expansion.

Tactical level problems include SND models and vehicle routing models. Service schedules can be modeled either decision variables of the SND model or can be determined by modeling the SND problem using time-space network representation (dynamic SND model), in which each node represents a station at a specific time.

SND problems arise in different freight network design contexts including ground, rail, air, maritime, and intermodal transportations. Main ground SND studies are less-than-truckload (LTL), truckload (TL), and motor carrier network design problems. In LTL transportation, the capacity of a vehicle is shared among different commodities. TL transportation generally operates with full vehicle loads and direct transports from origins to destinations. Motor carriers operate like LTL transportation but transport small parcels (postal deliveries and cargos). Compared to LTL transportation, delivery times become more important in motor carrier transportation, Campbell (2005).

Constructing vehicle routes to perform the services and after they perform the deliveries at the end of the planning period repositioning the empty vehicles are studied at tactical level. Same problems can also be modeled at operational level to determine the detailed schedules and vehicle movements. Whereas the planning concerns of tactical level is to fully utilize the vehicle capacities by minimizing the routing costs.

2.3 Freight Transportation Problems with Respect to Different Transportation Modes

Generally, most of freight transportation planning problems has direct applications in all transportation modes consisting of ground, rail, air, maritime, and intermodal. Intermodal transportation generally implies the transportation of freight/passengers using more than one transportation mode. Many definitions of intermodal transportation can be found in literature. Bontekoning et al. (2004) present 18 definitions ranging from general ones to specific ones. The definition that is admitted by European Conference of Ministers of Transport, in 1993 is

- "*The movement of goods in one and the same loading unit or vehicle, which uses successively several modes of transportation without handling the goods themselves in changing modes*".

This definition is the one that is used most commonly.

In intermodal transportation studies, multimodal transportation term is also used as an alternative definition, whereas they are not same. The main difference between two modes is highlighted by Pedersen (2005) stating that multimodal transportation means using more than one transportation mode but interoperability between different modes is not required. However, intermodal transportation covers all planning issues regarding the transfers between different modes.

In Table 1, for each transportation mode we list the most prominent planning problems with their specific and well-known definitions corresponding to each decision level. Since intermodal transportation covers a broader range of problems than multimodal transportation and the two terms are generally used interchangeably implying the intermodality, we include in Table 1 only intermodal transportation.

The information stated in Table 1 is constructed basically considering

- the classification schemes of freight transportation models, the reviews and the problems provided by Magnanti and Wong (1984), Crainic and Laporte (1997), Crainic (2000), Ghiani et al. (2004), Crainic and Kim (2007),

- the studies on railroad transportation provided by Crainic et al. (1984), Ahuja et al. (2005), Jha et al (2008), Liu et al. (2008), Zhu (2010),
- the studies on air transportation provided by Barnhart and Schenur (1996), Büedenbender et al. (2000), Barnhart et al. (2002), Armacost et al. (2004),
- the studies on maritime transportation provided by Lai and Lo (2004), Shintani et al. (2007), Imai et al. (2009), Gelareh et al. (2010), and
- the studies on intermodal and multimodal transportation provided by Kim et al. (1999), Jansen et al. (2004), Pedersen (2005), Crainic and Kim (2007).

Table 1. The Main Freight Transportation Problems of Each Decision Level with Respect to Transportation Modes

Transportation Mode	Strategic Level Problems	Tactical Level Problems	Operational Level Problems
Ground	- Acquisition of vehicles, fleet - Investments regarding infrastructure (highways, bridges) - Facility (terminals, plants, emergency centers) location - Facility layout and design	- Determine the station pairs to provide direct service - Determine the frequency of each service - Determine vehicle types (speed, capacity) used in each service - Determine freight routes (assignments to services/vehicles) - Reposition the empty vehicles	- Detailed vehicle scheduling to meet time windows/ delivery time requirements - Crew scheduling - Dynamic driver assignment
Rail	- Acquisition of (fast) trains, fleet - Investments regarding infrastructure (locating fast train lines, expensive equipments such as yard cranes used in yard operations) - Yard location - Capacity expansion (abandonment) of rail lines, yards	- Railroad blocking problem - Train scheduling/routing problem (determine the number of trains to run with the frequencies, the originating-ending and the intermediate stations of each train) - Assigning blocks to trains - Allocation of consolidation operations to yards	- Detailed train timetabling - Crew scheduling - Locomotive scheduling - Train dispatching problem - Empty railcar movement
Air	- Acquisition of aircrafts (airplanes, jets), fleet - Investments regarding infrastructure (expensive equipments-vehicles used in airports) - Airport location - Airport layout and design	- Determine which city pairs to provide direct flight - Determine aircraft types (maximum flying range, speed, capacity) - Determine aircraft routes/flights (determine delivery & pickup air-stops) - Fleet composition (determine fleet size) - Assignment of freight (time critical cargo) to flights	- Detailed flight scheduling (to meet time windows) - Assignment of fleet to flights - Constructing maintenance plans of aircrafts

Table 1. (continued)

Transportation Mode	Strategic Level Problems	Tactical Level Problems	Operational Level Problems
Maritime	- Acquisition of see vessels (ferries, ships, liner ships) - Investments regarding infrastructure (expensive equipments/platforms such as quay cranes used in ports, number of berths to be installed at ports) - Port/quay layout and design (location of berths)	- See vessel routing and scheduling of direct and multi-stop services - Determine the types of see vessels (capacity, speed) - See vessel scheduling - Assignment of freight to see vessels - Fleet composition (determine fleet size) - Empty container repositioning	- Crew scheduling - Detailed timetabling of service schedules - Constructing maintenance plans of see vessels
Intermodal	- Investments regarding infrastructure (expensive equipments and platforms such as quay/yard cranes to be installed at intermodal terminals, enlarging railway network to reach intermodal terminals) - Intermodal terminal (ports/ yards/ airports) location - Intermodal terminal layout and design (i.e. integrated design of the sea-side and the land-side of ports, location of gateways through which the transfers between different modes take place at airports) - Capacity expansion (abandonment) of intermodal terminals/platforms	- Determining intermodal service schedules (i.e. intermodal freight train scheduling) - Determine the types of vehicles (capacity, speed) - Establishing operating policies of intermodal terminals (i.e. consolidation strategies) - Assignment of freight to intermodal services	- Detailed scheduling of intermodal services (i.e. to meet time windows for time critical cargo/letter mail that is delivered via air-truck or air-train multimodal chain) - Container fleet management (empty container repositioning) - Berth scheduling (determining time and position of the arriving vehicles to each berth) - Quay crane scheduling (the detailed timetabling of quay cranes to perform load-unload operations from ships) - Stowage planning (determining the loading plan and placement of containers into the ships)

2.4 Conceptual Framework

In this section, we present conceptually where this thesis fits into the freight transportation. Since our model has similarities to hub location and railroad blocking problems, we state the basic issues in the consolidation structures and the objectives of both problems. Then, we indicate which properties and attributes of these models are similar to our main modeling concerns. Also, we state the differences of our model. We present the conceptual framework of this thesis in Figure 4.

Hub location problems are in the domain of location models class and railroad blocking problems are in the domain of service network design problems in tactical level. Two different modeling structures are generally applied in location and network design models. In location models, the transfers between node pairs are generally defined by assignment type decision variables:

- $x_{ij} \geq 0$, denoting the fraction of demand of node i served by the facility located at node j, or
- $x_{ij} \in \{0,1\}$, where x_{ij} equals to 1 if node i is directly assigned to node j for sending and receiving flow to other nodes and equals to 0 otherwise.

In network design models at strategic and tactical levels, the transfers between node pairs are generally modeled through multicommodity flow variables:

- $x_{ij}^{k} \geq 0$, denoting the fraction of demand of node i served by the facility located at node j, or
- $x_{ij}^{k} \in \{0,1\}$, where x_{ij}^{k} equals to 1 if commodity k flows through node i to node j or equals to 0 otherwise.

In this thesis, we focus on time efficient routing of commodities. We consider a commodity as homogenous freight that may either refer to cargo, containerized goods, letter mails, or small parcels. Each commodity is defined by an origin and a destination pair (O-D pair). Namely, all freight that is destined from the same O-D pair is treated as a single commodity.

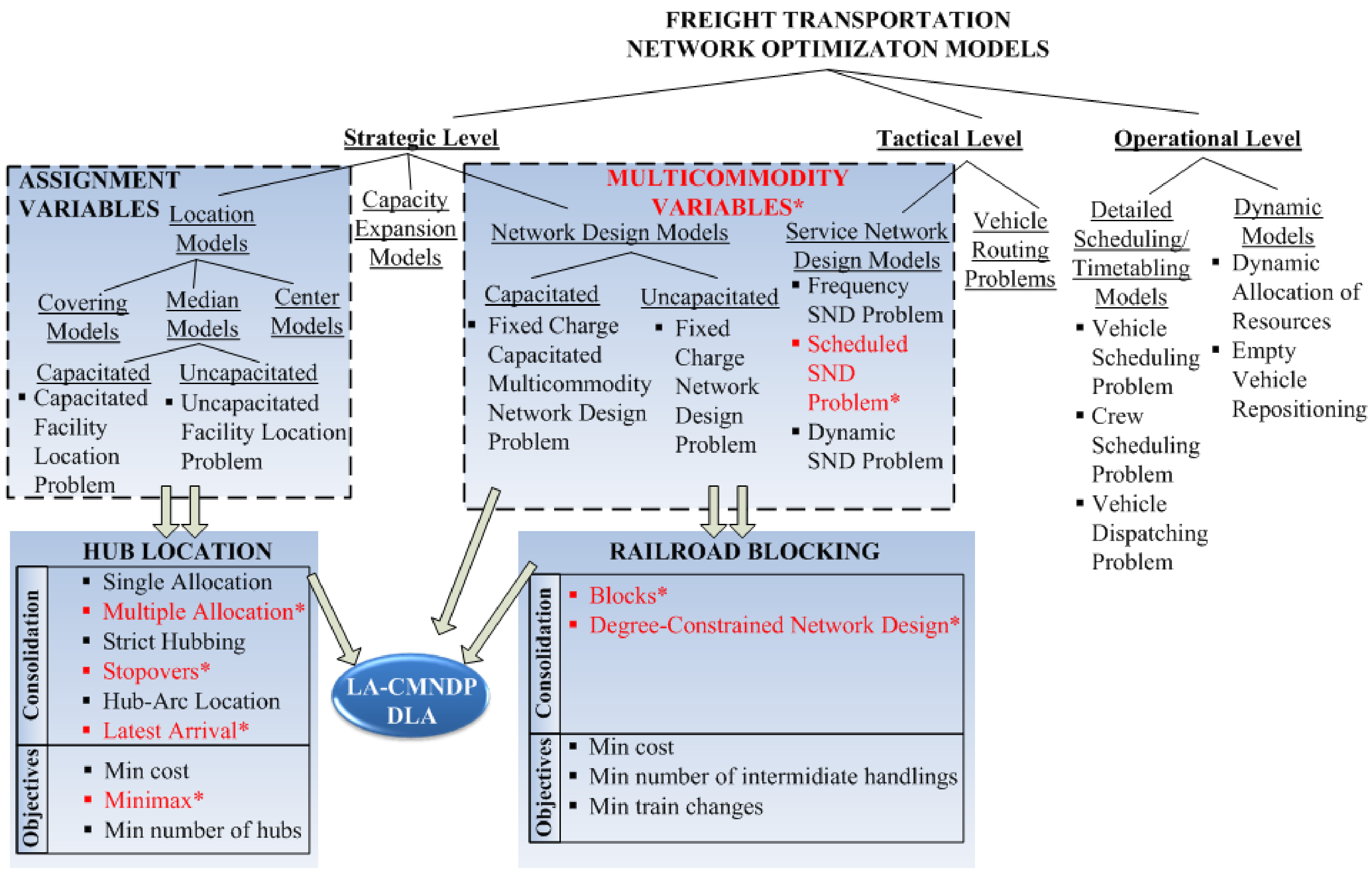


Figure 4. The Conceptual Framework of This Thesis in Freight Transportation

In Figure 4, we highlight the main attributes of our modeling properties by red ink and with a star. We indicate where they belong into freight transportation taxonomy and how they relate with hub location and railroad blocking problems. These attributes can be listed as follows:

- terminals are located to increase the level of consolidation throughout the service network,
- latest arrival concept of Kara and Tansel (2001) is applied,
- multiple allocation of nodes to other nodes/terminals are allowed,
- a similar routing structure to the stopovers in hub location problems is applied, in which a route can contain multiple stopping points to reach a terminal and in our modeling structure the routes that do not visit a terminal are also allowed,
- the objective function minimizes the maximum latest arrival time to destinations,
- commodity routes are modeled using multicommodity variables,
- a scheduled service network is designed,
- each arc in the service network defines a block as in the railroad blocking problems,
- there are limits on the number of incoming blocks to each node (degree-constrained network design).

The main differences of our model from the latest arrival hub location problems are that our model

- does not need full-cross-traffic assumption for the latest arrival time calculations,
- allows multiple allocation,
- can be applied to incomplete psychical networks as well as fully connected networks.

The main differences of our model from the railroad blocking problems are:

- the degree-constraints are defined on incoming arcs,
- the handling capacities of nodes/terminals are modeled implicitly through in-degree-constraints,

- the physical links are defined as uncapacitated,
- a minimax type objective function is used and to keep balance between the maximum latest arrival time and total travel times, limits are defined for the travel time of each commodity.

2.5 Relevance to Other Optimization Problems

In addition to freight transportation, the latest arrival consolidation structure we study in this thesis can also be applied to other optimization fields utilizing multicommodity flow network structure and involving delays due to transfers between node pairs of the network. One important application area of the latest arrival consolidation is the packet switching problem in telecommunication network optimization.

In communication networks, the data transmission (i.e. downloading a web page) is performed by first decomposing the whole content into smaller data packets and then transmitting each packet through several routers until it reaches its destination. The packets arriving to each router are stored instantly and then transmitted to another router, Pioro and Medhi (2004). During this transmission process, packets experience delays at routers. The major delays experienced by packets are stated by Kurose and Ross (2000) as nodal processing delay, queuing delay, transmission delay, and propagation delay. Time performance of data transmission is an important field of research in communication networks. Flow control techniques that focus on congestion control, providing fast packet switching, and the real time applications that require high-speed transmissions are studied by Pouzin (1981), Newman (1988), and Aras et al. (1994), respectively.

Since the transmission process of a content can only be completed until all packets of it reach the destination point, the latest arrival consolidation structure can be applied to packet switching problem. The modeling approach of LA-CMNDP can be used to determine the routes of transmissions and the related consolidation structure throughout the communication network for the purpose of minimizing the maximum latest arrival time of all transmissions, thereby the total duration of the transmission.

Also, the modeling structure of our second model DLA can be used to further minimize all delays experienced by the packets at the routers by re-adjusting the consolidation plan. Namely, at some of the routers, transmissions to the next router without waiting the latest arriving packet can be allowed with the purpose of decreasing the system wide delays.

CHAPTER 3

LITERATURE REVIEW

LA-CMNDP applies a consolidation structure that is similar to the blocking of railroad planning problems, utilizes multicommodity flow structure, and determines the arrival and departure times of consolidated blocks considering the latest arrivals. Although LA-CMNDP has features related to different problem domains, since it mainly constructs a service network and determines the schedules of consolidated blocks, LA-CMNDP belongs to the class of Service Network Design (SND) problems. In order to cover all features of LA-CMNDP, in this chapter we present a literature review of multicommodity network design problems in Section 3.1, SND problems in Section 3.2, railroad blocking problems in Section 3.3, and the latest arrival studies in Section 3.4 especially focusing on the ones that are related to LA-CMNDP.

We develop exact and heuristic solution procedures for LA-CMNDP. For the exact solution procedure, we implement Generalized Benders Decomposition procedure, which is a variant of Benders Decomposition procedure. We present the literature reviews of Benders Decomposition and Generalized Benders Decomposition procedures in sections 3.5 and 3.6, respectively. For a technical note on the fundamentals of Benders Decomposition procedure Appendix A can be referenced, and for the fundamentals and the technical details of Generalized Benders Decomposition procedure Section 5.1 and Appendix B can be referenced.

We develop a second model DLA for the delay management problem arising in service networks that apply latest arrival consolidation structure. The details of

DLA is presented in Chapter 7. In Section 3.7, we present the literature review of main delay management problems.

3.1 Multicommodity Network Design Problems

Network design problems arising in different application areas mainly utilize multicommodity flow structure. Fixed charge capacitated multicommodity network design (CMND) problems establish the major problem class of multicommodity network design problems since the CMND formulation represents a generic model for the planning problems in the construction, improvement, and operations of transportation, logistics, production, and telecommunication systems, (Crainic et al. (2000), Crainic and Gendreau (2002), Ghamlouche et al. (2004)).

In CMND problems each commodity $k \in K$ is defined by a demand value v^k and an origin $O(k)$ and a destination $D(k)$ pair. There is a fixed charge FC_{ij} for establishing a link between node pairs (i, j) and a unit routing cost c_{ij}^k for the flow of commodity k on arc (i, j) of the network that have a capacity limit represented by u_{ij}. Flow variables x_{ij}^k denote the flow of commodity k on arc (i, j) and design variables y_{ij} denote whether arc (i, j) exist in the resulting network or not. For a given directed network $G = (N, A)$ consisting of a set of nodes N and a set of arcs A, the arc-based formulation of CMND is as follows.

CMND

Minimize $\displaystyle\sum_{(i,j)\in A} FC_{ij}y_{ij} + \sum_{k\in K}\sum_{(i,j)\in A} c_{ij}^k x_{ij}^k$

subject to

$$\sum_{j\in N} x_{ij}^k - \sum_{j\in N} x_{ji}^k = \begin{cases} v^k & if\ i = O(k) \\ -v^k & if\ i = D(k) \\ 0 & otherwise \end{cases} \quad \forall i \in N, \forall k \in K, \tag{3.1}$$

$$\sum_{k\in K} x_{ij}^k \le u_{ij}y_{ij} \qquad \forall (i,j) \in A \tag{3.2}$$

$$x_{ij}^k \ge 0 \qquad \forall (i,j) \in A,\ \forall k \in K \tag{3.3}$$

$$y_{ij} \in \{0,1\} \qquad \forall (i,j) \in A \tag{3.4}$$

Constraints (3.1) are the flow balance constraints and ensure that the demand v^k of each k is delivered from $O(k)$ to $D(k)$. By constraints (3.2) commodity flows satisfy arc capacities u_{ij}. Majority of the CMND problems study on directed networks considering arc capacities. Whereas there are several studies in which undirected networks are addressed, (Alvarez et al. (2003), Zaleta and Socarras (2004), Alvarez et al. (2005a), Alvarez et al. (2005b)). In these studies, the capacity of an edge $\{i, j\}$ u_{ij} is shared among the commodities flowing in both directions on this edge, which is the case observed in telecommunication network design problems. For undirected networks, constraints (3.2) are replaced by (3.5).

$$\sum_{k \in K} (x_{ij}^k + x_{ji}^k) \leq u_{ij} y_{ij} \qquad \forall \{i, j\} \in E \tag{3.5}$$

CMND problems mostly assume that each commodity can be splittable to multiple paths and model x_{ij}^k variables as continuous. Yaghini and Kazamzadeh (2012) study the unsplittable CMND and present a model with binary flow variables. Since the splittable CMND problems are NP-hard, unsplittable CMND problems are even harder due to the increased number of binary variables. Frangioni and Gendron (2009) present a CMND model that allows multiple facilities on each $(i, j) \in A$ by defining the y_{ij} variables as integer variables.

Majority of the CMND problems are modeled as arc-based, but there are several studies applying path-based approach, (Crainic et al. (2000), Crainic and Gendreau (2002), Hewitt et al. (2010)). With respect to the LP relaxations of arc-based and path-based formulations, none of the formulations outperforms the other, as shown by Gendron et al. (1999).

CMND problems are NP hard problems necessitating exponential computational times to obtain the optimal solutions and LP relaxations do not provide tight lower bounds. Exact solution methods including relaxations and cutting plane algorithms are applied for CMND problems. Gendron et al. (1999) study continuous relaxations and Lagrangean relaxations of CMND, evaluate the quality of the lower bounds comparing them with Tabu Search heuristic and branch-and-bound results and concluded that none of the methods if applied alone to CMND is sufficient to solve large size problems. Gendron et al. (1994) show that the quality of the lower bounds obtained from relaxations can be improved by using valid knapsack inequalities.

Holmberg and Yuan (2000) presented Lagrangean relaxation based branch-and-bound method that can be used for obtaining good feasible solutions to large-size problems. Sridhar and Park (2000) present a branch-and-bound algorithm that applies Benders, cut set inequalities, and show that for small size networks Benders cuts are better for heavy traffic load and cut set inequalities are better for light traffic load. Costa et al. (2009) present a comprehensive comparison of Benders, metric and cut set inequalities and by strengthening Benders inequalities, they reduce the computational time to obtain feasible solutions. Crainic et al. (2001) present bundle-based relaxation methods that outperform the subgradient counterparts by converging faster and being more robust to initial parameter settings.

Chouman et al. (2003) and Chouman et al. (2009) study cutting plane algorithms by incorporating valid inequalities including strong and cut set inequalities. By this way, they obtain better lower bounds than the LP relaxation. Frangioni and Gendron (2009) study cutting plane algorithms and column and row generation methods for their reformulation of CMND problem as to form a starting point for constructing a MIP based heuristic method.

Exact solution methods cannot solve large size CMND problems. Therefore, heuristics are applied either solely or in conjunction with exact solution methods or other heuristics for large size problems. Solution methods of CMND problem that apply heuristic solution methods are presented in Section 3.1.1.

3.1.1 Heuristic Solution Methods

Neighborhood search algorithms establish the majority of the heuristic solution approaches applied for CMND problems. A review of CMND studies having heuristic based solution methods are presented in Table 2 with main modeling characteristics, solution methods, and computational results. Modeling characteristics are analyzed with respect to three criteria: whether commodities are modeled as splittable or unsplittable, an arc-based or a path-based formulation is used, and capacity limits are defined for arcs or edges. In computational results part, the test networks, test problem sizes and computational run time performances (CPU times) are listed. $|N|$ represents the number of nodes of the test problems and $|K|$ represents the number of commodities. Although CPU times depend on the hardware used, they provide an insight for the duration of the computation. Generally, each study presents a set of test networks. However, to compare the capabilities of the studies the largest test instances are presented in Table 2. For the same sized test networks, there are several instances due to variations of fixed charge to variable costs ratio and capacity levels. Therefore, whenever same size test network has different instances, the minimum and the maximum CPU times corresponding to those instances are presented in Table 2.

Table 2. Modeling Characteristics and Computational Performances of CMND Studies that Apply Heuristic Solution Methods

Study	Splittable/ Unsplittable	Form. Type	Arcs/ Edges	Solution Method	Test Network	Test Problem Sizes	CPU Times
Crainic et al. (2000)	splittable	path-based	arcs	A Simplex Based Tabu Search Method	Canad Test Networks	a) $\|N\|$=30, $\|K\|$=400 b) $\|N\|$=100, $\|K\|$= 30	a) 3 – 4.5 hr. b) 32 - 277 sec.
Crainic and Gendreau (2002)	splittable	path-based	arcs	Cooperative Parallel Tabu Search	Canad Test Networks	a) $\|N\|$=30, $\|K\|$=400 b) $\|N\|$=100, $\|K\|$= 30	a) 5 - 9 hr. b) 13 - 24 min.
Ghamlouche et al. (2003)	splittable	arc-based	arcs	Cycle Based Tabu Search	Canad Test Networks	$\|N\|$=30, $\|K\|$=400	3 – 4.5 hr.
Alvarez et al. (2003)	splittable	arc-based	edges	Scatter Search	Randomly generated test networks	$\|N\|$=20, $\|K\|$=100	120 - 180 sec.
Ghamlouche et al. (2004)	splittable	arc-based	arcs	Path Relinking Algorithm	Canad Test Networks	$\|N\|$=30, $\|K\|$=400	6 - 20 hr.
Crainc et al. (2004)	splittable	arc-based	arcs	A Slope Scaling and Lagrangean Perturbation Heuristic	Canad Test Networks	a) $\|N\|$=30, $\|K\|$=400 b) $\|N\|$=100, $\|K\|$= 30	Not stated
Zaleta and Socarras (2004)	splittable	arc-based	edges	Tabu Search	Randomly generated test networks	$\|N\|$=50, $\|K\|$=100	Not stated
Alvarez et al. (2005a)	splittable	arc-based	edges	Scatter Search	Randomly generated test networks	$\|N\|$={30, 50} $\|K\|$={30, 50, 100}	On average 113-170 sec.
Alvarez et al. (2005b)	splittable	arc-based	edges	GRASP Embedded Scatter Search	Randomly generated test networks	$\|N\|$={30, 50} $\|K\|$={30, 50, 100}	On average 54-181 sec.
Crainic et al. (2006)	splittable	arc-based	arcs	A Multilevel Cooperative Tabu Search	Canad Test Networks	$\|N\|$=30, $\|K\|$=400	Not Stated

Table 2. (continued)

Study	Splittable/ Unsplittable	Form. Type	Arcs/ Edges	Solution Method	Test Network	Test Problem Sizes	CPU Times
Crainic and Gendreau (2007)	splittable	arc-based	arcs	Scatter Search	Canad Test Networks	a) $\|N\|$=30,$\|K\|$=400 b)$\|N\|$=100,$\|K\|$=30	a) 8.6 – 36.5 hr. b)15 min.-1.2 hr.
Katayama et al. (2009)	splittable	arc-based	arcs	Capacity Scaling heuristic that applies column & row generation	Canad Test Networks	$\|N\|$=30, $\|K\|$=400	8 - 30 min.
Chouman and Crainic (2010)	splittable	arc-based	arcs	Combination of a MIP and Tabu Search	Canad Test Networks	a) $\|N\|$=30,$\|K\|$=400 b)$\|N\|$=100,$\|K\|$=30	a) 10-10.6 hr. b) 93 sec.-1.3 hr.
Hewitt et al. (2010)	splittable	path-based	arcs	Exact solution method within a Neighborhood Search	Random test networks	$\|N\|$=500, $\|K\|$=200	5 - 18 sec.
Rodrigez-Martin & Salazar Gonzalez (2010)	splittable	arc-based	arcs	Local Branching Heuristic	Canad Test Networks	a) $\|N\|$=30,$\|K\|$=400 b)$\|N\|$=100,$\|K\|$=30	a) & b) 600 sec.
Katayama and Yuritomo (2011)	splittable	arc-based	arcs	Capacity Scaling and Local Branching heuristics	Canad Test Networks	a) $\|N\|$=30,$\|K\|$=400 b)$\|N\|$=100,$\|K\|$=30	504 - 1550 sec.
Yaghini et al. (2012a)	splittable	arc-based	arcs	Simplex-Based Simulated Annealing Algorithm	Canad Test Networks	a) $\|N\|$=30,$\|K\|$=400 b)$\|N\|$=100,$\|K\|$=30	a)133 sec.-4.7 hr. b) 98 - 250 sec.
Yaghini & Akhavan Kazamzadeh (2012)	unsplittable	arc-based	arcs	Simulated Annealing	Canad Test Networks	$\|N\|$=30, $\|K\|$=400	20 min.
Paraskevopoulos et al. (2013)	splittable	arc-based	arcs	Evolutionary Algorithm that applies Scatter Search & Iterated Local Search	Canad Test Networks	a) $\|N\|$=30,$\|K\|$=400 b)$\|N\|$=100,$\|K\|$=30	a) and b) 600 - 900 sec.

As it can be seen from the computational performances, the difficulty of the CMND problems increase as the number of commodities increase.

CMND studies that consider capacitated edges use randomly generated test networks, generally apply Scatter Search and according to the presented average computational times, for networks with 50 nodes and 100 commodities near optimal solutions are generated around 3 minutes computation time.

Majority of the CMND studies that consider capacitated arcs, use the same test networks, which are randomly generated and introduced by Crainic et al. (2000). These test networks are called as Canad problems and include a set of instances having different fixed cost to variable cost ratios and different capacity levels.

Earlier studies using Canad test problems mostly apply Tabu Search based solution methods. Among them the first improvements are observed by cycle-based Tabu Search, Ghamlouche et al. (2003) and later Path Relinking algorithm proposed by Ghamlouche et al. (2004) outperformed the cycle-based Tabu Search. Crainic et al. (2006) present a multilevel cooperative Tabu Search method, which provides better results than the both methods. Crainic and Gendreau (2007) develop a Scatter Search algorithm and compare it with Path Relinking algorithm but the Scatter Search algorithm does not outperform the Path Relinking algorithm.

Chouman and Crainic (2010) present a combined MIP and Tabu Search method for developing good feasible solutions to CMND problem and they obtained better results than the ones provided by the optimization programs.

Hewitt et al. (2010) present a method that applies an exact solution method within a neighborhood search algorithm and for randomly generated test networks consisting of 500 nodes and 200 commodities they compare their method with cycle based Tabu Search and Path Relinking methods. The proposed combined model of Hewitt et al. (2010) outperforms both methods in solution quality and computational time performances.

Katayama et al. (2009) develop a capacity scaling heuristic that applies column and row generation. Testing the proposed method on Canad problems and comparing with previous studies simplex based Tabu Search of Crainic et al. (2000), cycle-based Tabu Search of Ghamlouche et al. (2003), Path Relinking of Ghamlouche et al. (2004), and multilevel cooperative Tabu Search of Crainic et al. (2006) revealed that the capacity scaling heuristic of Katayama et al. (2009) provides new best solutions in %70 of the test problems in the shortest computation times.

Rodrigez-Martin and Salazar Gonzalez (2010) and Katayama and Yuritomo (2011) apply local branching heuristic, Yaghini et al. (2012) and Yaghini and Kazamzadeh (2012) apply Simulated Annealing heuristic and they all shorten the computational times on Canad problems. Paraskevopoulos et al. (2013) present an Evolutionary Algorithm that applies Scatter Search and Iterated Local Search. In terms of solution quality, it is better than all the heuristics developed for CMND on average, but it is competitive with the method of Hewitt et al. (2010) and computational performance of Paraskevopoulos et al. (2013) is not better than that of Crainic et al. (2006) and Katayama et al. (2009).

3.2 Service Network Design Problems

SND problems apply multi commodity flow structure and have similarities to CMND problems. This similarity is also highlighted by Kim et al. (1997) and the author states that SND problems have additional complexity compared to CMND problems, since SND problems have to maintain a balance of service levels and transportation assets throughout the service network.

In SND literature, there are various modeling approaches including arc-based, path-based, and tree-based formulations, Kim et al. (1997). Since LA-CMNDP uses an arc-based formulation, we present a generic SND model that applies an arc-based formulation. For different modeling approaches in SND problems and for a review of SND literature Crainic (2000), Crainic (2005), and Wieberneit (2008) can be analyzed.

To state the arc-based SND problem, assume the following notation. For a given set of service types $f \in F$, let y_{ij}^{f} denote for arc (i, j) whether a service type of f is established or not. Commodity flows are denoted by x_{ij}^{k}. The capacity of service type f is represented by u^{f}, and the fixed cost of using arc (i, j) having service type f is represented by h_{ij}^{f}. Then the arc-based formulation of SND is as follows, Kim (1997).

SND-Arc

Minimize $\sum_{f \in F} \sum_{(i,j) \in A} h_{ij}^{f} y_{ij}^{f} + \sum_{k \in K} \sum_{(i,j) \in A} c_{ij}^{k} x_{ij}^{k}$

subject to

(3.1), (3.3)

$$\sum_{j \in N} y_{ij}^{f} - \sum_{j \in N} y_{ji}^{f} = 0 \qquad i \in N, f \in F \tag{3.6}$$

$$\sum_{k \in K} x_{ij}^{k} \leq \sum_{f \in F} u^{f} y_{ij}^{f} \qquad (i, j) \in A \tag{3.7}$$

$$y_{ij}^{f} \geq 0 \textit{ and integer} \qquad f \in F, (i, j) \in A \tag{3.8}$$

Constraints (3.1) and (3.3) are the *multicommodity flow conservation constraints*. Constraints (3.6) are the *design balance constraints* and ensure that the number of each service type *f* entering to node *i* is equal to the number leaving. Constraints (3.7) are the *capacity constraints* that limit the flow on each arc up to its service capacity. Constraints (3.8) ensure that for each arc an integer number of service of type *f* can be established, Kim (1997).

SND Problems are NP-hard problems consisting of many design issues. Besides the complexity of the models, considering the properties of a real life transportation network (node, arc, commodity numbers) and characteristics of modeling attributes (service level requirements, consolidation policies) SND problems require improved solution techniques. To obtain a near optimal solution within a reasonable computation time, heuristic solution methods are applied either solving the whole SND problem or using in conjunction with an exact solution method. Also Kim et al. (1999), Jarrah et al. (2009), and Pedersen (2005) emphasize the necessity of using heuristics in SND problems, which are NP-hard.

Neighborhood search algorithms are generally applied in LTL SND studies Equi et al. (1997), Farvolden and Powell (1994), Jarrah et al. (2009), Erera et al. (2012) and motor carrier SND studies Powell (1986a), Powell and Sheffi (1989), Powell and Koskosidis (1992), Barcos et al. (2010). Sung and Song (2003) present an SND model that both allows LTL and TL transportation and solve the model by a neighborhood search algorithm.

Main decision problems handled in rail freight SND studies include generation of train routes with frequencies and service levels, generation of freight routes with related train itineraries and intermediate stops, allocation of classification operations to yards in such a way to prevent congestion, and modeling the empty car flows. Yaghini and Akhavan (2012) analyze the railroad freight transportation studies that apply multicommodity network design problem structure. Recent CMND problems and major railroad planning problems having multicommodity structure are reviewed with their properties and solution methods.

Majority of rail SND studies apply heuristic based solution approaches, Crainic et al. (1984), Keaton (1989), Haghani (1989), Marin and Salmeron (1996a,b), Gorman (1998), Campetella et al. (2006), Zhu (2010), and Lin et al. (2012). There are several studies that apply exact solution methods, Kwon et al. (1998), and Pedersen and Crainic (2007).

Heuristic based solutions are applied in maritime SND studies. Lai and Lo (2004) study a dynamic ferry service network design problem and present a shortest path based heuristic algorithm. Shintani et al. (2007) apply genetic algorithm to a container shipping network design problem, which is modeled as a two level Knapsack problem.

Air SND studies deal with transportation of time critical shipments that are generally called as express shipment delivery problems. The shipments include small parcels, postal deliveries and have strict delivery time windows like next day delivery. To satisfy tight time windows, express shipment delivery necessitates modeling a multimodal SND that integrates ground and air transportation so as to guarantee point to point delivery. Armacost et al. (2002) and Barnhart and Shen (2005) focus on the air part of express shipment delivery problem. Armacost et al. (2002) extend the main model by a composite variable formulation that provides tighter LP bounds. Barnhart and Shen (2005) apply column generation. Büedenbender et al. (2000) also focus on the air part, but apply a hybrid Tabu Search and branch-and-bound algorithm.

Barnhart and Schneur (1996), Kim et al. (1999), and Barnhart et al. (2002) model the express shipment delivery problem considering both the ground and the air parts. Barnhart and Schneur (1996) apply column generation, Kim et al. (1999) apply a heuristic solution method that applies problem reduction and column generation techniques, and Barnhart et al. (2002) present an iterative solution procedure within a decomposition structure.

Multimodal SND studies also include models for combined ground and rail transportation networks, Jansen et al. (2004), Pedersen and Crainic (2007), and Andersan et al. (2009). Jansen et al. (2004) solve the main model by decomposing it into several sub problems. Pedersen and Crainic (2007) and Andersan et al. (2009) present dynamic SND models and apply exact solution methods.

To deal with demand variability over a planning horizon and to generate detailed schedules, dynamic SND models are developed that apply a time space network structure. To include time dimension, additional nodes and arcs have to be defined resulting in huge planning networks. This increase in network size brings additional complexity to already NP-hard SND problems.

Heuristic decomposition approaches are widely applied to these large and complicated time-space SND problems. The main decision problem is partitioned into several sub problems and solving the sub problems iteratively, a solution for the whole model is constructed. Local search algorithms, heuristic approximation algorithms, Slope Scaling heuristic, Tabu Search, and shortest path algorithms are used for solving the sub problems, Haghani (1989), Farvolden and Powell (1994), Kim et al. (1999), Jansen et al. (2004), Lai and Lo (2004), Jarrah et al. (2009), Zhu

(2010), Bai et al. (2010), Erera et al. (2012). Pedersen et al. (2009) apply Tabu Search algorithm to the main model without partitioning.

Several studies apply exact solution methods for time space SND problems, Kwon et al. (1998), Pedersen and Crainic (2007), Andersan et al. (2009). Kwon et al. (1998) present a dynamic routing and scheduling rail car model that incorporates delivery time windows and consider the priority of rail cars. The authors apply column generation and obtain the optimal plans for 12 terminals in reasonable computational times.

Pedersen and Crainic (2007) experiment with an optimization package to define the complexity of their intermodal freight train service design model. For 25 terminals and 90 commodities the authors could obtain only 3 feasible solutions out of 9 different scenarios within 90 hours of computation time, verifying the necessity of heuristics for larger scale networks.

Andersan et al. (2009) strengthen the LP relaxation of their dynamic SND model by defining valid inequalities. For 17 nodes and 40 commodities optimal solutions are obtained by an optimization program. The authors state the most promising solution methods for large networks as metaheuristics and column generation.

3.2.1 Heuristic Solution Methods

In order to report SND studies resembling the structure of LA-CMNDP*,* those that do not apply a time space network structure and those that utilize a heuristic solution method are reviewed in Table 3. The application areas (rail, LTL, ground, air), general description of the problem formulations, objective functions, main decisions of the models, the constraints, and computational results of the SND studies are presented in Table 3. If there are different instances for the same size test network then the maximum and minimum CPU time values of these instances are given in Table 3.

Table 3. Modeling Characteristics and Computational Performances of SND Studies that Apply Heuristic Solution Methods

Study	Application Area/ Problem Formulation	Objective Function	Decisions	Constraints	Solution Method	Test Network / Test Problem Sizes	CPU Times
Crainic et al. (1984)	Rail/ Capacitated Mixed Integer Multicommodity Flow	Minimize the sum of - train routing costs - classification costs - itinerary costs - delay costs	- to determine train routes with their frequencies and service levels - to determine traffic flow with their itineraries - to allocate classification operations to yards	- to satisfy capacity limits of each service - to prevent congestion at yards - to satisfy minimum block sizes	Decomposition and column generation within a heuristic algorithm	Western part of the Canadian National Railroads - 107 nodes, 735 traffic classes (commodities)	28 min.
Powell (1986a)	LTL Motor Carrier/ Fixed Charge Service Network Design	Minimize the sum of - transportation, - handing, - empty balancing costs	- to select terminal pairs to offer direct service between - to determine freight routes - to determine the flow of empty trailers	- to satisfy the minimum frequency requirements for direct service links	Local Improvement Add/Drop Heuristic	An extracted network from a real LTL motor carrier network - 140 nodes, 12 terminals (given)	750 sec.
Keaton (1989)	Rail/ Fixed Charge Capacitated Mixed Integer Multicommodity Flow	Minimize the sum of - fixed train costs - transportation costs	- to determine terminal pairs to offer direct service between - to determine traffic flow with assignments to trains - to determine train routes	- to satisfy capacity limits of terminals - to satisfy capacity limits of trains	Combination of a Lagrangean Decomposition and a Greedy Heuristic Algorithm	A generated network from Western part of Conrail System - 26 terminals, 333 commodities	407 sec.

Table 3 (continued)

Study	Application Area / Problem Formulation	Objective Function	Decisions	Constraints	Solution Method	Test Network / Test Problem Sizes	CPU Times
Powell and Sheffi (1989)	LTL Motor Carrier/ Fixed Charge Service Network Design	Minimize the sum of - transportation, - handing, - empty balancing costs	- to select terminal pairs to offer direct service between - to determine freight routes - to determine the flow of empty trailers	to satisfy the minimum frequency requirements for direct service links	An Interactive optimization system that applies a local add/drop heuristic	A real LTL motor carrier network - 300 nodes, 30 terminals (given)	Not stated
Powell and Koskosidis (1992)	LTL Motor Carrier/ Uncapacitated Mixed Integer Multicommodity Flow	Minimize the sum of - transportation costs	- to determine routing network - to determine freight routes	to satisfy the minimum frequency requirements for direct service links	Gradient based local improvement heuristic	An extracted network from a real LTL motor carrier-140 nodes, 12 terminals (given)	773 sec.
Marin and Salmeron (1996a,b)	Rail/ Capacitated Integer Programming Problem	Minimize the sum of - transportation costs - train frequency and service costs	- to determine train frequencies - to determine freight routes	- to satisfy capacity limits of yards - to satisfy capacity limits of trains	- Simulated Annealing Heuristic - Tabu Search Heuristic - Problem Specific Heuristic	A physical test network - 82 services, 61 yards, 150 commodities	12 hr.

Table 3 (continued)

Study	Application Area / Problem Formulation	Objective Function	Decisions	Constraints	Solution Method	Test Network / Test Problem Sizes	CPU Times
Equi et al. (1997)	Ground/ Combination of Integer Fixed Charge Capacitated Network Design and Circulation Problems	Minimize the sum of - fixed trip costs - transportation costs - truck scheduling costs	- to determine the trip routes together with number of trips - to determine flow of commodities - to assign the trucks to trips	to satisfy capacity limits of trucks	Lagrangean Decomposition and transportation sub problem is solved by a local search heuristic	A set of extracted networks from a real forestry industry - 635 nodes	22 min.
Gorman (1998)	Rail/ Fixed Charge Capacitated Integer Programming Problem	Minimize the sum of - fixed train costs - transportation costs	- to determine train routes - to determine freight routes with assignments to trains	- to satisfy customer service requirements - to satisfy capacity limits of trains	Tabu Enhanced Genetic Algorithm	Problems whose optimal solutions are known - 18 commodities, 4 train speeds	Not stated
Büedenbender et al. (2000)	Air/ Integer Fixed Charge Capacitated Mixed Integer Programming Problem	Minimize the sum of - fixed aircraft costs, - ferry costs - transportation costs of letter mails	- to determine flight schedules - to assign letter mail flows to flights	- to satisfy time windows - to satisfy capacity limits of aircrafts	A hybrid Tabu Search and branch-and-bound Algorithm	Generated Networks from the airmail network of Germany - 3,000 letter mail demands (commodities)	24 hr.

Table 3 (continued)

Study	Application Area / Problem Formulation	Objective Function	Decisions	Constraints	Solution Method	Test Network / Test Problem Sizes	CPU Times
Sung and Song (2003)	Ground/ Fixed Charge Mixed Integer Transportation Network Design	Minimize the sum of - locating CD (Cross Docking) centers - cost of vehicles	- to locate CD centers - to determine direct service links together with vehicle numbers - to determine freight demand routes	- to satisfy vehicle capacities - to satisfy service time restrictions of commodities	Tabu Search	Randomly generated test networks a) 50 Nodes, 12 CD centers, 960 commodities b) 50 Nodes, 12 CD centers, 1,156 commodities	a) 36 min. b) 51 min.
Campetella et al. (2006)	Rail/ Integer Fixed Charge Multicommodity Network Design	Minimize the sum of - fixed train costs - transportation costs	- to determine the service frequencies (number of trains) - to determine freight routes assigning them to services	to satisfy train capacities	Tabu Search	Extracted network from an Italian Railway Company - 39 nodes, 375 commodities	Not stated
Barcos et al. (2010)	LTL Motor Carrier/ Main model is not stated but empty vehicle balancing model is IP	Minimize the sum of - transportation costs - vehicle stop & handling costs	- to determine vehicle routes - to determine freight routes - to determine empty vehicle routes	to satisfy service levels	Ant Colony Metaheuristic	Spain and Portugal LTL Motor Carrier Network - 49 nodes, 2,352 commodities, 6 hubs (given)	3.2 hr.

Table 3 (continued)

Study	**Application Area / Problem Formulation**	**Objective Function**	**Decisions**	**Constraints**	**Solution Method**	**Test Network / Test Problem Sizes**	**CPU Times**
Bai et al. (2012)	General/ Fixed Charge Service Network Design	Minimize the sum of - fixed arc costs - transportation costs	- to determine service network - to determine freight routes	to satisfy vehicle capacities	Combination of Guided Local Search and Tabu Search	Canad Test Problems - 30 nodes, 400 commodities	104 - 785 sec.
Lin et al. (2012)	Rail/ Bilevel Programming of Capacitated Multicommodity Flow Problem	Minimize the sum of - operating costs - delay costs (accumulation, classification, assembly)	- to determine yard pairs to offer direct service - to determine freight routes - to determine train routes	- to satisfy yard capacities - to satisfy capacity of classification tracks - to prevent congestion at yards	Simulated Annealing	Railway Network of China - 127 yards, 14,440 shipments	6.4 hr.

Heuristic solutions developed for SND problems mostly apply neighborhood search algorithms. Majority of the models consider capacities of vehicles, consolidation centers, and services. Capacitated models aim determining the commodity and vehicle flows so as to prevent congestion and related delays in an implicit manner. For the same purpose, some SND studies apply different additional strategies. Powell (1986a), Powell and Sheffi (1989), and Powell and Koskosidis (1992) locate direct service links between terminal pairs, if the flow of commodities on this link satisfies the minimum frequency requirements. Crainic et al. (1984) restrict the minimum block sizes so as to fully utilize the classification infrastructure of yards and for the classification operations at yards apply a queuing model. In order to prevent congestion at yards, delays due to accumulation and classification at yards are penalized by a cost function.

Lin et al. (2012) present a rail service network design model that generates train services and frequencies. In the presented model, delay costs resulting from service accumulation, classification, and block formation operations at yards are considered in the objective function so as to distribute classification operations to yards in such a way to prevent congestion.

Büedenbender et al. (2000), Sung and Song (2003), and Barcos et al. (2010) consider service time requirements in their models and for problems with 3,000, 1,156, and 2,352 commodities the reported computational times are 24 hours, 51 minutes, and 3.16 hours respectively.

In computational experiments, Lin et al. (2012) consider the largest number of commodities with 14,440 and the computation time is 6.14 hours. Jarrah et al. (2009) consider the largest number of nodes with 725 nodes and the computation time is around 2 hours for 680 commodities.

3.3 Railroad Blocking Problems

Railroad blocking problem is the basic decision problem of railroads. Shipments, physical railroad network, and the yards constitute the major elements of a *railroad blocking problem*. A shipment consists of individual railcars and it may visit several yards on its route. In each yard, incoming shipments are reclassified to be placed on outgoing trains. To reduce the intermediate handlings taking place at yards, shipments are grouped together to form *blocks*. A block consists of individual shipments, whose origin and destinations need not be the same as the O-D pair of the block. Once a block is formed, it is not reclassified at yards until it reaches its destination. By this way, classification costs and delays occurring at yards are tried to be kept at minimum. Due to resource limits of yards (car handling capacities, working crew) and rail line capacities it is not possible to form an individual block for each shipment, Barnhart et al. (2000), Ahuja et al. (2005), Ahuja et al. (2007).

Blocking problem is a multicommodity network design problem, an arc-based formulation is presented by Ahuja et al. (2005) and a path-based formulation is presented by Newton et al. (1998). Path-based approach requires establishing all legal paths for each commodity beforehand. A priori path generation can require a lot of enumeration depending on the size of the network. Whereas by this way it is possible to handle more practical constraints by eliminating some paths that do not satisfy practical constraints like restrictions on allowable route lengths, Newton et al. (1998), Barnhart et al. (2000).

Let $G=(N,A)$ be the physical railroad network, where N represents the set of nodes denoting stations. At stations, individual shipments originate, terminate or switch trains. A represents the set of arcs of the physical railroad network. The sets of arcs that enter and emanate from station i are denoted by $In(i)$ and $Out(i)$, respectively. At each node $i \in N$, at most b_i *blocks* can be formed and d_i rail cars can be classified with unit classification cost of m_i. Then the arc-based formulation of railroad blocking problem is as follows.

Blocking-Arc

$$Minimize \quad \sum_{k \in K} \sum_{(i,j) \in A} c_{ij} x_{ij}^k + \sum_{i \in N} \sum_{k \in K} \sum_{(i,j) \in Out(i)} m_i x_{ij}^k$$

subject to

(3.1), (3.2), (3.4)

$$\sum_{(i,j) \in Out(i)} y_{ij} \le b_i \qquad \forall i \in N \tag{3.9}$$

$$\sum_{k \in K} \sum_{(j,i) \in In(i)} x_{ji}^k \le d_i \qquad \forall i \in N \tag{3.10}$$

$$x_{ij}^k \in \{0, v_k\} \qquad \forall (i,j) \in A, \ \forall k \in K \tag{3.11}$$

Objective function tries to minimize the sum of transportation costs and the classification costs. Constraints (3.1) and (3.11) are *multicommodity flow conservation constraints*. Due to constraints (3.11) the commodities are unsplittable. Constraints (3.2), (3.9), and (3,10) are arc capacity, blocking capacity, and car handling capacity constraints, respectively, Ahuja et al. (2005).

LA-CMNDP applies a consolidation structure that is similar to railroad blocking problem. Each arc in the solution network of LA-CMNDP defines a *block* and commodities are assumed to be unsplittable. Model LA-CMNDP does not consider arc capacities and car handling capacities. Whereas blocking capacity constraints (3.9) are handled in a different way by LA-CMNDP, in which capacity restrictions are applied to the number of incoming *blocks* to nodes.

Blocking problems generally use real railway networks or simulated networks generated from real railway data. Earlier studies on blocking problem apply exact solution methods, Bodin et al. (1980), Newton et al. (1998), Barnhart et al. (2000). Bodin et al. (1980) tested their model on a network having 33 classification yards. To reduce the problem size some variables are fixed heuristically and the reduced model is solved by an optimization program. Newton et al. (1998) apply a path-based formulation approach that enables reductions in problem size. For a network of 150 nodes and 1,300 commodities, the authors obtain solutions with 2% of lower bound by a branch and price algorithm that applies column generation. Barnhart et al. (2000) present a Lagrangean relaxation method that applies valid inequalities. The authors obtain tighter lower bounds than the LP relaxation on two test networks. The first network has 116 nodes with 7,170 commodities and the second one has 1,050 nodes with 12,110 commodities. For the larger network, they do not consider the car handling capacity restrictions.

Recent studies on railroad blocking problem apply neighborhood search and population based algorithms. Ahuja et al. (2005) and Ahuja et al. (2007) present a Very Large Scale Neighborhood (VLSN) search algorithm, Yaghini et al. (2011) and Yue et al. (2011) apply ant-colony and Yaghini et al. (2012b) apply genetic algorithm. In Section 3.3.1, railroad blocking studies that apply heuristic solution methods are analyzed with main modeling characteristics and computational results.

Railroad blocking problem is also a basic decision problem in Yard Location problem. Yards are the stations of railroads where cars from incoming trains are reclassified and assembled into outing blocks. Since important blocking operations take place at yards, the yard locations have great influence on the resulting blocking network structure, which necessitates giving yard location decisions considering the blocking problem. Ahuja et al. (2005) present a yard location formulation, which is based on the arc-based formulation of railroad blocking problem. The presented model assumes that the candidate yard locations and the desired number of yards is known beforehand and denoted by p. Let $\bar{N}$ denote the set of candidate yard nodes. If a node $i \in \bar{N}$ is selected as yard then $z_i = 1$, the blocking and car handling capacities of node i is increased by $\bar{b}_i$ and $\bar{d}_i$, respectively. Then, the arc-based formulation of yard location problem is as follows.

Yard Location-Arc

$$\textit{Minimize} \quad \sum_{k\in K}\sum_{(i,j)\in A} c_{ij}x_{ij}^k + \sum_{i\in N}\sum_{k\in K}\sum_{(i,j)\in Out(i)} m_i x_{ij}^k$$

subject to

(3.1), (3.2), (3.4), (3.11)

$$\sum_{(i,j)\in Out(i)} y_{ij} \le b_i \qquad \forall i \in N\backslash\bar{N} \tag{3.12}$$

$$\sum_{k\in K}\sum_{(i,j)\in In(i)} x_{ij}^k \le d_i \qquad \forall i \in N\backslash\bar{N} \tag{3.13}$$

$$\sum_{(i,j)\in Out(i)} y_{ij} \le b_i + \bar{b}_i z_i \qquad \forall i \in \bar{N} \tag{3.14}$$

$$\sum_{k\in K}\sum_{(j,i)\in In(i)} x_{ji}^k \le d_i + \bar{d}_i z_i \qquad \forall i \in \bar{N} \tag{3.15}$$

$$\sum_{i\in\bar{N}} z_i \le p \tag{3.16}$$

$$z_i \in \{0,1\} \qquad \forall i \in \bar{N} \tag{3.17}$$

Objective function tries to minimize the sum of transportation costs and the classification costs. Blocking capacity (3.9), and car handling capacity (3.10) constraints of blocking problem are rewritten for non-candidate yard nodes (3.12)-(3.13) and for candidate yard nodes (3.14)-(3.15). By constraints (3.16), at most p of the nodes can be selected as yards and the blocking and car handling capacities of these nodes are increased, Ahuja et al. (2005).

Like yard location problem, LA-CMNDP locates a predetermined number of terminals, considering the bottlenecks at nodes resulting from the incoming *block* capacity restrictions. By this way, the level of consolidation throughout the transportation network is tried to be increased.

For the solution of yard location problem, a neighborhood search algorithm is presented by Liu et al. (2008), which is presented in Section 3.3.1 with computational results.

3.3.1 Heuristic Solution Methods

Ahuja et al. (2005) present a VLSN search algorithm for the railroad blocking problem and give an overview of the solution approach. The details of the VLSN algorithm are presented by Ahuja et al. (2007). Considering a large U.S. railroad network, the authors state the potential of VLSN search algorithm to solve real life instances of railroad blocking problems, which may have 50,000 shipments. For the solution part, only blocking capacities of nodes (out-degree constraints of nodes) are considered, whereas car handling capacities of nodes and arc capacities are ignored. The authors state the reason of this assumption as even ignoring these capacities most nodes and arcs satisfy them. The VLSN search algorithm is tested using three major U.S. railroads' data. Compared to the current solutions of railroads, the VLSN method provides around 15% savings in reclassifications in 2 hours computation time.

Liu et al. (2008) apply the VLSN search algorithm in three greedy heuristic algorithms. All algorithms produce same results, though the simplest one is the drop algorithm. Since addition of yard location decisions increases the complexity of railroad blocking problem, the proposed solution method is experimented on a smaller network that has 39 yards. The proposed method provides a solution within 6 hours computation time to allow the decision maker analyze the problem from different perspectives.

Yaghini et al. (2011) and Yue et al. (2011) apply ant colony algorithm for blocking problem. For a test network of 50 nodes and 1114 commodities, Yaghini et al. (2011) obtain a feasible solution in 44 seconds. Yaghini et al. (2012b) present a Genetic algorithm that generates a near optimal solution within 81 seconds for a test network of 50 nodes and 1,114 commodities.

A review of railroad blocking studies having heuristic based solution methods are presented in Table 4 with main modeling characteristics of the studies and computational results including the size of the test networks and the solution time performances, CPU times.

Table 4. Modeling Characteristics and Computational Performances of Railroad Blocking Studies That Apply Heuristics

Study	Problem Type	Splittable/ Unsplittable	Formulation Type	Solution Method	Test Network	Test Problem Sizes	CPU Times
Ahuja et al. (2007)	Railroad Blocking	unsplittable	arc-based	VLSN algorithm	Rail Network of three major USA Railroad Companies	50,000 shipments	2 hr.
Liu et al. (2008)	Yard Location	unsplittable	arc-based	A Drop algorithm that applies a VLSN algorithm as a sub-routine	Rail Network of a major USA Railroad Company	39 yards	6 hr.
Yaghini et al. (2011)	Railroad Blocking	unsplittable	arc-based	Ant Colony algorithm	I) Simulated test networks II) Rail Network of Iran Railway Company	I) a) 30 nodes, 692 commodities I) b) 50 nodes, 1,114 commodities II) 170 terminals	I) a) 40 sec. I) b) 44 sec. II) 20 sec.
Yue et al. (2011)	Railroad Blocking	unsplittable	arc-based	Ant Colony algorithm	Coal Heavy Haul Rail Network of North China	21 nodes, 11 commodities	1 sec.
Yaghini et al. (2012b)	Railroad Blocking	unsplittable	path-based	Genetic algorithm	Simulated test networks	50 terminals, 521 commodities	81 sec.

3.4 Latest Arrival Problems

Latest arrival concept is first introduced by Kara and Tansel (2001) for single allocation hub location problem. Hubs are consolidation centers that collect shipments (goods, packages, messages, etc.) from their origins and distribute them to their destinations. Hub location problems consist of two main decisions; locating hub facilities and determining the allocation of non-hub nodes to hubs. The non-hub nodes are usually called as spokes. There are two allocation structures of spokes to hubs; i) *single allocation*, in which every spoke is assigned to exactly one hub and ii) *multiple allocation*, in which a spoke can be assigned to multiple hubs for different shipments.

Service times are handled in hub location literature in various forms. p-hub center and hub covering models are the first studies considering service times. Campbell (1994b) presents the first formulations of both problems. p-hub center problem locates p hubs in order to minimize the maximum travel time or distance between every origin destination pair. On the other hand, hub covering problem tries to cover all nodes by not exceeding a predetermined allowable travel time limit for all origin destination pairs by minimizing the number of hubs or total transportation cost.

Classical hub location models dealing with service times only consider travel times. However, delays occurring at hubs constitute an important part in total delivery times. Since hubs are consolidation centers, a significant amount of time is consumed for unloading the arriving vehicles, sorting the shipments, and loading the departing vehicles. Kara and Tansel (2001) consider the transient times spent at hubs by determining the departure time of a vehicle according to the latest arriving cargo that will be loaded to that vehicle.

The concept of latest arrival to a hub is studied in different hub location problems. Tan and Kara (2007) present latest arrival hub covering problem. Yaman et al. (2007) present latest arrival hub location problem with stopovers. Alumur (2009) use the latest arrival idea in multimodal hub location and hub network design problem for satisfying two different service levels. Yaman et al. (2012) present the release time scheduling and hub location for next day delivery problem, in which release times of trucks from each node are determined so as to maximize the total amount of cargo guaranteed to be delivered by the next day. The model applies the latest arrival idea to departures from hubs under single allocation structure.

LA-CMNDP applies the latest arrival consolidation structure defined by Kara and Tansel (2001). Different from latest arrival hub location problems, LA-CMNDP applies a multiple allocation structure, in which each node can be connected to multiple nodes. Moreover LA-CMNDP considers constant handling times for the processing operations performed at nodes. Among the latest arrival hub location problems, Yaman et al. (2007) is the only study considering constant processing times at stopovers. The main modeling characteristics of latest arrival studies are presented in Table 5.

Table 5. Modeling Characteristics of Latest Arrival Studies

Study	Problem Definition	Objective Function	Allocation Type	Consider Handling Times?	Constant or Variable Handling Times
Kara and Tansel (2001)	Hub Location Problem	Minimize Maximum Latest Arrival Time	single	no	-
Tan and Kara (2007)	Hub Covering Problem	Minimize The Number of Hubs to Locate	single	no	-
Yaman et al. (2007)	Hub Location Problem with Stopovers	Minimize Maximum Latest Arrival Time	single	only at stopovers	constant
Alumur (2009)	Multimodal Hub Location Problem	Minimize Fixed Network Design and Transportation Costs	single	no	-
Yaman et al. (2012)	Hub Location for Next Day Delivery	Minimize Transportation Costs	single	no	-

3.5 Benders Decomposition

Benders decomposition has successful applications in network design problems. The major survey paper on Benders decomposition is presented by Costa (2005) focusing on applications in fixed-charge network design problems. The survey includes 17 studies and in most of them Benders decomposition outperforms branch-and-bound and Lagrangean relaxation methods. In this section, we present a literature review of the main enhancement techniques developed for Benders decomposition. In the literature, generally hybrid methods that involve multiple enhancements are applied to accelerate Benders decomposition.

Considering the underlying structures of the main enhancement techniques, a classification scheme is proposed in Figure 5. Generalized Benders decomposition (GBD) is also included into the classification, since GBD provides an extension of Benders decomposition to non-linear programming problems. The literature review of GBD techniques is presented in detail in Section 3.6.

As presented in Figure 5, the enhancements in Benders decomposition are developed generally focusing on

i) effective solution techniques for the RMP and the SP,
ii) selecting/generating stronger cuts,
iii) generating multiple Benders cuts,
iv) extensions to classical Benders cuts, and
v) modeling structures.

The ultimate purpose of all techniques is to speed up the convergence, to increase the computational efficiency, and to solve larger size problems. However, each class of techniques put emphasis on different improvement areas. For instance, techniques in class ii) focus on choosing the cuts that have higher potential to better restrict the feasible region of the RMP and iii) focus on reducing the required number of RMP solutions. The explanations of the enhancement techniques together with the reviewed studies are listed in Table 6. In addition, a list of studies that apply the classical algorithm of Benders (1962) is provided in Table 7.

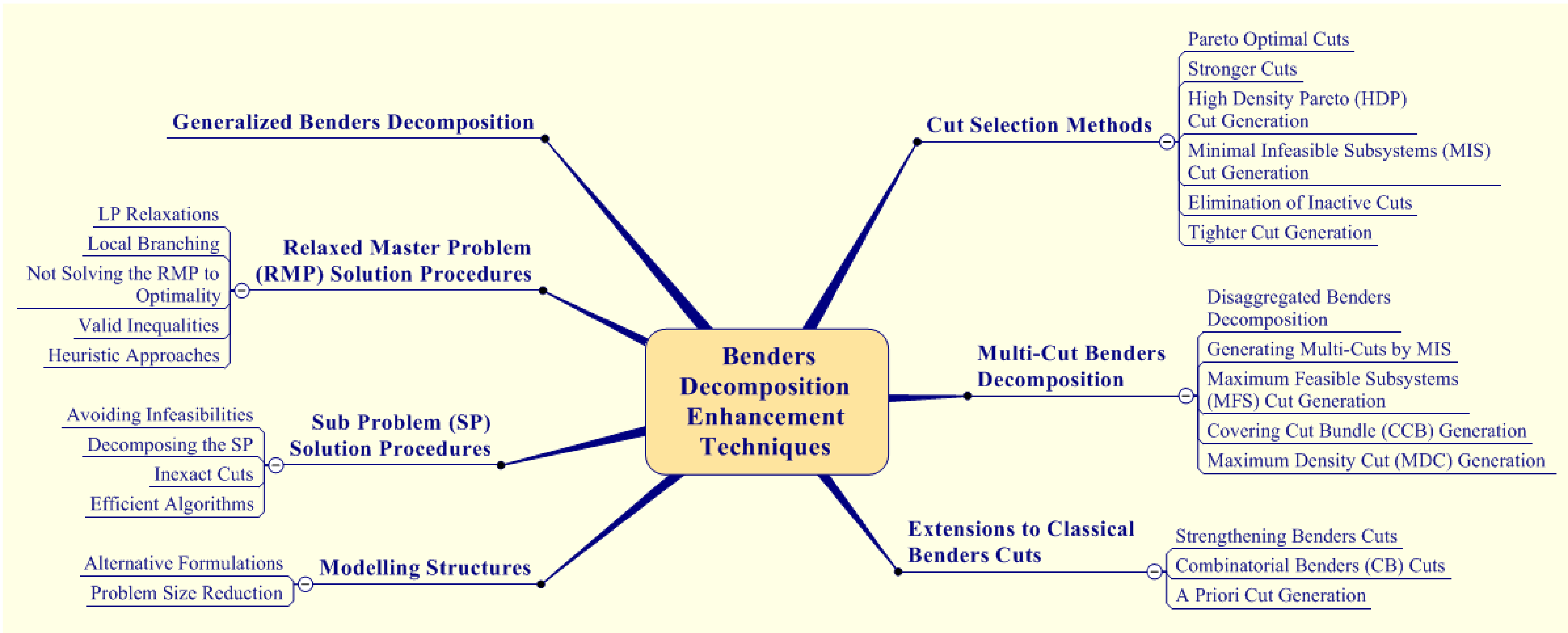


Figure 5. Classification of Main Enhancement Techniques for Benders Decomposition

Table 6. Explanations and Related Studies for Benders Decomposition Enhancement Techniques

Benders Decomposition Enhancement Techniques		Explanations	Studies
Major Class	**Sub Class**		
Cut Selection Methods	Pareto Optimal Cuts	When the SP has multiple optimal solutions (the case in most of the network optimization problems), among the alternative cuts the strongest one, namely the Pareto optimal cut is selected. The identification of these cuts within the Benders decomposition is performed by defining a core point of the RMP and then solving a Pareto Optimal Linear Program using the solution of the SP.	Magnanti and Wong (1981), Mercier et al. (2005), Cordeau et al. (2006), Papadakos (2008), Papadakos (2009), Contreras et al. (2011), Sa et al. (2013), Tang et al. (2013)
	Stronger Cuts	Instead of solving the Pareto optimal LP to optimality, it is solved approximately. The approximate solutions also provide strong cuts but not necessarily the Pareto optimal cuts.	Contreras et al. (2011)
	High Density Pareto (HDP) Cut Generation	High density Pareto optimal cuts (involving high number of decision variables of RMP) is obtained by lifting the Pareto optimal cuts.	Tang et al. (2013)
	Minimal Infeasible Subsystems(MIS) Cut Generation	This method selects the most violated Benders cut (either optimality or feasibility) among the alternatives by solving a Cut Generation LP, which is a feasibility seeking problem.	Fischetti et al. (2008), Fischetti et al. (2010), Sa et al. (2013)
	Elimination of Inactive Cuts	The multiple cuts generated by the SP are analyzed and the inactive cuts are eliminated before adding to the RMP.	Marin and Jaramillo (2009)

Table 6 (continued)

Benders Decomposition Enhancement Techniques		Explanations	Studies
Major Class	**Sub Class**		
Cut Selection Methods	Tighter Cut Generation	When it is difficult to obtain optimality cuts, this method provides the selection of tighter feasibility cut. Analyzing the distance between the infeasible point and the feasible points, the cut that is closer to the feasible region of the RMP is selected.	Yang and Lee (2011), Yang and Lee (2012)
Multi-cut Benders Decomposition	Disaggregated Benders Decomposition	In this method, the SPs are decomposed into sub problems and for each solution of the sub problems a separate cut is added to the RMP.	Cordeau et al. (2000), Camargo et. al. (2008), Gelareh and Nickel (2008), Marin and Jaramillo (2009), Contreras et al. (2011), Tang et al. (2013)
	Generating Multi-Cuts by MIS	In MIS method of Fischetti et al. (2008), when the SP is bounded, in addition to an optimality cut, a feasibility cut is also generated.	Fischetti et al. (2008)
	Maximum Feasible Subsystem (MFS) Cut Generation	This method is well suited to problems that necessitate more feasibility cuts than optimality cuts in Benders decomposition. When a feasibility cut is generated, MFS generates an additional optimality cut by identifying the maximum number of constraints that can become feasible from the infeasible SP.	Saharidis and Ierapetritou (2010)

Table 6 (continued)

Benders Decomposition Enhancement Techniques		Explanations	Studies
Major Class	**Sub Class**		
Multi-cut Benders Decomposition	Covering Cut Bundle (CCB) Generation	Using the RMP solutions, a bundle of low-density cuts (involving small part of decision variables of RMP) are generated through an auxiliary LP. The main principle in generation of multiple low-density cuts is to cover most of the decision variables of the RMP and thereby ensuring a stronger restriction on the solution space of the RMP.	Saharidis et al. (2010)
	Maximum Density Cut (MDC) Generation	MDC method can be applied either as a standalone method or together with the CCB method. The basic idea of the MDC method is to generate multiple cuts trying to cover maximum number of variables of the RMP using complementary slackness.	Saharidis and Ierapetritou (2013)
Relaxed Master Problem (RMP) Solution Techniques	LP Relaxations	The LP relaxations of the RMP are solved either in some iterations of Benders decomposition or in all iterations.	McDaniel and Devine (1977), Cordeau et al. (2000), Cordeau et al. (2006), Codato and Fischetti (2006)
	Local Branching	After each RMP is solved, local branching is applied. By this way different feasible solutions are searched and the possibility of obtaining better upper and lower bounds is increased.	Rei et al. (2008)

Table 6 (continued)

Benders Decomposition Enhancement Techniques		Explanations	Studies
Major Class	**Sub Class**		
Relaxed Master Problem (RMP) Solution Techniques	Not Solving the RMP to Optimality	The RMP is not solved to optimality, it is stopped when it finds a better solution that meets the required criterion (either a better solution than the incumbent one or (UB-ε))	Geoffrion and Graves (1974), Camargo et. al. (2008)
	Valid Inequalities	Valid inequalities are incorporated to the RMP, to increase the computational efficiency.	Cordeau et al. (2000), Cordeau et al. (2006), Saharidis et al. (2011), Tang et al. (2013)
	Heuristic Approaches	The RMP is solved heuristically.	Wentges (1998), Jiang et al. (2009), Poojari and Beasley (2009)
Sub Problem (SP) Solution Techniques	Avoiding Infeasibilities	Every time the SP is infeasible, feasibility Benders cuts have to be generated by identifying the extreme rays. To avoid these operations, the feasibility of the SP is guaranteed by incorporating problem specific feasibility ensuring cuts into the RMP formulation.	Camargo et al. (2009b)
	Decomposing the SP	The solution procedure of the SP is simplified by solving the SP for each commodity separately and then aggregating the individual solutions.	Geoffrion and Graves (1974), Camargo et al. (2009b)

Table 6 (continued)

Benders Decomposition Enhancement Techniques		Explanations	Studies
Major Class	**Sub Class**		
Sub Problem (SP) Solution Techniques	Inexact Cuts	When the SP is very large, instead of solving the SP to optimality, inexact cuts are generated through a primal-dual interior point method applied to the SP.	Zakeri et al. (1998)
	Efficient Algorithms	To speed up the SP solution time, efficient algorithms (like shortest path algorithm) are used instead of the simplex algorithm.	Wentges (1998), Marin and Jaramillo (2009), Camargo et al. (2009b)
Extensions to Classical Benders Cuts	Strengthening Benders Cuts	The strength of the Benders cuts are increased by an algorithm that modifies the optimal dual variables of the SP.	Van Roy (1986), Wentges (1998)
	Combinatorial (CB) Benders Cuts	CB cuts are alternative feasibility cuts and defined for problems whose objective function only depends on integer variables and the linkage of continuous and integer variables are in the form of big-M type logical constraints.	Codato and Fischetti (2006)
	A Priori Cut Generation	An initial set of Benders cuts are generated with the purpose of obtaining a better initial approximation of Benders reformulation and thereby reducing the required number of iterations.	Wentges (1998), Cordeau et al. (2000), Contreras et al. (2011)

Table 6 (continued)

Benders Decomposition Enhancement Techniques		Explanations	Studies
Major Class	**Sub Class**		
Modelling Structure	Alternative Formulations	In case of alternative formulations of a problem, selecting the most appropriate one can improve the computational performance of Benders decomposition.	Geoffrion and Graves (1974), Magnanti and Wong (1981), Papadakos (2009)
	Problem Size Reduction	Problem size is eliminated through reduction tests, in which the decision variables that would not appear in the optimal solution are identified using the *UB*, *LB* information and eliminated from the RMP and SP.	Contreras et al. (2011)

Table 7. List of Studies for Classical Benders Decomposition

Studies That Apply Classical Benders Decomposition
McDaniel and Devine (1977), Camargo et. al. (2008), Gelareh and Nickel (2008), Fischetti et al. (2008), Jiang et al. (2009), Marin and Jaramillo (2009), Fischetti et al. (2010), Saharidis et al. (2010), Saharidis and Ierapetritou (2010), Yang and Lee (2011), Yang and Lee (2012), Sharidis and Ierapetritou (2013), Tang et al. (2013)

Majority of the enhancements focus on the strength of the Benders cut set. By using more powerful cut sets, less number of RMPs would be required and the computational efficiency is expected to be increased. The trade-off between the required computational effort to increase the strength of the Benders cut set and the expected number of reduction in the RMPs may not be same for all problem structures or all instances. In fact, this observation holds for other enhancements presented in Table 6. Thus, before applying an enhancement the similarity of the problem structures should also be considered.

The initial Benders cuts have an important effect for the convergence of the Benders decomposition, Magnanti and Wong (1981). For most of the network optimization problems, due to network flow structure, mostly the SP is degenerate (having multiple optimal solutions). In such problems, the classical Benders decomposition generates the Benders cuts using the solution of the SP without considering the alternatives. Thus, to accelerate the Benders decomposition, selecting the strongest cut becomes a critical issue. Magnanti and Wong (1981) present the first tailored cut selection technique, in which the Pareto optimal cuts, namely the non-dominated cuts are selected through a Pareto Optimal LP using the solution of the SP and defining a core point of the RMP.

Fischetti et al. (2008) state the major drawbacks of Magnanti and Wong (1981) method as: (i) the SPs are required to be bounded, thus the method is not applicable to the unbounded cases, (ii) the quality of the cuts depend on the core points and it is difficult to define the core points, (iii) the method requires solving an additional time-consuming LP. The difficulty of defining the core points is also highlighted in Mercier (2005), Papadakos (2009), Sa et al. (2013) and an alternative way to define the core points is presented by Papadakos (2008). The enhancements of Papadakos (2008) also provide a solution procedure for the Pareto Optimal LP that does not depend on the SP solution.

Contreras et al. (2011) obtain stronger cuts through approximately solving the Pareto Optimal LP and by this way speed up the cut selection phase. Tang et al. (2013) present High Density Pareto (HDP) cut generation, in which the Pareto optimal cuts are lifted so as to include high number of decision variables of the RMP. For this purpose, the coefficient matrix of the Pareto Optimal LP is adjusted.

Fischetti et al. (2008) present the minimal infeasible subsystems (MIS) cut selection technique, in which the most violated optimality or feasibility cut is selected through a Cut Generation linear program. Cut selection is performed by establishing a correspondence between the minimal infeasible subsystems (MIS) of an infeasible LP and the vertices of the alternative polyhedron. The authors compare two variants; (i) single cut MIS, (ii) multi-cut MIS on a set of MIP problems. Multi-cut MIS forces to generate a feasibility cut also when the SP is bounded. The single cut MIS has lower CPU times on most of the instances, but for problems involving feasibility cuts multi-cut MIS becomes superior.

Sa et al. (2013) present a cut selection technique that applies the MIS cut selection scheme of Fischetti et al. (2008) and enhances the core point determination method of Papadakos (2008) to the cases when the SP is infeasible. On tree of hubs location problem, the method of Sa et al. (2013) significantly reduce the required number of iterations to converge and CPU times compared to the methods of Papadakos (2008) and Fischetti et al. (2008). In addition, on tree of hubs location problem, the MIS cut selection of Fischetti et al. (2008) outperforms the method of Papadakos (2008).

For the cases when it is difficult to obtain optimality cuts, Yang and Lee (2011) and Yang and Lee (2012) propose a tighter cut generation technique, that selects the strongest feasibility cut. In this method, the infeasible solutions are eliminated by analyzing the distance between the infeasible point and the feasible points and choosing the cut that is closer to the feasible region of the RMP. Tighter feasibility cut generation method outperforms the direct use of extreme rays in both CPU times and iteration numbers.

Another way of increasing the strength of the Benders cut set is to generate multiple cuts in each iteration. This method is widely applied in transportation problems, including hub location (Camargo et. al. (2008), Contreras et al (2011)), railroad planning problems (Cordeau et al. (2000)), logistics facility location problem with capacity expansion (Tang et al. (2013)), public transport (Gelareh and Nickel (2008)), and rapid transit network design problems (Marin and Jaramillo (2009)). In this method, the SP is decomposed into smaller problems, generally into sub SPs for each origin-destination pair. For individual solutions of the sub SPs a separate cut is added to the RMP and this method is called as disaggregated Benders decomposition.

Marin and Jaramillo (2009) effectively manage the number of constraints that are to be added in each iteration to the RMP by analyzing the disaggregated Benders cuts and eliminating the inactive ones. Contreras et al. (2011) generate multiple cuts by decomposing the SP to each potential hub node in an uncapacitated multiple allocation hub location problem. Thus, the number of sub SPs is decreased and the computational efficiency is increased compared to decomposing the SP to each *i-j* pair.

Saharidis and Ierapetritou (2010) generate multi-cuts by Maximum Feasible Subsystem (MFS) cut generation scheme for the particular problems that necessitate more feasibility cuts than optimality cuts. In this method, each time a feasibility cut is generated, an optimality cut is also generated through an auxiliary MIP that searches the minimum number of modifications that are necessary to convert an infeasible SP to a feasible one. Compared to the classical Benders decomposition, the MFS method significantly reduces CPU times on scheduling problem in a multi-purpose multi-product batch plant.

A novel multi-cut generation method that is called as Covering Cut Bundle (CCB) generation is proposed by Saharidis et al. (2010). The Benders cuts are stated as low-density cuts, namely to include a small set of decision variables of the RMP, thereby having low ability to restrict the feasible space of the RMP and necessitating lots of iterations. Considering the fact that a set of low-density cuts is stronger than the single cut that is obtained by summation of them, through an auxiliary LP a bundle of low-density cuts that cover most of the decision variables of the RMP are generated. In cut generation, the coverage of each variable is considered and priority is given to the uncovered ones. Compared to the classical Benders decomposition, the CCB significantly reduces CPU times on scheduling of crude oil problems.

Trying to cover all of the decision variables in CCB method may result in increased computational times. To overcome this, Maximum Density Cut (MDC) generation that intends to cover maximum number of decision variables of the RMP using complementary slackness theorem is proposed by Sharidis and Ierapetritou (2013). MDC generation can also be implemented together with the CCB method. On scheduling problem of multi-purpose multi-product batch plant and scheduling of crude oil problems significant improvements in CPU times are reported compared to the classical Benders decomposition.

In addition to cut selection and multi-cut Benders decomposition techniques, extensions to classical Benders cuts are also proposed in several studies. Van Roy (1986) proposes an algorithm for increasing the strength of Benders cuts obtained from the transportation SP of capacitated facility location problem. The optimal dual variables of the closed facilities are modified to obtain a stronger cut. Wentges (1998) propose alternative algorithms to strengthen the cuts on capacitated facility location problem and the proposed algorithms outperform the algorithm of Van Roy (1986).

Codato and Fischetti (2006) present combinatorial Benders (CB) cuts technique for problems whose objective function only depends on integer variables and the linkage of continuous and integer variables are in the form of big-M type logical constraints.

Instead of Benders feasibility cuts, CB cuts are added to the LP relaxation of the RMP to eliminate the minimal infeasible subsystems (MIS) from the RMP.

In Benders decomposition algorithm, the first RMP does not include any optimality or feasibility cuts. However, when it is easy to obtain feasible solutions for the main model, a priori generation of benders cuts can reduce the required Benders iterations. To generate an initial set of Benders cuts, Cordeau et al. (2000) and Cordeau et al. (2006), use a definite feasible point of the main model, and Wentges (1996) and Contreras et al. (2011) apply a heuristic algorithm.

Since the RMP is a MIP, efficient solution procedures are studied for increasing the computational performance. Geoffrion and Graves (1974) is the first study that proposes not solving the RMPs to optimality. The rationale of this technique is stated as the initial RMPs have too little information about transportation costs due to less number of cuts and need not be solved to optimality. McDaniel and Devine (1977), Cordeau et al. (2000), and Cordeau et al. (2006) solve the RMPs as an LP for the first l iterations, then solve them as an IP. Another extension for the RMP solution procedures is proposed by Rei et al. (2008), in which local branching is applied after each RMP is solved.

In order to increase the restriction on the solution space of the RMP, valid inequalities are also implemented. Cordeau et al. (2000) and Cordeau et al. (2006) use valid inequalities for strengthening the LP relaxation of the RMP. Saharidis et al. (2011) increase the convergence by using a set of valid inequalities on fixed-charge network problems. Tang et al. (2013) report significant improvements in lower bounds obtained from the RMP by using valid inequalities on logistics facility location problem with capacity expansion.

Heuristic algorithms are also used for solving the RMP. Wentges (1998) solve the RMP, which is a simple plant location problem, with a special purpose heuristic that is developed for capacitated facility location problems. Jiang et al. (2009) report up to 50% CPU time reduction on multi-product production distribution network design problems by using Tabu Search algorithm. Poojari and Beasley (2009) obtain better lower and upper bounds on a set of MIP problems with a Genetic algorithm.

Efficient solution procedures are also considered for the SP to speed up the solution time. To avoid the infeasibilities and thereby to avoid the identification of the related extreme rays of the SP, Camargo et al. (2009b) append cuts that ensure the installation of at least one hub into the RMP formulation of a hub location problem with economies of scale. To simplify the solution procedure of the SP, Geoffrion and Graves (1974) and Camargo et al. (2009b) decompose the SP to each commodity and obtain the ultimate solution by aggregating the individual solutions. For very large SPs, Zakeri et al. (1998) propose an inexact cut generation. In this method, instead of solving the SP to optimality, a feasible dual solution is generated by a primal-dual interior point method. The cuts generated through this way would be inexact but increase in computational efficiency is reported. The convergence of the algorithm for a sequence of inexact cuts is also shown.

Instead of the simplex method, efficient algorithms are also applied for solving the SP. For instance, to solve the SP Wentges (1996) use a standard transportation algorithm, Marin and Jaramillo (2009) use shortest path algorithm, and Camargo et al. (2009b) apply an inspection procedure that determines the dual variables by complementary slackness conditions and an all-pairs shortest path algorithm.

Among alternative formulations of a problem, one of them can be preferable for Benders decomposition. Geoffrion and Graves (1974) and Papadakos (2009) show that one of the alternative formulations is preferable in terms of computational efficiency. Magnanti and Wong (1981) present criterion of selecting the preferred formulations for Benders decomposition, comparing the Benders cuts of two alternative formulations using the dominance concept of Pareto optimal cuts.

Contreras et al. (2011) apply problem size reduction within Benders decomposition on uncapacitated hub location problem. Decision variables that would not appear in the optimal solution are identified using *UB* and *LB* information and eliminated from the RMP and the SP.

3.6 Generalized Benders Decomposition

GBD has a wide range of application areas including transportation (Florian and Nguyen (1974), França and Luna (1982), Federgruen and Zipkin (1984)), project scheduling (Erenguc et al. (1993)), chemical process design and control (Clasen (1984), Zhu and Kuno (2003), Liu et al. (2011)), electrical power systems planning (Benchakraun et al. (1997), Mahey et al. (2001), McCusker and Hobbs (2003), Marin and Salmeron (1998)), water resources management problems (Watkins and McKinney (1998), Cai et al. (2001)), and structural design problems arising in mechanical engineering (Munoz and Stolpe (2011)). In this section, we present a literature review of GBD solution techniques focusing on main variants and basic acceleration methods.

The solution techniques in GBD studies are presented with a classification scheme in Figure 6. Although most of the techniques in Figure 6 are only applicable to special type of problems and the main purpose of them is to extend the GBD method to the studied problem, to present a comprehensive review we include them into the classification scheme. In addition, some of the techniques are applied with the aim of accelerating the GBD algorithm and they are highlighted as enhancement techniques in Figure 6. The explanations of the GBD solution techniques together with the reviewed studies are listed in Table 8.

Most of the GBD studies focus on explicit determination of *L* functions, in order to obtain implementable algorithms. Another important issue in GBD studies is the problems involving nonconvexities, approximate approaches as well as exact GBD solution procedures are available for nonconvex NLP problems. Some of the GBD applications (i.e. nested GBD) necessitate specific problem structures. We classify such applications in “modeling structures” class.

Compared to Benders decomposition method, GBD provides a more flexible decomposition scheme since the projected y variables need not be integer. Therefore, depending on the modeling structure, alternative decompositions are available and due to different nonlinearity properties of f and G functions the decomposed problems, the RMP and the SP, may require tailored solution procedures.

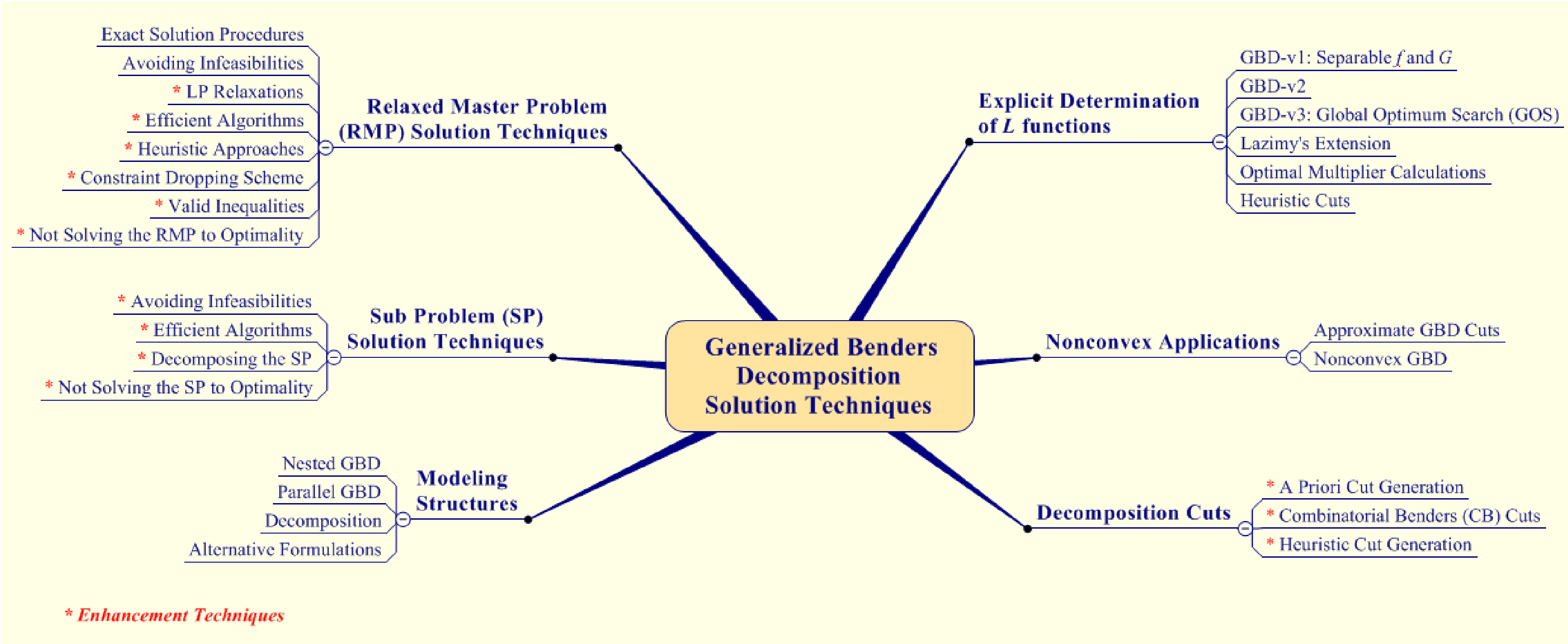


Figure 6. Classification of Main Solution Techniques for Generalized Benders Decomposition

Table 8. Explanations and Related Studies for Main Solution Techniques for Generalized Benders Decomposition

GBD Solution Techniques		Explanations	Studies
Major Class	**Sub Class**		
Explicit Determination of *L*-functions	GBD-v1: Separable f and G	This variant corresponds to the Property (P) of Geoffrion (1972). For problems, in which f and G are linearly separable in x and y, it becomes possible to determine the L functions independently of y.	Geoffrion (1972), Floudas (1995)
	GBD-v2	GBD-v2 is also applicable to the problems that do not satisfy Property (P). If the projected problem $v(y)$ is convex in y, then the optimal primal and dual solutions of $v(\overline{y})$ namely (x^*, u^*) and (x^*, λ^*) can be used to determine the functions $L^*(y,u)$ and $L_*(y,\lambda)$, respectively.	Floudas et al. (1989), Floudas and Aggarwal (1990), Floudas (1995), Watkins and McKinney (1998), Cai et al. (2001)
	GBD-v3: Global Optimum Search (GOS)	The decomposition is performed so as to satisfy the convexity assumptions and by this way global solutions are ensured to be obtained for the RMP and the SP. L functions are determined explicitly by GBD-v2. For problems with nonconvex $v(y)$, if f and G are not linearly separable in x and y, than GBD-v3 does not guarantee to obtain the global optimum solution.	Floudas et al. (1989), Floudas and Aggarwal (1990), Floudas (1995)
	Lazimy's Extension	Applying a conversion in the form of $z = \overline{y}$ is used to extend the GBD. By such an extension the nonlinearity of L functions (due to y variables) are omitted. Lazimy's extension does not require separability of f and G functions in x and y.	Lazimy (1986)

Table 8 (continued)

GBD Solution Techniques		Explanations	Studies
Major Class	**Sub Class**		
Explicit Determination of *L*-functions	Optimal Multiplier Calculations	Optimal multipliers are calculated by solving the Karush-Kuhn-Tucker optimality conditions of the SP rather than solving the SP to optimality.	Benchakraun et al. (1997)
	Heuristic Cuts	When the convexity assumptions or the Property (P) of Geoffrion (1972) do not hold for a problem, by determining the optimal multipliers with gradients heuristic GBD cuts are obtained.	McCusker and Hobbs (2003)
Modeling Structures	Nested GBD	This method is applicable to integrated models that involve separate decision problems. GBD is applied in a nested structure to an outer and an inner optimization problems. The SP of the outer optimization problem also includes an inner optimization problem, which is also solved by GBD. The solution of the outer RMP is used to solve the outer SP and the inner optimization problem.	McCusker and Hobbs (2003)
	Parallel GBD	The main problem is partitioned into separate problems that are all solved by GBD. The ultimate solution is obtained by aggregating the individual solutions of the partitioned problems.	Florian and Nguyen (1974)

Table 8 (continued)

GBD Solution Techniques		Explanations	Studies
Major Class	**Sub Class**		
Modeling Structures	Decomposition	Selection of complicating variables y is an important factor that determines the efficiency of the GBD. Moreover, in nonconvex problems, decomposition structure can also affect the convergence of the GBD algorithm.	Cai et al. (2001)
	Alternative Formulations	Considering alternative formulations of the original problem, efficient GBD applications could be obtained.	Munoz and Stolpe (2011)
Relaxed Master Problem (RMP) Solution Techniques	Exact Solution Procedures	RMP is solved by an exact solution method (i.e. branch and bound).	Erenguc et al. (1993)
	Avoiding Infeasibilities	When the RMP is infeasible, a feasible solution is obtained by an auxiliary feasibility seeking problem.	Zhu and Kuno (2003)
	LP Relaxations	The LP relaxations of the RMP are solved either in some or in all iterations.	Montemanni and Gambardella (2005)
	Efficient Algorithms	Efficient specialized algorithms are applied to solve the RMP.	Florian and Nguyen (1974), França and Luna (1982)
	Heuristic Approaches	The RMP is solved heuristically.	Hoang (1982), Sirikum et al. (2007)

Table 8 (continued)

GBD Solution Techniques		Explanations	Studies
Major Class	**Sub Class**		
Relaxed Master Problem (RMP) Solution Techniques	Constraint Dropping Scheme	The size of the RMP steadily increases as iterations progress. In this method, inactive constraints are identified and removed from the RMP so as to keep the RMP in manageable sizes and to speed up the algorithm.	Floudas and Aggarwal (1990), Marin and Salmeron (1998)
	Valid Inequalities	Valid inequalities are incorporated to the RMP, to increase the computational efficiency.	Mahey et al. (2001)
	Not Solving the RMP to Optimality	The RMP is not solved to optimality at some iterations of the GBD.	Oliveria et al. (1995), Munoz (2010), Munoz and Stolpe (2011)
Sub Problem (SP) Solution Techniques	Avoiding Infeasibilities	Every time the SP is infeasible, feasibility GBD cuts have to be generated by identifying the extreme rays. To avoid these operations either constraints of the SP are relaxed by adding to the objective function or feasibility of the SP is guaranteed by incorporating problem specific feasibility cuts into the RMP formulation.	Cai et al. (2001), Martinez-Crespo et al. (2007), Camargo et al. (2009a)
	Efficient Algorithms	Efficient specialized algorithms are applied to solve the SP.	Hoang (1982), França and Luna (1982), Erenguc et al. (1993), Mahey et al. (2001), Camargo et al. (2009a)

Table 8 (continued)

GBD Solution Techniques		Explanations	Studies
Major Class	**Sub Class**		
Sub Problem (SP) Solution Techniques	Decomposing the SP	The solution procedure of the SP is simplified by decomposing the SP into separate problems and obtaining the ultimate solution by aggregating the individual solutions of the separate SPs.	Hoang (1982), Camargo et al. (2009a)
	Not Solving the SP to Optimality	This method can be applied to the cases, in which slow convergence of the SP is observed. To speed up the SP solution time, it is sub optimized.	Clasen (1984)
Nonconvex Applications	Approximate GBD Cuts	Determining L functions by GBD-v2 technique would produce approximate cuts for problems that involve nonconvexities and do not satisfy the assumptions of GBD-v2. Although these approximate cuts are not necessarily valid for the original problem, good computational performance is reported in different studies.	Floudas et al. (1989), Floudas and Aggarwal (1990), Sahinidis and Grossman (1991), Cai et al. (2001), McCusker and Hobbs (2003)
	Nonconvex GBD	This method is developed for nonconvex stochastic MINLP problems, whose f and G functions are separable in integer and continuous variables. A convex programming problem is obtained by replacing the nonconvex functions with their convex relaxations. The derived nonconvex problem then can be solved by classical GBD.	Li et al. (2011), Li et al. (2012)

Table 8 (continued)

GBD Solution Techniques		Explanations	Studies
Major Class	**Sub Class**		
Decomposition Cuts	A Priori Cut Generation	An initial set of GBD cuts are generated with the purpose of obtaining a better initial approximation of GBD reformulation and thereby reducing the required number of iterations.	Montemanni and Gambardella (2005), Munoz (2010), Munoz and Stolpe (2011)
	Combinatorial Benders (CB) Cuts	CB cuts proposed by Codato and Fischetti (2006) are generated and added to the RMP whenever the SP is infeasible.	Munoz (2010), Munoz and Stolpe (2011)
	Heuristic Cut Generation	In this method, using an alternative formulation of the original problem a nonconvex $v(y)$ that produces heuristic GBD cuts is obtained. The GBD coefficients generated from nonconvex and convex $v(y)$ are smoothed to obtain one GBD cut (a heuristic cut).	Munoz (2010), Munoz and Stolpe (2011)

In addition to the three GBD variants (GBD-v1, GBD-v2, GBD-v3) that are presented in Appendix B, different methods are applied to determine the L functions explicitly. Lazimy (1986) presents an extension scheme for GBD that applies a conversion in the form of $z = \bar{y}$ and shows the computational benefits of the proposed extension on example problems. The Lazimy's extension provides improvements in two aspects; (i) the nonlinear L functions (due to y variables) are turned into linear functions, (ii) no further assumptions on f and G functions (i.e. the separability in x and y) are required to apply this extension.

In GBD, the SP can be an NLP that requires algorithmic solution procedures and it could be difficult to obtain optimal multipliers by optimization package solvers. Benchakraun et al. (1997) present an algorithmic solution procedure for the SP of a network design problem with underlying tree structure and obtain the optimal multipliers by solving the KKT optimality conditions of the SP.

McCusker and Hobbs (2003) present an integrated model for distributed power generation planning problems. The integrated model is solved by applying GBD twice in a nested structure. GBD is applied to the top level local planning problem, whose SP is a capacity planning and production costing problem, which is also solved by GBD. For the inner optimization problem, since convexity assumptions are not satisfied, the authors determine the optimal multipliers with gradients leading to heuristic GBD cuts. However, convergence to reasonable solutions is achieved.

Florian and Nguyen (1974) apply GBD to a problem that computes the equilibrium flows in a transportation network with elastic demands and congestion effects. The presented model has a convex objective function and is partitioned to each origin destination pair. Each partitioned problem is solved by GBD, in which the RMPs and the SPs are solved algorithmically. The ultimate solution is obtained by aggregating individual solutions. Although limited computational experiments are performed promising results are obtained.

The success of GBD depends on several aspects. The decomposed problems the RMP and the SP have to be easy to solve compared to the original problem. It must be possible to obtain optimal multipliers and extreme rays efficiently, Federgruen and Zipkin (1984). Therefore selecting the complicating y variables becomes a critical issue in GBD. Cai et al. (2001) present a supporting example for this case. On a nonconvex water resources management problem, the authors show that one decomposition structure does not even lead to convergence whereas another one produces good quality solutions.

Another aspect to be considered in modeling structures is the alternative formulations. Munoz and Stolpe (2011) report significant increase in computational times whit an alternative formulation to a structural design problem.

Erenguc et al. (1993) show the applicability of GBD to a time/cost trade-off problem with discounted cash flows. The RMP is solved by branch and bound that finds feasible solutions with a heuristic algorithm and the SP is solved algorithmically. With the proposed GBD method, fast convergence and low computational times are achieved on large number of test problems.

Zhu and Kuno (2003) present a hybrid branch and bound and GBD method to a nonconvex MINLP problem, whose RMP can turn out to be infeasible in some iterations. In order to overcome this situation, every time the RMP is infeasible, a feasible solution is obtained by solving an auxiliary feasibility seeking problem.

Many GBD studies focus on effective solution procedures for the RMP, to improve the overall computational efficiency. Montemanni and Gambardella (2005) solve LP relaxations of the RMP of a robust shortest path problem, to obtain a priori cuts before starting the main GBD algorithm. The authors experiment on the most effective number of a priori cuts that would increase the computational efficiency without increasing the RMP sizes.

When the RMP has a well-known structure, efficient algorithms are applied to increase the computational efficiency, Florian and Nguyen (1974) apply a shortest path based algorithm and França and Luna (1982) apply 0-1 based implicit enumeration algorithm. In addition, heuristic algorithms are also used. Hoang (1982) solve the RMP of a topological optimization problem heuristically and the authors remark that although it becomes difficult to obtain feasible solutions for the RMP as the gap between *UB* and *LB* decreases, good computational performance is achieved. Sirikum et al. (2007) solve the RMP of a power generation expansion planning problem with Genetic Algorithm (GA). The collaborative use of GA and GBD increases the computational efficiency especially for larger size problems.

In GBD, the size of the RMP gets larger steadily as iterations progress. To avoid excessive increase in size of the RMP and also to improve the computational performance, constraint dropping schemes are applied. Floudas and Aggarwal (1990) remove the GBD constraints that do not provide information about the main solution components. Likewise, Marin and Salmeron (1998) identify the inactive constraints and remove them from the RMP.

Mahey et al. (2001) apply GBD to capacitated multicommodity communication network flow problem, which is formulated as a MINLP. The solution performance of the RMP is increased by adding three sets of valid inequalities; (i) capacity cut set inequalities, (ii) spanning tree cuts, and (iii) efficient feasibility cuts.

Another enhancement technique applied to the RMP is not solving it to optimality. Oliveria et al. (1995) apply branch and bound method to obtain a feasible solution of the investment planning RMP of a transmission network planning problem. Munoz and Stolpe (2011) solve the RMP to optimality only every a fixed number of iterations to save computational time.

In GBD literature, the efficient solution procedures are studied for the SP as well. When the SP is infeasible, the feasibility GBD cuts have to be generated by identifying the extreme rays of the SP. To avoid these operations and to maintain the feasibility of the SP, Cai et al. (2001) relax the constraints of the SP by using slack variables and penalize them in the objective function of the SP. With this enhancement the quality of the solutions and the computational efficiency are significantly increased. Martinez-Crespo (2007) speed up the convergence of the GBD on a power scheduling problem by avoiding infeasibility of the SP. For this purpose, the authors incorporate problem specific feasibility ensuring cuts to the RMP formulation. Likewise, Camargo et al. (2009a) ensure the feasibility of the SP by adding valid inequalities to the RMP of a multiple allocation hub network design under hub congestion problem.

When the SP has a special structure well known algorithms are applied to it. Hoang (1982) apply a traffic assignment algorithm to the SP, which is a convex cost uncapacitated multicommodity flow problem. The computational efficiency is further increased by applying the algorithm to each *i-j* pair separately. Camargo et al. (2009a) decompose the SP into two separate transportation problems; (i) a linear transportation problem, (ii) a convex flow assignment transportation problem. The authors use a flow deviation algorithm for the former and solve the latter with an inspection procedure that determines the dual variables by complementary slackness conditions and the shortest path algorithm. França and Luna (1982) solve the convex stochastic transportation SP algorithmically. Mahey et al. (2001) apply a specially tailored algorithm to the convex cost multicommodity flow SPs.

Clasen (1984) apply GBD on chemical equilibrium planning problem and observe slow convergence of the SP. To overcome this, the SP is not solved to optimality and by this way the best computational performance is achieved.

Floudas et al. (1989) is the first to show that GBD can identify the global optimum even when the problem involves nonconvexities. For nonconvex NLP and MINLP problems, the authors present a Global Optimum Search (GOS) algorithm that aims to partition the main nonconvex problem into sub problems whose global optimums can be obtained efficiently. If the projected problem $v(y)$ is nonconvex, GOS does not guarantee to obtain the global optimum solution of the original problem and provide approximate GBD cuts, Floudas et al. (1989), Floudas and Aggarwal (1990). However, by using approximate GBD cuts good computational results and even convergence to global optimum is achieved in different studies, Floudas et al.

(1989), Floudas and Aggarwal (1990), Watkins and McKinney (1998), Cai et al. (2001), McCusker and Hobbs (2003).

Liu et al. (2011) extend the GBD to nonconvex stochastic MINLP problems, whose f and G functions are separable in integer and continuous variables. Using the separability property, it becomes possible to define the feasible regions of integer and continuous variables individually. Then, the nonconvex functions are replaced by their convex relaxations converting the original problem into a convex programming problem that can be solved by classical GBD. ε-convergence of the nonconvex GBD is also shown. Nonconvex GBD significantly improves the computational efficiency over optimization package solvers.

Munoz and Stolpe (2011) increase the computational performance of GBD on structural design problems by applying several enhancement techniques. A priori GBD cuts are generated by solving the continuous relaxation of the original problem. Using the dominance relationship of Magnanti and Wong (1981), the authors show that the continuous relaxation provides Pareto optimal cuts. In addition to GBD feasibility cuts, combinatorial cuts proposed by Codato and Fischetti (2005) are also generated and added to the RMP. Moreover, a GBD heuristic is proposed, in which heuristic GBD cuts are generated by applying alternative formulations.

3.7 Delay Management Problems

Delays occurring in transportation systems affect the service times and therefore the service quality. There are many causes of delays, including consolidation operations, vehicle arrivals, and unexpected break downs. The earliest studies of modeling delays in transportation systems can be found in railroad planning problems, Bodin et al. (1980), Crainic et al. (1984), Haghani (1989), and Ferreira (1997). An important amount of research is conducted on delays occurring in public transportation problems and detailed models are presented. In this section, we first briefly present the initial modeling approaches of delays that are studied in railroad planning problems and then we review the delay management studies in public transportation literature.

One of the earliest studies that model delays is presented by Bodin et al. (1980) in a railroad blocking problem. The authors model total delay cost, which is the cost of holding railcars at a yard before moving them to adjacent yards. This delay can be called as accumulation delay, the waiting time that is spent by railcars at yards until block formation.

Crainic et al. (1984) differentiate the delays occurring at yards into three classes; classification (time spent for blocking, inspection, and assembly operations), accumulation, and connection (the waiting time of railcars until the train becomes available) delays. Authors model total delay costs for each service by using a queuing model for classification delays and using cost estimates for the other delay types. Haghani (1989) model yard operations using a time space network structure, in which the queuing delays before classification operations and the connection delays are defined as links in the network.

Another classification of delays is presented by Ferreira (1997), in which delay sources are defined in three classes: (i) delays that occur due to track related problems like signal failures, (ii) delays due to train related problems like break down of a train, and (iii) delays occurring at terminals including loading/unloading operations, and crew changes.

In public transportation literature, delays due to vehicle arrivals and the resulting effects are studied extensively. When a delayed vehicle arrives at a station, there are two options for the connecting vehicles, to wait or to depart. In wait option, the passengers within the connecting vehicle and the passengers that will get on this vehicle later will be delayed. Whereas in depart option, the passengers within the delayed vehicle will miss their connection and have to wait for the next departing vehicle. Given a set of initial delays of specified vehicles at specified stations, the wait-depart decisions of all vehicles within a public transportation system is modeled as delay management problem, which is introduced by Schöbel (2001). The model minimizes the delay of all passengers and provides an updated timetable for all vehicles by determining the vehicle connections that would be maintained and that would be dropped from the current timetable. Since the delay management problem minimizes the sum of all delays over all passengers, it is called as total delay management problem by Schöbel (2006). In Schöbel (2001), procedures to calculate an upper and a lower bound for the delay management problem is also presented.

The complexity of the delay management problem is shown to be NP-hard by Gatto et al. (2005). In addition, the authors show that the restricted versions of the delay management problem, in which the maximum number of passenger transfers are limited or the cases including a specified delay of only one vehicle, are hard to solve. De Giovanni et al. (2005) present valid inequalities for the delay management problem and show that the inequalities are the facet defining inequalities of some special cases of the delay management problem. Heilporn et al. (2008) present two different MIP models for the delay management problem, which have less number of variables and propose a branch and bound algorithm and a constraint generation procedure to solve the models.

A comprehensive study on the main variants of the delay management problem and the corresponding solution procedures are presented by Schöbel (2006). The delay management problems presented in Schöbel (2006) all assume that the initial delays are given. The case, in which the delay is unknown is studied by Anderegg et al. (2009) in an online wait-depart decision problem and algorithms are proposed for two cases; (i) the case when the delay is unbounded and (ii) the case when the delay is expected to be within a specified bound.

Cicerone et al. (2008) introduce robust delay management problem for the cases that involve a single delay that can be at most α times. Authors show that the problem is NP-hard and provide a polynomial time algorithm. The robust timetabling algorithm guarantees that delay with α times would affect at most Δ arrivals and departures.

The capacities of rail tracks in delay management problem is studied by Scahatabeck and Schöbel (2008), Schöbel (2009), and Scahatabeck and Schöbel (2010). The authors present the delay management problem with priority decisions, in which the priorities of the trains using the same rail track is determined together with the wait-depart decisions and the updated timetables.

Delay management problems update an existing timetable in case of given delays. Constructing the timetables by considering the possible outcomes of the delay management problem is also studied, (Liebchen et al. (2010) and Dollevoet et al. (2012)). Liebchen et al. (2010) present a model that constructs delay resistant periodic timetables. The model optimizes the delay management and the timetabling problems in an integrated manner. The delay resistance property of the timetable is handled in the context of light robustness and modeled through analyzing the effects of different delay scenarios. Dollevoet et al. (2012) present an iterative optimization framework, in which the delay management problem is solved at macroscopic level and using the updated timetable the train scheduling problem is solved at microscopic level. The proposed framework is tested with several scenarios with different delay amounts on real-life railway data.

Dollevoet and Huisman (2013) model delay management problem by considering the rerouting decisions of passengers. Authors present three heuristic approaches and show that the one that solves the main delay management problem and reroutes the passengers in an iterative manner performs the best.

A review on the main delay management problems are presented in Table 9 including the formulation type, the objective functions, the assumptions, the decisions, and the solution procedures of the models.

Table 9. The Model Details of Main Delay Management Problems

Studies	Problem Definition	Objective Function	Assumptions	Decisions	Solution Procedures
a) Schöbel (2001) b) Gatto et al. (2005) c) De Giovanni et al. (2005) d) Schöbel (2006) e) Schöbel (2007)	Total Delay Management Problem	to minimize the delay of all passengers at their destinations	i) initial delays of specified vehicles at specified stations are known ii) the set of vehicles in the network is known with their routes and timetables iii) the set of passenger paths (including the connections from one vehicle to another) are known iv) the number of passengers using each passenger path is known	- to find wait-depart decisions for all vehicles - to calculate an updated timetable for all vehicles - to decide which connections in the passenger paths should be maintained and which can be dropped	a) a lower and an upper bounding procedure is presented b) show that the problem is NP-hard c) present valid inequalities d) linearize the model, present 2 heuristic algorithms, and propose a branch and bound algorithm together with lower and upper bounding procedures e) present path based & activity based IP models and propose a heuristic algorithm
Heilporn et al. (2008)	Total Delay Management Problem	to minimize the delay of all passengers at their destinations	assumptions ii)-iv) of Delay Management Problem are maintained v) the model is developed for a single specified delay case	Same as the total delay management problem introduced by Schöbel (2001)	* Authors present 2 models that include less number of variables than the base model * and solve the models with a branch and bound algorithm and a constraint generation procedure

Table 9 (continued)

Studies	Problem Definition	Objective Function	Assumptions	Decisions	Solution Procedures
Schöbel (2006)	Delay Management Problem with Fixed Connections	to minimize the delay of all vehicles at all stations	all assumptions i)-iv) of Delay Management Problem are maintained vi) a set of connections in passenger paths are fixed	- to find wait-depart decisions for all vehicles - to calculate an updated timetable for all vehicles	- it is shown that the constraint matrix of the main model is TU, therefore the LP relaxation would also provide an integer solution - making an analogy to the critical path method in project planning an algorithmic solution procedure is also presented
a) Schöbel (2006) b) Ginkel and Schöbel (2007)	Bicriteria Delay Management Problem	the model has two objectives: 1) to minimize the number of missed connections 2) to minimize the sum of all delays over all vehicles	assumptions i)-ii) of Delay Management Problem are maintained vii) a rough estimate of weighted passenger paths are given	Same as the total delay management problem introduced by Schöbel (2001)	a&b) show that the problem is NP-hard a&b) present an exact algorithm to identify all of the Pareto optimal solutions

Table 9 (continued)

Studies	Problem Definition	Objective Function	Assumptions	Decisions	Solution Procedures
Schöbel (2006)	General Delay Management Problem	the model has two objectives: 1) to minimize the number of customers missing a connection 2) to minimize the amount of additional delay of the remaining customers	- assumptions i)-ii) of Delay Management Problem are maintained - assumption vii) of bicriteria delay management problem is maintained	Same as the total delay management problem introduced by Schöbel (2001)	it is shown that total delay management, delay management with fixed connections, and bicriteria delay management problems are special cases of the general delay management problem
a) Cicerone et al. (2008), b) Cicerone et al. (2009)	Robust Delay Management Problem	to minimize the total travel time of all passengers	assumptions ii)-iv) of Delay Management Problem are maintained viii) the transportation system is assumed to involve a single delay that can be at most α times	a robust timetable, which guarantees that a delay with α times would affect at most Δ arrivals and departures in the transportation system	a)&b) show that the problem is NP-hard b) present a polynomial time algorithm

Table 9 (continued)

Studies	Problem Definition	Objective Function	Assumptions	Decisions	Solution Procedures
a) Scahatabeck and Schöbel (2008), b) Schöbel (2009) c) Scahatabeck and Schöbel (2010)	Delay Management with Priority Decisions	to minimize the sum of: 1) all delays of all passengers at their destinations 2) the number of missed connections	- all assumptions i)-iv) of Delay Management Problem are maintained - capacities of rail tracks are considered	- Same as the total delay management problem introduced by Schöbel (2001) - to determine the priority decisions, which set the sequence of the trains using the same rail track	a),b),c) present four heuristic algorithms that apply different rules for managing the priority decisions

CHAPTER 4

LATEST ARRIVAL CONSOLIDATED MULTICOMMODITY NETWORK DESIGN PROBLEM (LA-CMNDP)

The latest arrival consolidated multicommodity network design problem (LA-CMNDP) constructs a scheduled service network so as to minimize the maximum latest arrival time of commodities to destinations. The main entities of the service network obtained by LA-CMNDP are a consolidation plan that determines the delivery paths of commodities and a scheduling plan for each commodity including the arrival and departure times to stations. The schedules are determined by the latest arrival consolidation policy, in which the earliest departure times from stations are determined by the latest arriving commodities. The commodity flows are modeled by multicommodity network flow structure. In this chapter, we present the basic characteristics of LA-CMNDP in Section 4.1 together with the main planning issues concerned in service network design, the assumptions, and the deliverables of the model. We provide the mathematical model in Section 4.2, present the properties of the constructed service network in Section 4.3, and define the complexity of LA-CMNDP in Section 4.4.

4.1 Problem Definition

Let $G=(N,A)$ represent the physical transportation network, which is not necessarily complete. N is the set of nodes denoting the stations, where commodities are originated, consolidated, or terminated, and A is the set of arcs of the physical transportation network. The arcs that enter and emanate from station i are denoted by $In(i)$ and $Out(i)$, respectively. The set of commodities are defined by K, and each commodity $k \in K$ is defined by an origin station $O(k)$ and a destination station $D(k)$. For each commodity *k*, there is a ready time $r^k_{O(k)}$ that determines the earliest departure time of *k* from station $O(k)$.

In LA-CMNDP, commodities are consolidated at stations to form *blocks* so that they share the same routes on their delivery paths. A block consists of individual commodities, whose origin-destination pair need not be the same as the origin and the destination of the other commodities in the same block. Each arc $(i, j) \in A$ of the service network generated by LA-CMNDP represents a block, consisting of a set of commodities that are consolidated and routed together through station *i* to *j*. The transportation time from station *i* to *j* is denoted by t_{ij} and a positive handling time δ_i is associated to each station $i \in N$ for loading, unloading, and consolidation operations experienced by all departing blocks. To increase the level of consolidation throughout the service network, LA-CMNDP locates terminals that have higher consolidation capabilities compared to stations.

LA-CMNDP applies latest arrival consolidation policy, in which a consolidated block can depart from station *i* to *j* only after all commodities that will be routed through arc (i, j) arrive to station *i*. It is assumed that each station $i \in N$ can receive incoming blocks from B_i different stations $j \in In(i)$. For the stations that are selected as terminals, to increase the consolidation capabilities the B_i limits are increased by an amount of $\overline{B}_i$ so as to allow incoming blocks from all stations $j \in In(i)$. Thus, $\overline{B}_i$ values are defined as $B_i + \overline{B}_i = |In(i)|$.

The direct deliveries between stations *i* and *j* are assumed to be performed by the same vehicle and the vehicle capacities are not considered. Each commodity *k* is delivered through a single path, namely the flow of each individual commodity is unsplittable. There are no restrictions on the number of terminals and stations to be visited or the number of arcs, a delivery path can have.

Due to the latest arrival consolidation policy, the total travel time of a commodity from its origin to destination includes the waiting times spent at the stations for the late arrivals. LA-CMNDP has a minimax type objective function that minimizes the maximum latest arrival time to destinations. Therefore, each commodity might be routed through a longer path than its shortest delivery path as long as its total travel time is within the travel time of the latest arrival path. Let SP^k be the travel time along the shortest delivery path from $O(k)$ to $D(k)$. To keep the balance between the maximum latest arrival time and the total travel times of commodities, in LA-CMNDP each commodity k is routed through a path, whose travel time is at most β times $SP^k, \beta > 1.$ With this restriction, longer travel times would be avoided.

With the defined characteristics, LA-CMNDP

- designs a service network by determining which arcs of the physical transportation network are allowed to carry commodity flows,
- selects p of the stations as terminals to increase the level of consolidation,
- constructs a consolidation plan that includes an individual delivery path for each commodity, and
- determines arrival times of commodities to stations and departure times of blocks from stations, while minimizing the maximum latest arrival time to destinations.

LA-CMNDP has two types of decision variables: (i) binary network design variables y_{ij} for each $(i, j) \in A$ to construct the service network, routing variables x_{ij}^k for each $(i, j) \in A$ and $k \in K$ to construct the delivery paths of commodities, and H_i for each $i \in N$ to choose p of the stations as terminals; (ii) continuous variables DA_i^k for each $i \in N$ and $k \in K$ to represent the arrival time of commodity k to station *i*, DT_{ij} for each $(i, j) \in A$ to represents the departure time from station *i* to *j*, and LA that represents the maximum latest arrival time to destinations.

4.1.1 Derivation of Arrival and Departure Times

A representation of a consolidation plan generated by LA-CMNDP on an example transportation network consisting of 8 stations and 3 commodities is presented in Figure 7. Commodity 1 (K1) originates at station 1 and is destined to station 6, commodity 2 (K2) originates at station 2 and is destined to station 8. In delivery paths of these two commodities, they are consolidated at station 3 and transferred together from station 3 to 6.

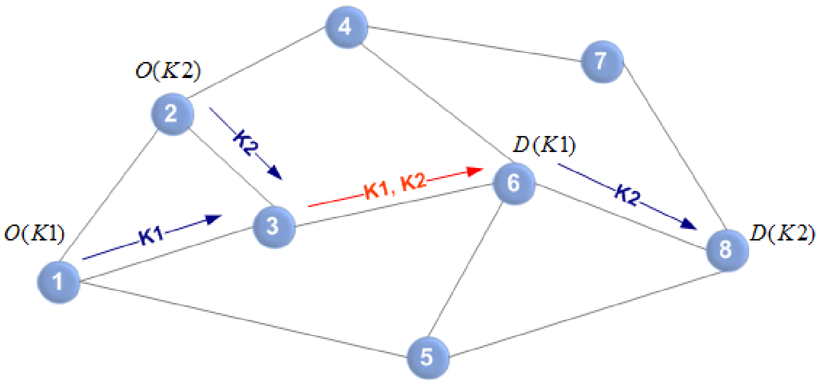


Figure 7. A Simple Representation of Consolidation in LA-CMNDP

Under the latest arrival consolidation structure, following observations can be stated for the consolidation plan presented in Figure 7.

- The block that is formed at station 3 can leave station 3, only after commodities K1 and K2 arrive to station 3.
- Since this block behaves as a single entity on its route from station 3 to 6, commodities K1 and K2 have to leave station 3 at the same time and arrive to station 6 at the same time. Therefore, there is a unique departure time DT_{ij} for all commodities departing from station i to j.
- Since station 6 is the destination of commodity K1, the block containing commodities K1 and K2 has to terminate at station 6 and commodity K2 has to be delivered to station 8 through another block. Therefore the arrival time of commodity K2 to station 6 has to be known individually to determine the departure time of a new block that contains commodity K2, which necessitates keeping arrival times to stations for all commodities separately, DA_j^k.

Arrival Times:

Each commodity k arrives to a station j through only one arc $(i,j) \in A$, (see Figure 8). The arrival time of commodity k to station j is determined by the departure time from station i to j and the travel time from station i to j by constraints (4.1). Since commodities can be included in the service network only after they are ready, the arrival time of commodities to their originating stations are set equal to their ready times by constraints (4.2).

$$DA_j^k \geq (DT_{ij} + t_{ij})x_{ij}^k \qquad \forall k \in K,\ \forall j \in N \setminus \{O(k)\}, \forall\, i \in N \setminus \{j\} \tag{4.1}$$

$$DA_{O(k)}^k = r_{O(k)}^k \qquad \forall k \in K \tag{4.2}$$

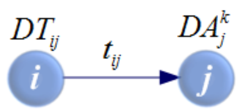


Figure 8. Arrival of Commodity k to Station j

Departure Times:

A departing block from station i to j includes a set of commodities, each of which might arrive to station i through different blocks, (see Figure 9). The departure time of a block from station i to j is determined by the latest arriving commodity that is to be placed into that block and the handling time occurring at station i by constraints (4.3).

$$DT_{ij} \geq (DA_i^k + \delta_i)x_{ij}^k \qquad \forall i \in N,\ \forall j \in N \setminus \{i\},\ \forall k \in K \tag{4.3}$$

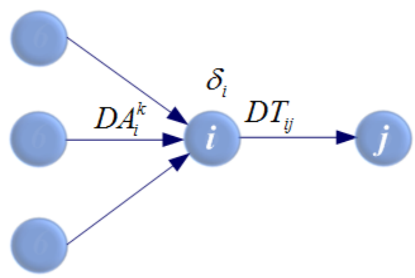


Figure 9. Departure from Station i to j

Constraints (4.1)-(4.3) construct the schedule of the service network designed by LA-CMNDP, in which the stations can be assigned directly to multiple other stations, the determination of arrival and departure times does not necessitate full-cross-traffic assumption, in which each node sends and receives flow from all other nodes of the transportation network, and there are no restrictions on the number of terminals and the stations to be visited by commodities. Whereas the latest arrival time constraints in hub location problem of Kara and Tansel (2001) are determined for single allocation of each non-hub node (spoke) to hubs, require full-cross-traffic assumption, and restrict each delivery path to consist of a sequence of nodes that have either *spoke-hub-hub-spoke* or *spoke-hub-spoke* structure.

The latest arrival model of Kara and Tansel (2001) do not include arrival time variables, rather it includes two different types of departure time variables: (i) DT_i the departure time from hub *i* to spokes and (ii) $\widehat{DT_i}$ the departure time from hub *i* to other hubs in the network. The full-cross-traffic and the single allocation assumptions result in a special structure in the departure times from hub *i*, so that for all departures to the spokes there is a unique DT_i for each hub *i* and for all departures to other hubs there is a unique $\widehat{DT_i}$ for each hub *i*.

Kara and Tansel (2001) model commodity flows by binary x_{ij} variables that equals to 1 if node *i* is assigned to hub *j* and 0 otherwise. Their model is nonlinear due to the following constraints, which determines the latest arrival time that is denoted as *Z*:

$$Z \geq (DT_i + t_{ij})x_{ij} \qquad \forall i, j \in N \tag{4.4}$$

Two different linearizations are proposed for constraints (4.4) as follows:

$$L1: \quad Z \geq DT_i + t_{ij}x_{ij} - M(1 - x_{ij}) \qquad \forall i \in N,\ \forall k \in K \tag{4.4$'$}$$

where *M* represents a large positive number and

$$L2: \quad Z \geq DT_i + t_{ij}x_{ij} \qquad \forall i \in N,\ \forall k \in K \tag{4.4$''$}$$

Kara and Tansel (2001) show that *L*1 and *L*2 are both valid for their model. The computational performance of *L*1 is stated as poor due to the *big M* structure but *L*2 performs well.

Our latest arrival time constraints (4.1) and (4.3) are also nonlinear and only $L1$ provides a valid linearization scheme for our model. Since in LA-CMNDP each node can be assigned directly to multiple other nodes, constraints (4.1) and (4.3) cannot be linearized by $L2$.

If we apply $L2$ to constraints (4.1) and (4.3):

$$DA_j^k \geq DT_{ij} + t_{ij}x_{ij}^k \qquad \forall k \in K,\ \forall j \in N \setminus \{O(k)\}, \forall\, i \in N \setminus \{j\} \qquad (4.1'')$$

$$DT_{ij} \geq DA_i^k + \delta_i x_{ij}^k \qquad \forall i \in N,\ \forall j \in N \setminus \{i\},\ \forall k \in K \qquad (4.3'').$$

For the cases $x_{ij}^k = 0,$ $(4.1'')$ and $(4.3'')$ would be $DA_j^k \geq DT_{ij}$ and $DT_{ij} \geq DA_i^k$, respectively. Therefore, $L2$ defining invalid constraints for DA and DT values.

However, if $L1$ is applied:

$$DA_j^k \geq DT_{ij} + t_{ij}x_{ij}^k - M(1 - x_{ij}^k) \qquad \forall k \in K,\ \forall j \in N \setminus \{O(k)\}, \forall\, i \in N \setminus \{j\} \qquad (4.1')$$

$$DT_{ij} \geq DA_i^k + \delta_i x_{ij}^k - M(1 - x_{ij}^k) \qquad \forall i \in N,\ \forall j \in N \setminus \{i\},\ \forall k \in K \qquad (4.3').$$

For the cases $x_{ij}^k = 0,$ $(4.1')$ and $(4.3')$ would be $DA_j^k \geq 0$ and $DT_{ij} \geq 0,$ respectively and for the cases $x_{ij}^k = 1$ $(4.1')$ and $(4.3')$ would be valid.

4.2 Mathematical Model

Using the latest arrival time constraints (4.1)-(4.3) defined in the previous section, we will present the mathematical model of LA-CMNDP as follows:

LA-CMNDP

Minimize LA

subject to

$$(4.1)-(4.3)$$

$$LA \geq DA^k_{D(k)} \qquad \forall k \in K \tag{4.5}$$

$$\sum_{(i,j)\in Out(i)} x^k_{ij} - \sum_{(j,i)\in In(i)} x^k_{ji} = \begin{cases} 1 & if\ i = O(k) \\ 0 & if\ i \neq O(k)\ and\ i \neq D(k), \\ -1 & if\ i = D(k) \end{cases} \quad \forall k \in K \tag{4.6}$$

$$x^k_{ij} \leq y_{ij} \qquad \forall (i,j) \in A,\ \forall k \in K \tag{4.7}$$

$$\sum_{(j,i)\in In(i)} y_{ji} \leq B_i + \overline{B}_i H_i \qquad \forall i \in N \tag{4.8}$$

$$\sum_{i\in N} H_i = p \tag{4.9}$$

$$\sum_{(i,j)\in A} (\delta_i + t_{ij}) x^k_{ij} \leq \beta\, SP^k \qquad \forall k \in K \tag{4.10}$$

$$x^k_{ij} \in \{0,1\},\ y_{ij} \in \{0,1\},\ H_i \in \{0,1\} \quad \forall k \in K, \forall i \in N, \forall (i,j) \in A \tag{4.11}$$

$$LA \geq 0,\ DA^k_i \geq 0,\ DT_{ij} \geq 0 \qquad \forall k \in K, \forall i \in N, \forall (i,j) \in A \tag{4.12}.$$

Constraints (4.1)-(4.3) determine the arrival and departure times as stated in detail in Section 4.1.1. Constraints (4.5) determine the maximum latest arrival time, *LA*, considering all arrivals to destinations. Constraints (4.6) are the multicommodity flow balance constraints and ensure that each commodity *k* is delivered from $O(k)$ to $D(k)$. Constraints (4.7) ensure that a commodity *k* can be sent from station *i* to *j*, if arc (i,j) is in the service network. By constraints (4.8) each station *i* can accept incoming blocks from at most B_i different stations and if station *i* is selected as a terminal, then it can accept incoming blocks from all $j \in In(i)$. Constraint (4.9) selects p of the stations as terminals. Constraints (4.10) ensure that in the optimal solution of LA-CMNDP, each commodity k is delivered through a path that is at most β times SP^k. Constraints (4.11) and (4.12) are the non negativity constraints.

Constraints (4.8) define capacity restrictions for stations by limiting the number of incoming arcs to each station and called as degree constraints by Şahin and Ahuja (2009). Constraints (4.8) are similar to the degree constraints used in yard location problem of Ahuja et al. (2005), since a consolidation center is located in both problems with the motivation of increasing the degree limits of the stations. Ahuja et al. (2005) define the degree constraints for the emanating arcs in order to model the block building capacities of yards. Whereas in LA-CMNDP, to increase the level of consolidation we define the degree constraints for the incoming arcs and increase the incoming arc limits of terminals.

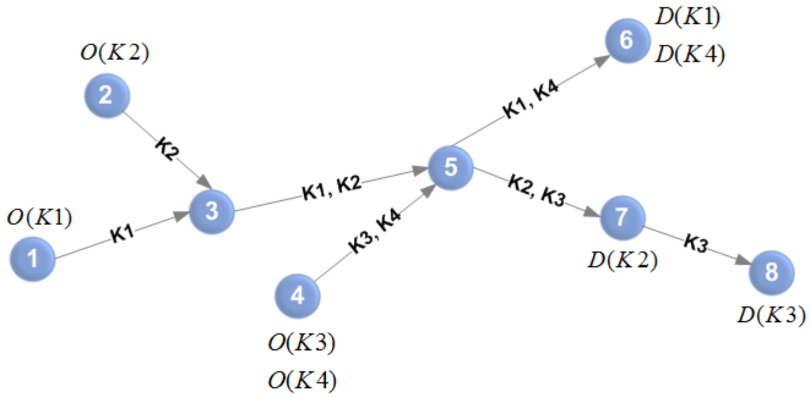
Defining capacity limits for the incoming flows is also applied in the capacitated hub location problems defined by Campbell (1994) and Ebery et al. (2000). The main reason for focusing on the incoming flow amounts is stated as after the incoming flows are consolidated at hubs they are not sorted again. Also another motivation for limiting the incoming flows could be the limited resources (handling operators) and the infrastructure (handling equipments, space limits) of consolidation centers.

In an optimum LA-CMNDP solution, to prevent long waiting times at nodes and thereby to reduce the maximum *LA* value, it may not be possible to send each commodity through its shortest path. Because, using the shortest paths for all commodities may result in higher *LA* values and for some problems even a feasible solution may not be obtained. The upper limits defined for B_i may also restrict the use of some shortest paths. Therefore, the optimum *LA* value is sure to be equal to or higher than the maximum shortest path length in the transportation network. Therefore, we can use the maximum shortest path length value as a lower bound for the optimum *LA* value of LA-CMNDP.

4.3 Properties of the LA-CMNDP Service Network

The main entities of a consolidation plan obtained by LA-CMNDP are the individual delivery paths of commodities and the consolidated paths.

Definition 4.1: A *Consolidated Path* (*CP*) is established, when the individual delivery paths of commodities form a sequence of stations that are linked to each other by arrival and departure times of blocks formed at stations. Consolidated paths may contain several commodities, whose origin and destination need not be the same as the beginning and the ending stations of the CP. An example LA-CMNDP service network having three consolidated paths is presented in Figure 10.



CP^1 : $1-3-5-6$
CP^2 : $4-5-6$
CP^3 : $2-3-5-7-8$

Figure 10. The Consolidated Paths in an Example LA-CMNDP Service Network

Definition 4.2: The *Latest Arrival Path* (*LA-Path*) is the CP that contains at least one commodity k, whose $D(k)$ is the same as the last node of the CP and $DA^k_{D(k)}$ is the maximum arrival time.

Lemma 4.1: A feasible solution of LA-CMNDP can contain at most K many different CPs if all commodities in K have distinct delivery paths.

Since we are interested in the CP that results with the highest latest arrival time for a commodity, in the remainder of the report, LA-path is used for referring to the CP that determines the maximum latest arrival time and CP is used for referring the other CPs in the service network.

The basic notations regarding an LA-Path are:

- N^{LA} : the set of nodes that are on the LA-path,
- A^{LA} : the set of arcs that are on the LA-path,
- P^{LA} : LA-path, $P^{LA} = (N^{LA}, A^{LA})$,

- K^{LA} : the set of commodities flowing on P^{LA},
- X^{LA} : the set of routing variables of $P^{LA} : X^{LA} = \{x_{ij}^{k} \mid (i,j) \in A^{LA}, k \in K^{LA}\}$,
- b^{LA} : the beginning node of P^{LA},
- e^{LA} : the ending node of P^{LA},
- i_{succ} : the successor node of node i on P^{LA},
- i_{pred} : the predecessor node of node i on P^{LA}.

An LA-path in a small sized LA-CMNDP service network consisting of 6 stations and 3 commodities is presented in Figure 11. Assume that $P^{LA} = \{1,(1,3),3,(3,5),5,(5,6),6\}$ in this network, then $K^{LA} = \{K1, K2, K3\}$.

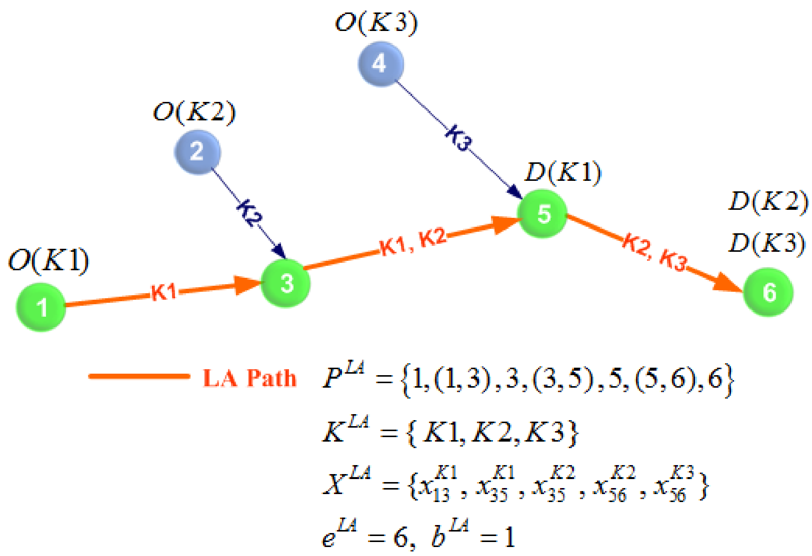


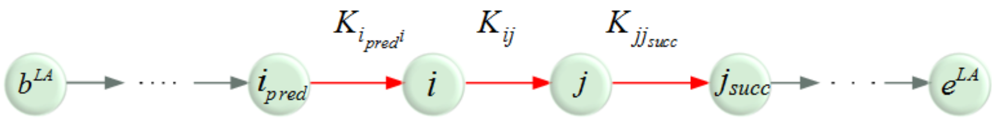
Figure 11. A Simple Representation of an LA-Path

4.3.1 Properties of an LA-Path

For a given set of routing variables X', arcs A' and nodes N' with corresponding arrival and departure times two conditions have to be satisfied to form P^{LA}; (i) linkage of commodities and (ii) linkage of arrival and departure times. *Condition 4.1* states a general condition for the linkage of commodities on any CP, and together with *Condition 4.2*, the given sets X', A', and N' define the P^{LA} and correspond to X^{LA}, A^{LA}, and N^{LA} respectively.

In order to satisfy the linkage of commodities on P^{LA}, the given routing variables $x_{ij}^{k} \in X'$ corresponding to the intermediate arcs $(i,j) \neq \{(b^{LA}, b_{succ}^{LA}), (e_{pred}^{LA}, e^{LA})\}$ of P^{LA} have to satisfy *Condition 4.1*.

Condition 4.1: For a given set of routing variables X', arcs A', and nodes N' of a transportation network, if $(i,j) \in A'$ then at least one of the commodities $k \in K$ flowing on arc (i,j) has to traverse $(i_{pred}, i) \in A'$, and at least one of the commodities $k \in K$ flowing on arc (i,j) has to traverse arc $(j, j_{succ}) \in A'$ to maintain the linkage of commodities on a CP. Let K_{ij} denote the set of commodities flowing on arc (i,j). At least one $k \in K_{ij}$ also has to be a member of $K_{i_{pred}i}$ and at least one $k \in K_{ij}$ has to be a member of $K_{jj_{succ}}$, (see Figure 12).



$$K_{ij} = \{ k \in K : x_{ij}^{k} = 1 \}$$
$$K_{i_{pred}i} = \{ k \in K : x_{i_{pred}i}^{k} = 1 \}$$
$$K_{jj_{succ}} = \{ k \in K : x_{jj_{succ}}^{k} = 1 \}$$

Figure 12. Linkage of Commodities on LA-Path

Condition 4.2: If a given set of routing variables X', arcs A', and nodes N' defines the P^{LA}, then for all $(i,j) \in A'$, $DT_{ij} = DT_{i_{pred}i} + t_{i_{pred}i} + \delta_i$.

If given sets X', A', N' together with the corresponding arrival and departure times satisfy both *Conditions 4.1* and *4.2*, then sets X', A', and N' define P^{LA}.

4.3.2 Applicability of LA-CMNDP

LA-CMNDP can only be applied to transportation networks, in which it is possible to obtain acyclic solutions. In some networks, when latest arrival consolidation structure is applied due to the parameter values of $(p, \beta, B_i, |K|)$ cyclic service networks can be obtained. In case of cycles, no feasible solution of LA-CMNDP can be constructed. We state the necessary condition for a solution to determine the applicability of LA-CMNDP in *Proposition 4.1*. We define a solution as an implementable solution, if it satisfies *Proposition 4.1*, namely if the latest arrival consolidation structure can be applied to the corresponding physical transportation network.

Before stating *Proposition 4.1*, a cycle in latest arrival service networks needs to be defined. A cycle is formed, when it is impossible to determine the departure time DT_{ij} for at least one arc $(i, j) \in A$. The departure time DT_{ij} cannot be determined, if the arrival time DA_i^k of at least one commodity $\{k \in K : x_{ij}^k = 1\}$ to node i depends on DT_{ij}. Such a situation causes a loop in the arrival and departure time calculations and the corresponding LA value equals to ∞. An example for a cyclic LA-CMNDP solution is presented in Figure 13.

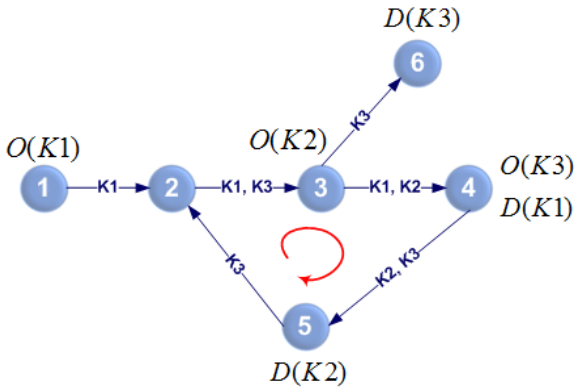


Figure 13. A Cyclic Solution

In latest arrival consolidation structure of LA-CMNDP, the transfers between node pairs are assumed to be performed by the same vehicle. If this assumption is relaxed so as to allow multiple vehicles to perform the direct rides, then no cycles would be formed. Therefore, by such an extension scheme, the latest arrival consolidation structure becomes applicable to all networks irrespective of the parameter values of $(p, \beta, B_i, |K|)$. Considering this extension, we develop a model that generates detailed vehicle schedules and routes and assigns commodity paths to vehicle routes by minimizing the sum of delays experienced by all commodities at stations due to late arrivals. We name this tactical level problem as delay management problem in service network design problems that apply latest arrival consolidation (DLA). We present the modeling details of DLA in Chapter 7.

Proposition 4.1: For an implementable LA-CMNDP solution, the service network should be acyclic.

Proof 4.1: Assume that the service network is cyclic. Then constraints (4.1)-(4.4) would result in an *LA* value of ∞ . □

To have an implementable LA-CMNDP solution, the service network established by X', A', N' should be acyclic. The formulation of LA-CMNDP does not necessitate incorporating additional sub-tour elimination constraints. However, in some problem instances, due to the parameter values of $(p, \beta, B_i, |K|)$ the LA-CMNDP service network might be cyclic and to obtain an implementable solution the parameter values should be adjusted. *Proposition 4.1* also states an important property that has to be considered while developing heuristic solution procedures for LA-CMNDP.

4.4 Complexity

In this section, we present that LA-CMNDP is NP-hard by showing that LA-CMNDP is a special case of uncapacitated multiple allocation p-hub center problem (UMApHCP), which is proven to be NP-hard by Ernst et al. (2009).

Proposition 4.2: LA-CMNDP is NP-hard, even if it is applied to a complete physical transportation network $G = (N, A)$, in which there is a positive flow between each node pair $(i, j) \in N \times N$, and $\beta = M$, $B_i = 1, \delta_i = 0$ $\forall i \in N$, $r_{O(k)} = 0$ $\forall k \in K$.

Proof 4.2:

Although the physical transportation network need not be complete for LA-CMNDP and the full-cross-traffic assumption is not required, we can apply LA-CMNP to a fully connected network, in which there is a positive flow between each node pair $(i, j) \in N \times N$ without loss-of generality. If we take $\beta = M, B_i = 1, \delta_i = 0$ $\forall i \in N, r_{O(k)} = 0$ $\forall k \in K$, then each non-terminal station *i* can only receive incoming flows from one other station *j*. Since there is a positive flow from all stations $j \in N \setminus \{i\}$ to station *i* and $B_i = 1$ for all non-terminal stations, the station *j* has to be a terminal. Therefore, each non-terminal station *i* receives incoming flows from all $j \in N \setminus \{i\}$ through one terminal. For the sake of brevity, the terminal that a station *i* is assigned to for receiving incoming flows is denoted as *H*(*i*).

Under these conditions, each non-terminal station *i* can be linked to multiple terminals in order to send flow to some stations $j \in N \setminus \{i\}$ as long as these assignments do not result in an increase in the *LA* value. The delivery paths of commodities can contain at most four stops that are in the form of *station-terminal-terminal-station* so as to minimize the maximum *LA* value, since the physical transportation network is fully connected. Flows from one terminal *l* to another terminal *m* can be directly sent from *l* to *m*, since the arc $(l, m) \in A$ and also the flows from one terminal *l* to each non-terminal station *i* can be directly sent to *H*(*i*). Since full-cross-traffic assumption holds and station *i* can only receive incoming flow from *H*(*i*), the departing vehicle from *H*(*i*) to station *i* has to wait for all the arrivals from other hubs and non-terminal stations. Therefore, the maximum *LA* value is determined by the (i, j) pair that has the longest travel time.

UMApHCP minimizes the maximum travel distance between all origin-destination pairs. Due to the minimax objective function of UMApHCP there would be multiple optimal solutions with different allocations of non-hub nodes to hubs.

Therefore, the service network structure of LA-CMNDP would correspond to one of the alternative solutions of UMApHCP, which is illustrated in Figure 14. If we take α as 1 in UMApHCP, the objective functions of both problems are determined by the maximum travel distance between all origin-destination pairs and the *LA* values of UMApHCP and LA-CMNDP would be the same. As a result, LA-CMNDP is a special case of UMApHCP and therefore LA-CMNDP is NP hard. □

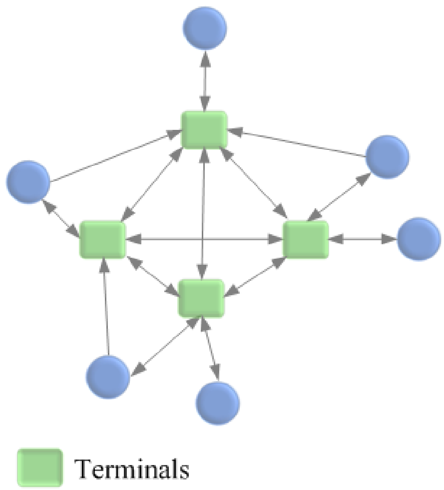


Figure 14. The Network Structure of LA-CMNDP and UMApHCP Under the Conditions Defined in Proof 4.2

CHAPTER 5

GENERALIZED BENDERS DECOMPOSITION FOR LA-CMNDP

The LA-CMNDP is a mixed integer non linear programming problem having two types of decision variables: (i) binary variables x_{ij}^k, y_{ij}, H_i, (ii) continuous variables LA, DT_{ij}, DA_i^k. When binary variables are known, the nonlinear constraints (4.1) and (4.3) are linearized, and LA-CMNDP reduces to a linear programming problem with only continuous variables. Thus, a decomposition structure that partitions LA-CMNDP into two parts; a multicommodity network design problem and a linear problem, would provide an efficient solution scheme. These observations motivated us to apply Benders decomposition to LA-CMNDP. Benders decomposition provides a simplified solution method and has many successful applications on network design and transportation problems. Due to the nonlinearity present in our model, we apply Generalized Benders Decomposition to LA-CMNDP.

In this chapter, we present the fundamentals of Generalized Benders decomposition algorithm in Sections 5.1. We present the GBD reformulation of LA-CMNDP, propose an alternative decomposition cut for LA-CMNDP in Section 5.2, and report the related computational experiments performed on a large number of test instances in Section 5.3.

5.1 Generalized Benders Decomposition Procedure

Benders decomposition is proposed by Benders in 1962 for solving mixed-integer programming problems that can be partitioned into two problems, a master problem and a sub problem. The master problem can be linear, non-linear or discrete programming problems including the complicating integer variables and the sub problem needs to be a linear programming problem. The partitioned problems are solved iteratively for finite number of times until a predetermined convergence criterion is met. For the fundamentals of Benders decomposition procedure, Appendix A can be referenced.

Geoffrion (1972) generalized Benders decomposition to nonlinear programming (NLP) problems. In this section, we present Generalized Benders Decomposition (GBD) algorithm for NLP problems having the following form:

$$
\begin{aligned}
(P) \qquad & \textit{Minimize} \quad f(x,y) \\
& \textit{subject to} \\
& \qquad G(x,y) \le 0 \\
& \qquad x \in X \subseteq \Re^{n_1} \\
& \qquad y \in Y \subseteq \Re^{n_2},
\end{aligned}
$$

where

- $G(x,y)$ represents the m-vector of constraints, defined on X and Y,
- y represents the complicating variables, namely for fixed y the original model becomes easier to solve.

GBD provides a simplified solution technique for the NLP problems, in which when y variables are fixed;

- (P) can be decomposed into separate optimization problems, each having a different subvector of x, or
- (P) reduces to a problem with a well-known special structure that has efficient solution algorithms, or
- (P) becomes convex in x, although the original (P) is nonconvex in x-y domain.

The partitioning of (P) in Benders decomposition is performed by projecting (P) onto the y-space. The details of the projection process, the finite convergence of GBD, and the variants of GBD procedure are presented in Appendix B. After the projection, for fixed y, (P) reduces to the following problem $P(y)$, which is the subproblem (SP) of GBD:

$$
\begin{array}{lll}
P(y) & Minimize & f(x,y) \\
 & subject\ to & \\
 & & G(x,y) \le 0 \\
 & & x \in X,
\end{array}
$$

and the following Generalized Benders reformulation is obtained:

$$
\begin{array}{l}
Minimize\ [\underset{u \ge 0}{\text{supremum}}[\underset{x \in X}{\text{infimum}}\,[f(x,y)+u^T G(x,y)]]] \\
subject\ to \\
\qquad \underset{x \in X}{\text{infimum}}[\lambda^T G(x,y)] \le 0 \qquad \forall \lambda \in \Lambda \\
\qquad y \in Y,
\end{array}
$$

where

- *u* is the optimal dual multipliers of $P(\overline{y})$,
- $\Lambda = \{\lambda \in R^m : \lambda \ge 0 \text{ and } \sum_{i=1}^{m} \lambda_i = 1\}$ and $\lambda \in \Lambda$ specifies the convex combination of the constraints that have no solution in *X*.

This reformulation is the Master Problem (*MP*) of (*P*). Since supremum states the *least upper bound*, the *MP* can be restated by using an auxiliary variable y_0:

$$
\begin{array}{llll}
(MP) & Minimize & y_0 & \\
 & subject\ to & & \\
 & & y_0 \ge \min_{x \in X} (f(x,y)+u^T G(x,y)) \qquad \forall u \ge 0 & (5.1) \\
 & & \min_{x \in X} (\lambda^T G(x,y)) \le 0 \qquad \forall \lambda \ge 0, \quad \lambda \in \Lambda & (5.2) \\
 & & y \in Y. &
\end{array}
$$

Constraints (5.1) and (5.2) have to be considered for all $u \ge 0$ (feasible SPs) and all extreme rays of the dual of the SP $\lambda \in \Lambda$ (infeasible SPs) resulting in a large number of constraints. Thus, the *MP* is solved considering a subset of the original set of constraints, by relaxation. When the solution of the relaxed *MP* (RMP) does not satisfy the ignored constraints, the violated constraints are generated, added to the RMP, and the RMP is solved again. This procedure continues until the optimal solution that satisfies all constraints of the *MP* or an ε-optimal solution is obtained.

To test the feasibility of an optimal RMP solution $(\hat{y}, \hat{y}_0)$ with respect to the ignored constraints and to generate the violated constraints, $P(\hat{y})$ is solved.

- If $P(\hat{y})$ is feasible, then using the optimal or a near optimal multiplier vector $\hat{u}$, a constraint in the form of (5.1) is added to the RMP.
- If $P(\hat{y})$ is infeasible, a vector $\hat{\lambda} \in \Lambda$ is obtained and a constraint in the form of (5.2) is added to the RMP.

It should be noted that constraints of type (5.1) correspond to the optimality constraints and (5.2) correspond to the feasibility constraints. $\hat{y}_0$ gives a lower bound and the objective function of $P(\hat{y})$ gives an upper bound on the optimal objective value of (*P*). Geoffrion (1972) states that the constraints (5.1) and (5.2) are usually the most (or nearly the most) violated among all violated constraints (Remark 2.4 in Geoffrion (1972)).

To state the main steps of the GBD algorithm for a minimization NLP problem, let functions $L^*(y,u)$ and $L_*(y,\lambda)$ represent the minimization problems in (5.1) and (5.2), respectively:

$$L^*(y,u) \equiv \min_{x \in X}(f(x,y) + u^T G(x,y)), \quad y \in Y, \quad u \geq 0$$

$$L_*(y,\lambda) \equiv \min_{x \in X}(\lambda^T G(x,y)), \quad y \in Y, \quad \lambda \geq 0$$

Then the RMP is:

$$\begin{array}{lll} \text{(RMP)} & \textit{Minimize} & y_0 \\ & \textit{subject to} & \\ & \quad y_0 \geq L^*(y,u_q) & q \in Q \\ & \quad L_*(y,\lambda_w) \leq 0 & w \in W \\ & \quad y \in Y. & \end{array}$$

The main steps of GBD algorithm for a minimization NLP problem are provided in Figure 15.

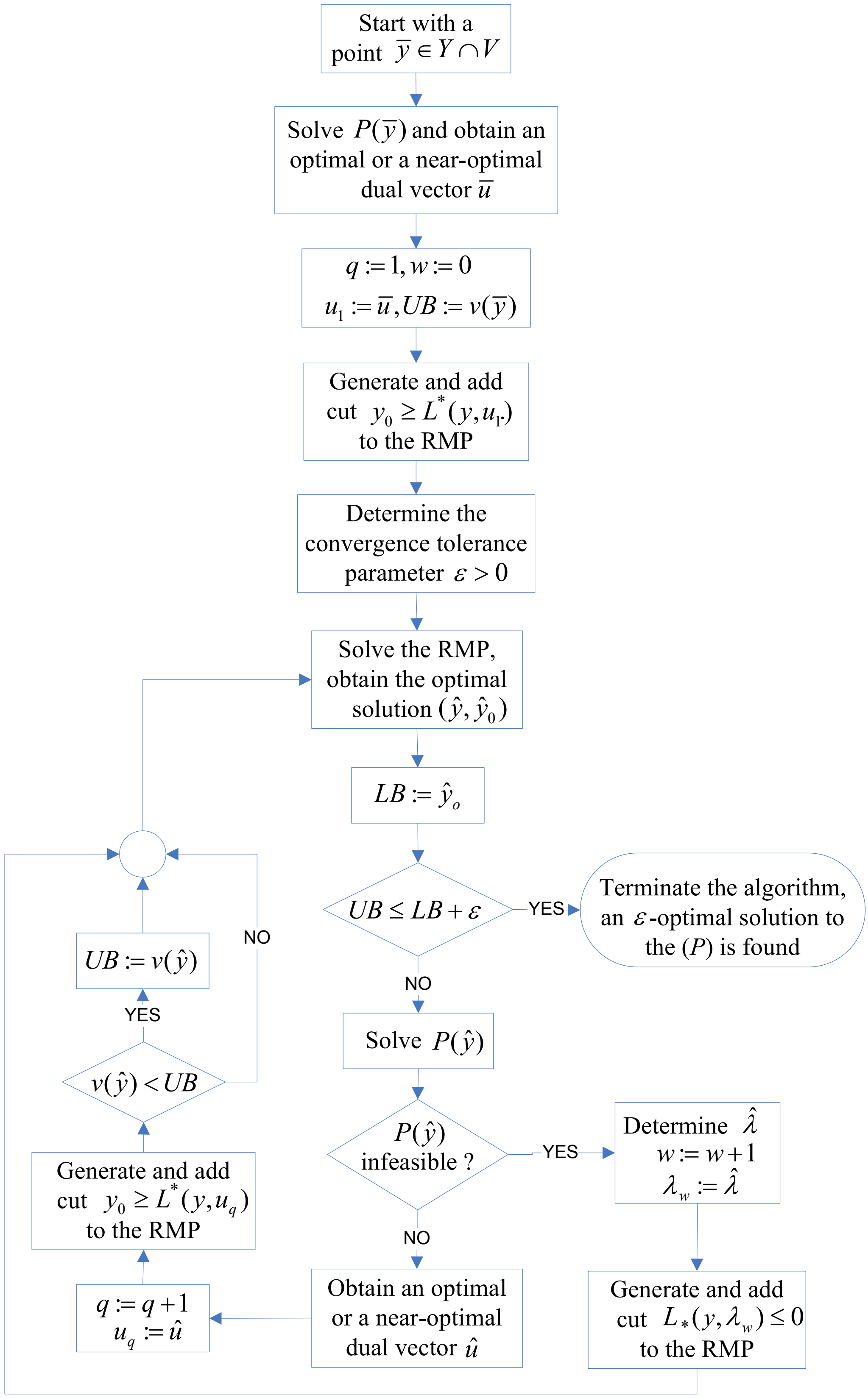


Figure 15. The Flow Chart of Generalized Benders Decomposition Algorithm

To present a brief comparison of Benders decomposition and Generalized Benders decomposition reformulations, the compact forms of the RMPs and the SPs of two methods are provided in Figure 16.

Benders Decomposition

$$\begin{array}{l} \text{Minimize } cx + f(y) \\ s.t. \\ \quad Ax + F(y) \geq b \\ \quad Dy \geq e \\ \quad x \geq 0 \\ \quad y \geq 0 \text{ and integer} \end{array}$$

Generalized Benders Decomposition

$$\begin{array}{l} \text{Minimize } f(x,y) \\ s.t. \\ \quad G(x,y) \leq 0 \\ \quad x \in X \\ \quad y \in Y \end{array}$$

RMP

$$\begin{array}{ll} \text{Minimize } f(y) + \eta & \\ s.t. & \\ \quad \eta \geq u_q^T (b - F(y)) & q \in Q \\ \quad \lambda_w^T (b - F(y)) \leq 0 & w \in W \\ \quad y \in Y,\ \eta \geq 0 & \end{array}$$

RMP

$$\begin{array}{ll} \text{Minimize } y_0 & \\ s.t. & \\ \quad y_0 \geq \underset{x \in X}{\text{minimum}} (f(x,y) + u_q^T G(x,y)) & q \in Q \\ \quad \underset{x \in X}{\text{minimum}}\, \lambda_w^T G(x,y) \leq 0 & w \in W \\ \quad y \in Y & \end{array}$$

SP

$$\begin{array}{l} \text{Minimize } cx \\ s.t. \\ \quad Ax \geq b - F(\hat{y}) \\ \quad x \geq 0 \end{array}$$

SP

$$\begin{array}{l} \text{Minimize } f(x,\hat{y}) \\ s.t. \\ \quad G(x,\hat{y}) \leq 0 \\ \quad x \in X \end{array}$$

Figure 16. Comparison of Benders and Generalized Benders Decomposition Reformulations

5.2 Generalized Benders Decomposition for LA-CMNDP

LA-CMNDP has two types of decision variables: (i) binary variables x_{ij}^k, y_{ij}, H_i and (ii) continuous variables LA, DA_i^k, DT_{ij}. We take binary variables as complicating variables, since for fixed binary variables LA-CMNDP reduces to an LP problem. In order to present the GBD reformulation of LA-CMNDP, we first state the compact form of LA-CMNDP as:

$$
\begin{aligned}
(P)\quad & \textit{Minimize} \quad f(z,y) \\
& \textit{subject to} \\
& \qquad G(z,y) \le 0 \\
& \qquad z \in Z \subseteq \Re^{n_1} \\
& \qquad y \in Y = \{0,1\}^{n_2},
\end{aligned}
$$

where

- z represents the continuous time variables LA, DA_i^k, DT_{ij},
- y represents the binary network variables x_{ij}^k, y_{ij}, H_i,
- $f(z,y) = LA$,
- $G(z,y) \le 0$ represent the *m*-vector of constraints
    - $t_{ij}x_{ij}^k - DA_j^k + x_{ij}^k DT_{ij} \le 0 \quad \forall k \in K,\ \forall j \in N \setminus \{O(k)\}, \forall\, i \in N \setminus \{j\}$,
    - $\delta_i x_{ij}^k - DT_{ij} + x_{ij}^k DA_i^k \le 0 \quad \forall i \in N,\ \forall j \in N \setminus \{i\},\ \forall k \in K$,
    - $DA_{D(k)}^k - LA \le 0 \quad \forall k \in K$,
- *Z* represents the set of constraints defined by (4.2) and (4.12),
- *Y* represents the set of constraints defined by (4.6)-(4.11),
- *Z* is a nonempty convex set,
- *Y* is a finite discrete set,
- f is convex on Z for each fixed $\bar{y} \in Y = \{0,1\}^{n_2}$, since f is a linear function,
- G is convex on Z for each fixed $\bar{y} \in Y = \{0,1\}^{n_2}$, since G is a bilinear function,
- the set $B_y = \{b \in R^m : G(z,\bar{y}) \le b$ for some $z \in Z\}$ is closed for each fixed $\bar{y} \in Y = \{0,1\}^{n_2}$, since *Z* is bounded and closed and $G(z,\bar{y})$ is continuous on z for each fixed $\bar{y} \in Y = \{0,1\}^{n_2}$.

For fixed $\bar{y} \in Y = \{0,1\}^{n_2}$, the following SP is obtained for LA-CMNDP:

$$
\begin{aligned}
P(\bar{y}): \quad & v(\bar{y}) \equiv \mathit{Minimize} \quad f(z,\bar{y}) \\
& \quad \mathit{subject\ to} \\
& \qquad G(z,\bar{y}) \le 0 \\
& \qquad z \in Z.
\end{aligned}
$$

$P(\bar{y})$ of LA-CMNDP is an LP problem and for each fixed $\bar{y} \in Y = \{0,1\}^{n_2}$, one of the following conditions hold:

- $v(\bar{y})$ is finite: then $P(\bar{y})$ gives us the optimal LA value. $v(\bar{y}) = LA$ for fixed $\bar{x}_{ij}^{k}, \bar{y}_{ij}, \bar{H}_i$ variables since $P(y)$ is an LP and $P(\bar{y})$ has an optimal multiplier vector $\hat{u}$ (dual variables of $P(\bar{y})$),
- $v(\bar{y}) = -\infty$: then we obtain the extreme ray vector $\hat{\lambda}$ from the unbounded dual of $P(\bar{y})$.

In GBD of LA-CMNDP, when $v(\bar{y})$ is finite we generate GBD optimality cuts and when $v(\bar{y}) = -\infty$ we generate GBD feasibility cuts and add each GBD cut to the following RMP of LA-CMNDP:

$$
\begin{aligned}
\text{(RMP)} \quad & \mathit{Minimize} \quad y_0 \\
& \mathit{subject\ to} \\
& \quad y_0 \ge \min_{z \in Z} \left(f(z,y) + u_q^T G(z,y) \right) && q \in Q \\
& \quad \min_{z \in Z} \left(\lambda_w^T G(z,y) \right) \le 0 && w \in W \\
& \quad y \in Y.
\end{aligned}
$$

The RMP of LA-CMDNP is a MIP problem that can be solved to optimality. So the assumptions of the GBD procedure, the details of which are presented in Appendix B are satisfied. The SP and the RMP formulations of LA-CMNDP are provided in explicit forms in Sections 5.2.1 and 5.2.2, respectively.

5.2.1 The Sub Problem

$SP(x)$ represents the compact form of the SP of LA-CMNDP. For fixed $\bar{x}$ variables that are obtained from the RMP, the SP becomes an LP problem. Taking ready times $r_{O(k)}^{k} = 0$ for $\forall k \in K$ without loss of generality, we can state the SP explicitly as:

$SP(\bar{x})$ *Minimize* LA

subject to

$$t_{ij}\bar{x}_{ij}^{k} - DA_j^k + \bar{x}_{ij}^{k} DT_{ij} \le 0 \qquad \forall k \in K,\ \forall j \in N \setminus \{O(k)\}, \forall\, i \in N \setminus \{j\} \qquad (5.3)$$

$$\delta_i \bar{x}_{ij}^{k} - DT_{ij} + \bar{x}_{ij}^{k} DA_i^k \le 0 \qquad \forall i \in N,\ \forall\, j \in N \setminus \{i\},\ \forall k \in K \qquad (5.4)$$

$$DA_{D(k)}^k - LA \le 0 \qquad \forall k \in K \qquad (5.5)$$

$$DA_{O(k)}^k = 0 \qquad \forall k \in K \qquad (5.6)$$

$$LA \ge 0,\ DA_i^k \ge 0,\ DT_{ij} \ge 0 \quad \forall k \in K, \forall i \in N, \forall (i,j) \in A \qquad (4.12).$$

We can obtain the optimal multiplier vector $\hat{u}$ by associating the dual variables π_{ij}^k to constraints (5.3), ω_{ij}^k to constraints (5.4), α^k to constraints (5.5), and μ^k to constraints (5.6) and taking the dual of SP (D-SP).

D-SP

$$\text{Maximize} \qquad \sum_{k\in K}\ \sum_{j\in N\setminus O(k)}\ \sum_{i\in N\setminus\{j\}} t_{ij}\bar{x}_{ij}^{k}\pi_{ij}^{k} + \sum_{k\in K}\sum_{i\in N}\ \sum_{j\in N\setminus\{i\}} \delta_i \bar{x}_{ij}^{k}\omega_{ij}^{k}$$

subject to

$$\sum_{k\in K}\alpha^k \le 1 \qquad (5.7)$$

$$\mu^k - \sum_{j\in N\setminus\{O(k)\}} \bar{x}_{O(k)j}^{k}\omega_{O(k)j}^{k} \le 0 \qquad \forall k \in K \qquad (5.8)$$

$$\sum_{j\in N\setminus\{D(k)\}} \pi_{jD(k)}^{k} - \sum_{j\in N\setminus\{D(k)\}} \bar{x}_{D(k)j}^{k}\omega_{D(k)j}^{k} - \alpha^k \le 0 \qquad \forall k \in K \qquad (5.9)$$

$$\sum_{j\in N\setminus\{i\}} \pi_{ji}^{k} - \sum_{j\in N\setminus\{i\}} \bar{x}_{ij}^{k}\omega_{ij}^{k} \le 0 \qquad \forall k \in K,\ \forall i \in N \setminus \{O(k), D(k)\} \qquad (5.10)$$

$$\sum_{k\in K}\omega_{ij}^{k} - \sum_{k\in K}\bar{x}_{ij}^{k}\pi_{ij}^{k} \le 0 \qquad \forall j \ne O(k),\ \forall i \in N \setminus \{j\} \qquad (5.11)$$

$$\sum_{k\in K}\omega_{ij}^{k} - \sum_{k\in K\setminus k:\{O(k)=j\}} \bar{x}_{ij}^{k}\pi_{ij}^{k} \le 0 \qquad \forall j = O(k),\ \forall i \in N \setminus \{j\} \qquad (5.12)$$

$$\pi_{ij}^k \ge 0,\ \mu^k urs,\ \omega_{ij}^k \ge 0,\ \alpha^k \ge 0 \qquad \forall i \in N,\ \forall j \in N,\ \forall k \in K \qquad (5.13)$$

In GBD algorithm,

- each time the $SP(\bar{x})$ is feasible, by using the optimal dual variables π_{ij}^{*k}, $\omega_{ij}^{*k}, \alpha^{*k}$, the following optimality cut is generated:

$$y_0 \ge \min_{LA,DA,DT\in Z}\Bigg(LA + \sum_{k\in K}\ \sum_{j\in N\setminus O(k)}\ \sum_{i\in N\setminus\{j\}} \left(\pi_{ij}^{*k}\left(t_{ij}x_{ij}^k - DA_j^k + x_{ij}^k DT_{ij}\right)\right) +$$

$$\sum_{k\in K}\sum_{i\in N}\ \sum_{j\in N\setminus\{i\}} \left(\omega_{ij}^{*k}\left(\delta_i x_{ij}^k - DT_{ij} + x_{ij}^k DA_i^k\right)\right) + \sum_{k\in K}\left(\alpha^{*k}\left(DA_{D(k)}^k - LA\right)\right)\Bigg) \qquad (5.14),$$

- each time the $SP(\bar{x})$ is infeasible, by using the extreme rays $\bar{\pi}_{ij}^k, \bar{\omega}_{ij}^k, \bar{\alpha}^k$, the following feasibility cut is generated:

$$\min_{LA,DA,DT\in Z}\left(\sum_{k\in K}\sum_{j\in N\setminus O(k)}\sum_{i\in N\setminus\{j\}}\left(\bar{\pi}_{ij}^k\left(t_{ij}x_{ij}^k - DA_j^k + x_{ij}^k DT_{ij}\right)\right)+\sum_{k\in K}\left(\bar{\alpha}^k\left(DA_{D(k)}^k - LA\right)\right)\right.$$
$$\left.\sum_{k\in K}\sum_{i\in N}\sum_{j\in N\setminus\{i\}}\left(\bar{\omega}_{ij}^k\left(\delta_i x_{ij}^k - DT_{ij} + x_{ij}^k DA_i^k\right)\right)\right)\leq 0 \qquad (5.15).$$

Optimality cuts (5.14) state a condition for the commodities that flow on the LA-path. Optimal dual variables $\pi_{ij}^{*k}, \omega_{ij}^{*k}, \alpha^{*k}$ denote which commodities have to be included in (5.14) for traversing arc $(i, j) \in A^{LA}$, for the handling time spent at node $i \in N^{LA}$, and for arriving the last node of LA-path, respectively. By *Proposition 4.1* for an implementable LA-CMNDP solution, the service network should be acyclic. Feasibility cuts (5.15) eliminate the cycles using the extreme rays $\bar{\pi}_{ij}^k, \bar{\omega}_{ij}^k, \bar{\alpha}^k$.

The minimization functions in (5.14) and (5.15) are determined by applying GBD-v2, the details of which are presented in Appendix B, since the SP is convex in x_{ij}^k:

$$L^*(x,\pi,w,\alpha) \equiv L(x, LA^*, DA^*, DT^*, \pi^*, w^*, \alpha^*),$$
$$L_*(x,\bar{\pi},\bar{\omega},\bar{\alpha}) \equiv L(x,\overline{LA},\overline{DA},\overline{DT},\bar{\pi},\bar{\omega},\bar{\alpha}).$$

Then (5.14) and (5.15) can be restated as:

$$y_0 \geq \left(LA^* + \sum_{k\in K}\sum_{j\in N\setminus O(k)}\sum_{i\in N\setminus\{j\}}\left(\pi_{ij}^{*k}\left(t_{ij}x_{ij}^k - DA_j^{*k} + x_{ij}^k DT_{ij}^*\right)\right)+\right.$$
$$\left.\sum_{k\in K}\sum_{i\in N}\sum_{j\in N\setminus\{i\}}\left(\omega_{ij}^{*k}\left(\delta_i x_{ij}^k - DT_{ij}^* + x_{ij}^k DA_i^{*k}\right)\right)+\sum_{k\in K}\left(\alpha^{*k}\left(DA_{D(k)}^{*k} - LA^*\right)\right)\right) \qquad (5.16),$$

and

$$\left(\sum_{k\in K}\sum_{j\in N\setminus O(k)}\sum_{i\in N\setminus\{j\}}\left(\bar{\pi}_{ij}^k\left(t_{ij}x_{ij}^k - \overline{DA}_j^k + x_{ij}^k \overline{DT}_{ij}\right)\right)+\sum_{k\in K}\left(\bar{\alpha}^k\left(\overline{DA}_{D(k)}^k - \overline{LA}\right)\right)\right.$$
$$\left.\sum_{k\in K}\sum_{i\in N}\sum_{j\in N\setminus\{i\}}\left(\bar{\omega}_{ij}^k\left(\delta_i x_{ij}^k - \overline{DT}_{ij} + x_{ij}^k \overline{DA}_i^k\right)\right)\right)\leq 0 \qquad (5.17).$$

5.2.2 The Relaxed Master Problem

RMP

Minimize y_0

subject to

$$y_0 \geq \Bigg(LA^{*q} + \sum_{k \in K} \left(\alpha^{*qk} \left(DA^{*qk}_{D(k)} - LA^{*q} \right) \right) +$$
$$\sum_{k \in K} \sum_{j \in N \setminus O(k)} \sum_{i \in N \setminus \{j\}} \left(\pi^{*qk}_{ij} \left(t_{ij} x^k_{ij} - DA^{*qk}_j + x^k_{ij} DT^{*q}_{ij} \right) \right) +$$
$$\sum_{k \in K} \sum_{i \in N} \sum_{j \in N \setminus \{i\}} \left(\omega^{*qk}_{ij} \left(\delta_i x^k_{ij} - DT^{*q}_{ij} + x^k_{ij} DA^{*qk}_i \right) \right) \Bigg) \qquad q \in Q \qquad (5.18)$$

$$\Bigg(\sum_{k \in K} \sum_{j \in N \setminus O(k)} \sum_{i \in N \setminus \{j\}} \left(\overline{\pi}^{wk}_{ij} \left(t_{ij} x^k_{ij} - \overline{DA}^{wk}_j + x^k_{ij} \overline{DT}^{w}_{ij} \right) \right) + \sum_{k \in K} \left(\overline{\alpha}^{wk} \left(\overline{DA}^{wk}_{D(k)} - \overline{LA}^{w} \right) \right)$$
$$\sum_{k \in K} \sum_{i \in N} \sum_{j \in N \setminus \{i\}} \left(\overline{\omega}^{wk}_{ij} \left(\delta_i x^k_{ij} - \overline{DT}^{w}_{ij} + x^k_{ij} \overline{DA}^{wk}_i \right) \right) \Bigg) \leq 0 \qquad w \in W \qquad (5.19)$$

$$\sum_{(i,j) \in Out(i)} x^k_{ij} - \sum_{(j,i) \in In(i)} x^k_{ji} = \begin{cases} 1 & if\ i = O(k) \\ 0 & if\ i \neq O(k)\ and\ i \neq D(k) \\ -1 & if\ i = D(k) \end{cases} \qquad \forall k \in K \qquad (4.6)$$

$$x^k_{ij} \leq y_{ij} \qquad \forall (i,j) \in A, k \in K \qquad (4.7)$$

$$\sum_{(j,i) \in In(i)} y_{ji} \leq B_i + \overline{B}_i H_i \qquad \forall i \in N \qquad (4.8)$$

$$\sum_{i \in N} H_i = p \qquad (4.9)$$

$$\sum_{(i,j) \in A} (\delta_i + t_{ij}) x^k_{ij} \leq \beta\, SP^k \qquad \forall k \in K \qquad (4.10)$$

$$x^k_{ij} \in \{0,1\} \quad y_{ij} \in \{0,1\} \quad H_i \in \{0,1\} \qquad \forall (i,j) \in A, k \in K \qquad (4.11)$$

The RMP is a MIP problem. y_0 is an under estimator for the optimal *LA* value of LA-CMNDP and is a lower bound on *LA*. The optimal solution of the SP LA^* provides an upper bound on *LA*. The main steps of the GBD algorithm of LA-CMNDP are stated in Figure 17. Although theoretically it is possible to obtain infeasible SPs, in GBD of LA-CMNDP, the SP turns out to be feasible at each iteration, therefore the algorithm steps related to infeasible SPs are not included in Figure 17. Since the SP only involves binary x^k_{ij} variables, to simplify the notation binary y_{ij}, H_i variables are not stated in Figure 17.

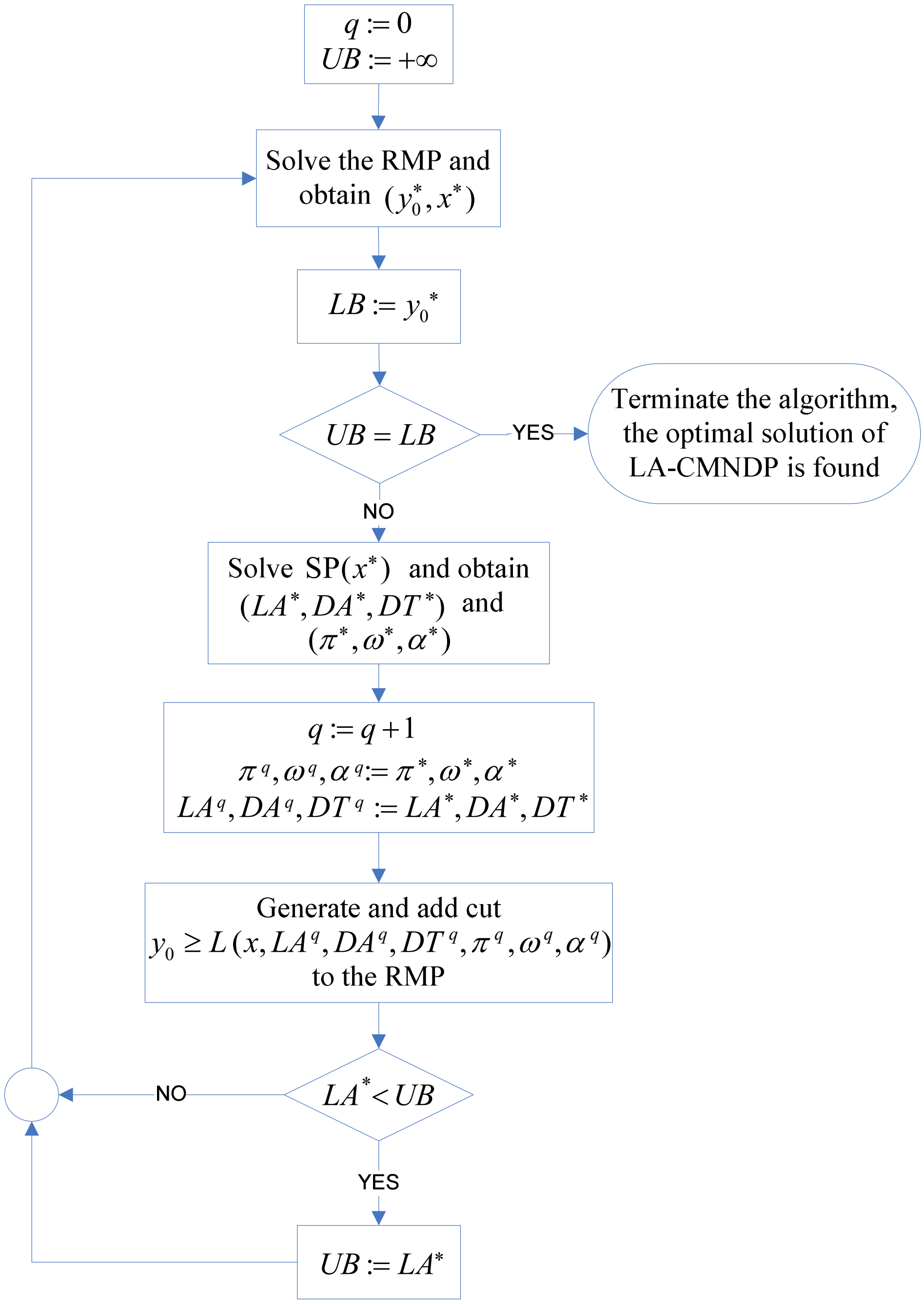


Figure 17. Main Steps of GBD Algorithm for LA-CMNDP

Before proceeding to the next section, we evaluate the applicability of enhancement techniques to our problem LA-CMNDP. Although, it is theoretically possible to obtain infeasible primal SPs, in GBD of LA-CMNDP the SP solutions always produce optimality cuts. Therefore, the enhancements regarding the feasibility cuts (i.e. Combinatorial Benders cuts) became not applicable to our problem. Since, the

SP of LA-CMNDP cannot be decomposed to sub problems we solve only one SP in each iteration.

Considering the properties of the GBD optimality cut, we develop an alternative decomposition cut for LA-CMNDP. The derivation of the alternative decomposition cut is presented in Section 5.2.3 and the computational comparison with the GBD cut is presented in Section 5.3. We also use valid inequalities in the RMP to speed up the convergence and report the application results in Section 5.3.

5.2.3 An Alternative Decomposition Cut

An alternative decomposition cut is developed for LA-CMNDP and is presented in this section in three steps. First (i) it is shown that D-SP of LA-CMNDP is Totally Unimodular, then (ii) the properties of the GBD cut $y_0 \geq L^*(x,\pi,w,\alpha)$ is stated, and (iii) the alternative decomposition cut is presented.

(i) The sufficient conditions for an $m \times n$ matrix *A* to be TU, Wolsey (1998):

" *(a)* $a_{ij} \in \{+1,-1,0\}$ *for all i, j.*

(b) Each column j contains at most two nonzero coefficients ($\sum_{i=1}^{m} |a_{ij}| \leq 2$).

(c) There exists a partition (M_1, M_2) *of the set* M *of rows such that each column j containing two nonzero coefficients satisfies* $\sum_{i \in M_1} a_{ij} - \sum_{i \in M_2} a_{ij} = 0$."

Proposition 5.1: The constraint matrix of D-SP is Totally Unimodular (TU), meaning that it always has an integral optimal solution.

Proof 5.1: Let *A* denote the constraint matrix of D-SP, whose formulation contains binary x_{ij}^k variables. Hence, *A* contains entries having binary values. Therefore, all coefficients a_{ij} of dual variables in D-SP are +1, -1, or 0 satisfying condition (a). In order for *A* to satisfy (c), the required row partition of *A* is $M_1 = A$ and $M_2 = \varnothing$. Therefore, if each column *j* of *A* satisfy (b), and $\sum_{i \in A} |a_{ij}| = 0$, then *A* will be TU.

In such a partition scheme each column of *A* can be analyzed separately:

- For dual variables $\alpha^k\ \forall k$, each corresponding column contains two non zero entries (one +1 and one -1) by constraints (5.7) and (5.9) respectively. Therefore all columns of *A* corresponding to dual variables α^k will satisfy (b) and (c).

- For dual variables $\mu^k\ \forall k$, each corresponding column contains one non zero entry (one +1) by constraints (5.8). Therefore all columns of *A* corresponding to dual variables μ^k will satisfy (b) and (c).

- For dual variables $\pi^k_{ji}\ \forall i \in N, \forall j \in N \setminus \{i\}, \forall k \in K$ each corresponding column contains one +1 by constraints (5.9) and (5.10) and one -1 by constraints (5.11) and (5.12). Therefore all columns of *A* corresponding to dual variables π^k_{ji} will satisfy (b) and (c).

- For dual variables $\omega^k_{ij}\ \forall i \in N, \forall j \in N \setminus \{i\}, \forall k \in K$ each corresponding column contains one -1 by constraints (5.8), (5.9), (5.10) and one +1 by constraints (5.11) and (5.12). Therefore all columns of *A* corresponding to dual variables ω^k_{ij} will satisfy (b) and (c). □

(ii) *The properties of the GBD cut* $y_0 \geq L^*(x, \pi, w, \alpha)$:
In iteration *l*, SP is solved fixing x^k_{ij} variables to x^{*lk}_{ij} (the optimal solution of the RMP at iteration *l*). The dual variables of the $SP(\bar{x})$ defines P^{LA}_l (the LA-path in solution x^{*lk}_{ij}) and the objective function equals to LA_l (the *LA* value of the solution x^{*lk}_{ij}). The GBD cut that is generated using the primal and the dual solutions of the $SP(\bar{x})$ includes a set of x^k_{ij} variables (X_l) that are necessary to define P^{LA}_l. It should be noted that X_l is indeed a subset of the set that includes all x^k_{ij} variables that flow on P^{LA}_l, namely $X_l \subset X^{LA}_l$.

The GBD cut of LA-CMNDP ensures that in the optimal solution of the RMP

- if every x_{ij}^k in X_l is equal to 1, then y_0 would be greater than or equal to LA_l ,
- if some of the x_{ij}^k in X_l are equal to 1, then y_0 would be greater than or equal to the maximum DA_i^k value that is attained by $x_{ij}^k = 1$ and $x_{ij}^k \in X_l$.

Proposition 5.2: The GBD cut $y_0 \geq L^*(x, \pi^q, w^q, \alpha^q)$ generated at iteration l includes the minimum set of x_{ij}^k variables that are sufficient to define P_l^{LA} .

Proof 5.2: Since the constraint matrix of D-SP is TU, D-SP always has an integral optimal solution, meaning that variables π_{ij}^k and ω_{ij}^k are either equal to 0 or 1 at optimality. For each $(i, j) \in A^{LA}$, the coefficient of π_{ij}^k in the objective function is t_{ij} for $\forall k \in K^{LA}$ and the coefficient of ω_{ij}^k is δ_i for $\forall k \in K^{LA}$. Since the objective function of D-SP provides the *LA* value and the constraint matrix of D-SP is TU;

- in the optimal solution of D-SP for each $(i, j) \in A^{LA}$ the corresponding dual variables π_{ij}^k and ω_{ij}^k can be equal to 1 for only one of the commodities $k \in K^{LA}$,
- thus the GBD cut $y_0 \geq L^*(x, \pi, w, \alpha)$ includes only one x_{ij}^k variable for traversing each $(i, j) \in A^{LA}$ and only one x_{ij}^k variable for the handling time δ_i spent at each $i \in N^{LA}$. □

(iii) *The Alternative Decomposition Cut:*

An alternative decomposition cut is developed for LA-CMNDP in a combinatorial structure. Similar to the GBD cut $y_0 \geq L^*(x, \pi, w, \alpha)$, the alternative decomposition cut also ensures that if in the optimal solution of the RMP the x_{ij}^k variables in X_l are equal to 1, then y_0 would be greater than or equal to LA_l . Since $SP(\bar{x})$ is feasible in each iteration of the GBD of LA-CMNDP, we can use the index subscript q, $(q \in Q)$ instead of *l*. Then the alternative decomposition cut can be stated as:

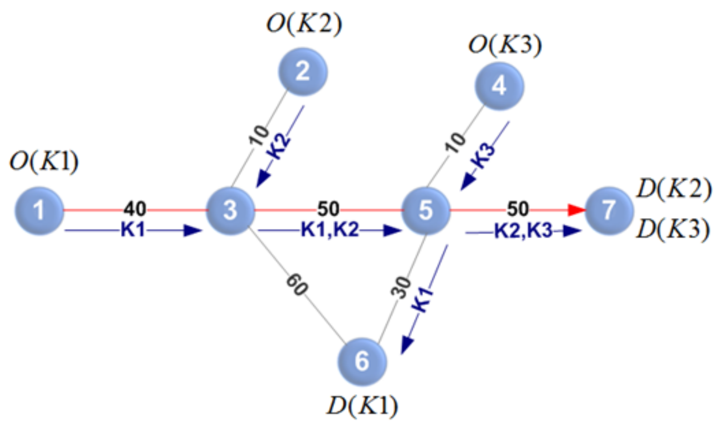
$$y_0 \geq \left(\sum_{x_{ij}^k \in X_q} x_{ij}^k - \left(|X_q| - 1 \right) \right) LA_q \qquad q \in Q \tag{5.20}$$

Proposition 5.3: The alternative decomposition cut stated by (5.20) is valid for the RMP of LA-CMNDP.

Proof 5.3: As shown in *Proof 5.2*, X_q consists of the minimum set of x_{ij}^k variables that are sufficient to define P_l^{LA}. In any feasible solution of the RMP,

- if all x_{ij}^k variables in X_q are equal to 1, then (5.20) would be $y_0 \geq LA_q$, and state the same condition on y_0 with the GBD cut (5.18).
- if at least one x_{ij}^k variable in X_q is not equal to 1, then (5.20) would be $y_0 \geq 0$, which is valid for the RMP. □

The GBD cut (5.18) and the alternative decomposition cut (5.20) are provided explicitly on a small example RMP solution presented in Figure 18.



$\delta_i = 10 \quad i = \{1,2,3,4,5,6,7\}$

Commodity	$O(k)$	$D(k)$	$Route(k)$	$DA_{D(k)}^k$
$K1$	1	2	$1-3-5-6$	150
$K2$	2	7	$2-3-5-7$	170
$K3$	4	7	$4-5-7$	170

Figure 18. A Simple Example Solution Obtained From the First Iteration of the RMP

If the D-SP is solved using the solution presented in Figure 18, then the following optimal dual variables $(\pi_{ij}^{*k}, \omega_{ij}^{*k}, \alpha^{*k})$ are obtained:

$$\begin{aligned}
&\pi_{13}^{1} = 1,\ \pi_{35}^{2} = 1,\ \pi_{57}^{3} = 1, \\
&\omega_{13}^{1} = 1,\ \omega_{35}^{1} = 1,\ \omega_{57}^{2} = 1, \\
&\alpha^{3} = 1,\ \mu^{1} = 1.
\end{aligned}$$

Using the optimal dual variables $(\pi_{ij}^{*k}, \omega_{ij}^{*k}, \alpha^{*k})$, the GBD cut for the solution presented in Figure 18 can be stated implicitly as follows:

$$\begin{aligned}
y_0 \geq (&LA^* + (DA_{D(3)}^{*3} - LA^*) + (t_{13}x_{13}^{1} - DA_3^{*1} + x_{13}^{1}DT_{13}^{*}) + (t_{35}x_{35}^{2} - DA_5^{*2} + x_{35}^{2}DT_{35}^{*}) + \\
&(t_{57}x_{57}^{3} - DA_7^{*3} + x_{57}^{3}DT_{57}^{*}) + (\delta_1 x_{13}^{1} - DT_{13}^{*} + x_{13}^{1}DA_1^{*1}) + (\delta_3 x_{35}^{1} - DT_{35}^{*} + x_{35}^{1}DA_3^{*1}) + \\
&(\delta_5 x_{57}^{2} - DT_{57}^{*} + x_{57}^{2}DA_5^{*2})) \qquad (5.21).
\end{aligned}$$

If we substitute the (LA^*, DA^*, DT^*) values into (5.21), we obtain the following GBD cut:

$$\begin{aligned}
y_0 \geq (&170 + (40x_{13}^{1} - 50 + 10x_{13}^{1}) + (50x_{35}^{2} - 110 + 60x_{35}^{2}) + \\
&(50x_{57}^{3} - 170 + 120x_{57}^{3}) + (10x_{13}^{1} - 10 + 0x_{13}^{1}) + (10x_{35}^{1} - 60 + 50x_{35}^{1}) + \\
&(10x_{57}^{2} - 120 + 110x_{57}^{2})) \qquad (5.22).
\end{aligned}$$

Using the optimal dual variables $(\pi_{ij}^{*k}, \omega_{ij}^{*k}, \alpha^{*k})$, the alternative decomposition cut for the solution presented in Figure 18 can be stated implicitly as follows:

$$y_0 \geq \left(x_{13}^{1} + x_{35}^{1} + x_{35}^{2} + x_{57}^{2} + x_{57}^{3} - 4\right)170 \qquad (5.23).$$

As shown in *Proof 5.3*, if variables $(x_{13}^{1}, x_{35}^{1}, x_{35}^{2}, x_{57}^{2}, x_{57}^{3})$ are all equal to 1 in any optimum solution of the RMP, then (5.22) and (5.23) would be the same.

5.2.4 Valid Inequalities for LA-CMNDP

Valid inequalities can be used to accelerate the solution time of the RMP. For this purpose, two sets of valid inequalities are defined:

$$\sum_{(i,j)\in Out(i)} x_{ij}^k \leq 1 \qquad \forall k \in K, \forall i \in N \tag{5.24}$$

$$\sum_{(j,i)\in In(i)} x_{ji}^k \leq 1 \qquad \forall k \in K, \forall i \in N \tag{5.25}$$

(5.24) and (5.25) are valid for LA-CMNDP, since each commodity is delivered through a single path. A commodity $k \in K$ can only enter to a node $i \in N$ by using one of the incoming arcs $(j,i) \in In(i)$ and can only leave that node by using one of the outgoing arcs $(i, j) \in Out(i)$.

5.3 Computational Experiments

Different test networks are used in hub location, multicommodity network design, and railroad blocking problems. Hub location studies generally use CAB (Civil Aeronautics Board) data set that is constructed using the Civil Aeronautics Board 1970 sample survey including the 25 intercity passenger data of United States, (O'Kelly (1986)). The most generally used data set in multicommodity network design problems is Canad data set, which is generated randomly and introduced by Crainic et al. (2000). Canad test networks include a set of problem instances having different fixed cost to variable cost ratios and different capacity levels. We do not consider CAB data set in our computational experiments, since CAB data set reflects the characteristics of passenger flows in air transportation. The focus of Canad data set is on flow amounts, capacities, and cost ratios. Since in this thesis we do not consider costs and capacities, we do not use Canad data set either.

In railroad blocking studies generally the real blocking data of different private and/or national railroad companies are used, (i.e. Crainic et al. (1984), Ahuja et al. (2005), Liu et al. (2008)). Since these data is private to companies we do not have an access to them. Through internet, some data about rail freight flow can be achieved. However, these data generally gives information about freight flow amounts according to different commodity types over a period of time.

Since the CAB, Canad, and railroad data sets does not fit our modeling concerns, we generate a new data set considering different regions of Turkey. For the inter city transportation times, we use the Turkish highway travel time data provided by Tan and Kara (2007). Cities of Turkey are provided in Appendix C together with region and license tag information. The regions and the size of the generated test networks are given in Table 10.

Table 10. The Regions and the Size of the Test Networks

Test Networks		Regions	Number of Cities
T1	T1-1	Marmara Region	11
	T1-2	Central Anatolia Region	13
T2	T2-1	Central Anatolia & Mediterranean Regions	21
	T2-2	Central Anatolia & Western and Middle Black Sea Regions	25
	T2-3	Central Anatolia & Western and Middle Black Sea & Mediterranean Regions	33
T3	T3-1	Central Anatolia & Mediterranean &Western and Middle Black Sea & Marmara & Aegean Regions	52
	T3-2	Central Anatolia & Mediterranean & Black Sea & Eastern and South Eastern Anatolia Regions	62
	T3-3	Turkey	81

In test networks, each city $i \in N$ is assumed to be connected to other cities $j \in N \setminus \{i\}$ that are within cover(i) travel time distance, Figure 19. The test networks that are generated in this way are provided in Appendix D, where the cities are indicated by their license tags.

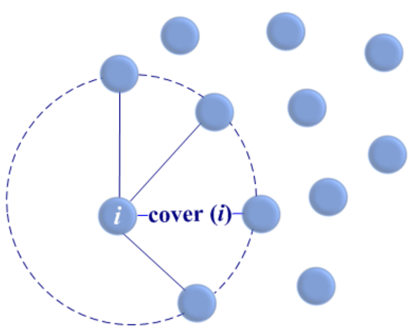


Figure 19. Connections of Stations in Test Networks

Due to terminal location constraints (4.8)-(4.9), each station $i \in N$ can accept incoming blocks from B_i different stations and can accept incoming blocks from all $j \in In(i)$ if it is chosen as a terminal. In any transportation network, if the number of arcs that are incident to station i is higher than B_i, then station i would be a candidate terminal node.

To compare the alternative decomposition cut (5.20) with the GBD cut (5.18) and to test the effectiveness of the valid inequalities (5.24) and (5.25), eight different Benders decomposition algorithms are constructed. The optimality cuts and the valid inequalities used in the algorithms are presented in Table 11.

Table 11. Variants of the Benders Decomposition Algorithms Applied to LA-CMNDP

<table>
<tr><th colspan="2">Variants of Benders Decomposition Algorithms</th><th>Valid Inequalities</th><th>Optimality Cut</th></tr>
<tr><td rowspan="4">GBD-1</td><td>GBD-1-1</td><td>No Valid Inequalities</td><td rowspan="4">GBD Optimality Cut (5.18)</td></tr>
<tr><td>GBD-1-2</td><td>(5.24)</td></tr>
<tr><td>GBD-1-3</td><td>(5.25)</td></tr>
<tr><td>GBD-1-4</td><td>(5.24) & (5.25)</td></tr>
<tr><td rowspan="4">GBD-2</td><td>GBD-2-1</td><td>No Valid Inequalities</td><td rowspan="4">Alternative Decomposition Cut (5.20)</td></tr>
<tr><td>GBD-2-2</td><td>(5.24)</td></tr>
<tr><td>GBD-2-3</td><td>(5.25)</td></tr>
<tr><td>GBD-2-4</td><td>(5.24) & (5.25)</td></tr>
</table>

All computations are performed on a computer with Intel® Core™ i7-2620M CPU @ 2.7GHz and 2.94 GB RAM. All variants of the Generalized Benders decomposition algorithms of LA-CMNDP are implemented in C# using CPLEX 12.5. To compare the GBD algorithm results with the optimal values, model LA-CMNDP is also solved to optimality by CPLEX 12.5.

To solve the SP efficiently, we develop an algorithm but it did not outperform the simplex algorithm. Thus, we solve the SPs by the simplex algorithm. In preliminary runs, we generate multiple valid GBD cuts by using the feasible points that are obtained from the RMP, but this significantly slows down the convergence and increase the number of iterations in all test instances. Therefore, in all GBD algorithms we solve the RMP to optimality.

In computational experiments of the GBD algorithms, the following parameter setting is applied in all instances. B_i is set equal to the average number of incident arcs of $\forall i \in N$, and β is taken as 1.1. Handling times δ_i are determined randomly using different lower and upper bounds considering the total amounts of flow that originates and that is destined to each station $i \in N$. Namely for the stations with higher flow amounts higher δ_i values and for the stations with lower flow amounts lower δ_i values are determined randomly.

As expected, the complexity of LA-CMNDP increases as the network size gets larger and it becomes difficult to test various levels of all parameters on all instances. Therefore, separate experimental designs are constructed for T1, T2, and T3 networks. The rest of this section is organized as follows. For each test network T1, T2, and T3, first the experimental design is explained then the analysis of the computational results is provided.

For T1 test networks, 6 instances are generated for all combinations of $(|N|, p)$ where $|N| \in \{11,13\}$ and $p \in \{2,3,4\}$. For each of these 6 instances, $|K|$ is tested at 5 levels $|K| \in \{0.20L, 0.40L, 0.60L, 0.80L, L\}$, where $L = |N| \times (N| - 1)$ and L denotes the number of commodities for the full-cross-traffic case. For each of the commodity levels $|K| \in \{0.20L, 0.40L, 0.60L, 0.80L\}$, selecting the $O(k)$, $D(k)$ pairs randomly from $k \in K$, 5 different instances are generated. Therefore, the total number of instances generated for T1 networks equals to 126. In T1 test networks, the highest p value, p=4 equals to the number of candidate terminal nodes and corresponds to imposing no limits on the number of incoming blocks to nodes.

Therefore, the T1 instances with p=4 are solved with removing the terminal location constraints (4.8)-(4.9).

In Table 12, for T1 test networks, the computational performances of GBD cut (GBD-1-1) and the alternative decomposition cut (GBD-2-1) are compared with each other and with the optimal LA-CMNDP solutions that are obtained from CPLEX. For each solution method, the CPU times are reported. The instances that do not have an implementable LA-CMNDP solution are stated with "No Solution" remark in computational results. For GBD-1-1 and GBD-2-1 algorithms, the numbers of Benders cuts that are generated are also presented in Table 12. The results corresponding to the commodity levels $|K| \in \{0.20L, 0.40L, 0.60L, 0.80L\}$ in Table 12 represent averages over 5 random instances. For T1 networks, the computational results of each random instance and the instances, in which $|K| = L$ are provided in Appendix E.

Table 12.Computational Results of GBD-1-1 and GBD-2-1 on T1 Instances

Test Problem	|N|	p	|K|	CPLEX CPU Time (s)	GBD-1-1 # of Cuts	GBD-1-1 CPU Time (s)	GBD-2-1 # of Cuts	GBD-2-1 CPU Time (s)
T1-1	11	2	22	0.5	3	0.6	3	0.4
			44	0.8	4	1.2	7	1.3
			66	1.4	6	2.5	6	1.8
			88	1.7	4	2.4	4	1.4
			110	No Solution				
			Average	**1.1**	**4**	**1.7**	**5**	**1.2**
		3	22	0.5	3	0.6	3	0.4
			44	0.8	4	1.2	7	1.3
			66	1.4	6	2.6	8	2.1
			88	1.9	6	3.1	7	2.4
			110	No Solution				
			Average	**1.1**	**5**	**1.9**	**6**	**1.5**
		4	22	0.4	3	0.6	3	0.4
			44	0.7	5	1.2	4	0.8
			66	1.3	4	1.8	4	1.2
			88	1.7	5	2.5	5	1.7
			110	2.8	7	4.3	7	2.8
			Average	**1.4**	**5**	**2.1**	**5**	**1.4**
T1-2	13	2	31	0.7	4	0.8	4	0.6
			62	1.3	7	2.7	6	1.7
			94	2.1	8	4.4	9	3.3
			125	2.3	5	3.9	6	3.3
			156	No Solution				
			Average	**1.6**	**6**	**2.9**	**6**	**2.2**
		3	31	0.6	4	0.8	4	0.6
			62	1.2	7	2.8	6	1.8
			94	2.2	8	4.7	7	2.9
			125	2.9	13	9.2	12	5.8
			156	No Solution				
			Average	**1.7**	**8**	**4.4**	**7**	**2.8**
		4	31	0.6	3	0.7	3	0.5
			62	1.3	5	2.0	5	1.4
			94	2.0	8	3.9	7	2.2
			125	2.8	9	5.9	11	4.5
			156	2.3	13	10.6	6	2.1
			Average	**1.8**	**8**	**4.6**	**6**	**2.1**

According to the results presented in Table 12, all T1 instances that have an implementable LA-CMNDP solution could be solved to optimality by GBD-1-1 and GBD-2-1. On T1 networks, GBD-2-1 reduces the CPU times compared to GBD-1-1. Only on few instances, GBD-2-1 has shorter CPU times than CPLEX. On most of the T1 instances, CPLEX provides solutions in shorter CPU times than GBD-1-1 and GBD-2-1. For the instances that correspond to the full- cross-traffic case, in which $|K|=|N|\times(|N|-1)$, an implementable LA-CMNDP solution can be obtained only when p=4, which equals to the number of candidate terminal nodes. The reason of this is that as number of commodities increase to have an implementable LA-CMNDP solution the terminal location constraints (4.8)-(4.9) are generally need to be relaxed.

The combined and the individual effects of the valid inequalities (5.24) and (5.25) are tested separately in GBD-1-1 and GBD-2-1 algorithms on T1 instances $|N| \in \{11,13\}$ with p=4 and $|K| \in \{0.20L, 0.40L, 0.60L, 0.80L, L\}$, and the computational results are presented in Table 13 and in Table 14, respectively. Since there are 5 different instances for each of the commodity levels $|K| \in \{0.20L, 0.40L, 0.60L, 0.80L\}$, valid inequalities are tested on 42 instances in total. For each of the Benders decomposition variant, the number of cuts and the CPU times in seconds are presented. The computational results of each random instance are provided in Appendix F.

Table 13. Computational Results of GBD-1 Variants on T1 Instances

Test Problem	|*N*|	|*K*|	GBD-1-1 # of Cuts	GBD-1-1 CPU Time (s)	GBD-1-2 # of Cuts	GBD-1-2 CPU Time (s)	GBD-1-3 # of Cuts	GBD-1-3 CPU Time (s)	GBD-1-4 # of Cuts	GBD-1-4 CPU Time (s)
T1-1	11	22	3	0.6	3	0.6	2	0.4	4	0.6
		44	5	1.2	3	1.0	2	0.7	4	1.0
		66	4	1.8	4	1.8	3	1.5	5	1.9
		88	5	2.6	5	2.7	4	2.5	5	2.8
		110	7	4.3	4	2.3	4	2.9	7	4.3
		Average	**5**	**2.1**	**4**	**1.7**	**3**	**1.6**	**5**	**2.1**
T1-2	13	31	3	0.7	3	0.7	3	0.7	3	0.7
		62	5	2.0	5	1.9	5	1.9	6	2.4
		94	8	3.9	7	4.0	7	4.1	5	3.2
		125	9	5.9	11	7.3	10	6.8	9	6.7
		156	13	10.6	17	15.2	10	8.6	13	11.4
		Average	**8**	**4.6**	**9**	**5.8**	**7**	**4.4**	**7**	**4.9**

Table 14. Computational Results of GBD-2 Variants on T1 Instances

Test Problem	|N|	|K|	GBD-2-1 # of Cuts	GBD-2-1 CPU Time (s)	GBD-2-2 # of Cuts	GBD-2-2 CPU Time (s)	GBD-2-3 # of Cuts	GBD-2-3 CPU Time (s)	GBD-2-4 # of Cuts	GBD-2-4 CPU Time (s)
T1-1	11	22	3	0.4	3	0.4	3	0.4	3	0.5
		44	4	0.8	4	0.8	4	0.8	4	0.8
		66	4	1.2	5	1.3	4	1.2	5	1.4
		88	5	1.7	5	1.7	4	1.6	5	1.8
		110	7	2.8	3	1.5	4	1.9	7	2.9
		Average	**5**	**1.4**	**4**	**1.2**	**4**	**1.2**	**5**	**1.5**
T1-2	13	31	3	0.5	3	0.6	3	0.5	3	0.6
		62	5	1.4	5	1.4	5	1.4	5	1.5
		94	7	2.2	6	2.4	6	2.2	7	2.7
		125	11	4.5	10	5.0	11	5.2	10	4.8
		156	12	6.5	15	8.5	14	8.2	17	10.3
		Average	**8**	**3.0**	**8**	**3.6**	**8**	**3.5**	**8**	**4.0**

According to the results presented in Table 13 and Table 14, for T1 instances adding valid inequalities in the RMP formulation result in minor decreases in the CPU times of some instances.

In T2 test networks, the number of commodities increases considerably and all GBD variants could only solve instances with limited percentage of the commodities that correspond to the full-cross-traffic case. Changing the number of terminals in a problem instance, which have low number of commodities compared to the commodity number of the full-cross-traffic case, does not have an impact on the resulting solutions. Therefore, terminal location constraints (4.8) and (4.9) of LA-CMNDP are disregarded in T2 and also in T3 computations. Each T2 test network $|N| \in \{21, 25, 33\}$ is tested for 6 levels of $|K| \in \{25, 50, 75, 100, 125, 150\}$. For each level of $|K|$, selecting the $O(k), D(k)$ pairs randomly 5 different instances are generated. Therefore, the total number of instances generated for T2 networks equals to 90.

All GBD variants are tested on T2 instances. According to the computational results, GBD-1-4 and GBD-2-4 outperform the variants GBD-1-2, GBD-1-3, GBD-2-2, and GBD-2-3. Namely, incorporating valid inequalities (5.24) and (5.25) simultaneously into the RMP produce better results than its counterparts that incorporate only one of the valid inequalities. Therefore, to evaluate the impact of the valid inequalities we report only the GBD-1-4 and GBD-2-4 variants. In Table 15 and Table 16 the computational results of GBD-1-1, GBD-1-4 and GBD-2-1, GBD-2-4 on T2 instances are reported respectively and compared with CPLEX solutions. All solution methods are implemented without imposing a time limit on them. Some of the T2 instances cannot be solved to optimality and the solution methods terminate due to out of memory restriction for these instances. For each of the solution method, the number of instances that are solved to optimality, the optimality gap of the solution method, which is stated as (*UB*-*LB*)/*UB*, and the CPU times are reported. For GBD variants, the number of cuts that are generated and the deviation of the *LA* values of the GBD variants from the *LA* value of CPLEX are also reported. The data in Table 15 and Table 16 represent the results over 5 random instances. The computational results of each random instance are provided in Appendix G.

Table 15. Computational Results of GBD-1-1 and GBD-1-4 on T2 Instances

Test Problem	$\lvert N \rvert$	$\lvert K \rvert$	CPLEX			GBD-1-1					GBD-1-4				
			# of Opt. Instances	(UB-LB)/ UB	CPU Time (s)	# of Opt. Instances	# of Cuts	% Dev. from CPLEX	(UB-LB)/ UB	CPU Time (s)	# of Opt. Instances	# of Cuts	% Dev. from CPLEX	(UB-LB)/ UB	CPU Time (s)
T2-1	21	25	5	0.0%	2.9	5	14	0.0%	0.0%	3.8	5	15	0.0%	0.0%	4.8
		50	5	0.0%	19.1	5	19	0.0%	0.0%	11.2	5	16	0.0%	0.0%	10.4
		75	5	0.0%	26.9	5	48	0.0%	0.0%	49.5	5	30	0.0%	0.0%	31.6
		100	5	0.0%	56.4	3	85	0.1%	17.6%	124.4	3	77	1.0%	9.8%	127.4
		125	5	0.0%	86.5	3	84	0.6%	13.5%	148.5	5	83	0.0%	0.0%	159.7
		150	5	0.0%	164.4	0	113	2.9%	39.2%	256.0	2	92	2.6%	29.3%	222.1
		Average	**5.0**	**0.0%**	**59.4**	**3.5**	**61**	**0.6%**	**11.7%**	**98.9**	**4.2**	**52**	**0.6%**	**6.5%**	**92.7**
T2-2	25	25	5	0.0%	4.4	5	11	0.0%	0.0%	3.8	5	11	0.0%	0.0%	4.6
		50	5	0.0%	17.8	5	47	0.0%	0.0%	33.9	5	42	0.0%	0.0%	31.7
		75	5	0.0%	32.8	4	82	0.6%	20.0%	90.9	4	86	0.2%	16.1%	116.6
		100	5	0.0%	75.2	2	113	0.3%	30.7%	193.3	0	120	1.1%	62.4%	226.9
		125	5	0.0%	261.1	0	113	1.4%	84.8%	241.5	1	112	1.1%	74.5%	269.8
		150	5	0.0%	979.6	1	98	3.4%	69.8%	235.1	1	89	3.1%	80.0%	246.4
		Average	**5.0**	**0.0%**	**228.5**	**2.8**	**77**	**0.9%**	**34.2%**	**133.1**	**2.7**	**77**	**0.9%**	**38.8%**	**149.3**
T2-3	33	25	5	0.0%	18.6	5	56	0.0%	0.0%	32.9	5	48	0.0%	0.0%	31.4
		50	5	0.0%	359.0	1	116	1.1%	62.8%	168.1	1	117	1.6%	60.8%	188.2
		75	1	16.0%	284.1	0	115	6.7%	100.0%	230.4	0	116	5.1%	100.0%	255.6
		100	1	7.8%	1,084.4	0	108	8.5%	98.2%	283.3	0	108	9.0%	100.0%	313.3
		125	0	10.8%	1,714.9	0	106	11.8%	100.0%	332.9	0	103	13.0%	89.2%	356.3
		150	0	26.6%	1,950.6	0	97	8.6%	90.9%	365.2	0	98	5.7%	100.0%	417.1
		Average	**2.0**	**10.2%**	**901.9**	**1.0**	**100**	**6.1%**	**75.3%**	**235.5**	**1.0**	**98**	**5.7%**	**75.0%**	**260.3**

Table 16. Computational Results of GBD-2-1 and GBD-2-4 on T2 Instances

			CPLEX			GBD-2-1					GBD-2-4				
Test Problem	|*N*|	|*K*|	# of Opt. Instances	(UB-LB)/ UB	CPU Time (s)	# of Opt. Instances	# of Cuts	% Dev. from CPLEX	(UB-LB)/ UB	CPU Time (s)	# of Opt. Instances	# of Cuts	% Dev. from CPLEX	(UB-LB)/ UB	CPU Time (s)
T2-1	21	25	5	0.0%	2.9	5	14	0.0%	0.0%	2.6	5	15	0.0%	0.0%	3.4
		50	5	0.0%	19.1	5	19	0.0%	0.0%	7.1	5	20	0.0%	0.0%	8.9
		75	5	0.0%	26.9	5	45	0.0%	0.0%	26.2	5	30	0.0%	0.0%	20.3
		100	5	0.0%	56.4	4	125	0.0%	20.0%	113.8	3	99	0.5%	40.0%	106.6
		125	5	0.0%	86.5	3	118	0.0%	40.0%	131.2	4	97	0.3%	20.0%	125.5
		150	5	0.0%	164.4	1	167	3.0%	80.0%	245.2	3	100	1.3%	40.0%	161.6
		Average	**5.0**	**0.0%**	**59.4**	**3.8**	**81**	**0.5%**	**23.3%**	**87.7**	**4.2**	**60**	**0.4%**	**16.7%**	**71.0**
T2-2	25	25	5	0.0%	4.4	5	11	0.0%	0.0%	2.5	5	11	0.0%	0.0%	3.1
		50	5	0.0%	17.8	5	41	0.0%	0.0%	17.8	5	45	0.0%	0.0%	24.6
		75	5	0.0%	32.8	3	120	0.5%	40.0%	88.6	4	98	0.2%	20.0%	86.3
		100	5	0.0%	75.2	2	154	1.6%	60.0%	161.0	1	180	1.4%	80.0%	226.5
		125	5	0.0%	261.1	0	174	1.1%	100.0%	243.1	1	163	1.4%	80.0%	263.6
		150	5	0.0%	979.6	0	168	3.2%	100.0%	285.0	1	134	2.1%	80.0%	304.8
		Average	**5.0**	**0.0%**	**228.5**	**2.5**	**111**	**1.1%**	**50.0%**	**133.0**	**2.8**	**105**	**0.8%**	**43.3%**	**151.5**
T2-3	33	25	5	0.0%	18.57	5	58	0.0%	0.0%	22.0	5	56	0.0%	0.0%	24.8
		50	5	0.0%	359.0	2	153	0.8%	60.0%	125.1	2	167	1.3%	60.0%	162.7
		75	1	16.0%	284.1	0	174	5.4%	100.0%	233.0	0	173	5.3%	100.0%	271.0
		100	1	7.8%	1,084.4	0	167	7.8%	100.0%	311.2	0	165	8.6%	100.0%	364.9
		125	0	10.8%	1,714.9	0	160	12.7%	100.0%	372.2	0	154	13.6%	100.0%	421.4
		150	0	26.6%	1,950.6	0	151	5.6%	100.0%	422.2	0	147	5.9%	100.0%	493.0
		Average	**2.0**	**10.2%**	**901.9**	**1.2**	**144**	**5.4%**	**76.7%**	**247.6**	**1.2**	**144**	**5.8%**	**76.7%**	**289.6**

According to the results presented in Table 15 and Table 16, with respect to the number of optimum instances and the percentage deviation from CPLEX values, GBD-2-1 outperforms GBD-1-1 for $|N|=\{21,33\}$ and GBD-1-1 outperforms GBD-2-1 for $|N|=25$. By incorporating valid inequalities into the RMP, the optimum solutions could be obtained on average for 5 more T2 instances by GBD-1-4 and for 6 more T2 instances by GBD-2-4. As network size gets larger, especially for the instances with $|N|=25$, $|K|=150$ and $|N|=33$, $|K|\geq 100$ GBD variants produce a solution that is on average %8 higher than the CPLEX solution in considerably shorter CPU times.

For T3 test networks, 12 instances are generated for all combinations of $(|N|,|K|)$ where $|N|\in\{52,62,81\}$ and $|K|\in\{50,100,150,200\}$. For each of the commodity levels 5 different instances are generated selecting the $O(k), D(k)$ pairs randomly. Therefore, the total number of instances generated for T3 networks equals to 60.

Since incorporating both of the valid inequalities outperform the individual uses as number of nodes and commodities increase, on T3 instances GBD-1-4 and GBD-2-4 are tested. In Table 17 and Table 18 the computational results of GBD-1-1, GBD-1-4 and GBD-2-1, GBD-2-4 on T3 instances are reported respectively and compared with CPLEX solutions. The comparison structure of Table 15 and Table 16 is used basically in Table 17 and Table 18. Since none of the instances could be solved to optimality by any of the solution methods, the number of instances in which a feasible solution could be obtained is reported instead. As number of commodities increase no feasible solution could be obtained by CPLEX. To evaluate the solution quality of GBD algorithms on these instances, the maximum shortest path length values are used as lower bounds as the details are explained in Chapter 4. All solution methods are implemented without imposing a time limit on them. The data in Table 17 and Table 18 represent the results over 5 random instances. The computational results of each random instance are provided in Appendix H.

Table 17. Computational Results of GBD-1-1 and GBD-1-4 on T3 Instances

			CPLEX			GBD-1-1						GBD-1-4					
Test Problem	|*N*|	|*K*|	# of Instances with a Solution	(UB-LB) /UB	CPU Time (s)	# of Instances with a Solution	# of Cuts	% Dev. from CPLEX	% Dev. from Max. SP Length	(UB-LB) /UB	CPU Time (s)	# of Instances with a Solution	# of Cuts	% Dev. from CPLEX	% Dev. from Max. SP Length	(UB-LB) /UB	CPU Time (s)
T3-1	52	50	5	55.3%	492	5	101	2.2%	2.5%	91.6%	415	5	100	1.3%	1.7%	87.9%	440
		100	5	54.4%	541	5	71	-3.5%	9.2%	96.8%	586	5	72	-1.0%	11.9%	100.0%	631
		150	3	60.5%	2,388	5	44	9.6%	30.3%	98.1%	574	5	47	9.2%	28.2%	96.0%	660
		200	0	-	555	5	45	-	38.8%	79.0%	794	5	43	-	35.3%	91.8%	800
		Ave.	**3.3**	**56.7%**	**994**	**5.0**	**65**	**2.8%**	**20.2%**	**91.4%**	**592**	**5.0**	**65**	**3.2%**	**19.3%**	**93.9%**	**633**
T3-2	62	50	5	57.4%	490	5	89	1.6%	3.3%	100.0%	434	5	90	1.6%	3.3%	100.0%	479
		100	3	56.0%	2,202	5	55	2.7%	6.0%	100.0%	537	5	58	2.1%	5.4%	100.0%	629
		150	0	-	3,042	5	49	-	13.2%	100.0%	740	5	51	-	12.1%	100.0%	845
		200	0	-	407	5	29	-	24.5%	100.0%	608	5	31	-	16.8%	100.0%	706
		Ave.	**2.0**	**56.7%**	**1,535**	**5.0**	**55**	**2.1%**	**11.8%**	**100.0%**	**580**	**5.0**	**57**	**1.8%**	**9.4%**	**100.0%**	**665**
T3-3	81	50	5	74.5%	1,344	5	78	-4.8%	4.7%	100.0%	523	5	79	-2.6%	5.2%	100.0%	596
		100	0	-	1,502	5	51	-	13.2%	100.0%	759	5	53	-	13.8%	100.0%	873
		150	0	-	152	5	27	-	18.8%	99.5%	640	5	28	-	20.7%	100.0%	745
		200	0	-	87	5	21	-	25.8%	98.9%	699	5	20	-	24.6%	100.0%	750
		Ave.	**1.3**	**74.5%**	**771**	**5.0**	**45**	**-4.8%**	**15.6%**	**99.6%**	**655**	**5.0**	**45**	**-2.6%**	**16.1%**	**100.0%**	**741**

Table 18. Computational Results of GBD-2-1- and GBD-2-4 on T3 Instances

			CPLEX			GBD-2-1						GBD-2-4					
Test Problem	**|*N*|**	**|*K*|**	**# of Instances with a Solution**	**(UB-LB) /UB**	**CPU Time (s)**	**# of Instances with a Solution**	**# of Cuts**	**% Dev. from CPLEX**	**% Dev. from Max. SP Length**	**(UB-LB) /UB**	**CPU Time (s)**	**# of Instances with a Solution**	**# of Cuts**	**% Dev. from CPLEX**	**% Dev. from Max. SP Length**	**(UB-LB) /UB**	**CPU Time (s)**
T3-1	52	50	5	55.3%	492	5	158	2.0%	2.3%	100%	468	5	155	0.6%	0.9%	100%	519
		100	5	54.4%	541	5	120	0.1%	13.1%	100%	757	5	114	0.3%	13.2%	100%	813
		150	3	60.5%	2,388	5	77	8.7%	24.7%	100%	791	5	80	10.0%	25.2%	100%	906
		200	0	-	555	5	82	-	36.4%	100%	1,160	5	75	-	36.5%	100%	1,146
		Ave.	**3.3**	**56.7%**	**994**	**5.0**	**109**	**3.6%**	**19.1%**	**100%**	**794**	**5.0**	**106**	**3.6%**	**19.0%**	**100%**	**846**
T3-2	62	50	5	57.4%	490	5	140	1.6%	3.3%	100%	506	5	127	0.1%	1.8%	100%	583
		100	3	56.0%	2,202	5	93	0.1%	4.5%	100%	734	5	96	2.7%	5.8%	100%	852
		150	0	-	3,042	5	88	-	13.5%	100%	1,089	5	88	-	11.1%	100%	1,221
		200	0	-	407	5	58	-	19.4%	100%	1,017	5	58	-	16.2%	100%	1,105
		Ave.	**2.0**	**56.7%**	**1,535**	**5.0**	**95**	**0.8%**	**10.2%**	**100%**	**837**	**5.0**	**92**	**1.4%**	**8.7%**	**100%**	**940**
T3-3	81	50	5	74.5%	1,344	5	126	-3.3%	4.4%	100%	659	5	127	-4.1%	3.6%	100%	809
		100	0	-	1,502	5	90	-	11.8%	100%	1,068	5	88	-	11.6%	100%	1,217
		150	0	-	152	5	56	-	19.2%	100%	1,089	5	56	-	19.6%	100%	1,241
		200	0	-	87	5	48	-	24.7%	100%	1,323	5	44	-	24.7%	100%	1,356
		Ave.	**1.3**	**74.5%**	**771**	**5.0**	**80**	**-3.3%**	**15.0%**	**100%**	**1,035**	**5.0**	**79**	**-4.1%**	**14.8%**	**100%**	**1,156**

According to the results presented in Table 17 and Table 18, considering the average performances, on T3 instances the minimum deviations in *LA* values from the maximum SP length is obtained by GBD-2-4. On average, GBD-2-1 outperforms GBD-1-1 for the deviations from the maximum SP length. However, GBD-1-1 outperforms the GBD-2-1 on average CPU times. Although incorporating valid inequalities into the RMP increase computational times, better solutions with lower *LA* values could be obtained. Compared to T1 and T2 instances, on T3 instances as number of commodities increase all GBD variants provide a feasible solution, whereas CPLEX cannot. With the use of a GBD algorithm, for 34 instances, which the CPLEX cannot provide any solution, feasible solutions are obtained. This outcome is the most important factor that shows the effectiveness of the GBD algorithms over CPLEX especially on larger networks.

CHAPTER 6

LARGE NEIGHBORHOOD SEARCH ALGORITHM FOR LA-CMNDP (LNS-LA)

The computational results presented in Section 5.3 show that the exact solution procedure GBD has limited performance on LA-CMNDP. To solve larger size problems efficiently, tailored metaheuristic algorithms are required. We develop a Large Neighborhood Search (LNS) algorithm for LA-CMNDP (LNS-LA), considering the structure of a feasible solution. In this chapter, we present the fundamentals of an LNS metaheuristic in Section 6.1 and the main structure of LNS-LA together with sub algorithm details in Sections 6.2-6.4. In computational experiments, (i) the performance of LNS-LA is experimented on different sized test networks, (ii) LA-CMNDP is compared with single allocation latest arrival p-hub location problem (SLApHLP) of Kara and Tansel (2001), and (iii) sensitivity analysis regarding the changes in terminal locations are performed and results are presented in Section 6.5.

6.1 Large Neighborhood Search

LNS metaheuristic is in the class of Very Large Scale Neighborhood Search (VLSN) algorithms. Therefore, in this section we first provide the basic concepts of the neighborhood search method and then present the VLSN and LNS techniques.

6.1.1 Neighborhood Search

Among the heuristic solution methods developed for large scale optimization problems, *neighborhood search algorithms* (*local search algorithms*) establish an important class due to their computational performances and structured mechanisms that can be tailored to meet various modeling aspects. Neighborhood search algorithms start with a feasible solution and iteratively improve it by searching its neighborhood solutions. The neighborhood structure and the search strategy that guides the selection of the next solution for continuing the search are the main elements for defining a neighborhood search algorithm. To give the basic definitions, first a *Combinatorial Optimization Problem* needs to be defined.

A *Combinatorial Optimization Problem* determines the values of a set of decision variables $X = \{x_1, ..., x_m\}$ with variable domains $D_1, ..., D_m$ so as to satisfy the problem constraints and to minimize the objective function $f(x)$:

$$\begin{aligned} &\textit{Minimize} \quad f(x) \\ &\textit{subject to} \\ &\qquad\qquad\qquad x \in S, \end{aligned}$$

where S represents the set of all feasible assignments of decision variables in X, $S = \{s = (x_1, v_1), ..., (x_m, v_m) \mid v_i \in D_i,\ s \text{ satisfies all the constraints}\}$, and S is usually called as the *search space*, Blum and Roli (2003).

A *neighborhood structure* $\mathcal{N}$ is defined by a point to a set assignment function. A *neighborhood function* assigns a set of *neighbors* $\mathcal{N}(s) \subseteq S$ to every $s \in S$. The set of neighbors $\mathcal{N}(s)$ establish the *neighborhood* of s, Ahuja et al. (2002), Blum and Roli (2003).

Since neighborhood search algorithms are in the class of metaheuristic algorithms, the important design issues regarding metaheuristics have to be considered while designing a neighborhood search algorithm. Thus, we first introduce the most prevalent design issues that ascertain the performance of a metaheuristic algorithm; (i) intensification and (ii) diversification. As remarked by Blum and Roli (2003), although the focuses of intensification and diversification are contrary, they describe search strategies that are complementary to each other. To guarantee a thorough search of the entire search space, the balance between these two strategies becomes critical. Intensification aims to quickly discover the regions of the search space that include high quality solutions and search them thoroughly. On the other hand diversification aims to search different regions that may lead to better solutions and not to waste too much time in the regions those are already searched or do not produce good quality solutions.

The performance of a neighborhood search algorithm mainly depends on the neighborhood structure. As the neighborhood size gets larger, it becomes more probable to obtain better solutions, whereas the neighborhood search time increases. To set a balance between the computational time and the quality of the solutions, and to tailor the neighborhood structure for the purposes of intensification and diversification, different neighborhood size management strategies are available. The most common strategies include problem dependent techniques that provide fast evaluation of neighbors, and specialized neighborhood search techniques like Variable Neighborhood Search (VNS) or Very Large Scale Neighborhood search (VLSN).

VNS metaheuristic is proposed for combinatorial optimization problems by Mladenovic and Hansen (1997). VNS provides an efficient neighborhood management structure that aims to prevent getting stuck at a local minimum. For this purpose, use of multiple neighborhood structures $\mathcal{N}_1, \mathcal{N}_2, ..., \mathcal{N}_r$ are allowed and systematic change of neighborhoods is performed within the search. As stated by Blum and Roli (2003), a solution which does not direct the search trough a local minimum with respect to one neighborhood structure can be a good starting point for another neighborhood structure. The success of VNS metaheuristic for different combinatorial optimization problems including the travelling salesman problem, the p-median problem, and the minimum sum-of-squares clustering problem is presented by Hansen and Mladenovic (2001).

When the neighborhood size gets exponential as the instance size increases or the neighborhood is too large to search explicitly, improved search techniques are required. VLSN techniques provide specialized search strategies that identify the good solutions in a large neighborhood within reasonable computational times.

6.1.2 Very Large Scale Neighborhood Search

VLSN technique is first introduced by Ahuja et al. (2000) and a comprehensive survey is presented by Ahuja et al. (2002). VLSN deals with very large neighborhoods and covers a broad range of techniques for searching the good neighbors without explicitly evaluating all neighbors of a neighborhood. Exponential neighborhood structures constitute an important class of VLSN. A neighborhood structure $\mathcal{N}$ is exponential, if the number of neighbors $|\mathcal{N}(s)|$ grows exponentially with the instance size. The search techniques applied for the neighborhoods that are too large to search explicitly are also in the domain of VLSN, Ahuja et al. (2002).

VLSN techniques are grouped in three major classes by Ahuja et al. (2002):

- Variable depth neighborhood search (VDNS) methods,
- Network flow based improvement methods,
- Methods that base the neighborhood structure on special cases that are solvable in polynomial time.

VDNS methods search a set of deeper neighborhoods $\mathcal{N}_1, \mathcal{N}_2, ..., \mathcal{N}_r$ heuristically. Transitions from $\mathcal{N}_{j-1}$ to $\mathcal{N}_j$ is performed by a *Move* function, which is executed *r* iterations and defined as $(x_j, f(x_j)) = Move(x_{j-1}, f(x_{j-1}))$. The value of *r* is guided by the search algorithm. In VDNS, each neighborhood $\mathcal{N}_j$ is searched partially; by this way the computational time required for the neighborhood search is reduced, Ahuja et al. (2002).

The main difference of VDNS and VNS is that the former searches the same neighborhood with variable depths, on the other hand the latter searches structurally different neighborhoods. The illustration comparing the neighborhood structures of VDNS and VNS that is presented by Psinger and Ropke (2010) is given in Figure 20 and the main steps of VDNS presented by Ahuja et al. (2002) are provided in Figure 21.

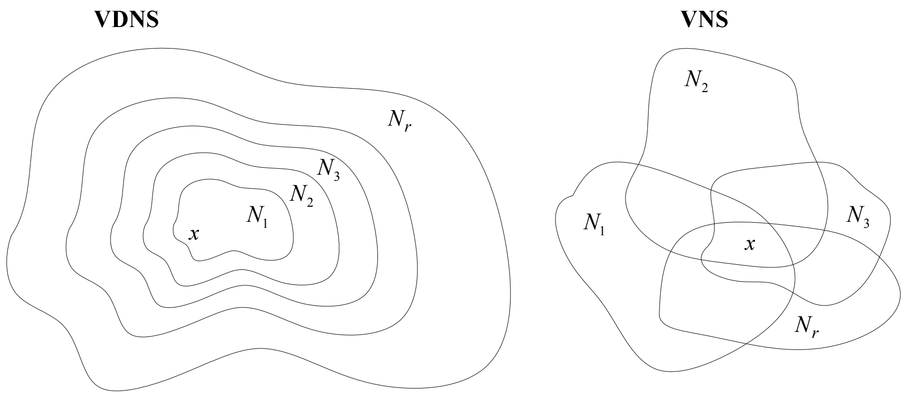


Figure 20. The Illustration Presented by Psinger and Ropke (2010) for Comparing the Neighborhood Structures of VDNS and VNS

algorithm Variable Depth Neighborhood Search
begin
    input: x_1 = a feasible solution;
    for j=2 to r
        $(f(x_j), x_j) = Move(f(x_{j-1}), x_{j-1})$;
    end;
    return the x_j that minimizes $(f(x_j) : 1 \leq j \leq r)$;
end;

Figure 21. Main Steps of Variable Depth Neighborhood Search Method

In network flow based VLSN methods, efficient search of large neighborhoods is performed by identifying improving neighbors with network flow techniques. These techniques are grouped in three main categories by Ahuja et al. (2002).
Methods that define improving neighbors by;

- finding minimum cost cycles,
- applying shortest path or dynamic programming based methods,
- finding minimum cost assignments and matchings.

Special cases of some NP-hard combinatorial optimization problems can be solved in polynomial time. By restricting the problem topology or adding constraints to the main model, these special cases can be derived from the main NP-hard problem. If the neighborhood structure is based on such an efficiently solvable special case, then the large neighborhood can be searched in polynomial time, Ahuja et al. (2002).

6.1.3 Large Neighborhood Search

LNS is proposed by Shaw (1998) with application results for vehicle routing problems, in which a routing plan that serves a set of customers with a fleet of vehicles is determined. The proposed LNS method performs two main operations; (i) remove some of the customers from the routing plan and (ii) re-optimize the routing plan by re-inserting these removed customers. The size of the neighborhood (number of customers to be removed, r) is managed throughout the algorithm so as to favor small r values to speed up the algorithm and to favor increments in r to increase the quality of the solutions and to escape from local minimum points. Computational results reveal that LNS is extremely competitive with the best known metaheuristic methods on vehicle routing problems.

LNS is a metaheuristic method that belongs to the class of VLSN techniques. Psinger and Ropke (2010) describe the fundamentals of LNS metaheuristic and present the application areas. LNS improves an initial solution by applying two main operations. First the solution is destroyed resulting in an infeasible solution and then this infeasible solution is repaired by re-optimizing the destroyed components to reach a feasible solution. Different from most neighborhood search algorithms, in LNS the neighborhood is defined implicitly by these *destroy* and *repair* operations. Accordingly, the neighborhood $\mathcal{N}(x)$ of a solution x consists of all the solutions that can be reached by the consecutive application of the *destroy* and *repair* operations. LSN metaheuristic does not search all neighbors in a neighborhood; rather it samples the neighborhood.

The *destroy* operation and its implementation determines the search strategy. Destroy operations should be defined so as to search a broad solution space especially focusing on parts that may include global optimum. Furthermore to prevent both cycling and getting stuck at a local minimum, the destroy operation should make it possible to destroy different parts of a solution throughout the LNS algorithm. The degree of destruction should be managed considering the characteristics of the search space. Low degree destructions cannot benefit from the advantages of large neighborhood search. On the other hand, high degree destructions require high computational times and may provide poor quality solutions.

The *repair* operation can be performed by a heuristic method or an optimization method. A heuristic method identifies better solutions in reasonable computational times. On the other hand an optimization method increases computational time but good quality solutions can be obtained. Compared to a heuristic method, an optimization based *repair* operation cannot guarantee to search a set of diverse solutions. The repaired solution is called as x^{move}. After obtaining x^{move}, it is either accepted and replaced by the incumbent solution *x*, or it is rejected and the search continues with *x*. The acceptance criterion is defined by a function $accept(x^{move}, x)$. In LNS depending on problem characteristics, either only improving solutions are accepted or non-improving solutions are allowed under specified circumstances for the purpose of diversification. The main steps of LNS metaheuristic are presented in Figure 22.

algorithm Large Neighborhood Search
begin
    input: x := a feasible solution;
    $x^{best} := x$;
    repeat
        x^{move} := *repair*(*destroy*(*x*));
        if accept(x^{move} , *x*) **then**
            $x := x^{move}$;
        end;
        if *cost*(x^{move}) < *cost*(x^{best}) **then**
            $x^{best} := x^{move}$;
        end;
    until stop criterion is met;
    return x^{best} ;
end;

Figure 22. Main Steps of Large Neighborhood Search

6.2 LNS-LA

A neighborhood structure and a search strategy have been used by neighborhood search algorithms in network problems. The simplest of the neighborhood search moves in network problems is the add/drop structure, in which new solutions are searched by either adding or dropping an arc from the current network configuration. Powell (1986a) and Powell and Sheffi (1989) apply add/drop structure for LTL transportation. Extending the idea of basic add/drop structure, similar neighborhood moves are defined in various modeling contexts. Several examples can be listed as: Crainic et al. (1984), Marin and Salmeron (1996a, b), and Equi et al. (1997) increase/decrease the frequencies of selected services, Gorman (1998) add/delete a vehicle to the current schedule, or slide the schedule of a vehicle, Jansen et al. (2004) swap the elements of separate solution components of operational plans.

As modeling structure gets complicated neighborhood moves like add/drop and swap operations may not guarantee a thorough search. Simple neighborhood moves generally perform well in location problems, which deliver commodities through paths consisting of few number of arcs, Ghamlouche et al (2003). But this is not the case for multicommodity network flow problems involving commodity paths consisting of many arcs. In these problems, when a change regarding the flow on a single arc is performed, mostly an equivalent solution to the current one is obtained, Ghamlouche et al (2003). Therefore, for these problems, neighborhood moves that deal with changes in several components of a solution are required to design an effective search.

Path based and cycle based neighborhood structures perform higher level of change in searched solutions compared to simple neighborhood moves. In the path based neighborhood structure, the path of one commodity is changed in each local move, Crainic (2000). A cycle-based neighborhood structure includes and excludes several arcs from the current solution simultaneously by redirecting flow around cycles considering capacity restrictions, Ghamlouche et al. (2003), Pedersen (2005), Pedersen et al. (2009), and Zhu (2010).

The main components in a feasible solution of LA-CMNDP are the delivery paths of commodities and the consolidated paths. Since the problem is uncapacitated, the decision variables determining the commodity paths x_{ij}^{k} do not include information about the flow amounts. Due to the minimax objective function, the arcs and the commodities that constitute the LA-path have higher potential to obtain a different LA-CMNDP solution compared to other consolidated paths. For instance, although the delivery paths of commodities flowing on other consolidated paths are changed, the *LA* value of the current solution may not change. Therefore we focus on the LA-path in our search algorithm and define our main neighborhood move as "to break the LA-path" of the incumbent solution by rerouting a set of commodities. This move can provide solutions with a lower LA value. Although it is also possible to obtain a solution with a higher LA value, iteratively applying the same neighborhood move can direct the search towards a better solution and can provide escaping from local minimums.

The metaheuristic we design is LNS, in which the neighborhood moves first destroy the current solution and then repair the destroyed solution to obtain a feasible solution by searching a large neighborhood. The LNS algorithm developed for LA-CMNDP (LNS-LA) begins with an initial feasible solution and iteratively improves the incumbent solution by applying two different destroy and repair operations. In Section 6.2.1 the initial solution generation routine is presented and in Section 6.2.2 the neighborhood search part is presented in detail. The subroutines used in LNS-LA are provided in Section 6.3.

6.2.1 Initial Solution Generation

Throughout the document *k* is used to define an individual commodity and $\mathbb{k}$ is used to denote the parameter in $\mathbb{k}$-shortest paths. Due to constraints (4.10), the delivery path from $O(k)$ to $D(k)$ for a commodity k is restricted to be within β times the shortest path of commodity k ($\beta \geq 1$). The initial solution generation routine considers $\mathbb{k}$-shortest paths for routing commodities. The $\mathbb{k}$-shortest path algorithm of Yen (1971) is used in the routine, which has a complexity of $O(N^2 + \mathbb{k}N)$. At the end of the routine a set of alternative paths $\mathbb{P}^k = \{ P^1, P^2, ..., P^{\Bbbk} \}$ is obtained for each $k \in K$.

In initial solution generation routine, by routing each commodity through a path that is randomly chosen from set the $\mathbb{P}^k$, a set of solutions $\mathbb{S} = \{S^1, S^2, ..., S^I\}$ are generated, where I represents the number of solutions to be generated and is determined according to the instance size. After obtaining $\mathbb{S}$, the feasibility of each solution in $\mathbb{S}$ is checked with respect to the terminal location constraints (4.8)-(4.9) and infeasible solutions are eliminated from $\mathbb{S}$. Next, for the solutions in $\mathbb{S}$, the LA values and DA, DT vectors are determined and the infeasible solutions with respect to the latest arrival constraints (4.1)-(4.3) and (4.5) are eliminated from $\mathbb{S}$. Among the remaining solutions in $\mathbb{S}$, the one that attains the lowest LA value is chosen as the initial solution. If $\mathbb{S}$ becomes empty upon elimination of infeasible solutions, the routine terminates stating that “no feasible solution is found”. The pseudo-code of the routine is provided in Figure 73 in Appendix I and the flow chart is presented in Figure 23 including the subroutine calls. The subroutines are provided in Section 6.3. The complexity of the subroutine is $O(KN^2)$ due to *Latest Arrival* subroutine that is presented in Section 6.3.1 in detail.

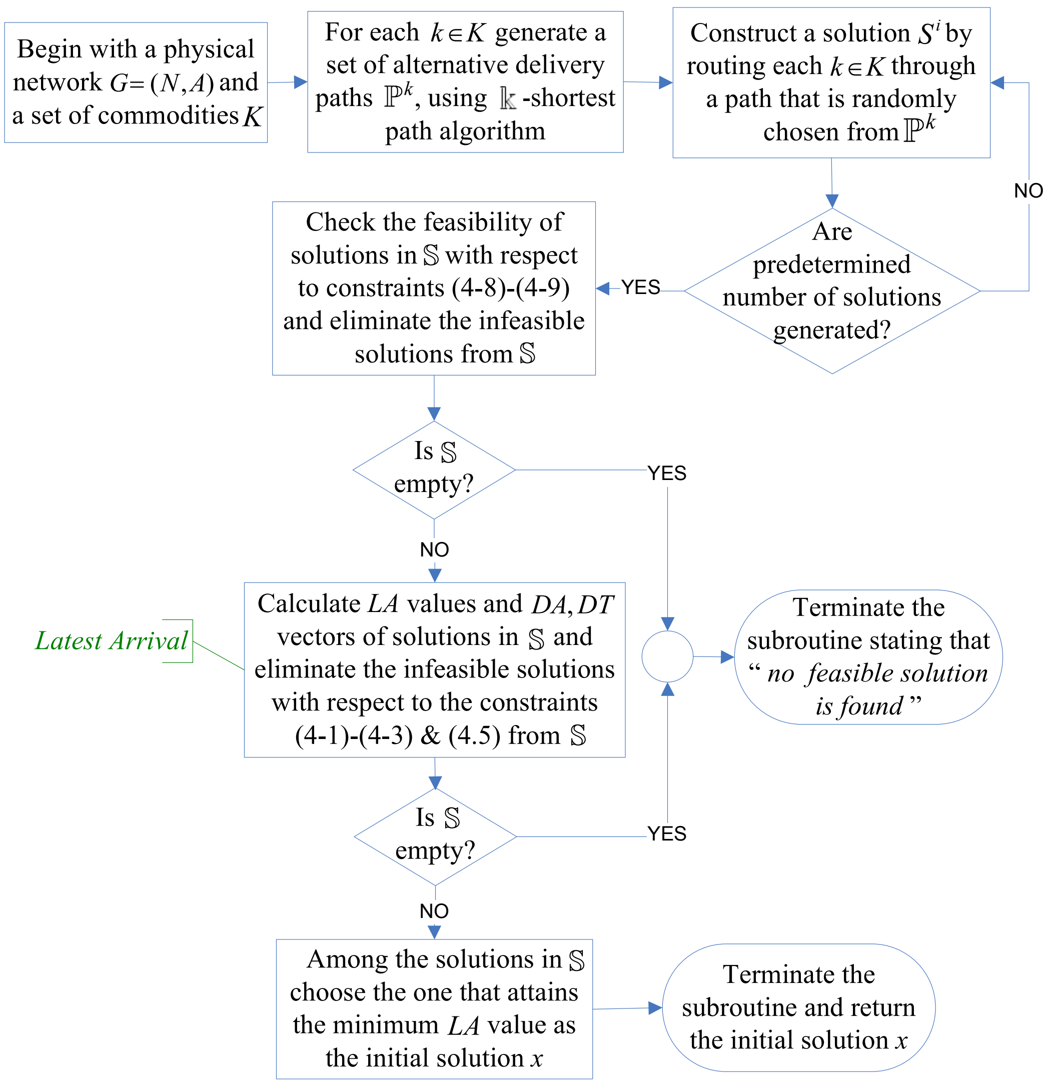


Figure 23. The Flow Chart of Initial Solution Generation Routine

6.2.2 Neighborhood Search

LNS-LA applies two neighborhood structures defined as $\mathcal{N}_1 = repair^1(destroy^1(x))$ and $\mathcal{N}_2 = repair^2(destroy^2(x))$ in two separate phases: (i) the $\mathcal{N}_1$-phase and (ii) the $\mathcal{N}_2$-phase. These phases are applied consecutively until the predetermined iteration number is reached. The algorithm begins with the $\mathcal{N}_1$-phase. If the best solution x^{best} cannot be updated by applying $\mathcal{N}_1$ consecutively for a predetermined number of times, then to prevent getting stuck at a local optimum $\mathcal{N}_1$-phase terminates. To jump to another region of the search space the algorithm enters the $\mathcal{N}_2$-phase. The beginning solution for the $\mathcal{N}_2$-phase is chosen by a function $choose(x^{best}, x)$. After obtaining a new solution x^{move} by $\mathcal{N}_1$ and $\mathcal{N}_2$, LNS-LA either continues the search with x^{move}, x, or x^{best}. The decision of which solution to continue with is given by functions $accept^1(x^{move}, x)$ for $\mathcal{N}_1$ and by $accept^2(x^{move},\ x^{best})$ for $\mathcal{N}_2$. The main steps of LNS-LA are presented in Figure 24 and the flow chart is presented in Figure 25.

algorithm LNS-LA
begin
  input: x= a feasible solution of LA-CMNDP;
  $x^{best} := x$;
  repeat
    $x^{move} := repair^1(destroy^1(x))$;
    if $accept^1(x^{move}, x) = x^{move}$ **then** $x := x^{move}$; $\mathcal{N}_1$-phase
    if $LA(x) < LA(x^{best})$ **then** $x^{best} := x$;
    if x^{best} cannot be updated for predetermined number of times **then**
      $x := choose(x^{best}, x)$;
      $x^{move} := repair^2(destroy^2(x))$;
      if $LA(x^{move}) < LA(x^{best})$ **then** $x^{best} := x^{move}$;
      if $accept^2(x^{move}, x^{best}) = x^{move}$ **then** $x := x^{move}$; $\mathcal{N}_2$-phase
      else $x := x^{best}$;
      end;
    end;
  until maximum iteration number is reached;
  return x^{best};
end;

Figure 24. Main Steps of LNS-LA Algorithm

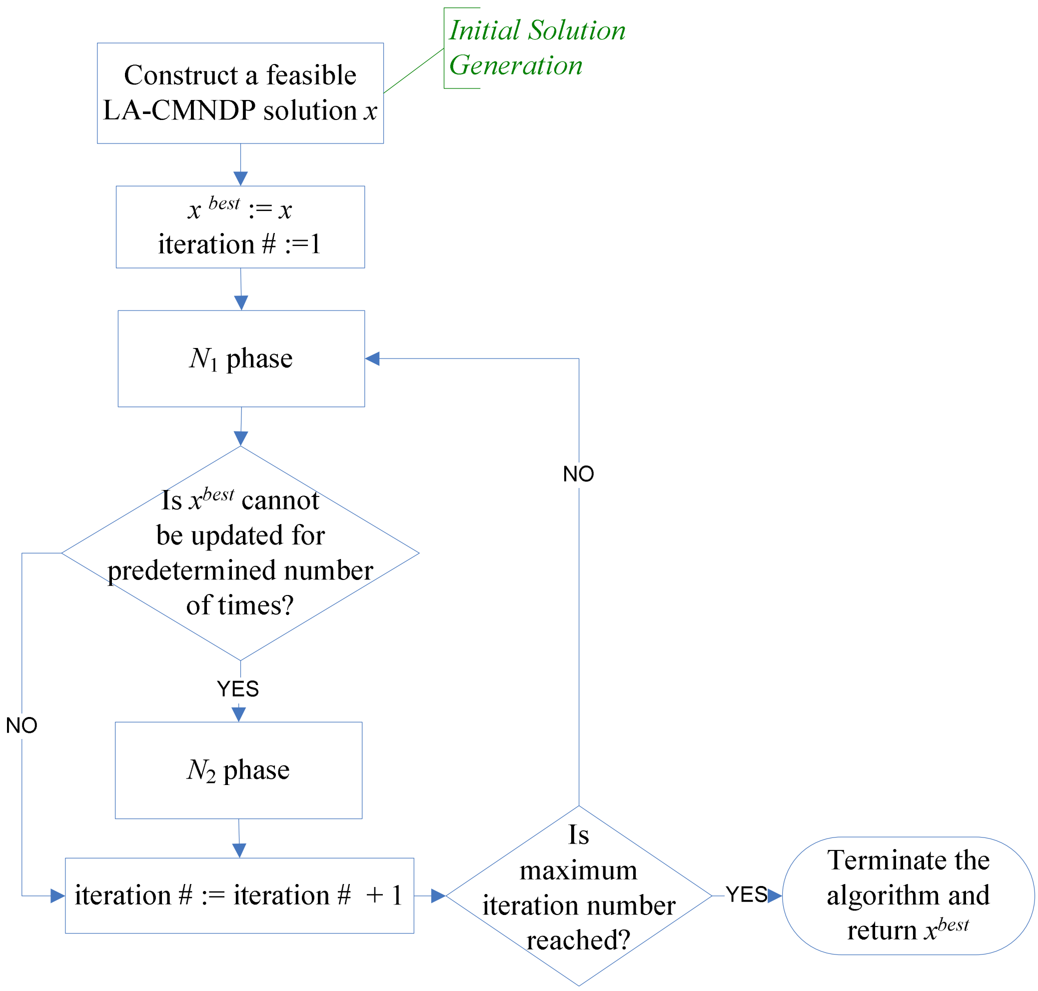


Figure 25. The Flow Chart of LNS-LA Algorithm

The main idea of the first neighborhood structure $\mathcal{N}_1 = repair^1(destroy^1(x))$ can be stated as "*to break the LA-path*" and is defined as follows:

Definition 6.1: To break the LA-path, at least one arc $(i,j) \in A^{LA}$ has to be selected as the break point arc. Then among the commodities $k \in K^{LA}$, the ones that flow on $(i_{pred}, i) \in A^{LA}$ and (i,j) are re-routed through alternative paths that do not contain arcs $\{(i_{pred}, i), (i,j)\}$ simultaneously.

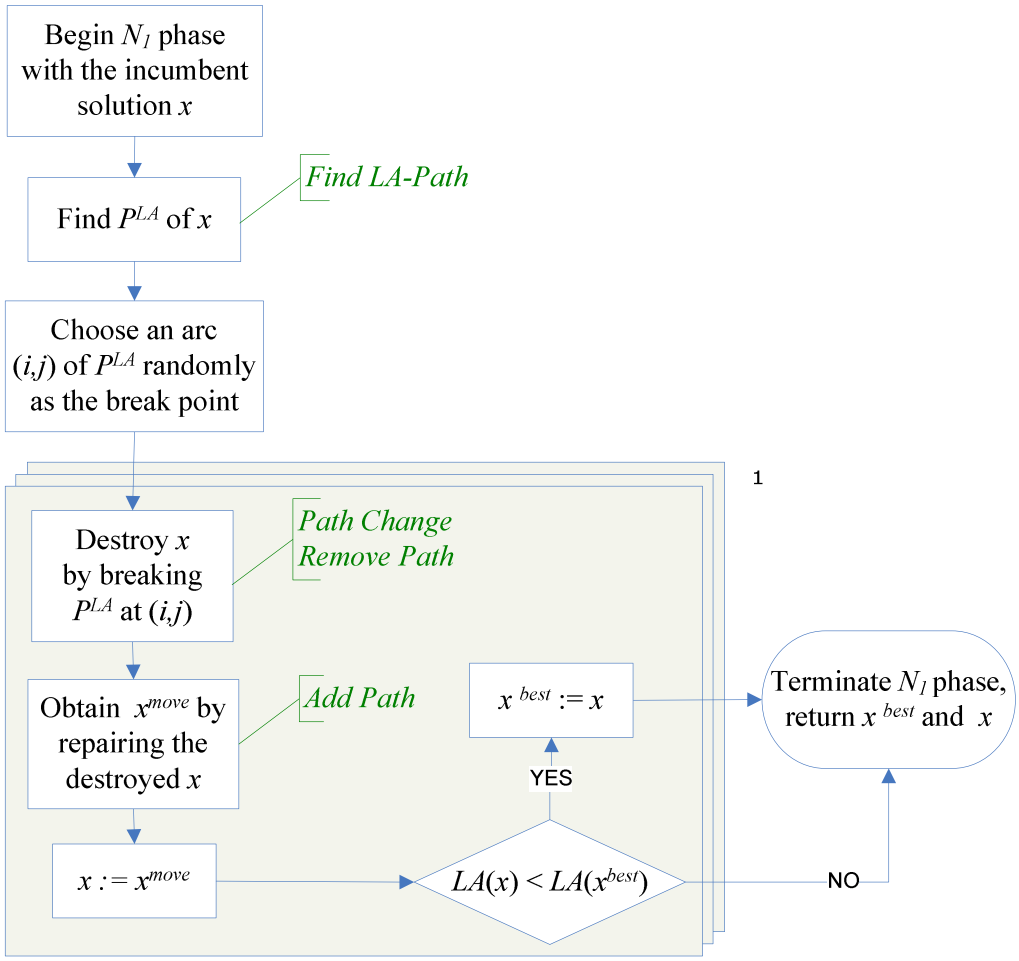
Break point arc selections are performed randomly by $\mathcal{N}_1$ so as to reach different portions of the search space. For a break point arc (i,j), for each k with $x_{ij}^k = 1$, if there are 2 alternative paths, then the related neighborhood size would be exponential and can be expressed as 2^l, where $l = \{k \in K \mid x_{ij}^k = 1\}$.

Let K^R denote the set of commodities that are to be removed from the current solution. In LNS-LA, K^R is determined considering the commodities flowing on $\{(i_{pred}, i), (i, j)\} \in A^{LA}$, due to the linkage property of P^{LA} presented in *Condition 4.1*. It should be noted that, P^{LA} can also be broken if K^R is determined only considering the commodities flowing on arc (i, j), but this increases the number of commodities to be re-routed, since a new set of commodities that flow on arc (i, j) but that do not flow on (i_{pred}, i) is added to the K^R. Whereas the set $K^R = \{k \mid k \in K^{LA}$ and $x_{i_{pred} i}^k = x_{ij}^k = 1\}$ defines the minimum sufficient set of commodities to break P^{LA} and thereby reduces the required computational search time to obtain a new LA-CMNDP solution.

$destroy^1$ operation selects the break point arc, determines the set K^R, and removes the commodities $k \in K^R$ from P^{LA}. Then $repair^1$ operation generates new LA-CMNDP solutions by searching the alternative paths of commodities in K^R. Due to the large neighborhood size of $\mathcal{N}_1$, $repair^1$ operation searches neighboring solutions heuristically, in which the 2-shortest paths among alternatives are considered.

When $\mathcal{N}_1$ is applied to the incumbent solution *x,* either an improving or a non-improving x^{move} is obtained. By $accept^1(x^{move}, x)$ function all x^{move} are accepted to continue the search. Non-improving solutions are also accepted, since the solutions with higher *LA* values provide escaping from local minimum and direct the search through unexplored regions of the solution space where further improvements can be found. If the *LA* value of x^{move} ($LA(x^{move})$) is less than $LA(x^{best})$ then x^{best} is updated with x^{move}. The flow chart of $\mathcal{N}_1$-phase is presented in Figure 26 including the subroutine calls. The subroutines are provided in detail in Section 6.3.

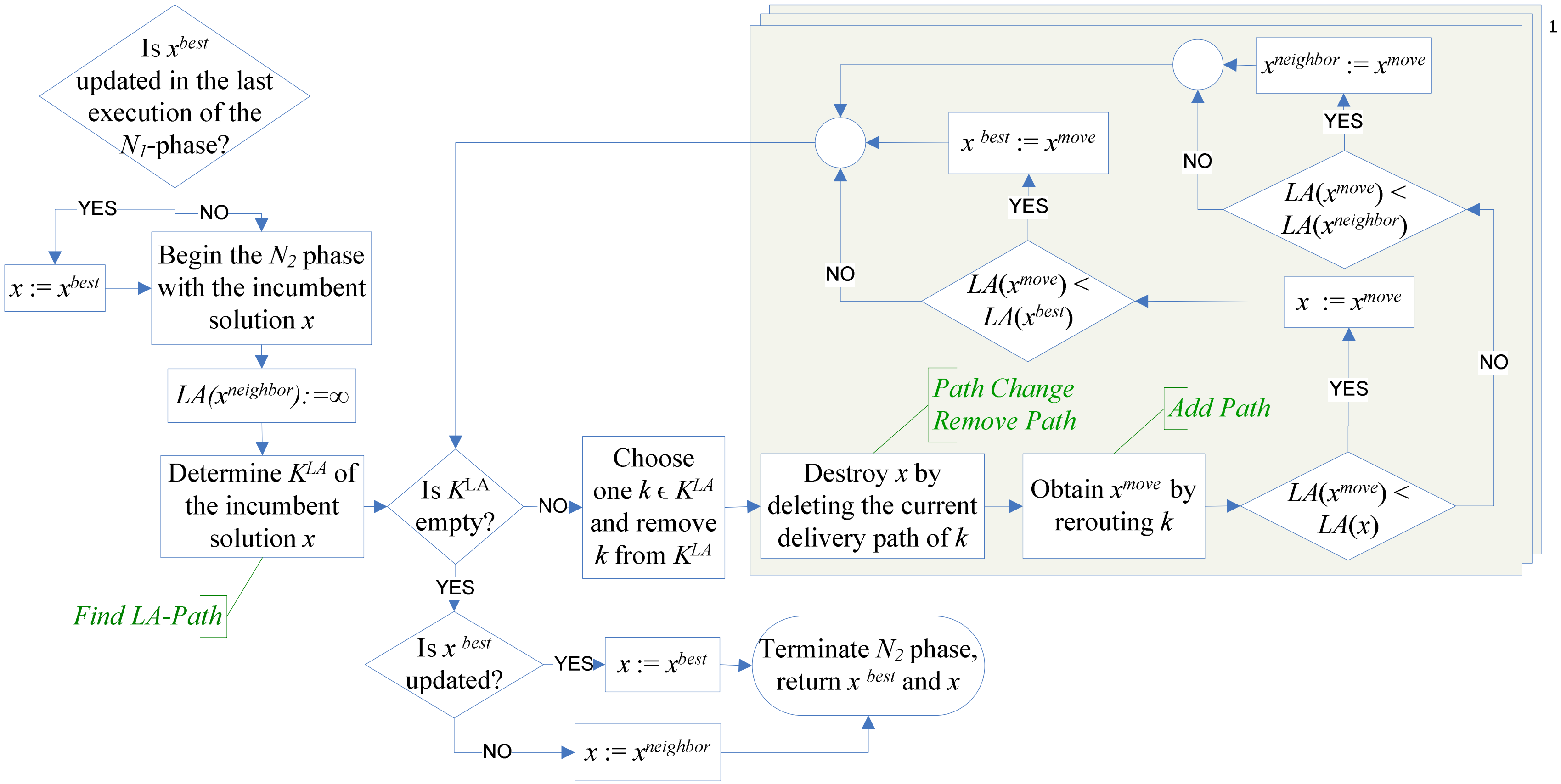
The $\mathcal{N}_1$-phase produces two solutions as output; the incumbent solution x and x^{best}. If x^{best} is updated in the last execution of the $\mathcal{N}_1$-phase, the $\mathcal{N}_2$-phase begins with x^{best}, for the purpose of intensification Otherwise, the $\mathcal{N}_2$-phase begins with x.



[1] The processes in ( ) are executed multiple times during the neighborhood search.

Figure 26. The Flow Chart of $\mathcal{N}_1$-phase

The $\mathcal{N}_1$-phase destroys the LA-path of a solution at a single arc. Although the break point arcs are selected randomly, after some iterations the $\mathcal{N}_1$-phase can switch among similar solutions. To prevent getting stuck at local optimums, the second neighborhood structure $\mathcal{N}_2 = repair^2(destroy^2(x))$ with an increased level of destruction is applied. In the $\mathcal{N}_2$-phase, the path of each commodity $k \in K^{LA}$ is destroyed and then repaired. To maintain diversification throughout the LNS-LA, $repair^2$ operation considers all alternative delivery paths while re-routing commodities. After all $k \in K^{LA}$ are re-routed by $\mathcal{N}_2$, if x^{best} is updated then the incumbent solution x is replaced by x^{best}, otherwise x is replaced by the best neighbor $x^{neighbor}$, which may have a higher LA value. Like the first neighborhood structure, $\mathcal{N}_2$ also allows non-improving solutions. The flow chart of the $\mathcal{N}_2$-phase is presented in Figure 27 including the subroutine calls, the details of which are provided in Section 6.3.



[1] The processes in ( ) are executed multiple times during the neighborhood search.

Figure 27. The Flow Chart of the $\mathcal{N}_2$-phase

6.3 Subroutines Used in LNS-LA

To conduct an efficient search, tailored algorithms are required to manage latest arrival time calculations. For this purpose, a set of subroutines are used within LNS-LA. These subroutines establish the basic algorithmic operations of latest arrival time calculations in multicommodity network flow problems and can also be used in different modeling contexts dealing with late arrivals. The inputs, the outputs, and the main functions of the subroutines are presented in this section.

Given a solution $(\bar{x}, \bar{y}, \bar{H})$, *Latest Arrival* subroutine identifies the feasibility of the solution and determines the LA value with corresponding DA and DT vectors. Given the $(LA, \overline{DA}, \overline{DT})$, *Find LA-Path* subroutine identifies P^{LA}, K^{LA}, X^{LA}. *Latest Arrival* subroutine performs all time calculations from scratch; therefore if it is used in the neighborhood search part it would necessitate an important amount of computational time. However, after determining the time values of a solution *x* by *Latest Arrival* subroutine, the neighboring solutions can be searched by update subroutines that focus only on the components of *x* that would be affected upon path changes. To perform updates, three subroutines are applied consecutively: *Path Change*, *Remove Path*, and *Add Path*.

Path Change subroutine compares the current path p^C and a new path p^N of a commodity k and provides the sets of arcs and nodes that have to be updated for removing p^C and adding p^N to the current solution. The removal process is performed by *Remove Path* subroutine and the addition process is performed by *Add Path* subroutine.

6.3.1 Latest Arrival Subroutine

The routing variables $\bar{x}$ establish the main input of *Latest Arrival* subroutine. Considering $\bar{x}$, the arcs with positive flow are put in active arcs set A^{AC}, and *DT*, *DA* calculations are performed considering the commodities that flow on A^{AC}. The sequence between *DT* and *DA* calculations for each $(i, j) \in A^{AC}$ are arranged by a parameter called as $ready(i, j)$ that indicates whether all arrival times to node *i* that are needed to determine DT_{ij} are known. When $ready(i, j) = 1$ the corresponding arc (i, j) can be selected for time calculations. The flow chart of the routine that determines and updates the $ready(i, j)$ parameters is presented in Figure 28. The pseudo-code is provided in Figure 74 in Appendix I and the complexity of the routine is $O(KN^2)$.

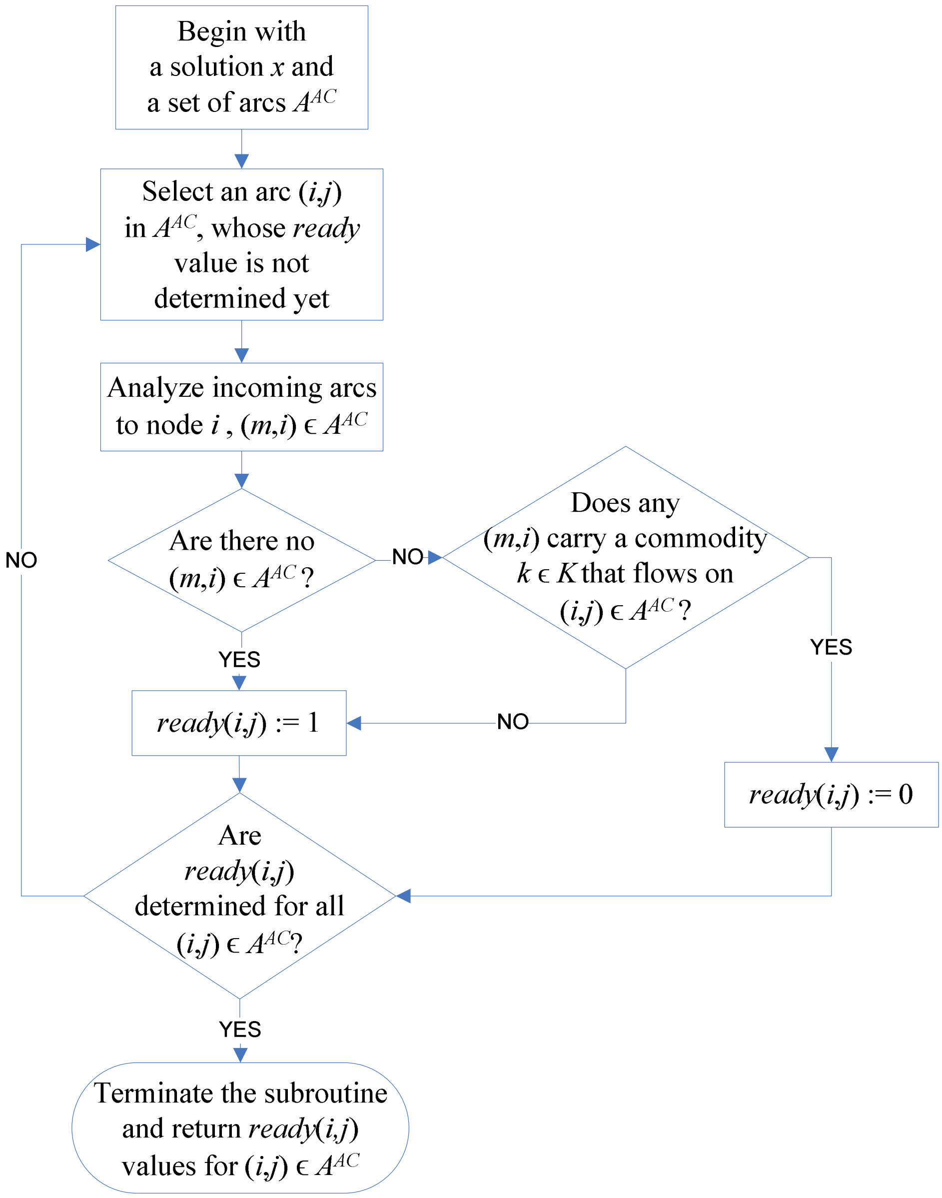


Figure 28. The Flow Chart of Ready Subroutine

To facilitate the selection of arcs in A^{AC}, a set of active nodes N^{AC} is also kept. Upon calculating DT_{ij}, arc (i,j) is removed from A^{AC}, node j is added to N^{AC}, and node i is removed from N^{AC} if none of the emanating arcs from node i is in A^{AC}. At any point of the routine, if $ready(i,j)$=0 for all arcs in A^{AC}, then no arc can be selected for time calculations. This means that the corresponding solution x is cyclic, namely feasibility is violated and the routine terminates stating that x is infeasible. The flow chart of *Latest Arrival* subroutine is presented in Figure 29. The pseudo-code is provided in Figure 75 in Appendix I and the complexity of the subroutine is $O(KN^2)$.

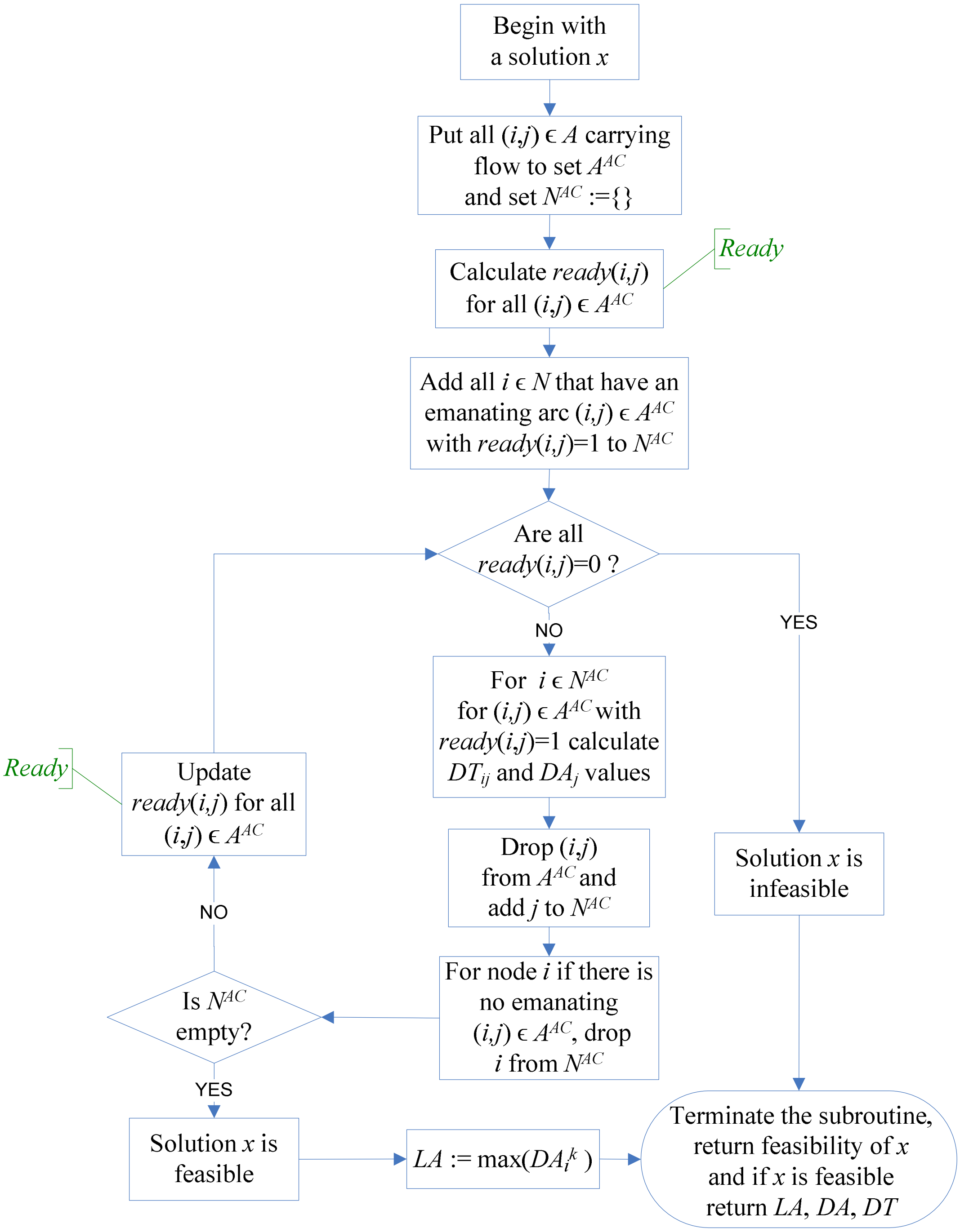


Figure 29. The Flow Chart of Latest Arrival Subroutine

6.3.2 Find LA-Path Subroutine

Find LA-Path subroutine identifies the components of P^{LA} by applying a backward search beginning from the destination node $i = D(k)$ that satisfies $DA^{k}_{D(k)} = LA$. In the first step, node *i* is marked as the last node of P^{LA}. Then the preceding node of node *i* on P^{LA} is identified by analyzing the commodity flows on $(j,i) \in A$ with corresponding DT_{ji} values. The node *j* that satisfies *Conditions 4.1* and *4.2* is selected as the preceding node of node *i* on P^{LA}. This backward procedure is repeated until the beginning node of P^{LA} is identified.

Find LA-Path subroutine can also be used to identify all distinct Consolidated Paths in a feasible solution of LA-CMNDP by conducting separate backward searches for each distinct $DA^{k}_{D(k)}$ value. The flow chart of *Find LA-Path* subroutine is presented in Figure 30. The pseudo-code of the subroutine is provided in Figure 76 in Appendix I and the complexity is $O(KN)$.

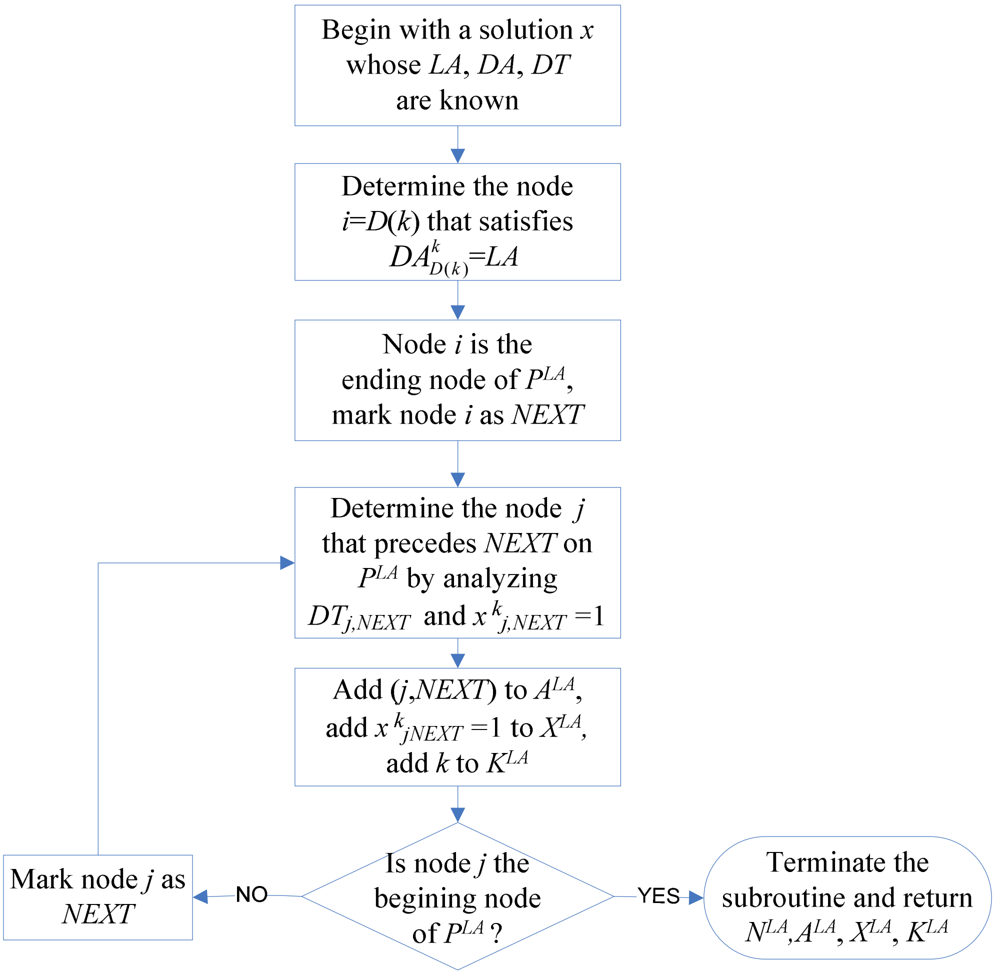
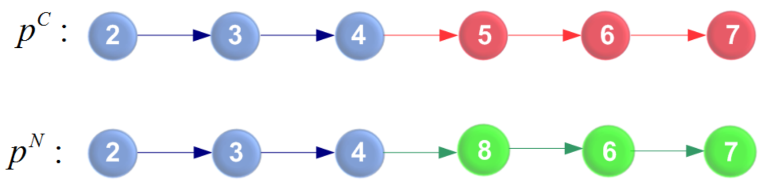


Figure 30. The Flow Chart of Find LA-Path Subroutine

6.3.3 Path Change Subroutine

To change the current path p^C of a commodity k with a new path p^N, for a set of arcs $(i, j) \in A$ and a set of nodes $i \in N$ the corresponding DT_{ij} and DA_i values have to be updated. Let A^{REM}, N^{REM} denote the sets that have to be considered for updates regarding the removal of p^C from the current solution. Let A^{ADD}, N^{ADD} denote the sets that have to be considered for updates regarding the addition of p^N to the current solution. The process performed by *Path Change* subroutine is shown on an example presented in Figure 31.



$A^{REM} = \{(4,5),(5,6),(6,7)\}$ $\qquad$ $A^{ADD} = \{(4,8),(8,6),(6,7)\}$

$N^{REM} = \{5,6,7\}$ $\qquad$ $N^{ADD} = \{8,6,7\}$

Figure 31. An Example for the Inputs and the Outputs of Path Change Subroutine

To change the p^C with p^N of commodity k, $x_{45}^k, x_{56}^k, x_{67}^k$ should be set equal to 0 and $x_{48}^k, x_{86}^k, x_{67}^k$ should be set equal to 1. Therefore $DT_{23}, DA_3^k, DT_{34}, DA_4^k$ will not be affected. Whereas DT_{45} has to be updated since the departing *block* from node 4 to node 5 will no longer have to wait for the arrival of k to node 4. The change in the value of DT_{45} will cause a change in the departure and arrival time calculations of the forward paths emanating from node 4. These changes are handled by *Remove Path* and *Add Path* subroutines. The flow chart of *Path Change* subroutine is presented in Figure 32. The pseudo-code of the subroutine is provided in Figure 77 in Appendix I and the complexity is $O(N)$.

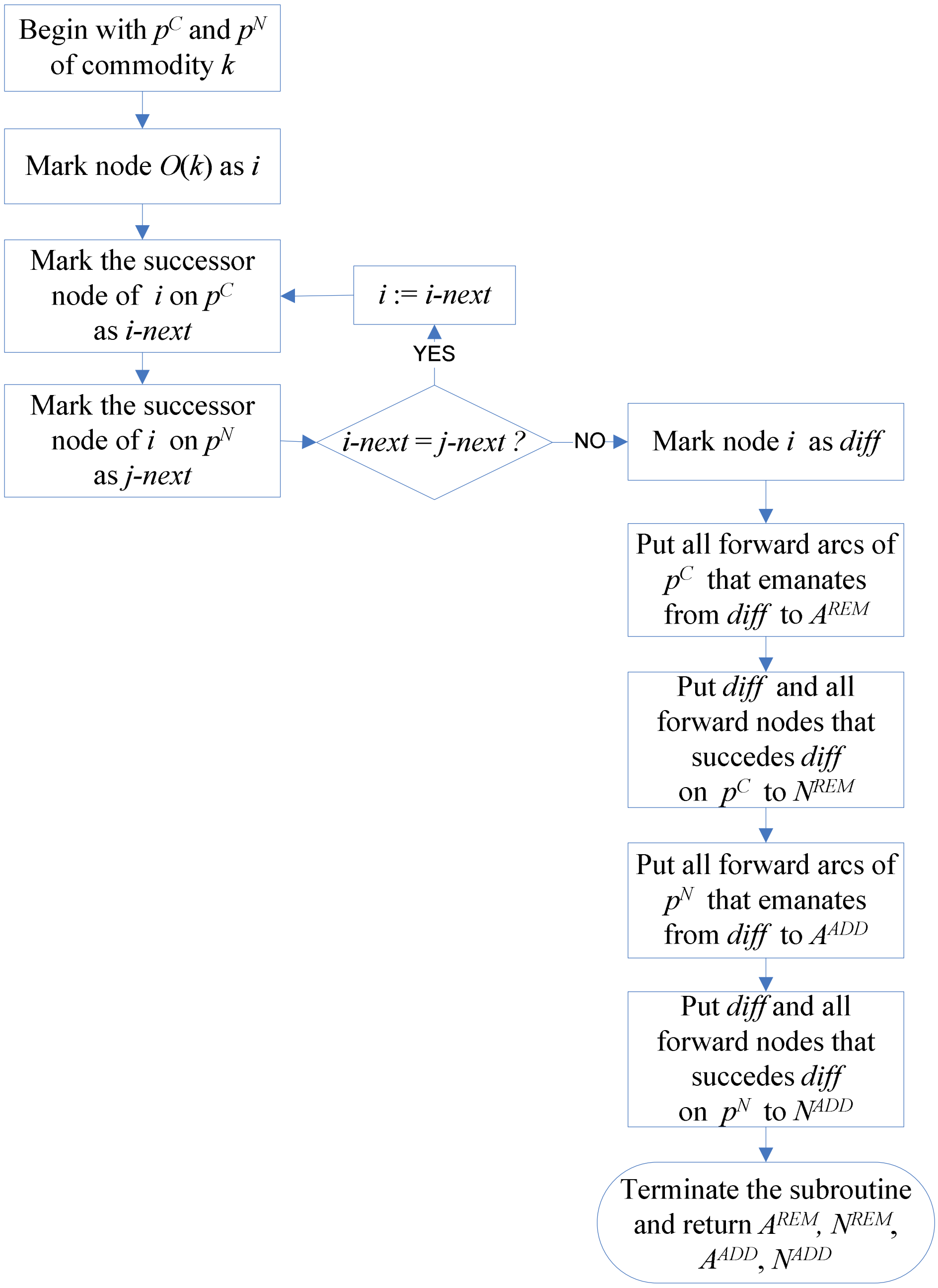


Figure 32. The Flow Chart of Path Change Subroutine

6.3.4 Remove Path Subroutine

Remove Path subroutine updates the *DA*, *DT* values that would be affected upon removing commodity *k* from the arcs in A^{REM}. As indicated in Section 6.3.3, this removal process affects the time calculations for the set of commodities in $K \setminus \{k\}$ and the set of arcs in $A \setminus \{A^{REM}\}$ that form the emanating blocks from nodes in N^{REM}. *Remove Path* subroutine also performs the required updates regarding these emanating blocks.

Since removing a commodity from some arcs of a feasible solution of LA-CMNDP does not produce a cycle causing infeasibility in time calculations, feasibility check with respect to constraints (4.1)-(4.3) and (4.5) is not needed to be performed. Due to latest arrival consolidation structure, the sequence of *DT* and *DA* updates is arranged with *ready* parameter. The flow chart of *Remove Path* subroutine is presented in Figure 33, where A^{UP}, N^{UP}, K^{UP} denotes the sets for time calculation updates and *p* denotes the commodities in $K \setminus \{k\}$. The pseudo-code is provided in Figure 78 in Appendix I and the complexity is $O(KN^2)$.

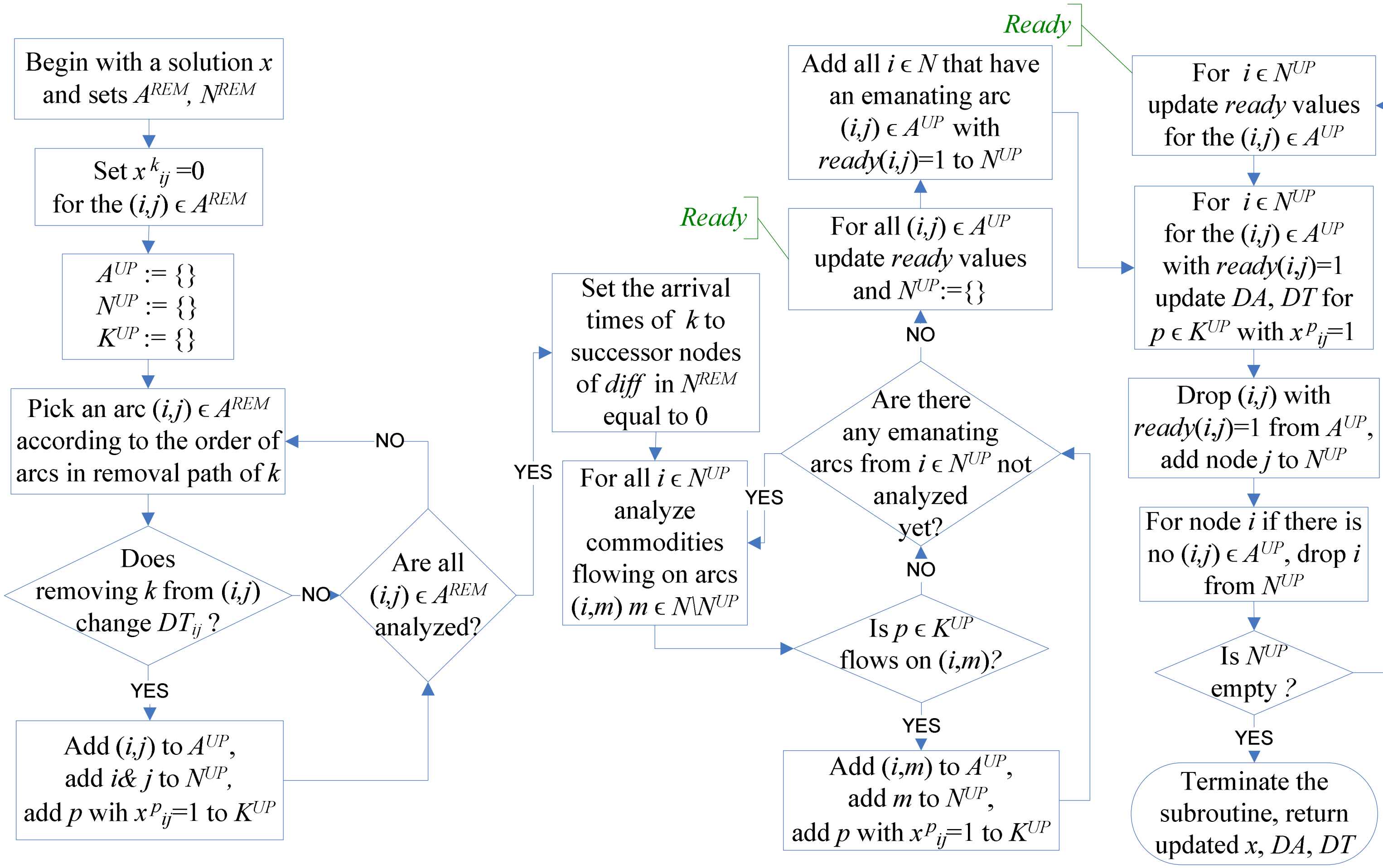


Figure 33. The Flow Chart of Remove Path Subroutine

6.3.5 Add Path Subroutine

Add Path subroutine updates the *DA*, *DT*, and *LA* values that would be affected upon adding commodity k to the arcs in A^{ADD}. Similar to the *Remove Path* counterpart, the required updates regarding the emanating blocks from nodes in N^{ADD} are also performed and the sequence of *DT* and *DA* updates is arranged with *ready* parameter. In any feasible solution of LA-CMNDP, after the removal of a commodity k from a set of arcs, the flow balance constraints of commodity k become violated. *Add Path* subroutine guarantees to satisfy these constraints. But a change in the delivery path of a commodity does not guarantee to obtain a solution that is feasible with respect to the time calculations. Thus, the feasibility check with respect to constraints (4.1)-(4.3) and (4.5) is performed within the subroutine. The flow chart of *Add Path* subroutine is presented in Figure 34, where A^{UP}, N^{UP}, K^{UP} denotes the sets for time calculation updates and p denotes the commodities in $K \setminus \{k\}$. The pseudo-code is provided in Figure 79 in Appendix I and the complexity is $O(KN^2)$.

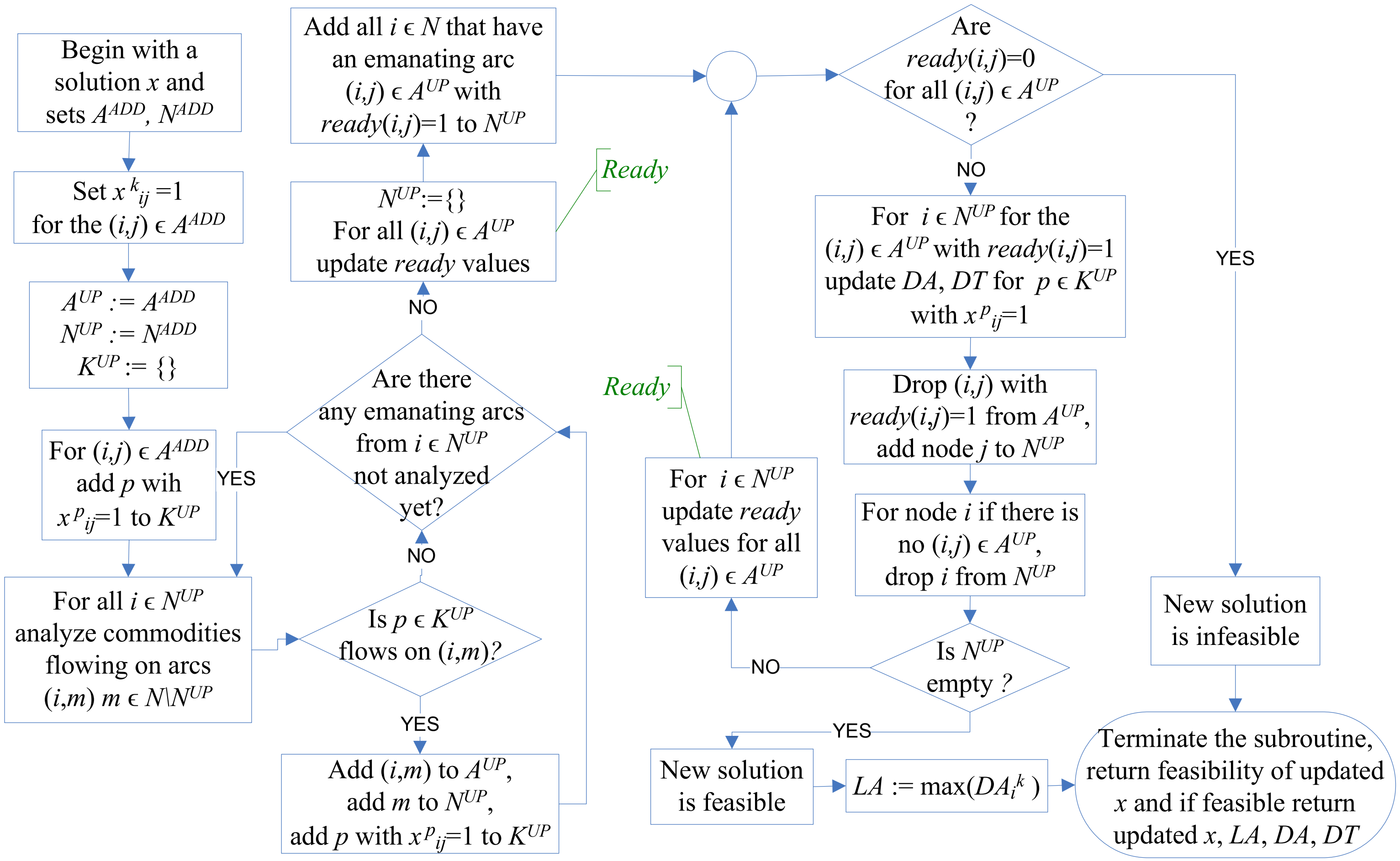


Figure 34. The Flow Chart of Add Path Subroutine

6.4 Complexity of LNS-LA

The complexity of LNS-LA and the complexities of the main parts of LNS-LA including the related subroutine calls are presented in Table 19, in which I represents the predetermined number of solutions that are to be constructed to generate a feasible initial solution and depends on the instance size. The upper limit for the subroutine calls in the $\mathcal{N}_1$-phase equals to 2^7 and the upper limit for the number of alternative paths of commodity k equals to 9. It is important to note that, in majority of the iterations of LNS-LA, the realized subroutine calls in the $\mathcal{N}_1$-phase and the number of alternative paths of a commodity are considerably lower than their upper limits. Since the complexity of an algorithm is defined by the worst case computational operations, the upper limits are provided in Table 19.

Table 19. Complexity of LNS-LA

	Main Parts of LNS-LA		
Subroutines of LNS-LA	Initial Solution Generation	$\mathcal{N}_1$-phase	$\mathcal{N}_2$-phase
Latest Arrival $O(KN^2)$	I	-	-
Find LA-Path $O(KN)$	-	1	1
Path Change $O(N)$	-	2^7	$9\|K^{LA}\|$
Remove Path $O(KN^2)$	-	2^7	$9\|K^{LA}\|$
Add Path $O(KN^2)$		2^7	$9\|K^{LA}\|$
Complexities of Main Parts	$I\,O(KN^2)$	$2^7 O(KN^2)$	$O(K^2N^2)$
Complexity of LNS-LA	$2^7 O(KN^2)+O(K^2N^2)$		

6.5 Computational Experiments

All computational experiments are performed by using the test networks (T1, T2, T3), the structures of which are presented in Section 5.3. The performance of LNS-LA algorithm is analyzed in Section 6.5.1. LA-CMNDP is compared with SLApHLP in Section 6.5.2 and the sensitivity of LA-CMNDP solutions to terminal locations are analyzed in Section 6.5.3.

6.5.1 Computational Results of LNS-LA

All computations are performed on a computer with Intel® Core™ i7-2620M CPU @ 2.7GHz and 2.94 GB RAM. LNS-LA algorithm is coded using MATLAB 7.11.0 (R2010b). To compare the LNS-LA results with the optimal values, model LA-CMNDP is also solved by CPLEX 12.

The performance of LNS-LA is analyzed according to three aspects; (i) the quality of the solutions, (ii) the percentage improvements over the initial solution, and (iii) the CPU times. By using CPLEX, the optimal LA-CMNDP solutions can be generated for majority of the T1 networks but for larger networks (T2, T3) having realistic number of commodities even an integer solution cannot be obtained by CPLEX. Due to this reason, the *LA* values of the solutions obtained by LNS-LA are compared with the optimal *LA* values in T1 networks and in some of the T2 and T3 instances of Chapter 5.

For T2 and T3 networks, two separate analyses are performed. In the first one, LNS-LA is tested on the T2 and T3 instances of Chapter 5 with the purpose of comparing the performances of LNS-LA and the best performing GBD algorithm. Considering the average performances reported in Chapter 5, GBD-2-4 is the best GBD algorithm for both of the T2 and the T3 instances. In the second analysis, LNS-LA is tested on newly generated T2 and T3 instances having significantly higher number of commodities compared to the instances of Chapter 5. For assessing the quality of the solutions, the maximum shortest path length values are used as lower bounds, the details of this lower bound is explained in Chapter 4.

Since it is difficult to test various levels of all parameters on all instances, separate experimental designs are constructed for T1, T2, and T3 networks. In all instances B_i is set equal to the average number of incident arcs of $\forall i \in N$. The rest of this section is organized as follows. For each test networks T1, T2, and T3, first the experimental design is explained then the analysis of computational results is provided.

For T1 test networks, 30 instances are generated for all combinations of $(p, \beta, |N|)$ where $p \in \{2,3,4\}$, $\beta \in \{1.1, 1.2, 1.3, 1.4, 1.5\}$, $|N| \in \{11, 13\}$. All instances are solved for full-cross-traffic case. The computational results of T1 instances are presented in Table 20. For the CPLEX computations, the optimum *LA* values, the percentage differences of the optimum *LA* values from the maximum shortest path lengths, and the CPU times are presented. The comparison with the maximum shortest path lengths is provided to evaluate the effects of p and β on optimal *LA* values. For LNS-LA computations, the initial and the final *LA* values, the percentage improvements over the initial *LA* values, the optimality gaps, and the CPU times are presented.

According to the results presented in Table 20, for all instances that can be solved to optimality, the LNS-LA finds the optimum solution. LNS-LA performs a significant improvement over the initial solution, which equals to 33% on average. The computational times of LNS-LA are much shorter than those of the CPLEX. Therefore, for T1 networks LNS-LA presents a good overall performance. For some of the instances no LA-CMNDP solution exists. Namely all of the commodities $k \in K$ cannot be delivered from their origin to destination due to the βSP^k restrictions on the delivery paths and the B_i limits imposed on nodes $i \in N$. To reach a feasible LA-CMNDP solution, either p or β values have to be increased. For the instances with the same $(|N|, |K|, p)$, as β is increased longer delivery paths for commodities are allowed and for the instances with the same $(|N|, |K|, \beta)$, as p is increased new routing arcs are introduced. By increasing either β or p, it becomes more likely to reach feasibility and to lower *LA* values. When β is increased from 1.1 to 1.2, a feasible solution can be obtained for most of the instances and increasing β from 1.2 to 1.3 does not decrease the *LA* values considerably. Thus, in the computational experiments of T2 and T3 test networks β value is taken as 1.2.

Table 20. Computational Results of LNS-LA on T1 Test Networks

Test Problem	Max. SP Length	|*N*|	|*K*|	*p*	β	CPLEX Results			LNS-LA Results				
						LA	% Difference from Max. SP Length	CPU Time (s)	Initial *LA*	Final *LA*	% Improvement Over Initial *LA*	Optimality Gap %	CPU Time (s)
T1-1	575	11	110	2	1.1		No Solution				No Solution		
					1.2								
					1.3	579	0.7%	161.1	1,168	579	50.4%	0.0%	2.3
					1.4	575	0.0%	2,378.4	965	575	40.4%	0.0%	2.2
					1.5	575	Out of Memory		970	575	40.7%	0.0%	3.9
				3	1.1		No Solution				No Solution		
					1.2	605	5.2%	27.4	883	605	31.5%	0.0%	1.4
					1.3	575	0.0%	940.6	955	575	39.8%	0.0%	2.0
					1.4	575	0.0%	27,160.6	884	575	35.0%	0.0%	2.8
					1.5	575	0.0%	4,065.8	947	575	39.3%	0.0%	2.6
				4	1.1	809	40.7%	2.2	835	809	3.1%	0.0%	1.3
					1.2	575	0.0%	53.7	736	575	21.9%	0.0%	1.3
					1.3	575	0.0%	219.0	845	575	32.0%	0.0%	2.1
					1.4	575	0.0%	1,033.4	883	575	34.9%	0.0%	2.0
					1.5	575	0.0%	114,743.0	883	575	34.9%	0.0%	2.5
T1-2	840	13	156	2	1.1		No Solution				No Solution		
					1.2	885	5.4%	108.3	1,872	885	52.7%	0.0%	6.0
					1.3		Out of Memory		1,788	885	50,5%	-	8.9
					1.4				2,005	885	55.9%	-	17.0
					1.5				1,721	868	49.6%	-	18.8
				3	1.1		No Solution				No Solution		
					1.2	840	0.0%	89.5	1,600	840	47.5%	0.0%	6.4
					1.3	840	0.0%	3,043.7	1,231	840	31.8%	0.0%	11.0
					1.4	840	Out of Memory		1,295	840	35.1%	0.0%	16.0
					1.5	840			1,348	840	37.7%	0.0%	17.3
				4	1.1	906	7.9%	4.4	958	906	5.4%	0.0%	2.5
					1.2	840	0.0%	60.4	973	840	13.7%	0.0%	6.9
					1.3	840	0.0%	1,031.8	973	840	13.7%	0.0%	10.7
					1.4	840	0.0%	12,584.0	973	840	13.7%	0.0%	12.9
					1.5	840	Out of Memory		973	840	13.7%	0.0%	21.1

As we stated in the beginning of this section, we performed two separate computational experiments of LNS-LA regarding T2 and T3 networks. First, we report the computational experiments of LNS-LA on the T2 and T3 instances of Chapter 5, comparing the LNS-LA results with the results of the CPLEX and the best performing GBD algorithm (GBD-2-4) on these instances. For the details of the related experimental designs, Chapter 5 can be referenced. Then we present a new experimental design regarding T2 and T3 networks having higher number of commodities and report the related computational results of LNS-LA.

The comparison of LNS-LA with CPLEX and GBD-2-4 algorithm on T2 instances is presented in Table 21 and on T3 instances is presented in Table 22. All the computational results in Table 21 and Table 22, represent the averages over five random instances. The computational results regarding each random instance are provided in Appendix J.

In Table 21, for the CPLEX and the GBD-2-4 algorithm the optimality gaps of the solution methods, which is stated as (*UB*-*LB*)/*UB*, are presented. For GBD-2-4 the number of Benders cuts is presented. For all solution methods, the number of optimum instances and the CPU times are presented. The *LA* value of GBD-2-4 is compared with the *LA* value of CPLEX and the *LA* value of LNS-LA is compared with the *LA* values of the CPLEX and the GBD-2-4 algorithm.

The comparison structure of Table 22 is basically same as the structure of Table 21. Since for most of the T3 instances CPLEX cannot provide any solution, in Table 22, the number of instances with a feasible solution of each solution method is presented. The maximum shortest path lengths are used as lower bounds in Table 22, for the explanations of this lower bound Chapter 4 can be referenced.

Table 21. Comparison of LNS-LA with GBD-2-4 Algorithm on T2 Instances

			CPLEX			GBD-2-4					LNS-LA			
Test Problem	**|*N*|**	**|*K*|**	**# of Opt. Instances**	**(UB-LB)/ UB**	**CPU Time (s)**	**# of Opt. Instances**	**# of Cuts**	**% Dev.from CPLEX**	**(UB-LB)/UB**	**CPU Time (s)**	**# of Opt. Instances**	**% Dev. from CPLEX**	**% Dev. From GBD-2-4**	**CPU Time (s)**
T2-1	21	25	5	0.0%	2.9	5	15	0.0%	0.0%	3.4	5	0.0%	0.0%	0.9
		50	5	0.0%	19.1	5	20	0.0%	0.0%	8.9	5	0.0%	0.0%	1.5
		75	5	0.0%	26.9	5	30	0.0%	0.0%	20.3	5	0.0%	0.0%	2.2
		100	5	0.0%	56.4	3	99	0.5%	40.0%	106.6	5	0.0%	-0.5%	3.3
		125	5	0.0%	86.5	4	97	0.3%	20.0%	125.5	5	0.0%	-0.3%	4.8
		150	5	0.0%	164.4	3	100	1.3%	40.0%	161.6	5	0.0%	-1.2%	5.9
		Ave.	**5.0**	**0.0%**	**59.4**	**4.2**	**60**	**0.4%**	**16.7%**	**71.0**	**5.0**	**0.0%**	**-0.3%**	**3.1**
T2-2	25	25	5	0.0%	4.4	5	11	0.0%	0.0%	3.1	5	0.0%	0.0%	1.0
		50	5	0.0%	17.8	5	45	0.0%	0.0%	24.6	5	0.0%	0.0%	2.2
		75	5	0.0%	32.8	4	98	0.2%	20.0%	86.3	5	0.0%	-0.2%	3.2
		100	5	0.0%	75.2	1	180	1.4%	80.0%	226.5	5	0.0%	-1.3%	4.4
		125	5	0.0%	261.1	1	163	1.4%	80.0%	263.6	5	0.0%	-1.4%	6.0
		150	5	0.0%	979.6	1	134	2.9%	80.0%	304.8	5	0.0%	-2.8%	6.9
		Ave.	**5.0**	**0.0%**	**228.5**	**2.8**	**105**	**1.0%**	**43.3%**	**151.5**	**5.0**	**0.0%**	**-1.0%**	**3.9**
T2-3	33	25	5	0.0%	18.57	5	56	0.0%	0.0%	24.8	5	0.0%	0.0%	2.7
		50	5	0.0%	359.0	2	167	1.3%	60.0%	162.7	5	0.0%	-1.2%	4.4
		75	1	16.0%	284.1	0	173	5.3%	100.0%	271.0	1	0.0%	-4.9%	6.7
		100	1	7.8%	1,084.4	0	165	8.6%	100.0%	364.9	1	-0.1%	-7.9%	8.9
		125	0	10.8%	1,714.9	0	154	13.6%	100.0%	421.4	0	0.0%	-11.9%	13.1
		150	0	26.6%	1,950.6	0	147	5.9%	100.0%	493.0	0	-6.3%	-11.6%	14.6
		Ave.	**2.0**	**10.2%**	**901.9**	**1.2**	**144**	**5.8%**	**76.7%**	**289.6**	**2.0**	**-1.1%**	**-6.2%**	**8.4**

Table 22. Comparison of LNS-LA with GBD-2-4 Algorithm on T3 Instances

Test Problem	|N|	|K|	CPLEX: # of Instances with a Soln.	CPLEX: (UB-LB)/UB	CPLEX: CPU Time (s)	GBD-2-4: # of Instances with a Soln.	GBD-2-4: # of Cuts	GBD-2-4: % Dev. from CPLEX	GBD-2-4: % Dev. from Max.SP Length	GBD-2-4: (UB-LB)/UB	GBD-2-4: CPU Time (s)	LNS-LA: # of Instances with a Soln.	LNS-LA: % Dev. from CPLEX	LNS-LA: % Dev. From GBD-2-4	LNS-LA: % Dev. from Max.SP Length	LNS-LA: CPU Time (s)
T3-1	52	50	5	55.3%	492	5	155	0.6%	0.9%	100%	519	5	0.0%	-0.6%	0.3%	14
		100	5	54.4%	541	5	114	0.3%	13.2%	100%	813	5	-7.4%	-7.6%	4.4%	21
		150	3	60.5%	2,388	5	80	10.0%	25.2%	100%	906	5	-4.0%	-12.5%	9.5%	29
		200	0	-	555	5	75	-	36.5%	100%	1,146	5	-	-17.7%	11.7%	40
		Average	**3.3**	**56.7%**	**994**	**5.0**	**106**	**3.6%**	**19.0%**	**100%**	**846**	**5.0**	**-3.8%**	**-9.6%**	**6.5%**	**26**
T3-2	62	50	5	57.4%	490	5	127	0.1%	1.8%	100%	583	5	-1.3%	-1.3%	0.4%	15
		100	3	56.0%	2,202	5	96	2.7%	5.8%	100%	852	5	-1.0%	-5.3%	0.2%	29
		150	0	-	3,042	5	88	-	11.1%	100%	1,221	5	-	-9.2%	0.9%	41
		200	0	-	407	5	58	-	16.2%	100%	1,105	5	-	-12.5%	1.7%	58
		Average	**2.0**	**56.7%**	**1,535**	**5.0**	**92**	**1.4%**	**8.7%**	**100%**	**940**	**5.0**	**-1.1%**	**-7.1%**	**0.8%**	**36**
T3-3	81	50	5	74.5%	1,344	5	127	-4.1%	3.6%	100%	809	5	-6.9%	-3.0%	0.5%	28
		100	0	-	1,502	5	88	-	11.6%	100%	1,217	5	-	-8.9%	1.6%	58
		150	0	-	152	5	56	-	19.6%	100%	1,241	5	-	-13.3%	3.6%	90
		200	0	-	87	5	44	-	24.7%	100%	1,356	5	-	-14.1%	7.1%	116
		Average	**1.3**	**74.5%**	**771**	**5.0**	**79**	**-4.1%**	**14.8%**	**100%**	**1,156**	**5.0**	**-6.9%**	**-9.8%**	**3.2%**	**73**

According to the computational results presented in Table 21, on T2 instances LNS-LA outperforms GBD-2-4 with respect to the solution quality and the computational times. GBD-2-4 cannot provide the optimum solution in 23 instances, whereas LNS-LA always identifies the optimum solution. Moreover, for the instances that the CPLEX cannot provide the optimum solution, LNS-LA provides better solutions.

According to the computational results presented in Table 22, on T3 instances LNS-LA provides considerably better solutions that are on average 4% lower than CPLEX, and 9% lower than GBD-2-4. As network size gets larger the computational efficiency of LNS-LA becomes more prominent with an average of less than 1 minute CPU time.

The highest number of commodities in T2 and T3 instances of Chapter 5 are 150 and 200, respectively. These numbers correspond to a very low percentage of the commodities in the full-cross-traffic case. To test LNS-LA on instances that have realistic commodity numbers, we generate new T2 and T3 instances.

For T2 test networks, 9 instances are generated for all combinations of $(|N|, p)$ where $|N| \in \{21, 25, 33\}$ and $p \in \{ 0.20 \text{ x } |N|, 0.24 \text{ x } |N|, 0.28 \text{ x } |N|\}$. For each of these 9 instances, $|K|$ is tested at three levels $|K| \in \{0.35\,L, 0.65\,L, L\}$ where L denotes the number of commodities for the full-cross-traffic case and $L = |N| \text{ x } (|N|-1)$. For each of the commodity levels $|K| \in \{0.35\,L, 0.65\,L\}$, selecting the $O(k), D(k)$ pairs randomly 5 different instances are generated. Therefore, the total number of instances generated for T2 networks equals to 99. The computational results of LNS-LA on T2 instances are presented in Table 23 with the percentage improvements over the initial *LA* values, the difference between the *LA* values of LNS-LA and the maximum shortest path lengths, and the CPU times. The results corresponding to the commodity levels $|K| \in \{0.35\,L, 0.65\,L\}$ in Table 23 represent the averages over five random instances. The computational results of the random instances and the instances, in which $|K| = L$ for T2 networks are provided in Appendix K.

Table 23. Computational Results of LNS-LA for T2 Test Networks

Test Problem	$\|N\|$	p	Commodity Density	$\|K\|$	LNS-LA Results		
					% Improvement Over Initial *LA*	% Difference from Maximum SP Length	CPU Time (s)
T2-1	21	4	35%	147	30.1%	3.9%	10.1
			65%	273	53.3%	5.8%	20.2
			100%	420	No Solution		
		5	35%	147	32.6%	3.8%	10.7
			65%	273	39.9%	5.5%	20.4
			100%	420	52.2%	4.9%	57.4
		6	35%	147	33.4%	3.4%	10.4
			65%	273	34.5%	3.5%	18.6
			100%	420	36.3%	4.9%	41.6
T2-2	25	5	35%	210	32.9%	9.9%	18.5
			65%	390	No Solution		
			100%	600			
		6	35%	210	30.8%	8.5%	19.2
			65%	390	40.7%	8.4%	38.7
			100%	600	No Solution		
		7	35%	210	30.9%	8.4%	17.7
			65%	390	30.8%	7.8%	41.2
			100%	600	30.2%	1.7%	78.5
T2-3	33	7	35%	370	41.9%	8.9%	55.5
			65%	686	No Solution		
			100%	1,056			
		8	35%	370	41.5%	8.2%	51.6
			65%	686	60.8%	2.5%	162.1
			100%	1,056	No Solution		
		9	35%	370	41.0%	7.8%	56.7
			65%	686	60.9%	2.3%	157.4
			100%	1,056	67.6%	4.2%	184.3

According to the results presented in Table 23, LNS-LA provides solutions with *LA* values that are on average 6% higher than the maximum SP length. In LA-CMNDP problems as network size increases, the difference between the *LA* values and the maximum SP length tend to increase. Therefore, a difference of 6% can be accepted as an indicator of good quality solutions. LNS-LA improves the initial solutions by an amount of 41% on average and the computing times are less than 3 minutes for all instances. Considering these observations, LNS-LA presents a good overall performance for T2 test networks.

In T2 instances as p is increased, it becomes more likely to obtain solutions with lower LA values and to reach feasibility. This is an expected outcome, since by locating more terminals alternative routing arcs become available and better solutions are likely to be obtained.

As $|N|$ and $|K|$ increases, the limitations due to terminal location constraints (4.8)-(4.9) become more restrictive for obtaining a feasible solution. According to the computational results of T1 and T2 networks, only for the highest p values, a feasible solution can be obtained for all levels of $(|N|,|K|,\beta)$. Actually, these highest p values equal to the number of alternative terminal nodes and correspond to imposing no limits on the number of incoming blocks to nodes. Since T3 test networks are larger than T1 and T2 networks and include significantly higher number of commodities, terminal location constraints (4.8)-(4.9) of LA-CMNDP are disregarded in T3 computations.

T3 test networks consist of 3 networks, $|N| \in \{52, 62, 81\}$. For each of these 3 networks, $|K|$ is tested at 4 levels $|K| \in \{0.25\,L, 0.50\,L, 0.75L, L\}$ where L denotes the number of commodities for the full-cross-traffic case and $L = |N| \text{ x } (|N|-1)$. Since the number of commodities is increased considerably compared to T2 instances, the commodity levels are increased to obtain more information about high number of commodities. For each of the commodity levels $|K| \in \{0.25\,L, 0.50\,L, 0.75L\}$, five different instances are generated selecting the $O(k), D(k)$ pairs randomly. Therefore, the total number of instances generated for T3 networks equals to 48. The computational results of LNS-LA on T3 instances are presented in Table 24 with the percentage improvements over the initial LA values, the difference between the LA values of LNS-LA and the maximum shortest path lengths, and the CPU times. The results corresponding to the commodity levels $|K| \in \{0.25\,L, 0.50\,L, 0.75L\}$ in Table 24 represent the averages over five random instances. The computational results of the random instances are provided in Appendix L.

Table 24. Computational Results of LNS-LA for T3 Test Networks

Test Problem	$\|N\|$	Commodity Density	$\|K\|$	LNS-LA Results		
				% Improvement Over Initial LA	% Deviation from Max. SP Length	CPU Time (s)
T3-1	52	25%	663	34%	7%	227
		50%	1,326	53%	13%	553
		75%	1,989	55%	21%	646
		100%	2,652	No Solution		
T3-2	62	25%	946	39%	8%	510
		50%	1,891	45%	30%	863
		75%	2,837	44%	41%	1,319
		100%	3,782	No Solution		
T3-3	81	25%	1,620	35%	20%	1,233
		50%	3,240	45%	21%	2,446
		75%	4,860	50%	22%	3,548
		100%	6,480	No Solution		

According to the results presented in Table 24, as number of commodities increase over 1,000, the percentage deviations from maximum shortest path lengths tend to increase, which is an expected outcome. The average percentage deviations for the lowest commodity level, $|K| = 0.25L,$, equals to 12% and for the highest commodity level $|K| = 0.75L,$ this value equals to 28%. Considering the related instance sizes these deviations can be accepted as an indicator of good quality solutions. LNS-LA improves the initial solutions by an amount of 44% on average and the computational times are less than 11 minutes for the instances with $|N| = 52$, 22 minutes for the instances with $|N| = 62$, and 1 hour for the instances with $|N| = 81$. As a result, LNS-LA presents a good overall performance for T3 test networks.

6.5.2 Comparison of LA-CMNDP and SLApHLP

In this section, LA-CMNDP is compared with the single allocation latest arrival p-hub location problem (SLApHLP) of Kara and Tansel (2001). Both models try to minimize the maximum latest arrival time to destinations, whereas use different consolidation structures and assumptions.

The consolidation structure and the assumptions that effect time calculations in SLApHLP can be stated as follows:

- each non-hub-node (spoke) can only be assigned to a single hub,
- there are two possible delivery path structures for the commodities: (i) spoke-hub-spoke, (ii) spoke-hub-hub-spoke,
- the transfers between hubs are discounted with an amount α, $(0 < \alpha < 1)$,
- a fully-connected transportation network is assumed,
- full-cross-traffic is assumed.

Compared to the consolidation structure of SLApHLP, LA-CMNDP allows more flexible routing possibilities for commodities. In LA-CMNDP, each node can be directly assigned to multiple other nodes to send and receive commodity flows. The delivery paths of commodities do not have to satisfy a special structure and do not have to include a terminal station. LA-CMNDP can be applied to incomplete transportation networks as well as the fully-connected ones. In addition, LA-CMNDP does not apply a discount factor α, rather it restricts the delivery paths of commodities to be at most β times their shortest delivery paths.

Another main difference of both models is that LA-CMNDP does not use the full-cross-traffic assumption. This assumption has a simplifying effect on time calculations and when combined with single allocation structure SLApHLP does not have to model arrival times and can have a linear model. Whereas LA-CMNDP has to model arrival times and necessitates more complicated time calculations. The largest test instance solved by SLApHLP consists of 25 nodes and 600 commodities, (Kara and Tansel (2001)). Later studies on single allocation latest arrival hub location problems report larger networks like the Turkey network consisting of 81 nodes and 6,480 commodities, (Yaman et al. (2007)).

To analyze the effects of the consolidation structures of models SLApHLP and LA-CMNDP on the latest arrival times and the total travel times we compare both problems on the same network. For this purpose, we use T1 test networks, since for majority of the T1 instances the optimum LA-CMNDP solutions can be obtained. Since model SLApHLP assumes a fully-connected structure, in T1 networks the travel times t_{ij} between the node pairs (i, j), that are not physically connected by a direct link are set equal to the corresponding shortest paths in the incomplete T1 networks.

To observe the effects of selecting different values of β and α, SLApHLP is tested for $\alpha \in \{1, 0.9, 0.8, 0.7, 0.6\}$ and LA-CMNDP is tested for $\beta \in \{1.1, 1.2, 1.3, 1.4, 1.5\}$ on test instances that are generated for all combinations of $(|N|, p)$ where $|N| \in \{11, 13\}$ and $p \in \{2, 3, 4\}$.

Comparing LA-CMNDP and SLApHLP when $\alpha = 1$ would enable us to evaluate the effect of networking structures of both problems on travel times. In Table 25, *LA* values and the total travel times of both problems are provided. For LA-CMNDP, the percentage of decrease in *LA* and the percentage of decrease in total travel times over SLApHLP are also given.

Considering the results presented in Table 25, when $|N|=11$ LA-CMNDP provides an average of 15% decrease in *LA* and an average of 26% decrease in total travel times over SLApHLP. As $|N|$ is increased from 11 to 13, LA-CMNDP again provides solutions with lower time values. The main reason of the decreases in travel times is that LA-CMNDP allows a more flexible networking structure compared to SLApHLP.

Table 25. Comparison of LA-CMNDP and SLApHLP When α=1

Test Problem	*p*	SLApHLP *LA*	SLApHLP Total Travel Times	LA-CMNDP β	LA-CMNDP *LA*	LA-CMNDP Total Travel Times	% Decrease in *LA*	% Decrease in Total Travel Times
T1-1	2	519	48,114	1.1	457	33,828	12%	30%
				1.2	417	32,525	20%	32%
				1.3	417	33,484	20%	30%
				1.4	417	33,431	20%	31%
				1.5	417	33,793	20%	30%
	3	485	44,796	1.1	456	33,203	6%	26%
				1.2	417	31,708	14%	29%
				1.3	417	32,678	14%	27%
				1.4	417	32,423	14%	28%
				1.5	417	32,300	14%	28%
	4	464	40,673	1.1	456	32,771	2%	19%
				1.2	417	31,743	10%	22%
				1.3	417	32,852	10%	19%
				1.4	417	32,660	10%	20%
				1.5	417	32,773	10%	19%
T1-2	2	762	79,010	1.1	727	74,088	5%	6%
				1.2	727	74,357	5%	6%
				1.3	727	75,276	5%	5%
				1.4	727	75,652	5%	4%
				1.5	727	75,331	5%	5%
	3	693	77,761	1.1	689	72,111	1%	7%
				1.2	689	71,987	1%	7%
				1.3	689	72,785	1%	6%
				1.4	689	72,643	1%	7%
				1.5	689	72,812	1%	6%
	4	689	74,790	1.1	689	72,494	0%	3%
				1.2	689	72,367	0%	3%
				1.3	689	73,298	0%	2%
				1.4	689	73,611	0%	2%
				1.5	689	73,739	0%	1%

α value of 1 corresponds to applying no discounts for the inter hub transfers, which may be the case depending on the type of the vehicles used but considering the main motivation of hub location problems $\alpha = 1$ is not a practical case. Yaman et al. (2007) state that $\alpha = 0.9$ reflects the realistic discount amount in travel times considering the trucking operations of cargo companies in Turkey and apply this α value in their latest arrival hub location with stopovers problem. Taking this real life application into consideration, we compare LA-CMNDP and SLApHLP when $\alpha = 0.9$ in Table 26. The comparison structure of Table 26 is same with the structure of Table 25.

Table 26. Comparison of LA-CMNDP and SLApHLP When α=0.9

Test Problem	p	SLApHLP LA	SLApHLP Total Travel Times	LA-CMNDP β	LA-CMNDP LA	LA-CMNDP Total Travel Times	% Decrease in LA	% Decrease in Total Travel Times
T1-1	2	498	43,747	1.1	457	33,828	8%	23%
				1.2	417	32,525	16%	26%
				1.3	417	33,484	16%	23%
				1.4	417	33,431	16%	24%
				1.5	417	33,793	16%	23%
	3	464	42,144	1.1	456	33,203	2%	21%
				1.2	417	31,708	10%	25%
				1.3	417	32,678	10%	22%
				1.4	417	32,423	10%	23%
				1.5	417	32,300	10%	23%
	4	430	39,530	1.1	456	32,771	-6%	17%
				1.2	417	31,743	3%	20%
				1.3	417	32,852	3%	17%
				1.4	417	32,660	3%	17%
				1.5	417	32,773	3%	17%
T1-2	2	720	73,807	1.1	727	74,088	-1%	0%
				1.2	727	74,357	-1%	-1%
				1.3	727	75,276	-1%	-2%
				1.4	727	75,652	-1%	-2%
				1.5	727	75,331	-1%	-2%
	3	644	72,286	1.1	689	72,111	-7%	0%
				1.2	689	71,987	-7%	0%
				1.3	689	72,785	-7%	-1%
				1.4	689	72,643	-7%	0%
				1.5	689	72,812	-7%	-1%
	4	620	71,336	1.1	689	72,494	-11%	-2%
				1.2	689	72,367	-11%	-1%
				1.3	689	73,298	-11%	-3%
				1.4	689	73,611	-11%	-3%
				1.5	689	73,739	-11%	-3%

Considering the results presented in Table 26, although LA-CMNDP does not apply a discount factor, LA-CMNDP provides an average of 8% decrease in *LA* and an average of 21% decrease in total travel times over SLApHLP when $|N|$=11. In addition, the *LA* values of LA-CMNDP when $|N|$=13 and p={3,4} are equal to the maximum SP lengths, namely the best possible *LA* values. When $|N|$=13, due to the discount factor α, SLApHLP provide solutions with lower time values. As α values get lower than 0.9 and p increases, SLApHLP gets an advantage for further decreasing the time values. The *LA* values of SLApHLP and LA-CMNDP are compared for different values of α and β in Figure 35, Figure 36, Figure 37 for p=2, 3, and 4 respectively when $|N|=11$; and in Figure 38, Figure 39, Figure 40 for p=2, 3, and 4 respectively when $|N|=13$.

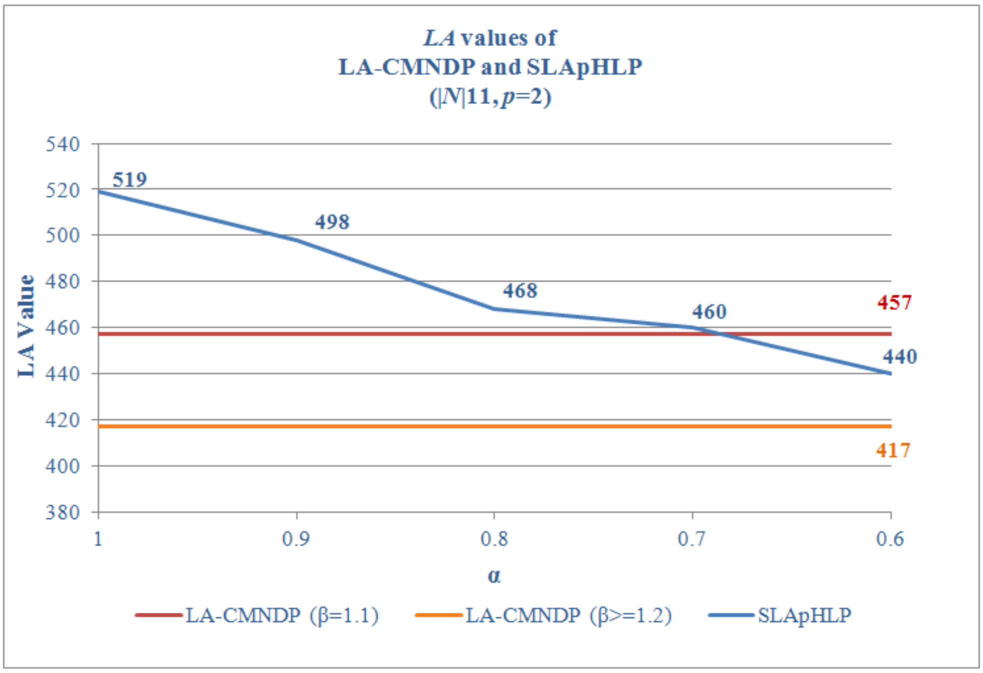


Figure 35. *LA* Values of LA-CMNDP and SLApHLP When |*N*|=11, *p*=2

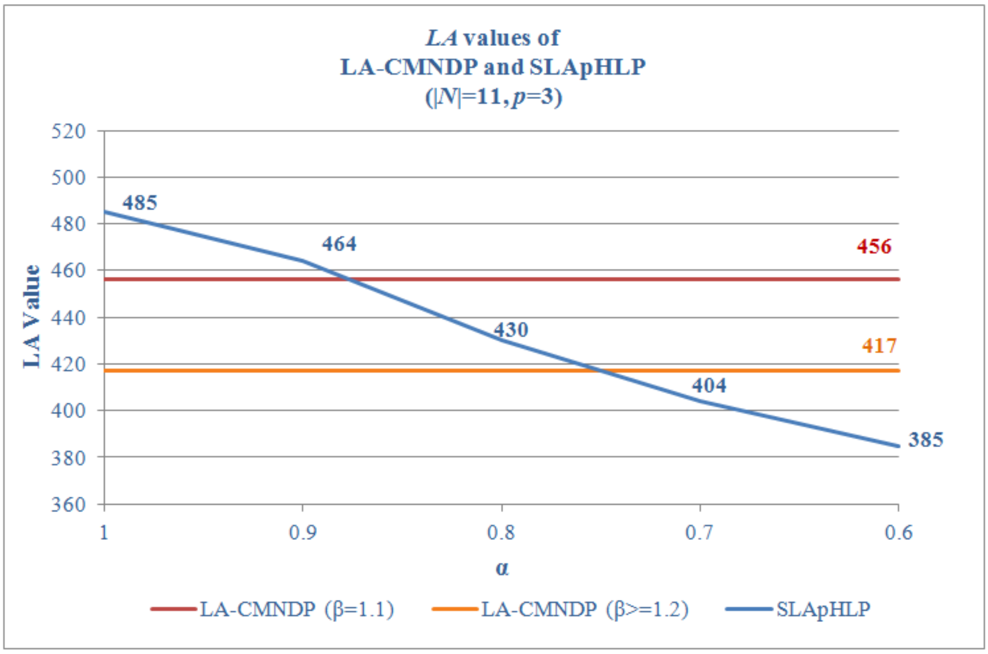


Figure 36. *LA* Values of LA-CMNDP and SLApHLP When |*N*|=11, *p*=3

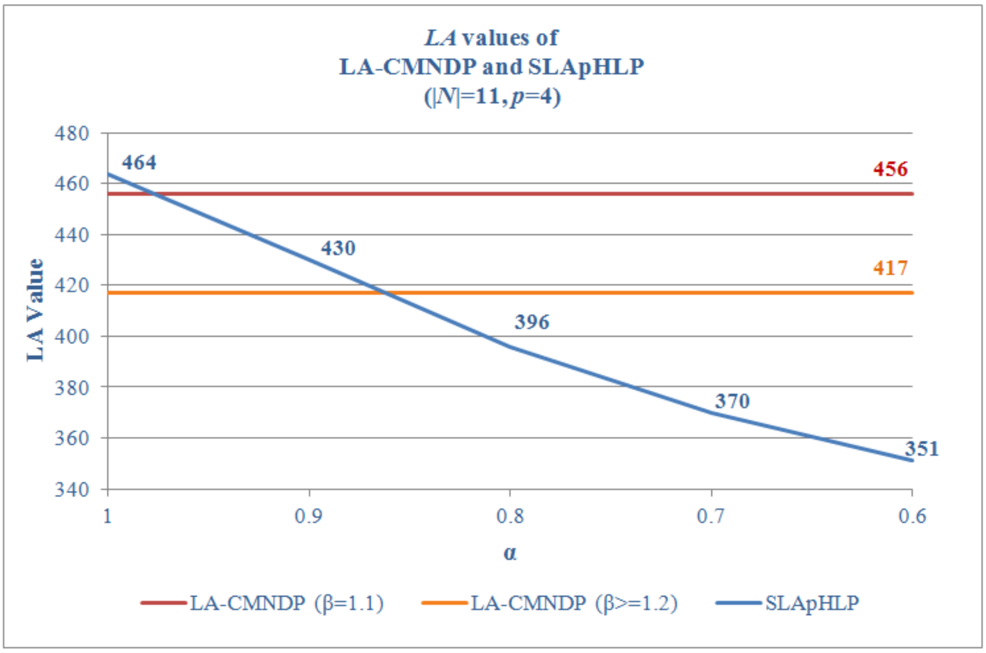


Figure 37. *LA* Values of LA-CMNDP and SLApHLP When |*N*|=11, *p*=4

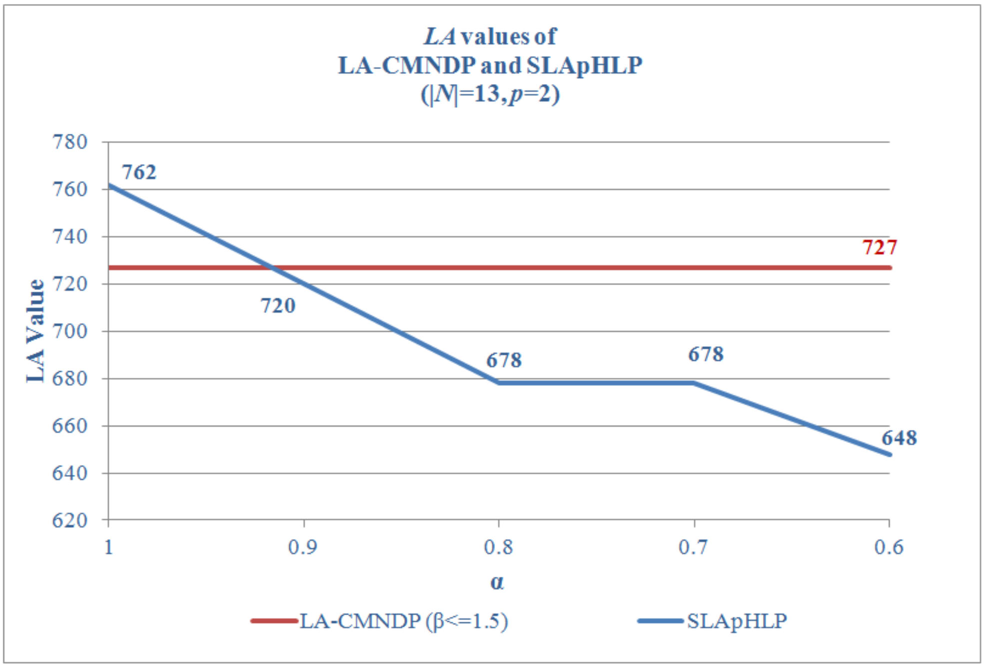


Figure 38. *LA* Values of LA-CMNDP and SLApHLP When |*N*|=13, *p*=2

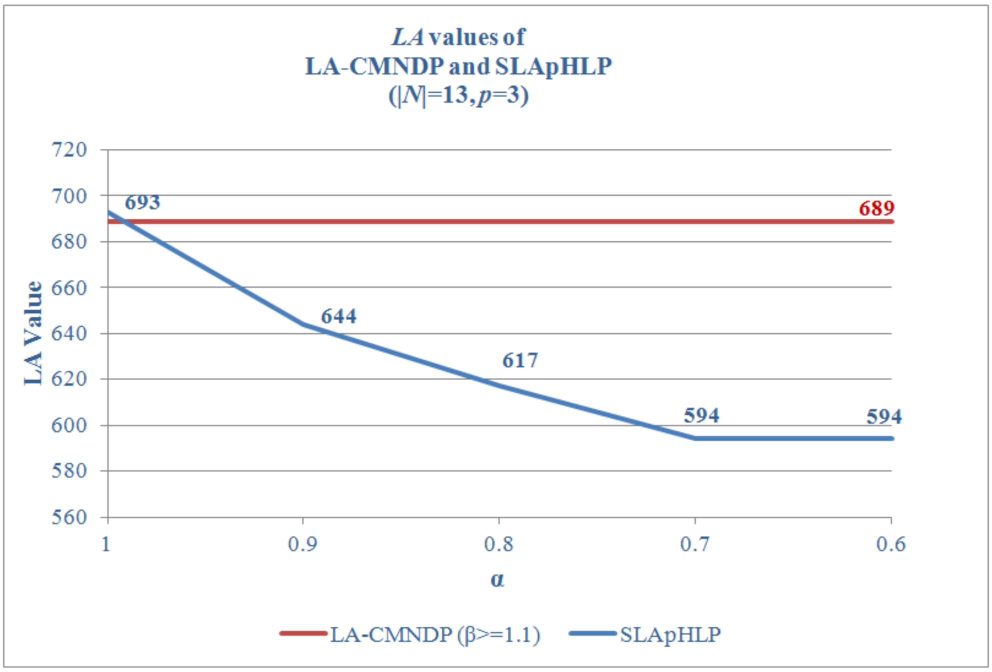


Figure 39. *LA* Values of LA-CMNDP and SLApHLP When |*N*|=13, *p*=3

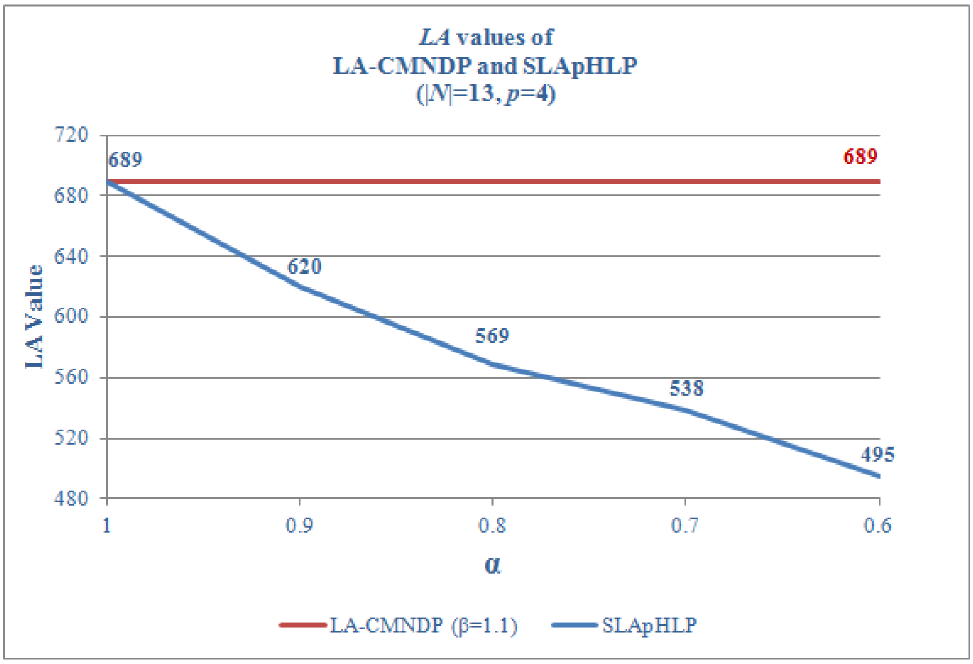


Figure 40. *LA* Values of LA-CMNDP and SLApHLP When |*N*|=13, *p*=4

Considering the Figures 35 through 37, although LA-CMNDP does not apply a discount factor, when $|N|=11$ LA-CMNDP outperforms SLApHLP for $\alpha \geq 0.8$ on all values of p. Considering the Figures 38 through 40, when $|N|=13$ and p={3,4}, LA-CMNDP provides solutions that are equal to or better than SLApHLP and for $\alpha \leq 0.9$ SLApHLP outperforms LA-CMNDP. As p increases, the number of inter hub links also increases causing considerable decreases in LA values.

In Figures 41 and 42 total travel times of LA-CMNDP and SLApHLP are compared for $|N|=11$ and $|N|=13$ respectively. Since different values of (p, β) do not cause significant amount of changes in total travel times of solutions obtained by LA-CMNDP, the average total travel times over all (p, β) are presented in Figures 41 and 42 for LA-CMNDP. When $|N|=11$, LA-CMNDP produces solutions with lower total travel times compared to SLApHLP for majority of the test instances. When $|N|=13$, due to the increased problem size the effect of applying discounts on the inter hub links become more observable and for $\alpha \leq 0.9$ SLApHLP produces solutions with lower total travel times.

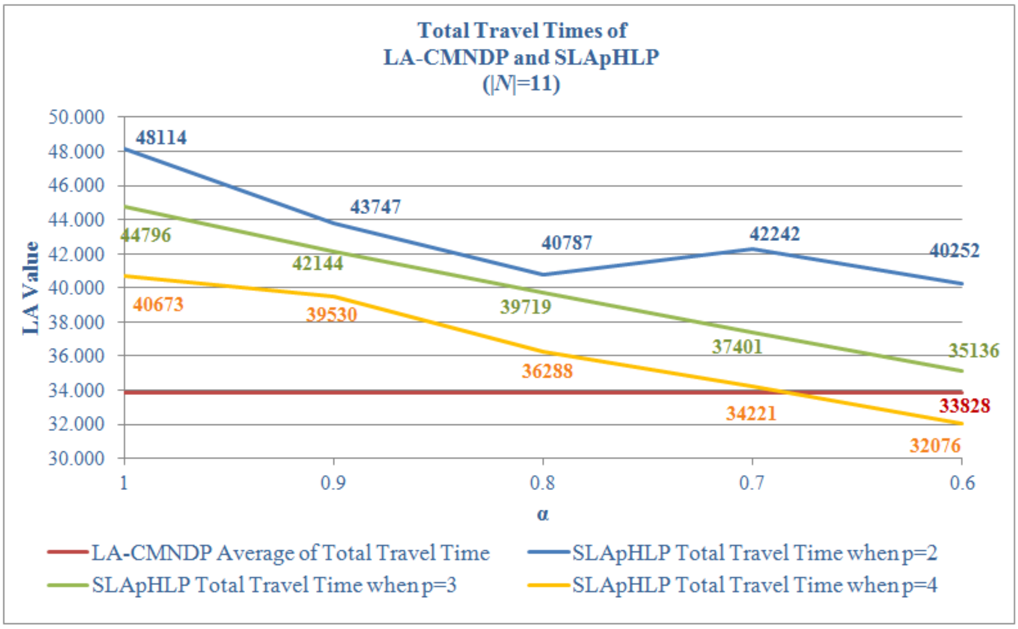


Figure 41. Total Travel Times of LA-CMNDP and SLApHLP When |N|=11

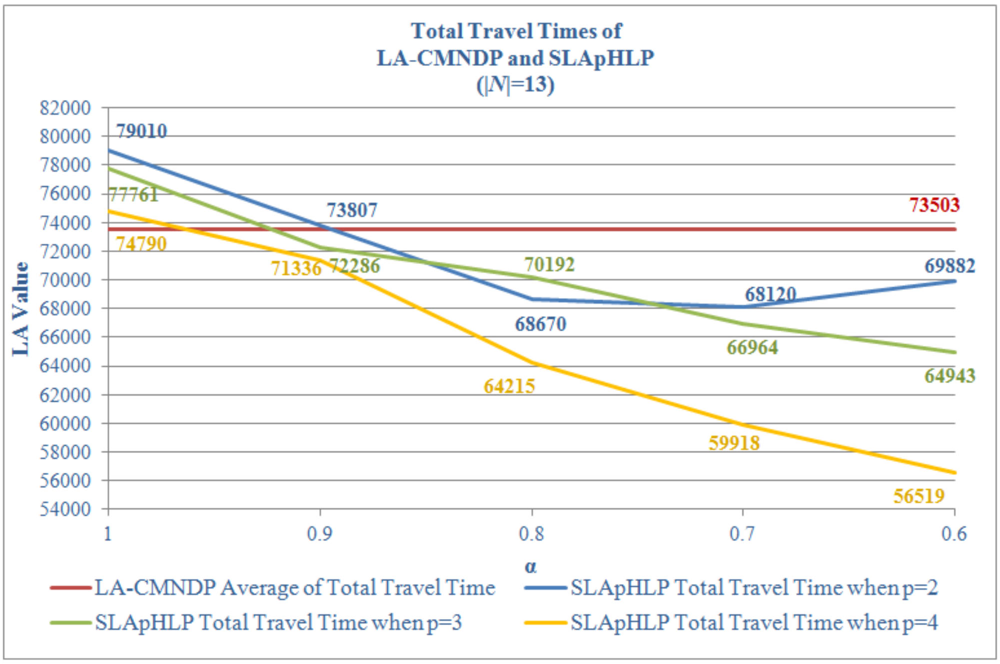


Figure 42. Total Travel Times of LA-CMNDP and SLApHLP When |*N*|=13

6.5.3 Sensitivity Analysis of Terminals

The main motivation of locating terminals in LA-CMNDP is to increase the level of consolidation throughout the transportation system. As expected, the stations that are chosen as terminals have an important effect on the consolidation plans and therefore the arrival times. Also due to terminal location constraints (4.8)-(4.9), for some LA-CMNDP instances an implementable solution cannot be obtained. For examples the computational results in Section 6.5.1 can be referenced. To analyze the effects of the terminal locations in an optimum LA-CMNDP solution on *LA* values and on obtaining an implementable solution, we perform a sensitivity analysis.

We use T1 test networks in the sensitivity analysis, in which we first record the terminal locations in an optimum solution. Then, we re-solve the related instance by applying an additional restriction on the terminal locations such that the restricted node or the nodes cannot be selected as a terminal. Finally, for each instance we report the new *LA* values and the new terminal locations. When a node, which is selected as a terminal by LA-CMNDP, is restricted for terminal location, and LA-CMNDP is re-solved with this restriction one of the other candidate terminal nodes has to be selected as a terminal. Candidate terminal nodes are presented in Figure 43 and Figure 44 for test networks with $|N|=11$ and $|N|=13$, respectively.

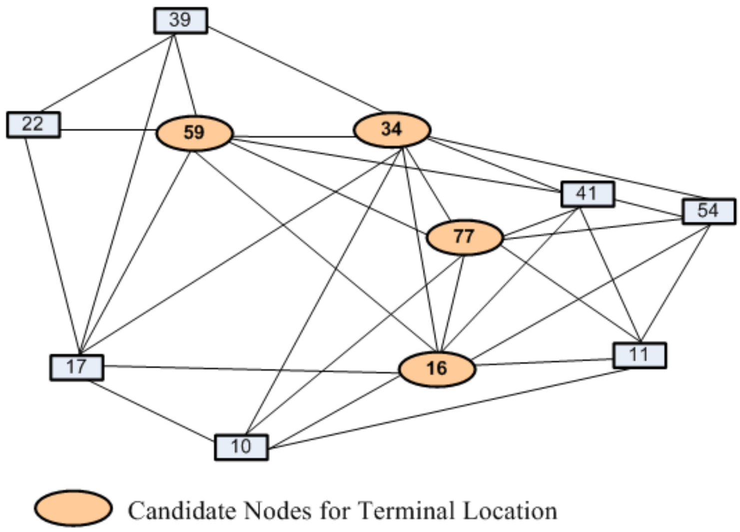


Figure 43. Candidate Terminal Nodes for $|N|=11$

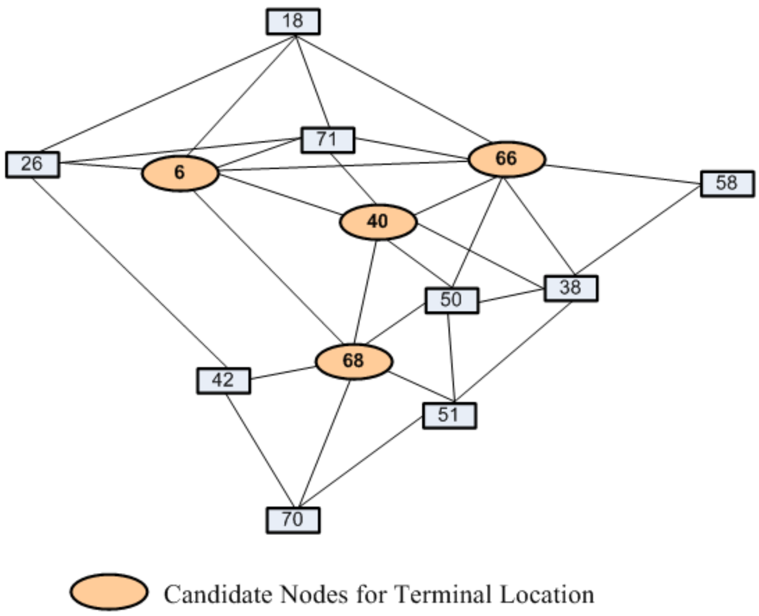


Figure 44. Candidate Terminal Nodes for $|N|=13$

The restrictions are applied to either one candidate terminal node or a couple of nodes if necessary. The sensitivity analysis is only performed on T1 instances that yield an implementable LA-CMNDP solution. When p=4, none of the terminals can be selected for removal, since p=4 means that locating a terminal at all candidate terminal locations for $|N| \in \{11,13\}$. Therefore, the sensitivity analysis is performed only for $p \in \{2,3\}$.

In Table 27, three separate sensitivity analysis results are presented. The *LA* values and the terminals in the optimum solution of LA-CMNDP under no restrictions (Original Model) is presented in the first column of Table 27. Each 'Terminals' column in Table 27 indicates which nodes are selected as terminals in the corresponding solutions. Total travel time values are not reported since restricting a candidate terminal node for terminal location do not result in a considerable effect on total travel time values. The instances that are remarked as "Not Applicable" refers to the cases, in which the restricted node is not selected as a terminal in the optimal solution of the LA-CMNDP (Original Model).

According to the results presented in Table 27, the obtaining an implementable LA-CMNDP solution is highly sensitive to the changes in terminal locations especially on instances with lower values of p and β and the *LA* values are moderately sensitive to the changes in terminal locations.

For instance, when either node 59 or node 77 is restricted for terminal location in T1-1 network, a feasible LA-CMNDP solution cannot be generated for p=2, β=1.2 and p=3, $\beta \in \{1.2, 1.3\}$. In addition, restricting node 40 or node 68 for terminal location in T1-2 network increases *LA* values for p=3, $\beta \in \{1.2, 1.3, 1.4, 1.5\}$ and for p=3 and $\beta \in \{1.2, 1.3, 1.4\}$, respectively.

Table 27. Sensitivity Analysis of Terminals in LA-CMNDP on T1 Instances

Test Problem	p	β	LA-CMNDP Optimal Values (Original Model)		Sensitivity Analysis - I LA-CMNDP Optimal Values (with restrictions on terminal nodes)				Sensitivity Analysis - II LA-CMNDP Optimal Values (with restrictions on terminal nodes)				Sensitivity Analysis - III LA-CMNDP Optimal Values (with restrictions on terminal nodes)			
			LA	Terminals	*LA*	Difference of LA from Original Model	Terminals	Restricted Nodes for Terminal Location	*LA*	Difference of LA from Original Model	Terminals	Restricted Nodes for Terminal Location	*LA*	Difference of LA from Original Model	Terminals	Restricted Nodes for Terminal Location
T1-1	**2**	1.3	579	59, 77	No Solution			59	No Solution			77	No Solution			59, 77
		1.4	575	34, 59	581	1.0%	16, 34		Not Applicable				581	1.0%	16, 34	
		1.5	575	59, 77	578	0.5%	34, 77		575	0.0%	16, 59		578	0.5%	16, 34	
	3	1.2	605	16, 59, 77	No Solution			59	No Solution			77	No Solution			16
		1.3	575	34, 59, 77									Not Applicable			
		1.4	575	16, 34, 59	581	1.0%	16, 34, 77		Not Applicable				575	0.0%	34, 59, 77	
		1.5	575	16, 34, 59	578	0.5%	16, 34, 77						575	0.0%	34, 59, 77	
T1-2	**2**	1.2	885	40, 66	885	0.0%	66, 68	40	No Solution			66	No Solution			40, 66
		1.3	885	40, 66	885	0.0%	66, 68									
		1.4	885	40, 66	885	0.0%	66, 68									
		1.5	877	40, 66	877	0.0%	66, 68									
	3	1.2	840	40, 66, 68	885	5.4%	6, 66, 68	40	No Solution			66	885	5.4%	6, 40, 66	68
		1.3	840	40, 66, 68	885	5.4%	6, 66, 68						885	5.4%	6, 40, 66	
		1.4	840	40, 66, 68	885	5.4%	6, 66, 68		840	0.0%	6, 40, 68		885	5.4%	6, 40, 66	
		1.5	840	40, 66, 68	850	1.2%	6, 66, 68		840	0.0%	6, 40, 68		840	0.0%	6, 40, 66	

CHAPTER 7

DELAY MANAGEMENT IN LATEST ARRIVAL CONSOLIDATION

LA-CMNDP provides delivery paths for all commodities together with a scheduling plan that involves arrival and departure times to stations. Due to the latest arrival consolidation structure, some commodities may have to wait at some stations of the transportation network for the late arriving commodities. These waiting times cause delays affecting all arrivals to destinations. In LA-CMNDP, each direct ride between stations is assumed to be performed by the same vehicle. If we extend this assumption in such a way to allow multiple vehicles to perform the direct rides, the delays occurring at stations then could be minimized. This extension arise a new decision issue of which vehicles wait at the stations for the arrival of late commodities and which vehicles depart without waiting. The minimum number of vehicles required within the service network can be determined for each feasible LA-CMNDP solution. Without exceeding these minimum numbers, the wait-depart decisions of each vehicle for the stations on its route and the corresponding schedules namely the timetables can be generated so as to minimize the waiting time of all commodities at all stations. In this chapter, we present the basic decision issues and the modeling framework that we use for the delay management problem in latest arrival consolidation problems in Section 7.1. In Section 7.2, we present the mathematical model and discuss on the possible solution procedures.

7.1 Delay Management in Latest Arrival Consolidation

After LA-CMNDP has been solved, the next planning problem is to determine the vehicle routes for carrying the commodities through their delivery paths. By analyzing any feasible solution, the minimum number of vehicles required to perform the deliveries could be determined together with the initiating station for each vehicle. LA-CMNDP constructs the commodity paths by assuming that each direct ride between stations is performed by the same vehicle. Although the maximum latest arrival time to destinations is minimized, at some stations of the service network departing vehicles may have to wait for the late arriving commodities. These waiting times delay the departure times of the vehicles and the commodities arrive to destinations with these delays.

Without exceeding the minimum vehicle numbers, if multiple vehicles are allowed to perform the direct rides between the stations, the delays experienced by commodities could be minimized. The objective of minimizing the sum of all delays over all commodities force the vehicles depart as early as possible without waiting for the late arrivals. On the other hand, it may not be possible for all vehicles to depart from stations without waiting for the late arrivals, since the number of vehicles is limited.

For the delay management problem in latest arrival consolidation (DLA), in Section 7.1.1 we present the basic decision issues including the notation. To construct the mathematical model, we use event-activity-network representation and in Section 7.1.2 we present the related event-activity-network structure in detail.

7.1.1 Basic Decision Issues

LA-CMNDP provides a delivery path P^k for each commodity $k \in K$ in a transportation network $G=(N,A)$. Let V be the set of vehicles that are required to deliver all commodities in K. For each vehicle $v \in V$, the originating station $O(v)$, and a set of alternative routes R^v can be identified by analyzing any feasible LA-CMNDP solution. Each individual route $r \in R^v$ consists of a set of stations $N^v \subseteq N$ which vehicle v runs through. We assume that individual routes of vehicles include each station $i \in N$ only once as Schöbel (2006), in order to avoid a time dependent representation of stations.

The wait-depart decisions of vehicles at stations establish the timetables and affect the arrival times of commodities to their destinations. To manage the delays within the transportation system, the arrival and the departure times of vehicles to stations have to be modeled. Let DA_i^v denote the arrival time of vehicle v to station i and DT_i^v denote the departure time of vehicle v from station i. We assume the same constant handing time δ_i is spent for the load-unload operations by all vehicles stopping at station i.

There are three basic operations for a vehicle v passing through a station i, illustrated in Figure 45:

(i) stop and wait for the late arriving commodities,

(ii) stop and collect/deliver commodities,

(iii) continue the route without stopping.

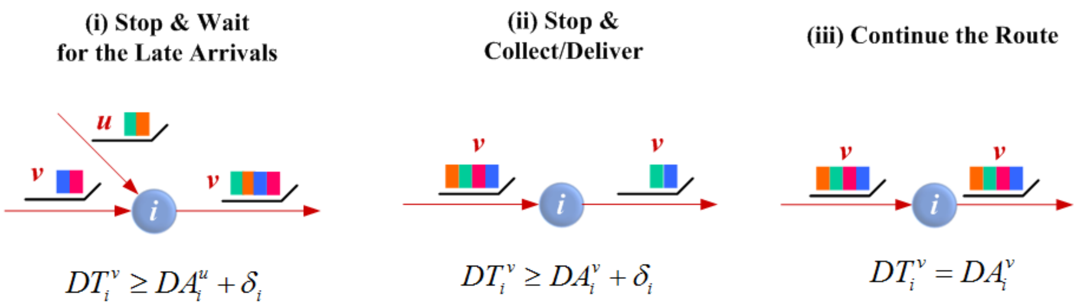


Figure 45. Possible Operations for a Vehicle v Passing Through a Station

It should also be noted that, operations (i) and (ii) can be performed together by a vehicle stopping at a station.

(i) *Stop & Wait for the Late Arrivals:*

If vehicle v waits at station i for the arrival of vehicle u, then vehicle v can depart only after vehicle u arrives and the unload/load operations are performed:

$$DT_i^v \geq DA_i^u + \delta_i \qquad (7.1).$$

(ii) *Stop & Collect/Deliver:*

In this option,

- new commodities can be loaded to v,

- some of the commodities on v can be unloaded.

The new commodities have two sources, either they are unloaded from a vehicle that departs before v arrives to station i or station i can be the originating station of the new commodities. The unloaded commodities can be loaded to other vehicles departing from station i or station i can be the destination of these commodities. Vehicle v can depart from station i after experiencing δ_i amount of time for the corresponding load/unload operations:

$$DT_i^v \geq DA_i^v + \delta_i \tag{7.2}$$

(ii) *Continue the Route:*

In this option, vehicle v does not stop at station i, does not experience any handling time, and does not wait for the other vehicles. Therefore, the departure time of vehicle v from station i equals to the arrival time of vehicle v to station i:

$$DT_i^v = DA_i^v \tag{7.3}$$

All vehicles cannot always choose the depart option at all stations. Since there is limited number of vehicles within the transportation system, at some stations some of the vehicles have to wait for the late arrivals. The wait option is depicted in Figure 46 on a small network consisting of 5 stations, 3 commodities (a, b, c), and 2 vehicles (v, u).

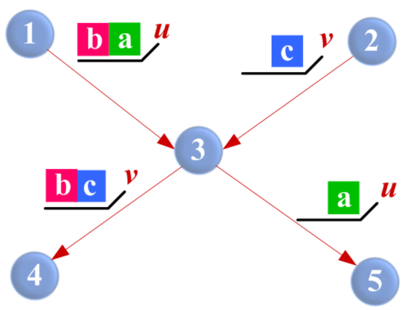


Figure 46. An Example Representation for the Wait Option

If vehicle v arrives at station 3 earlier than vehicle u, $DA_3^v < DA_3^u$, then v has to wait at station 3 for the arrival of u, since the delivery from station 3 to station 4 has to be performed by one vehicle.

7.1.2 Event Activity Network Representation

DLA uses the following inputs obtained and derived from the solution of LA-CMNDP:

- the delivery paths P^k of all commodities in K,
- the minimum number of vehicles $|V|$ required to deliver all commodities in K through their delivery paths P^k,
- the originating stations $O(v)$ of each vehicle in V,
- the set of alternative routes R^v for each vehicle in V.

DLA

- determines the wait-depart decisions of each vehicle at each station,
- determines the vehicle routes together with the timetables indicating the arrival and departure times to stations,
- determines a routing plan for each commodity, namely assigns each commodity k to vehicle routes on its delivery path P^k.

The input-output relationship of LA-CMNDP and DLA is presented in Figure 47.

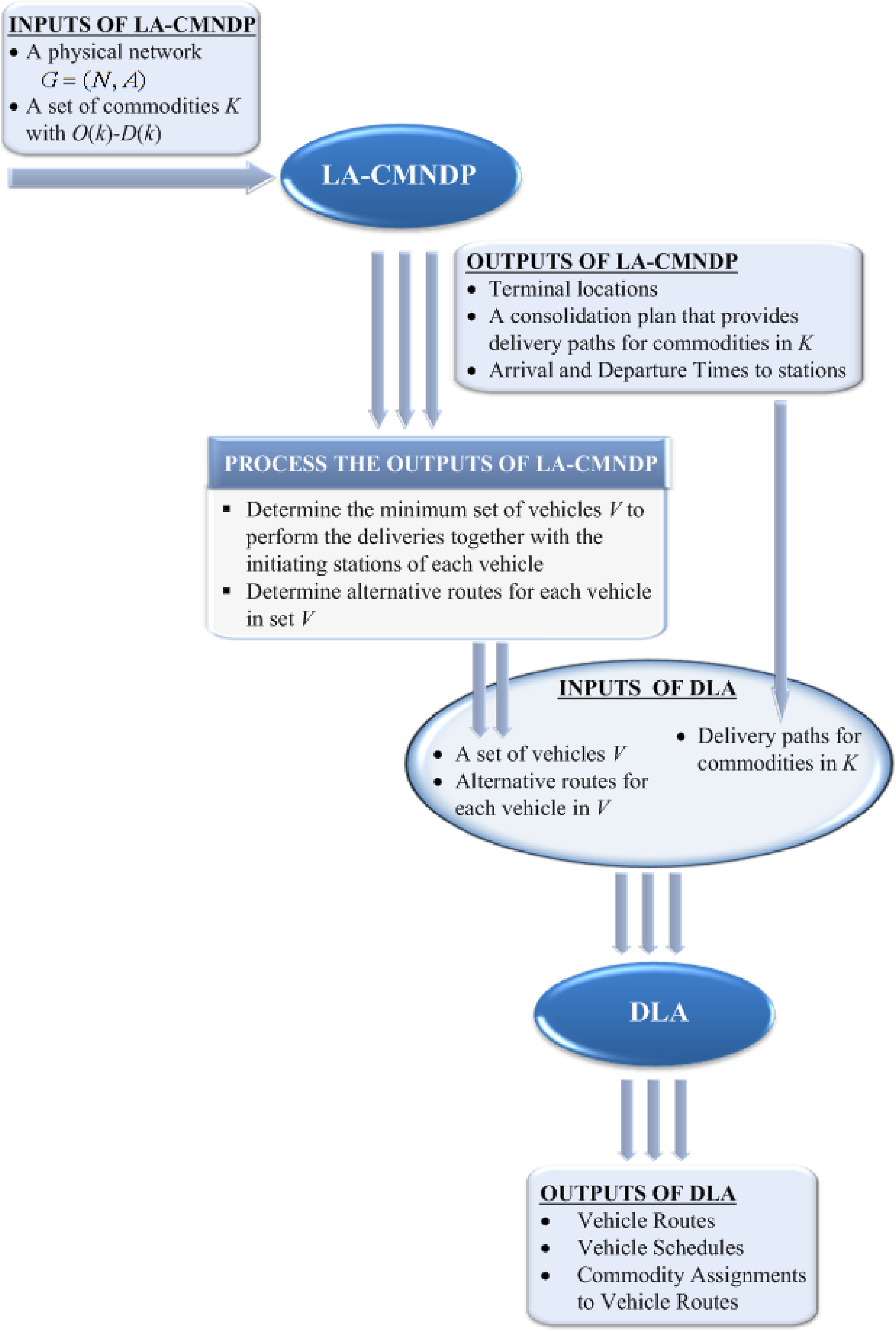


Figure 47. The Input-Output Relationship of LA-CMNDP and DLA

In delay management and timetabling studies, the activity on arc project network representation is generally used to model the problems, (Schöbel (2001), Schöbel (2006), Ginkel and Schöbel (2007), Heilporn et al. (2008)). Likewise, we generate the modeling structure of DLA, by using an event activity network representation that is similar to the one presented by Schöbel (2006).

Let $EAN = (E, \theta)$ represent the event activity network of DLA. E is the set of nodes that represents the arrival and the departure events of the vehicles, and θ is the set of directed arcs that represents the driving, changing, and passing activities. The sets of arcs in θ that enter and emanate from event $g \in E$ are denoted by $In(g)$ and $Out(g)$, respectively. Using the event activity network structure of Schöbel (2006), the events and the activities in $EAN = (E, \theta)$ is defined as follows:

- $E = E_{arr} \cup E_{dep}$
- $E_{arr} = \{(v,i,arr): the\ arrival\ event\ of\ vehicle\ v \in V\ to\ station\ i \in N\}$
- $E_{dep} = \{(v,i,dep): the\ departure\ event\ of\ vehicle\ v \in V\ from\ station\ i \in N\}$
- E^v_{arr}: the set of nodes in E_{arr} representing the arrival events of vehicle v
- E^v_{dep}: the set of nodes in E_{dep} representing the departure events of vehicle v
- $E^v = E^v_{arr} \cup E^v_{dep}$
- $\theta = \theta_{drive} \cup \theta_{pass} \cup \theta_{change}$
- $\theta_{drive} = \{((v,i,dep),(v,j,arr)) \in E_{dep} \times E_{arr} : vehicle\ v\ drives\ from\ station\ i\ to\ j\}$
- $\theta_{pass} = \{((v,i,arr),(v,i,dep)) \in E_{arr} \times E_{dep} : vehicle\ v\ passes\ through\ station\ i\}$
- $\theta_{change} = \{((v,i,arr),(u,i,dep)) \in E_{arr} \times E_{dep} : commodities\ change\ from\ vehicle\ v\ to\ u\ at\ station\ i\}$

Driving and passing activities are performed by both the vehicles and the commodities, whereas the changing activities are only performed by commodities. The passing activities θ_{pass} represent the possible options for a vehicle passing through a station. As previously discussed in Section 7.1.1, these options are:

(i) stop & wait for the late arrivals,

(ii) stop & collect/deliver,

(iii) continue the route.

For an example transportation network given in Figure 48, the corresponding event activity network representation $EAN = (E, \theta)$ is presented in Figure 49.

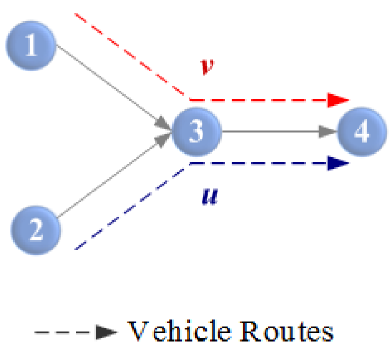


Figure 48. An Example Transportation Network with Vehicle Routes

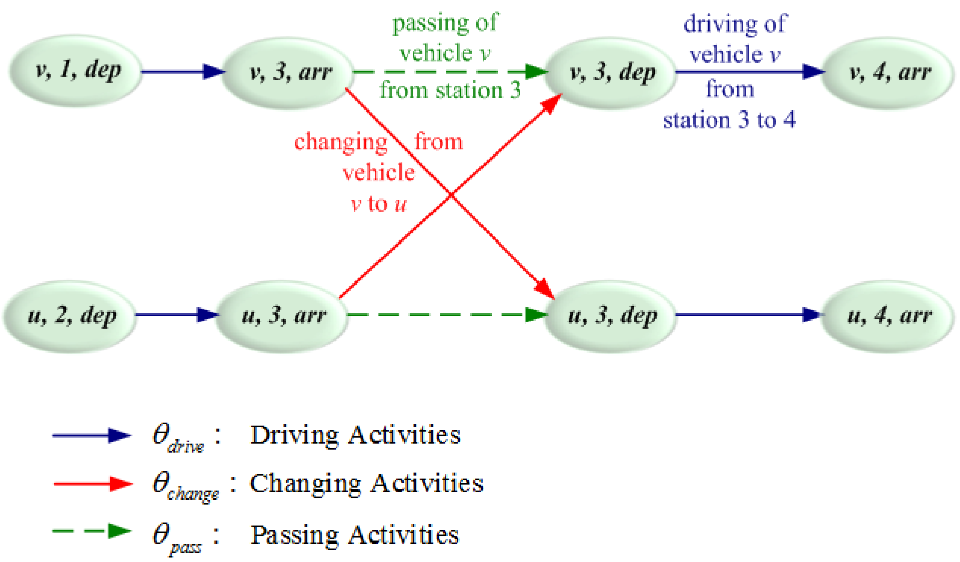


Figure 49. The *EAN* Representation of the Transportation Network Presented in Figure 48

To model the vehicle routes and the commodity assignments to vehicles in DLA, we define the following decision variables:

- $y_{gh} = \begin{Bmatrix} 1 & \textit{if the activity } (g,h) \in \theta \textit{ is performed} \\ 0 & \textit{otherwise} \end{Bmatrix}$
- $x_{gh}^{k} = \begin{Bmatrix} 1 & \textit{if commodity } k \in K \textit{ performs the activity } (g,h) \in \theta \\ 0 & \textit{otherwise} \end{Bmatrix}$
- $z_{gh} = \begin{Bmatrix} 1 & \textit{if handling operations are required to perform} \\ & \textit{the activity } (g,h) \in \theta_{pass} \\ 0 & \textit{otherwise} \end{Bmatrix}$
- Π_g : the schedule of event $g \in E$
- DA^k : the arrival time of commodity $k \in K$ to its destination $D(k)$.

Π_g denotes the

- arrival time DA of vehicles to stations for $g \in E_{arr}$,
- departure time DT of vehicles from stations for $g \in E_{dep}$.

In a feasible solution of DLA; the y_{gh} variables give us the route $r \in R^v$ of each vehicle v together with the related wait-depart decisions and the operations performed by each vehicle v at stations on its route, the x_{gh}^k variables give us the commodity assignments to vehicle routes, the Π_g variables give us the timetable of the transportation system, and DA^k variables give us the arrival time of commodities to their destinations.

To obtain a feasible timetable, the constraints that have to be satisfied by Π_g variables of DLA are defined in (7.4)-(7.12) considering the precedence relations of the events in $EAN = (E, \theta)$ that are determined by the performed activities in θ and the commodity assignments to the performed activities.

In DLA, the ready times of the commodities at their origins are assumed to be zero. The initiating stations $i = O(v)$ of vehicles can be of two types. In the first type, vehicle v begins its route at station i by loading only the commodities k, whose $O(k) = i$. In the second type, vehicle v begins its route at station i by loading the commodities, whose $O(k) \neq i$. The initiating stations of the vehicles in the first set is denoted as N^b and the second set is denoted as N^m. The Π_g of the arrival events of the vehicles in V, whose $O(v) \in N^b$, is equal to zero. Since these vehicles can start their routes without waiting for the arrival of other vehicles:

$$\Pi_g = 0 \quad \forall g \in E_{arr} = \{(v,i,arr) \mid i = O(v) \; and \; i \in N^b\} \tag{7.4}$$

The vehicles v, whose $O(v) \in N^m$, have to wait at $O(v)$ for the arrival of other vehicles u, (see Figure 50). Therefore, the arrival time of the vehicles v are set greater than or equal to the arrival time of the vehicles u:

$$\Pi_g \geq \Pi_f y_{fh} \quad \forall (g,h) \in \theta_{pass} = \{((v,i,arr),(v,i,dep)) \mid i = O(v) \; and \; i \in N^m\},$$
$$\forall (f,h) \in \theta_{change} = \{((u,i,arr),(v,i,dep)) \mid i = O(v) \; and \; i \in N^m\} \tag{7.5}$$

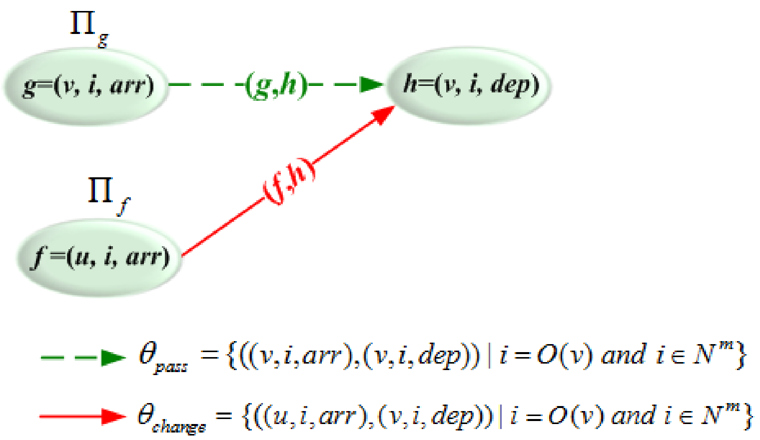


Figure 50. The *EAN* Representation for a Vehicle *v* That Can Start Its Route at Station *i*=*O*(*v*) After the Arrival of Vehicle *u*

The possible operations that can be performed by a vehicle passing through a station have different effects on the schedules that are defined by constraints (7.6)-(7.12).

In *stop & wait for the late arrivals* case, vehicle v waits at station i for the late arriving commodities and a vehicle change happens from the late arriving vehicle u to vehicle v, (see Figure 51). Vehicle v can depart from station i only after vehicle u arrives to station i and the load/unload operations are performed:

$$\Pi_h \geq (\Pi_f + \delta_i) y_{fh} \qquad \forall (f,h) \in \theta_{change} = \{((u,i,arr),(v,i,dep))\} \qquad (7.6).$$

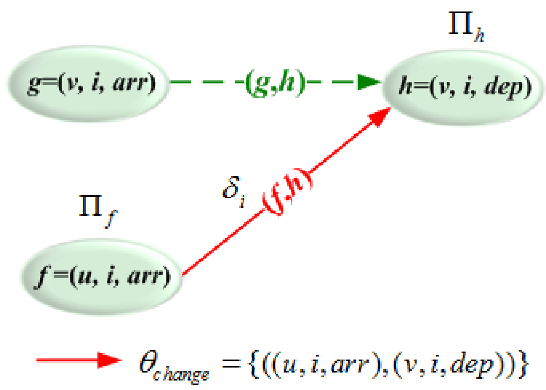


Figure 51. The *EAN* Representation for a Vehicle v That Waits at Station i for the Late Arriving Vehicle u

In *stop & collect/deliver* case, the handling time δ_i spent by vehicle v at station i for the load operations (*collect*) are considered by constraints (7.7)-(7.9):

$$\Pi_h \geq \Pi_g + \delta_i z_{gh} \qquad \forall (g,h) \in \theta_{pass} = \{((v,i,arr),(v,i,dep))\} \qquad (7.7)$$

$$z_{gh} \geq x_{gh}^k \qquad \forall k \in K,\ (g,h) \in \theta_{pass} = \{(v,i,arr),(v,i,dep) \mid i = O(k)\} \qquad (7.8)$$

$$\Pi_h \geq (\Pi_g + \delta_i) y_{fh} \qquad \forall (g,h) \in \theta_{pass} = \{((v,i,arr),(v,i,dep))\}, \quad \forall (f,h) \in \theta_{change} = \{((u,i,arr),(v,i,dep))\} \qquad (7.9).$$

Together with constraints (7.7) and (7.8) define the load operations of the new commodities k that start their route at station $i = O(k)$, (see Figure 52).

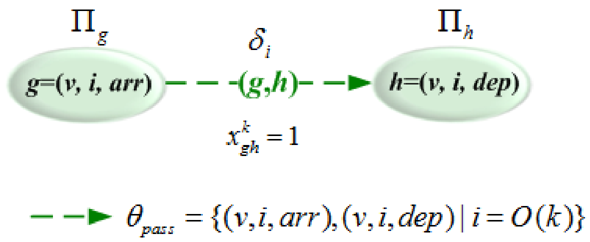


Figure 52. The *EAN* Representation for the Load Operations of Vehicle *v* at Station *i* for The New Commodities *k* With *i*=*O*(*k*)

Constraints (7.9) define the load operations of the commodities that have arrived to station *i* before vehicle *v* arrives, (see Figure 53).

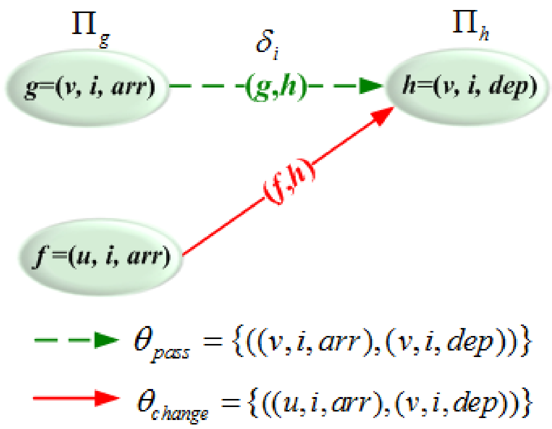


Figure 53. The *EAN* Representation for the Load Operations of Vehicle *v* at Station *i* for the Commodities That Have Arrived to Station *i* Before Vehicle *v*

In *stop & collect/deliver* case, the handling time δ_i spent by vehicle *v* at station *i* for the unload operations (*deliver*) are considered by constraints (7.7) and (7.10)-(7.11):

$$\Pi_h \geq \Pi_g + \delta_i z_{gh} \quad \forall (g,h) \in \theta_{pass} = \{((v,i,arr),(v,i,dep))\} \tag{7.7}$$

$$z_{gh} \geq x^k_{fg} \quad \forall k \in K, (f,g) \in \theta_{drive} = \{(v,j,dep),(v,i,arr) \mid i = D(k)\} \tag{7.10}$$

$$\Pi_h \geq (\Pi_g + \delta_i) y_{gf} \quad \forall (g,h) \in \theta_{pass} = \{((v,i,arr),(v,i,dep))\}, \quad \forall (g,f) \in \theta_{change} = \{((v,i,arr),(u,i,dep))\} \tag{7.11}$$

Together with constraints (7.7) and (7.10) define the unload operations of the commodities k that arrive to their destination at station $i = D(k)$, (see Figure 54).

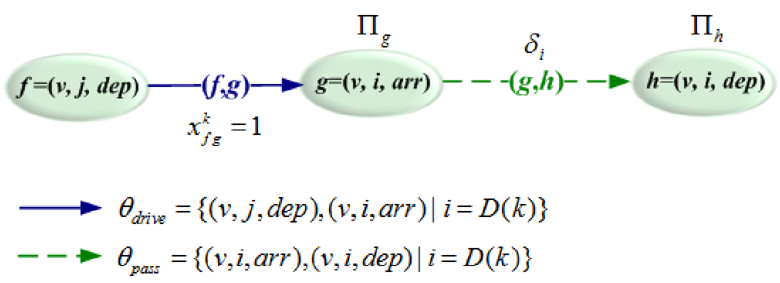


Figure 54. The *EAN* Representation for the Unload Operations of Vehicle v at Station i for the Commodities k With i=D(k)

Constraints (7.11) define the unload operations of the vehicle v at station i for the vehicle changes from vehicle v to u, (see Figure 55).

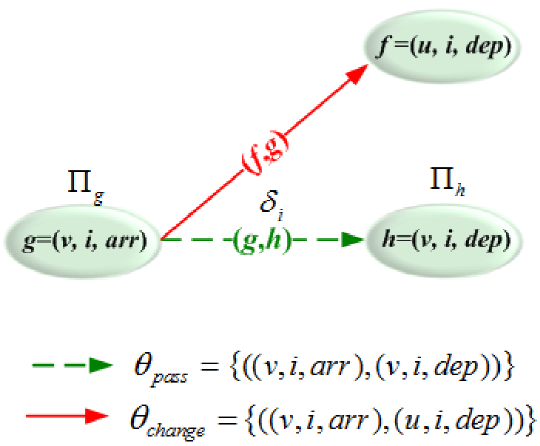


Figure 55. The *EAN* Representation for the Unload Operations of Vehicle v at Station i for the Vehicle Changes From v to u

If vehicle v does not stop at station i and continues its route, then $z_{gh} = 0$ and constraint (7.7) would be $\Pi_h \geq \Pi_g$ corresponding to *continue the route* case.

The departing vehicle v from station i can reach the next station j on its route only after experiencing the travel time t_{ij} between stations, (see Figure 56).

$$\Pi_g \geq (\Pi_h + t_{ij}) y_{hg} \qquad \forall (h,g) \in \theta_{drive} = \{((v,i,dep),(v,j,arr))\} \tag{7.12}$$

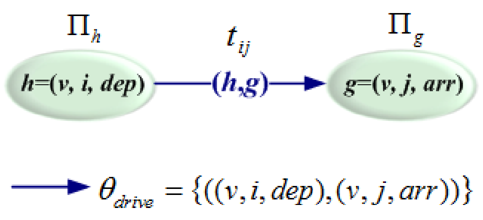


Figure 56. The *EAN* Representation for the Driving of Vehicle *v* From Station *i* to *j*

The arrival times of commodities to their destinations is obtained by the following sets of constraints:

$$DA^k \geq \Pi_g \qquad \forall k \in K, g \in E_{arr} = \{(v,i,arr) \mid i = D(k)\} \tag{7.13}$$

7.2 Mathematical Model

To model the vehicle routes in DLA, a modeling structure that is similar to the one presented by Ahuja et. al. (2005) for train scheduling problem is used. The train scheduling problem uses the blocking plan as an input and determines the number of trains to be scheduled; the arrival and the departure times of each train to each station; the route of each train including the initiating and the final stations; and assignments of blocks to the trains. Although DLA is similar to the train scheduling problem, there are differences between the two problems. In DLA, the minimum number of vehicles that are required to perform the deliveries and the initiating stations of the vehicles are known, whereas they are decision variables in train scheduling problem.

In modeling network structure of Ahuja et. al. (2005), two dummy nodes; a source node ϕ representing the supply of trains and a sink node s representing the train terminations are added to the scheduling network with the assumption that at source node ϕ a unit flow of distinct commodities each representing a train is available to be sent to the stations of the transportation network $G = (N, A)$, (see Figure 57). The trains that are to be scheduled are sent from node ϕ to their initiating stations, which are determined by the mathematical model. The scheduled and the unscheduled trains terminate at node s. The arc from node ϕ to a station $i \in N$ is used for the scheduled trains and the arc (ϕ, s) is used for the unscheduled trains.

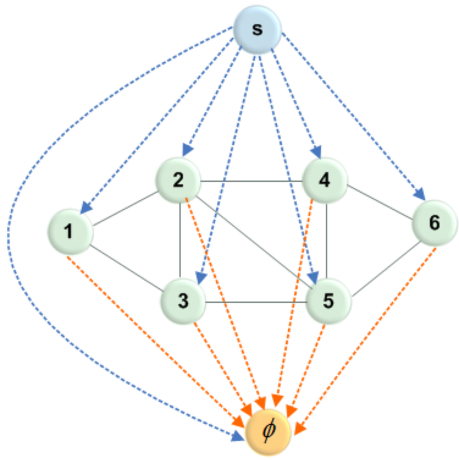
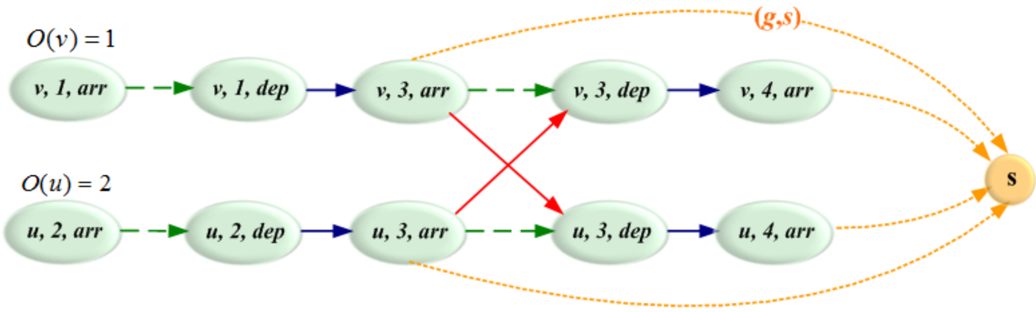


Figure 57. The Network Structure of Ahuja et al. (2005) Used For Modeling Train Routes

Since in DLA the minimum number of vehicles required to perform the deliveries, $|V|$, are known with the initiating stations $O(v)$ for each vehicle, only the dummy sink node s is added to the EAN as a termination event. Each vehicle v begins its route at $O(v)$, passes through a set of stations $i \in N$, and terminates its route at s. In order to model the vehicle routes from stations $O(v)$ to s in EAN, an arc from each arrival event in $g \in E^{*}_{arr} = \{(v,i,arr) \mid i \neq O(v)\}$ to s is added to the EAN and the arcs (g,s) define the termination activities $\theta_{terminate}$:

$$\theta_{terminate} = \{((v,i,arr),s) \mid i \neq O(v) : \textit{vehicle v terminates its route at station i}\}.$$

The extended EAN structure that includes the termination event s and the termination activities $\theta_{terminate}$ is presented in Figure 58. It should be noted that termination activities are only performed by vehicles. Each commodity k terminates its route at one of the arrival events of its destination station, $g \in E^{**}_{arr} = \{(v,i,arr) \mid i = D(k)\}$.



$\theta_{terminate}$: Termination Activities

$E := E \cup \{s\}$

$\theta := \theta \cup \theta_{terminate}$

$(g, s) \in \theta_{terminate} = \{((v, i, arr), s) \mid i \neq O(v)\}$

Figure 58. The Extended *EAN* Structure Used to Model the Vehicle Routes in DLA

Using the extended *EAN* structure, DLA is formulated so as to minimize the waiting times of all commodities at all stations. For this purpose, the objective function of DLA minimizes the sum of arrival times to destinations of all commodities, $\sum_{k \in K} DA^k$.

DLA

Minimize $\sum_{k \in K} DA^k$

subject to

$(7.4) - (7.13)$

$$\sum_{(g,h) \in Out(g)} y_{gh} - \sum_{(h,g) \in In(g)} y_{hg} = \begin{cases} 1 & \text{if } g \in E^v_{arr} = \{(v,i,arr) \mid i = O(v)\} \\ 0 & \text{if } g \in E^v = \{(v,i,:) \mid i \neq O(v)\} \\ -1 & \text{if } g = s \end{cases} \quad ,\forall v \in V \quad (7.14)$$

$$\sum_{g \in E_{arr} = \{(v,i,arr) \mid i = O(k)\}} \left(\sum_{(g,h) \in Out(g)} x^k_{gh} - \sum_{(h,g) \in In(g)} x^k_{hg} \right) = 1$$
$$\forall k \in K,\ \forall (g,h),(h,g) \in E \times E = \{((v,i,:),(v,j,:)) \mid \{i,j\} \in P^k\} \quad (7.15)$$

$$\sum_{g \in E_{arr} = \{(v,i,arr) \mid i = D(k)\}} \left(\sum_{(g,h) \in Out(g)} x^k_{gh} - \sum_{(h,g) \in In(g)} x^k_{hg} \right) = -1$$
$$\forall k \in K,\ \forall (g,h),(h,g) \in E \times E = \{((v,i,:),(v,j,:)) \mid \{i,j\} \in P^k\} \quad (7.16)$$

$$\sum_{(g,h) \in Out(g)} x^k_{gh} - \sum_{(h,g) \in In(g)} x^k_{hg} = 0 \quad \forall k \in K,$$
$$\forall g \in E = \{g = (v,i,:) \mid i \in P^k,\ g \neq (v, O(k), arr) \text{ and } g \neq (v, D(k), arr)\},$$
$$\forall (g,h),(h,g) \in E \times E = \{((v,i,:),(v,j,:)) \mid \{i,j\} \in P^k\} \quad (7.17)$$

$$x^k_{gh} \leq y_{gh} \qquad \forall k \in K, \forall (g,h) \in E \times E = \{((v,i,:),(v,j,:)) \mid \{i,j\} \in P^k\} \quad (7.18)$$

$$x^k_{gh} \in \{0,1\},\ y_{gh} \in \{0,1\},\ z_{gh} \in \{0,1\} \qquad \forall k \in K,\quad \forall (g,h) \in \theta \quad (7.19)$$

$$\Pi_g \geq 0 \qquad \forall g \in E \quad (7.20)$$

The objective function minimizes the sum of the arrival times of all commodities to their destinations. Constraints (7.4)-(7.12) state the feasibility of the timetable of the transportation system. Constraints (7.13) determine the arrival time of commodities to their destinations. The vehicle routes and the assignments of commodities to the vehicle routes are modeled as multicommodity network flows using flow balance constraints. Constraints (7.14) are the flow balance of vehicle routes and constraints (7.15)-(7.17) are the flow balance of commodity assignments. By constraints (7.18), each commodity *k* can only be assigned to an activity *g*, if *g* is selected by the model and *g* is an activity of the stations that are on P^k. Constraints (7.19)-(7.20) are the nonnegativity constraints. DLA is a nonlinear mixed integer programming problem, due to the timetable constraints of (7.5)-(7.7), (7.9), (7.11), and (7.12).

7.2.1 Flow Balance of Vehicle Routes and Commodity Assignments

In this section, the explicit forms of the flow balance constraints of the vehicle routes and the commodity assignments are stated on small example transportation networks. In Figure 59, a set of example routes are given for a vehicle v and in Figure 60 the corresponding *EAN* representation is presented.

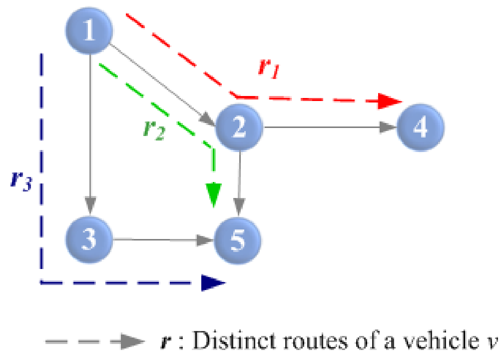


Figure 59. A Set of Example Routes for a Vehicle v

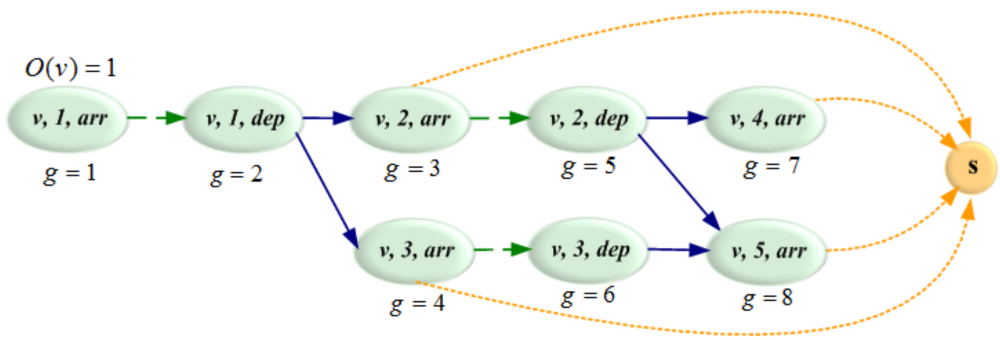


Figure 60. The *EAN* Representation of the Vehicle Routes Given in Figure 59

Constraints (7.14) ensure that each vehicle in V can only be routed through one of its alternative routes R^v. The explicit forms of the constraints (7.14) for the *EAN* given in Figure 60 are as follows:

$$
\begin{array}{lll}
g = O(v): & \{ g = 1 & y_{12} = 1 \\
g \neq O(v) \text{ and } g \neq s: & \begin{cases} g = 2 \\ g = 3 \\ g = 4 \\ g = 5 \\ g = 6 \end{cases} & \begin{array}{l} y_{23} + y_{24} - y_{12} = 0 \\ y_{35} + y_{3s} - y_{23} = 0 \\ y_{46} + y_{4s} - y_{24} = 0 \\ y_{57} + y_{58} - y_{35} = 0 \\ y_{68} - y_{46} = 0 \end{array} \\
g \neq O(v) \text{ and } g \neq s: & \begin{cases} g = 7 \\ g = 8 \end{cases} & \begin{array}{l} y_{7s} - y_{57} = 0 \\ y_{8s} - y_{58} - y_{68} = 0 \end{array} \\
g = s: & \{ g = s & -y_{3s} - y_{4s} - y_{7s} - y_{8s} = -1.
\end{array}
$$

In Figure 61, a set of example vehicle routes are given for a commodity k and in Figure 62 the corresponding *EAN* representation is presented.

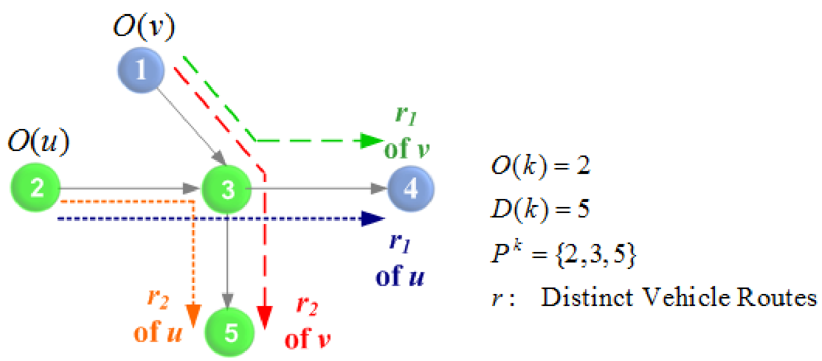


Figure 61. A Set of Alternative Vehicle Routes for a Commodity k

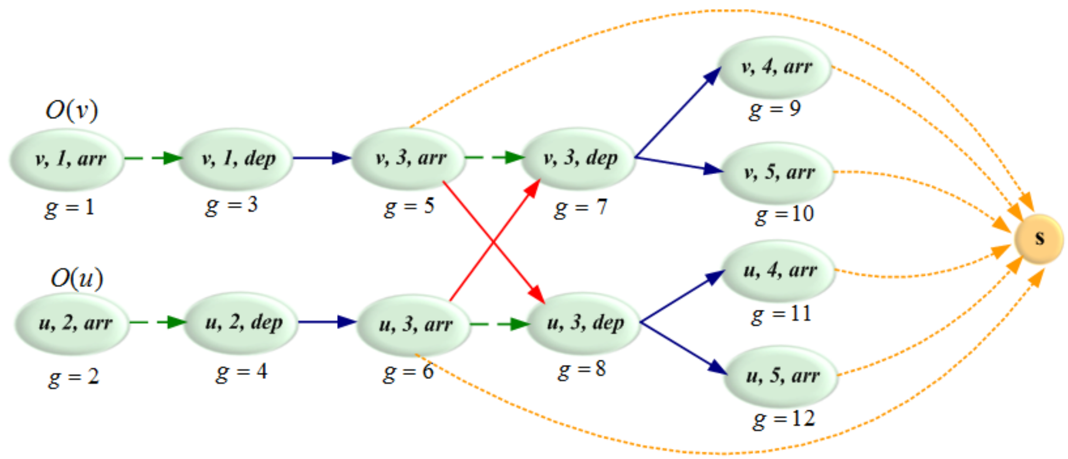


Figure 62. The *EAN* Representation of the Vehicle Routes Given in Figure 61

Constraints (7.15)-(7.17) assign commodities to vehicle routes such that each commodity is routed through its delivery path P^k following a sequence of vehicle routes. Constraints (7.15) and (7.16) state the flow balance of a commodity k for the events with $(v, O(k), arr)$ and $(v, D(k), arr)$, respectively. The explicit forms of the constraints (7.15) and (7.16) for the *EAN* given in Figure 62 are as follows:

$$g \in \{(v,i,arr) \mid i = O(k)\}: \quad g = \{2\} \quad x^k_{2,4} = 1$$
$$g \in \{(v,i,arr) \mid i = D(k)\}: \quad g = \{10,12\} \quad -x^k_{7,10} - x^k_{8,12} = -1.$$

Constraints (7.17) state the flow balance of the a commodity k for the events in $E \setminus \{(v,O(k),arr),(v,D(k),arr)\}$. The explicit forms of the constraints (7.17) for the *EAN* given in Figure 62 are as follows:

$$g \neq (v,O(k),arr) \; and \; g \neq (v,D(k),arr): \begin{cases} g = (u,2,dep) & x^k_{4,6} - x^k_{2,4} = 0 \\ g = (u,3,arr) & x^k_{6,7} + x^k_{6,8} - x^k_{4,6} = 0 \\ g = (v,3,dep) & x^k_{7,10} - x^k_{6,7} = 0 \\ g = (u,3,dep) & x^k_{8,12} - x^k_{6,8} = 0. \end{cases}$$

7.2.2 Solution Procedures

We solve DLA on a small example problem by applying the linearization structure presented in Section 4.1. We generate the problem instance using T1-1 network that has $|N| = 11$ and we generate 5 commodity paths on this network, ($|K| = 5$). Even for this small network, the minimum number of vehicles required to deliver the 5 commodities is equal to 4 and there are 11 alternative vehicle routes. The corresponding *EAN* consists of 34 nodes and 48 arcs. We obtain 45% decrease in total waiting times by DLA, showing the potential of our model to reduce the delays within the transportation system.

To solve large instances, the properties of DLA has to be considered. DLA is a nonlinear mixed integer programming problem. The nonlinearity is due to the constraints (7.5)-(7.7), (7.9), (7.11), (7.12) that state the feasibility conditions of the timetable. DLA has a similar structure to LA-CMNDP. When the binary variables x^k_{gh}, y_{gh}, z_{gh} are known, the nonlinear constraints become linear and DLA reduces to a linear programming problem that can be solved efficiently. Therefore, decomposition methods like Generalized Benders Decomposition (GBD) procedure can be applied to DLA. To increase the computational efficiency valid inequalities could also be incorporated into the decomposition scheme.

Also to reduce the size of the event-activity network in DLA, the shrinking method that is used by Ginkel and Shöbel (2007) could be applied. By this way, the same model would be used in an efficient structure that reduces the number of decision variables.

Tailored metaheuristic algorithms could also provide efficient solution procedures for DLA, especially for problems with realistic sizes. After constructing an initial routing plan for vehicles and a corresponding timetable algorithmically, improvements could be achieved by neighborhood search algorithms. Changes in vehicle routes and the wait-depart decisions of the vehicles at stations could be searched by tailored neighborhood moves.

CHAPTER 8

CONCLUSION

In this chapter, we first present a brief summary of this thesis stating our main findings and results in Section 8.1. Then, in Section 8.2 we list the main contributions of this thesis. Lastly, in Section 8.3 we provide possible future research directions.

8.1 Summary

In this thesis, we study on a multicommodity network design problem that applies latest arrival consolidation. We name this problem as LA-CMNDP and define the main properties of the problem and the resulting service network. We present a literature survey of the studies that are related to LA-CMNDP, including multicommodity network design problems, service network design problems, railroad planning problems, and the latest arrival problems. We present the properties and the capabilities of the models developed for these problems.

For LA-CMNDP, we develop exact and heuristic solution procedures and analyze their capabilities on different size test networks by performing extensive computational experiments. For this purpose, we generate 8 different-sized test networks considering different regions of Turkey.

In the exact solution procedures, we implement Generalized Benders Decomposition (GBD). We develop 8 different GBD algorithms for LA-CMNDP. We propose an alternative GBD cut and show the effectiveness of this alternative cut over the original GBD cut by performing computational experiments on 276 test instances. We also incorporate different sets of valid inequalities into the GBD algorithms and identify the ones that increase the quality of the solutions.

All GBD variants provide the optimum solutions within seconds for the test instances with $|N|=\{11,13\}$ and the commodity numbers satisfying the full-cross-traffic case. For the test instances having $|N|=\{21,25\}$, if we consider the average performances CPLEX provide better solutions than all GBD variants. However, as number of nodes and commodities increase, CPLEX cannot provide even an integer solution, whereas all GBD variants provide a feasible solution. These instances can be listed as: $|N|=33$, $|K|\geq 75$; $|N|=52$, $|K|\geq 200$; $|N|=62$, $|K|\geq 150$; $|N|=81$, $|K|\geq 100$.

Although on larger size instances all GBD algorithms outperform CPLEX, the computational times of GBD algorithms are around 12 minutes on average. The computational times and the quality of the solutions are further improved with a tailored heuristic algorithm. We develop an LNS metaheuristic algorithm for LA-CMNDP, which we call as LNS-LA.

We prove the effectiveness of LNS-LA by performing computational experiments on 327 test instances. We compare the computational performances of the best GBD algorithm and LNS-LA on test instances with $|N|=\{21,25,33,52,62,81\}$. In all instances, LNS-LA provides better solutions in significantly shorter computational times than the best performing GBD algorithm. Moreover, for the instances, which CPLEX cannot provide the optimal solution, LNS-LA provides solutions with better objective functions in less than 2 minutes computation time.

Under all different parameter settings of $(p,\beta,|K|)$, LNS-LA provides the optimum solutions in seconds for test networks with $|N|=\{11,13\}$. LNS-LA provides good quality solutions for test networks with $|N|=\{21\}$, $|K|\leq 420$; $|N|=\{25\}$, $|K|\leq 600$; $|N|=\{33\}$, $|K|\leq 1{,}026$ in computational times of less than 3 minutes. LNS-LA provides good quality solutions for test networks with $|N|=\{52\}$ and $|K|\leq 1{,}989$ in less than 10 minutes, for test networks with $|N|=\{62\}$ and $|K|\leq 2{,}837$ in less than 22 minutes, and for test networks with $|N|=\{81\}$ and $|K|\leq 4{,}860$ in less than an hour.

We compare LNS-LA and the single allocation latest arrival p-hub location problem (SLApHLP) of Kara and Tansel (2001) with respect to the effect of their networking structures on travel times. Although LA-CMNDP does not apply a discount factor to any of the arcs in the service network, even for $\alpha = 0.9$, LNS-LA provide solutions with lower *LA* values and lower total travel times. This is due to the flexible networking structure of LA-CMNDP. Compared to the strict hubbing structure of SLApHLP, in LA-CMNDP a node can be linked to multiple other nodes and the commodities are allowed to be delivered through paths consisting even 1 or 2 arcs as long as other constraints hold.

We analyze the effects of terminal locations on the service network of LA-CMNDP by performing a sensitivity analysis on test networks having $|N| \in \{11,13\}$. According to the sensitivity analysis results, obtaining an implementable LA-CMNDP solution is highly sensitive and the *LA* values are moderately sensitive to the locations of terminals.

We present a model for the delay management problem arising in service network design problems that apply latest arrival consolidation, which we call as DLA. Using event-activity-network representation of project networks, we develop a model that simultaneously determine the vehicle routes with schedules and the assignments of commodity paths to the vehicle routes. We present the computational results of DLA on a small transportation network and discuss on possible solution procedures.

8.2 Contributions

The main contributions of this thesis can be listed as follows:

- A taxonomy of network optimization problems in freight transportation is presented together with the main strategic, tactical, and operational planning problems in each transportation mode (ground, rail, air, maritime, and intermodal).
- A mathematical model for service network design problems that apply latest arrival consolidation is presented. The proposed model (LA-CMNDP)

allows more flexible routing than its counterparts in hub location studies; allows multiple allocation; does not require full-cross-traffic assumption for the latest arrival time calculations, can be both applied to incomplete and complete physical networks.

- Comprehensive literature reviews for Benders Decomposition and Generalized Benders Decomposition procedures are presented.
- For the main solution and enhancement techniques of BD and GBD procedures, classification schemes, which provide researchers a guideline for the possible set of techniques and the required problem features for applicability, are proposed.
- For the solution of LA-CMNDP, eight GBD based algorithms are developed. An alternative decomposition cut to the classical GBD cut is proposed. The proposed alternative decomposition cut improves the computational time performance and the solution quality of the classical GBD procedure. Further enhancements in GBD algorithms are analyzed by incorporating different sets of valid inequalities and experimenting on large number of test instances.
- A Large Neighborhood Search (LNS) metaheuristic is developed for LA-CMNDP. With the proposed metaheuristic algorithm, moderate sized test networks having up to 200 commodities can be solved significantly faster than CPLEX and all GBD variants. Larger networks (81 nodes, 4,860 commodities) that cannot be solved by CPLEX and GBD algorithms are solved in reasonable computing times.
- The proposed metaheuristic includes a set of interrelated subroutines that establish the basic algorithmic operations of latest arrival time calculations in multicommodity network flow problems. The presented subroutines can also be used in different modeling contexts (i.e. the packet switching problem in telecommunication networks) dealing with late arrivals.
- An integrated mathematical model for the delay management problem in service networks that apply latest arrival consolidation is proposed. The integrated modeling of vehicle routes and commodity assignments to vehicle routes are achieved by defining flow balance constraints on event-activity-networks.

8.3 Future Research Directions

Several enhancements can be incorporated into the latest arrival consolidation structure and new models under these considerations can be developed. Minimum allowable waiting time limits, in which all the departures from a station wait for all the arrivals within this limit, can be modeled. Priorities of the commodities can be considered in consolidation operations. Also, delivery time restrictions of commodities can be incorporated into the consolidation structure and a service network providing different levels of services can be modeled.

LA-CMNDP can be re-modeled considering the capacity restrictions of vehicles, the transportation infrastructure (terminals, handling equipments), and the resources (crew, handling operators).

The modeling concerns of DLA can be enhanced by incorporating time window constraints. Allowable waiting times can be defined for all vehicles stopping at a station. The effect of applying soft and hard time window constraints on the system wide delays can be analyzed.

Developing exact and heuristic solution procedures for model DLA can be studied. Generalized Benders Decomposition procedure can be implemented with possible enhancement techniques like incorporating valid inequalities, developing efficient solution procedures for the sub and the relaxed master problems. In addition, tailored heuristics can be developed for DLA.

An integrated model that develops the commodity paths, the vehicle routes, and the corresponding timetables simultaneously with latest arrival consolidation can be studied. Namely, the problems LA-CMNDP and DLA can be modeled together. For this integrated model, decomposition structures that partition the main model into sub problems and solve the sub problems in an iterative manner can be developed.

REFERENCES

1. Ahuja, R.K., Cunha, C.B., Şahin, G. (2005). Network Models in Railroad Planning and Scheduling. Tutorials in Operations Research, INFORMS, doi: 10.1287/educ.1053.0013.

2. Ahuja, R.K., Ergun, O., Orlin, J.B., Punnen, A.P. (2002). A Survey of Very Large-Scale Neighborhood Search Techniques. *Discrete Applied Mathematics*, 123, 75-102.

3. Ahuja, R.K., Jha, K.C., Liu, J. (2007). Solving Real Life Railroad Blocking Problems. *Interfaces*, 37 (5), 404-419.

4. Ahuja, R.K., Orlin, J.B., Sharma, D. (2000). Very Large-Scale Neighborhood Search. *International Transactions in Operational Research*, 7, 301-317.

5. Alumur, S.A. (2009). Hub Location and Hub Network Design. *Ph.D. Thesis*, Bilkent University Industrial Engineering Department, 06800 Bilkent, Ankara, Turkey.

6. Alvarez, A.M., Gonzalez, J.L., De-Alba, K. (2003). Scatter Search for a Network Design Problem. Technical Report, San Nicolas de los Ganza University, Mexico.

7. Alvarez, Gonzalez-Velarde, J.L., De-Alba, K. (2005a). Scatter Search for a Network Design Problem. *Annals of Operations Research*, 138, 159–178.

8. Alvarez, Gonzalez-Velarde, J.L., De-Alba, K. (2005b). GRASP Embedded Scatter Search for the Multicommodity Capacitated Network Design Problem. *Journal of Heuristics*, 11, 233-257.

9. Anderegg, L., Penna, P, Widmayer, P. (2009). Online Train Disposition: To Wait or Not to Wait? Robust and Online Large-Scale Optimization, 387-398. Springer-Verlag Berlin Heidelberg.

10. Andersan, J., Crainic, T.G., Christiansen, M. (2009). Service Network Design with Management and Coordination of Multiple Fleets. *European Journal of Operational Research*, 193, 377-389.

11. Aras, C.M., Kurose, J.F., Reeves, D.S., Schulzrinne, H. (1994). Real-Time Communication in Packet-Switched Networks. *Proceedings of the IEEE*, 82 (1), 122-139.

12. Armacost, A.P., Barnhart, C., Ware, K. A. (2002). Composite Variable Formulations for Express Shipment Service Network Design. *Transportation Science*, 36 (1), 1-20.

13. Armacost, A.P., Barnhart, C., Ware, K. A., Wilson, A.M. (2004). UPS Optimizes its Air Network. *Interfaces*, 34 (1), 15-25.

14. Bagajewicz, M.J., Manousiouthakis, V. (1991). On the Generalized Benders Decomposition. *Computers and Chemical Engineering*, 15 (10), 691-700.

15. Bai, R., Kendall, G., Li, J. (2010). An Efficient Guided Local Search Approach for Service Network Design Problem with Asset Balancing. *Logistics Systems and Intelligent Management, International 2010 Conference*, *IEEE*, 1, 110-115.

16. Bai, R., Kendall, G., Qu, R., Atkin, J.A.D. (2012). Tabu Assisted Guided Local Search Approaches for Freight Service Network Design. *Information Sciences*, 189, 266-281.

17. Barcos, L., Rodriguez, V., Alvarez, M.J., Robuste, F. (2010). Routing Design for Less-Than-Truckload Motor Carriers Using Ant Colony Optimization. *Transportation Research Part E*, 46, 367-383.

18. Barnhart, C., Schneur, R.R. (1996). Air Network Design for Express Shipment Service. *Operations Research*, 44 (6), 852-863.

19. Barnhart, C., Jin, H., Vance, P.H. (2000). Railroad Blocking: A Network Design Application. *Operations Research*, 48 (4), 603-614.

20. Barnhart, C., Krishnan, N., Kim, D., Ware, K. (2002). Network Design for Express Shipment Delivery. *Computational Optimization and Applications*, 21, 239-262.

21. Barnhart, C., Shen, S. (2005). Logistics Service Network Design for Time-Critical Delivery. *Practice and Theory of Automated Timetabling V*, Springer Berlin Heidelberg.

22. Benchakraun, V.G.A., Ferland, J., Gascon, V. (1997). Benders Decomposition for Network Design Problems with Underlying Tree Structure. *Investigacion Operativa*, 6 (1), 165-180.

23. Benders, J. F. (1962). Partitioning Procedures for Solving Mixed-Variables Programming Problems. *Numerische Mathematik*, 4 (1), 238-252.

24. Blum, C., Roli, A. (2003). Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison. *ACM Computing Surveys Archive,* 35 (3), 268-308.

25. Bodin, L.D., Bruce, L.G., Allan, D.S. (1980). A Model for the Blocking of Trains. *Transportation Research Part B*, 14, 115-120.

26. Bontekoning, Y.M., Macharis, C., Trip, J.J. (2004). Is a new applied transportation field emerging? – A Review of Intermodal Rail-Truck Freight Transport Lietarure. *Transportation Research Part A*, 38, 1-34.

27. Büedenbender, K., Grünert, T., and Sebastian, H.-J. (2000). A Hybrid Tabu Search/Branch and Bound Algorithm for the Direct Flight Network Design Problem. *Transportation Science*, 34 (4), 364–380.

28. Cai, X., McKinney, D.C., Lasdon, L.S., Watkins Jr, D. W. (2001). Solving Large Nonconvex Water Resources Management Models Using Generalized Benders Decomposition. *Operations Research*, 49 (2), 235-245.

29. Camargo, R. S., Miranda, G., Luna, H. P. (2008). Benders Decomposition for the Uncapacitated Multiple Allocation Hub Location Problem. *Computers and Operations Research*, 35, 1047-1064.

30. Camargo, R. S., Miranda, G., Luna, H. P. (2009a). Multiple Allocation Hub and Spoke Network Design Under Hub Congestion. *Computers and Operations Research*, 36, 3097-3106.

31. Camargo, R. S., Miranda, G., Luna, H. P. (2009b). Benders Decomposition for Hub Location Problems with Economies of Scale. *Transportation Science*, 43 (1), 86-97.

32. Campbell, J.F. (1994b). Integer programming formulations of discrete hub location problems. *European Journal of Operational Research*, 72 (2), 387-405.

33. Campbell, J.F. (2005). Strategic Network Design for Motor Carriers. In Logistics Systems: Design and Optimization, Springer US.

34. Campbell, J.F., Ernst, A.T., Krishnamoorthy, M. (2005a). Hub Arc Location Problems: Part I – Introduction and Results. *Management Science*, 51 (10), 1540–1555.

35. Campbell, J.F., Ernst, A.T., Krishnamoorthy, M. (2005b). Hub Arc Location Problems: Part II – Formulations and Optimal Algorithms. *Management Science*, 51 (10), 1556–1571.

36. Campetella, M., Lulli, G., Pietropaoli, U., Ricciardi, N. (2006). Freight Service Design for an Italian Railways Company. 6th Workshop on Algorithmic Methods and Models for Optimization of Railways, Switzerland.

37. Chouman, M., Crainic, T.G., Gendron, B. (2003). A Cutting-Plane Algorithm Based on Cutset Inequalities for Multicommodity Capacitated Fixed Charge Network Design. Technical Report, Centre for Research on Transportation, University of Montreal, Canada.

38. Chouman, M., Crainic, T.G., Gendron, B. (2009). A Cutting-Plane Algorithm for Multicommodity Capacitated Fixed Charge Network Design. Technical Report, Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation, Canada.

39. Chouman, M., Crainic, T.G. (2010). A MIP-Tabu Search Hybrid Framework for Multicommodity Capacitated Fixed-Charge Network Design. Technical

Report, Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation, Canada.

40. Cicerone, S., D'Angelo, G., Di Stefano, G., Frigioni, D., Navarra, A. (2008). Delay Management Problem: Complexity Results and Robust Algorithms. Combinatorial Optimization and Applications, 458-468. Springer Berlin Heidelberg.

41. Cicerone, S., D'Angelo, G., Di Stefano, G., Frigioni, D., Navarra, A. (2009). Recoverable Robust Timetabling for Single Delay: Complexity and Ploynomial Algorithms for Special Cases. *Journal of Combinatorial Optimization*, 18 (3), 229-257.

42. Clasen, R.J. (1984). The Solution of the Chemical Equilibrium Programming Problem with Generalized Benders Decomposition. *Operations Research*, 32 (1), 70-79.

43. Codato, G., Fischetti, M. (2006). Combinatorial Benders' Cuts for Mixed-Integer Linear Programming. *Operations Research*, 54 (4), 756-766.

44. Contreras, I., Cordeau, J.F., Laporte, G. (2011). Benders Decomposition for Large Scale Uncapacitated Hub Location. *Operations Research*, 59 (6), 1477-1490.

45. Cordeau, J.F., Soumis, F., Desrosiers, J. (2000). A Benders Decomposition Approach for the Locomotive and Car Assignment Problem. *Transportation Science*, 34 (2), 133–149.

46. Cordeau, J.F., Pasin, F., Solomon, M.M. (2006). An Integrated Model for Logistics Network Design. *Annals of Operations Research*, 144, 59-82.

47. Costa, A.M., (2005). A Survey on Benders Decomposition Applied to Fixed-Charge Network design Problems. *Computers and Operations Research*, 32 (6), 1429-1450.

48. Costa, A.M., Cordeau, J.F., Gendron, B. (2009). Benders, Metric and Cutest Inequalities for Multicommodity Capacitated Network Design. *Computational Optimization and Applications*, 42 (3), 371-392.

49. Crainic, T.G. (2000). Service Network Design in Freight Transportation. *European Journal of Operational Research,* 122, 272-288.

50. Crainic, T.G. (2005). A Survey of Optimization Models for Long-Haul Freight Transportation. Appeared as: Handbook of Transportation Science R.W. Hall (Ed.), 2nd Edition, 451-516, Kluwer, 2003.

51. Crainic, T.G., Ferland, J.-A., Rousseau, J.-M (1984). A Tactical Planning Model for Rail Freight Transportation. *Transportation Science*, 18 (2), 165–184.

52. Crainic, T.G., Laporte, G. (1997). Planning models for freight transportation. *European Journal of Operational Research*, 97, 409-438.

53. Crainic, T.G., Gendreau, M., Farvolden, J. M. (2000). A Simplex-Based Tabu Search Method for Capacitated Network Design. *INFORMS Journal on Computing*, 12 (3), 223–236.

54. Crainic, T.G., Frangioni, A., Gendron, B. (2001). Bundle-Based Relaxation Methods for Multicommodity Capacitated Fixed Charge Network Design. *Discrete Applied Mathematics*, 112, 73-99.

55. Crainic, T.G., Gendreau, M. (2002). Cooperative Parallel Tabu Search for Capacitated Network Design. *Journal of Heuristics,* 8, 601-627.

56. Crainic, T.G., Gendreau, M., Hernu, G. (2004). A Slope Scaling/Lagrangean Perturbation Heuristic with Long-Term Memory for Multicommodity Capacitated Fixed-Charge Network Design. *Journal of Heuristics,* 10, 525-545.

57. Crainic, T.G., Li, Y., Toulouse, M. (2006). A First Multilevel Cooperative Algorithm for Capacitated Multicommodity Network Design. *Computers & Operations Research*, 33, 2602-2622.

58. Crainic, T.G., Gendreau, M. (2007). A Scatter Search Heuristic for the Fixed Charge Capacitated Network Design Problem, Springer US. 25-40.

59. Crainic, T.G., Kim, K.H. (2007). Intermodal Transportation. In Handbooks in Operations Research and Management Science, vol. 14 Transportation, North Holland, Elsevier.

60. De Giovanni, L., Heilporn, G., Labbe, M. (2005). Polyhedral Study for the Delay Management Problem in Public Transportation. Technical Report, Université Libre de Bruxelles.

61. Dollevoet, T., Corman, F., D'Ariano, A., Huisman, D. (2012). An Iterative Optimization Framework for Delay Management and Train Scheduling. *Flexible Services and Manufacturing Journal*, 1-26.

62. Dollevoet, T., Huisman, D. (2013). Fast Heuristics for Delay Management with Passenger Rerouting. *Public Transport*, 6 (1-2), 67-84.

63. Ebery, J., Krishnamoorthy, M., Ernst, A., Boland, N. (2000). The Capacitated Multiple Allocation Hub Location Problem: Formulations and Algorithms. *European Journal of Operational Research*, 120, 614-631.

64. Equi, L., Gallo, G., Marziale, S., Weintraub, A. (1997). A Combined Transportation and Scheduling Problem. *European Journal of Operational Research*, 97, 94-104.

65. Erenguc, S.S., Tufekci, S., Zappe, C.J. (1993). Solving Time/Cost Trade-off Problems with Discounted Cash Flows Using Generalized Benders Decomposition. *Naval Research Logistics*, 40 (1), 25-50.

66. Erera, A., Hewitt, M., Savelsbergh, M., Zhang, Y. (2012). Improved Load Plan Design Through Integer Programming Based Local Search. *Transportation Science*, Articles in Press, 1–16.

67. Ernst, A.T., Hamacher, H., Jiang, H., Krishnamoorthy, Woeginger, G. (2009). Uncapacitated Single and Multiple Allocation p-hub Center Problems. *Computers and Operations Research*, 36, 2230-2241.

68. Farvolden, J.M., Powell, W.B. (1994). Subgradient Methods for the Service Network Design Problem. *Transportation Science*, 28 (3), 256–272.

69. Federgruen, A., Zipkin, P. (1984). A Combined Vehicle Routing and Inventory Allocation Problem. *Operations Research*, 32 (5), 1019-1037.

70. Ferreira, A., (1997). Planning Australian Freight Rail Operations: An Overview. *Transportation Research Part A*, 31 (4), 335-348.

71. Fischetti, M., Salvagnin, D., Zanette, A. (2008). Miminal Infeasible Subsystems and Benders Cuts. Technical Report, University of Padova, Padua, Italy.

72. Fischetti, M., Salvagnin, D., Zanette, A. (2010). A Note on the Selection of Benders' Cuts. *Mathematical Programming*, 124, 175-182.

73. Florian, M., Nguyen, S. (1974). A Method for Computing Network Equilibrium with Elastic Demands. *Transportation Science*, 8 (4), 321-332.

74. Floudas, C.A., Aggarwal, A., Ciric, A.R. (1989). Global Optimum Search for Nonconvex NLP and MINLP Problems. *Computers and Chemical Engineering,* 13 (10), 1117-1132.

75. Floudas, C.A., Aggarwal, A. (1990). A Decomposition Strategy for Global Optimum Search in the Pooling Problem. *ORSA Journal on Computing*, 2 (3) 225-235.

76. Floudas, C.A. (1995). Nonlinear and Mixed Integer Optimization. Oxford University Press, New York.

77. França, P.M., Luna, H.P.L. (1982). Solving Stochastic Transportation Location Problems by Generalized Benders Decomposition. *Transportation Science*, 16 (2), 113-126.

78. Frangioni, A., Gendron, M. (2009). 0-1 Reformulations of the Multicommodity Capacitated Network Design Problem. *Discrete Applied Mathematics*, 157, 1229-1241.

79. Gatto, M., Jacob, R., Peeters, L., Schöbel, A. (2005). The Computational Complexity of Delay Management. Graph Theoretic Concepts in Computer Science, Springer, Berlin Heidelberg, 227-238.

80. Gelareh, S., Nickel, S., (2008). A Benders Decomposition for Hub Location Problems Arising In Public Transport. *Operations Research Proceedings*, vol. 2007, Springer, Berlin Heidelberg, 129–134.

81. Gelareh, S., Nickel, S., Psinger, D. (2010). Liner Shipping Hub Network Design in A Competitive Environment. *Transportation Research Part E*. 46, 991-1004.

82. Gendron, B., Crainic, T.G., Frangioni, A. (1999). Multicommodity Capacitated Network Design. Chapter 1 in *Telecommunications Network Planning,* Norwell, M.A: Kluwer Academics Publishers, 1-19.

83. Geoffrion A.M. (1972). Generalized Benders Decomposition. *Journal of Optimization Theory and Applications*, 10 (4), 237-260.

84. Geoffrion A.M., Graves G.W. (1974). Multicommodity Distribution System Design by Benders Decomposition. *Management Science*, 20 (5), 822-844.

85. Ghamlouche, I., Crainic, T.G., Gendreau, M. (2003). Cycle-Based Neighborhoods for Fixed-Charge Capacitated Multicommodity Network Design. *Operations Research*, 51 (4), 655–667.

86. Ghamlouche, I., Crainic, T.G., Gendreau, M. (2004). Path Relinking, Cycle-Based Neighborhoods and Capacitated Multicommodity Network Design. *Annals of Operations Research*, 131, 109-133.

87. Ghiani, G., Laporte, G., Musmanno, R. (2004). Introduction to Logistics System Planning and Control, John Wiley & Sons, England.

88. Ginkel, A., Schöbel, A. (2007). To Wait or Not to Wait? The Bicriteria Delay Management Problem in Public Transportation. *Transportation Science*, 41 (4), 527-538.

89. Gorman, M.F. (1998). An Application of Genetic and Tabu Searches to the Freight Railroad Operating Plan Problem. *Annals of Operations Research*, 78, 51–69.

90. Grossman, I.E. (2002). Review of Nonlinear Mixed Integer and Disjunctive Programming Techniques. *Optimization and Engineering*, 3, 227-252.

91. Haghani, A.E. (1989). Formulation and Solution of A Combined Train Routing and Makeup, and Empty Car Distribution Model. *Transportation Research Part B*, 23 (6), 433–431.

92. Hansen P., Mladenovic N. (2001). Variable Neighborhood Search: Principles and Applications. *European Journal of Operational Research*, 130, 449-467.

93. Heilporn, G., De Giovanni, L., Labbe, M. (2008). Optimization Models for the Single Delay Management Problem in Public Transportation. *European Journal of Operational Research*, 189, 762-774.

94. Hewitt, M., Nemhauser, G.L., Savelsbergh, M.W. (2010). Combining Exact and Heuristic Approaches for the Capacitated Fixed Charge Network Flow Problem. *INFORMS Journal on Computing*, 22 (2), 314-325.

95. Hoang, H.H. (1982). Topological Optimization of Networks: A Nonlinear Mixed Integer Model Employing Generalized Benders Decomposition. *IEEE Transactions on Automatic Control*, 27 (1), 164-169.

96. Holmberg, K., Yuan, D. (2000). A Lagrangean Heuristic Based Branch-and-Bound Approach for the Capacitated Network Design Problem. *Operations Research*. 48 (3), 461-481.

97. Imai, A., Shintani, K., Papadimitriou, S. (2010). Multi-Port vs. Hub-and-Spoke Port Calls by Containerships. *Transportation Research Part E*, 45, 740-757.

98. Jansen, B., Swinkels, P.C.J., Teeuwen, G.J.A., van Antwerpen de Fluiter, B., Fleuren, H.A. (2004). Operational planning of a Large-Scale Multi-Modal Transportation System. *European Journal of Operational Research*, 156, 41-53.

99. Jarrah, A.I., Johnson, E., Neubert, L.C. (2009). Large-Scale, Less-than-Truckload Service Network Design. *Operations Research*, 57 (3), 609–625.

100. Jha, K.C., Ahuja, R.K., Şahin, G. (2008). New Approaches for Solving the Block-to-Train Assignment Problem. *Networks*, 51 (1), 48-62.

101. Jiang, W., Tang, L., Xue, S. (2009). A Hybrid Algorithm of Tabu Search and Benders Decomposition for Multi-Product Production Distribution Network Design. *Proceedings of the IEEE International Conference on Automation and Logistics Shenyang, China, August 2009*, 79-84.

102. Kara B.Y., Tansel B.C. (2001). The Latest Arrival Hub Location Problem. *Management Science*, 47 (10), 1408–1420.

103. Katayama, N., Chen, M., Kubo, M. (2009). A Capacity Scaling Heuristic for the Multicommodity Capacitated Network Design Problem. *Journal of Computational and Applied Mathematics*, 232, 90-101.

104. Katayama, N., Yuritomo, S. (2011). Combining Capacity Scaling and Local Branch Approaches for the Logistics Network Design Problem. *21st International Conference on Production Research*, Stuttgart, Germany.

105. Keaton, M.H. (1989). Designing Optimal Railroad Operating Plans: Lagrangean Relaxation and Heuristic Approaches. *Transportation Research Part B*, 23 (6), 415–431.

106. Kim, D. (1997). Large Scale Transportation Service Network Design: Models, Algorithms, and Applications. *Ph.D. Thesis,* Massachusetts Institute of Technology, Civil and Environmental Engineering Department, USA.

107. Kim, D., Barnhart, C., Ware, K., Reinhardt, G. (1999). Multimodal Express Package Delivery: A Service Network Design Application. *Transportation Science*, 33 (4), 391–407.

108. Kwon, O.K., Martland, C.D., Sussman, J.M. (1998). Routing and Scheduling Temporal and Heterogeneous Freight Car Traffic on Rail Networks. *Transportation Research Part E*, 34 (2),101 –115.

109. Kuby, M.J., Gray, R.G. (1993). The Hub Network Design Problem with Stopovers And Feeders: The Case of Federal Express. *Transportation Research Part A*, 27 (1), 1-12.

110. Kurose, J.F., Ross, W. (2000). Computer Networking A Top Down Approach Featuring the Internet. Addison-Wesley.

111. Lai, M.F., Lo, K. (2004). Ferry Service Network Design: Optimal Fleet Size, Routing, and Scheduling. *Transportation Research Part A*, 38, 305–328.

112. Lazimy, R. (1986). Extension of the Generalized Benders Decomposition. *Communications in Applied Numerical Methods*, 2, 195-203.

113. Li, X., Chen, Y., Barton, P.I. (2012). Nonconvex Generalized Benders Decomposition With Piecewise Convex Relaxations for Global Optimization of Integrated Process Design and Operations Problem. *Industrial and Engineering Chemistry Research,* 51, 7287-7299.

114. Li, X., Tomasgard, A., Barton, P.I. (2011). Nonconvex Generalized Benders Decomposition for Stochastic Separable Mixed Integer Nonlinear Programs. *Journal of Optimization Theory and Applications*, 151 (3), 425-454.

115. Liebchen, C., Schachtebeck, M., Schöbel, A., Stiller, S., Prigge, A. (2010). Computing Delay Resistant Railway Timetables. *Computers & Operations Research*, 37, 857-868.

116. Lin, B.L., Wang, Z.M., Ji, L.J., Tian, Y.M., Zhou, G.Q. (2012). Optimizing the Freight Train Connection Service Network of a Large-Scale Rail System. *Transportation Research Part B*, 46 (5), 649–667.

117. Liu, J., Ahuja, R.K., Sahin, G. (2008). Optimal Network Configuration and Capacity Expansion of Railroads. *Journal of the Operational Research Society*, 59, 911-920.

118. Luss, H. (1982). Operations Research and Capacity Expansion Problems: a Survey. *Operations Research*, 30 (5), 907-947.

119. Magnanti, T.L., Wong, R.T. (1981). Accelerating Benders Decomposition: Algorithmic Enhancement and Model Selection Criteria. *Operations Research*, 29 (3), 464-484.

120. Magnanti, T.L., Wong, R.T. (1984). Network Design and Transportation Planning: Models and Algorithms. *Transportation Science*, 18 (1), 1-55.

121. Mahey, P., Benchakroun, A., Boyer, F. (2001). Capacity and Flow Assignment of Data Networks by Generalized Benders Decomposition. *Journal of Global Optimization*, 20, 173-193.

122. Marin, A., Jaramillo, P. (2009). Urban Rapid Transit Network Design: Accelerated Benders Decomposition. *Annals of Operations Research*, 169 (1), 35-53.

123. Marin, A., Salmeron, J. (1996a). Tactical Design of Rail Freight Networks. Part I: Exact and Heuristic Methods. *European Journal of Operational Research*, 90 (1), 26-44.

124. Marin, A., Salmeron, J. (1996b). Tactical Design of Rail Freight Networks. Part II: Local Search Methods with Statistical Analysis. *European Journal of Operational Research*, 94, 43-53.

125. Marin, A., Salmeron, J. (1998). Electric Capacity Expansion Under Uncertain Demand: Decomposition Approaches. *IEEE Transactions on Power Systems*, 13 (2), 333-339.

126. Martinez-Crespo, J., Usaola, J., Fernandez, J.L. (2007). Optimal Security-Constrained Power Scheduling by Benders Decomposition. Electric Power Systems Research, 77, 739-753.

127. Mercier, A., Cordeau, J.F., Soumis, F. (2005). A Computational Study of Benders Decomposition for the Integrated Aircraft Routing and Crew Scheduling Problem. *Computers and Operations Research*, 32, 1451-1476.

128. McCusker, S., Hobbis, B.F. (2003). A Nested Benders Decomposition Approach to Locating Distributed Generation in a Multi Area Power System. *Networks and Spatial Economics*, 3 (2), 197-223.

129. Mcdaniel, D., Devine M. (1977). A Modified Benders' Partitioning Algorithm for Mixed Integer Programming. *Management Science*, 24 (3), 312-319.

130. Mladenovic N., Hansen P. (1997). Variable Neighborhood Search. *Computers and Operations Research*, 24 (11), 1097-1100.

131. Montemanni, R., Gambardella, L.M. (2005). The Robust Shortest Path Problem with Interval Data via Benders Decomposition. *4OR*, 3 (4), 315-328.

132. Munoz, E. (2010). Global Optimization for Structural Design by Generalized Benders Decomposition. *Ph.D. Thesis,* Technical University of Denmark, Department of Mathematics, Lyngby, Denmark.

133. Munoz, E., Stolpe, M. (2011). Generalized Benders Decomposition for Topology Optimization Problems. *Journal of Global Optimization*, 51 (1), 149-183.

134. Nemhauser, G.L., Wolsey, L.A. (1998). Integer and Combinatorial Optimization, Wiley. John Wiley & Sons, USA.

135. Newman, P. (1988). A Fast Packet Switch for the Integrated Services Backbone Network. *IEEE Journal on Selected Areas in Communications*, 6 (9), 1468-1479.

136. Newton, H.N., Barnhart, C., Vance, P.H. (1998). Constructing Railroad Blocking Plans to Minimize Handling Costs. *Transportation Science*, 32 (4), 330–345.

137. O'Kelly, M.E. (1986). The Location of Interacting Hub Facilities. *Transportation Science*, 20 (2), 92-106.

138. Oliveria, G.C., Costa, A.P.C., Binato, S. (1995). Large Scale Transmission Network Planning Using Optimization and Heuristic Techniques. *IEEE Transactions on Power Systems*, 10 (4), 1828-1834.

139. Papadakos, N. (2008). Practical Enhancements to Magnanti-Wong Method. *Operations Research Letters*, 36, 444-449.

140. Papadakos, N. (2009). Integrated Airline Scheduling. *Computers and Operations Research*, 36, 176-195.

141. Parakevopoulos, D.C., Bektas, T., Crainic, T.G., Potts, C.N. (2013). A Cycle-Base Evolutionary Algorithm for the Fixed-Charge Capacitated Multi-Commodity Network Design Problem. Technical Report,

Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation, Canada.

142. Pedersen, M.B. (2005). Optimization models and solution methods for intermodal transportation. *Ph.D. Thesis,* Centre for Traffic and Transport, Technical University of Denmark, Denmark.

143. Pedersen, M.B., Crainic, T.G. (2007). Optimization of Intermodal Freight Service Schedules on Train Canals. Technical Report, Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation, Canada.

144. Pedersen, M.B., Crainic, T.G., Madsen, O.B.G., (2009). Models and Tabu Search Metaheuristics for Service Network Design with Asset-Balance Requirements. *Transportation Science*, 43 (2), 158–177.

145. Pioro, M., Medhi, D. (2004). Routing, Flow, and Capacity Design in Communication and Computer Networks. Morgan Kaufmann Publishers, Elsevier.

146. Pisinger, D., Ropke, S., (2010). Large Neighborhood Search. *Handbook of Metaheuristics, International Series in Operations Research & Management Science*, 146, 399-421.

147. Poojari, C.A., Beasley, J.E. (2009). Improving Benders Decomposition Using a Genetic Algorithm. *European Journal of Operational Research*, 199 (1), 89-97.

148. Pouzin, L. (1981). Methods, Tools, and Observations on Flow Control in Packet-Switched Data Networks. *IEEE Transactions on Communications*, 29 (4), 413-426.

149. Powell, W.B. (1986a). A Local Improvement Heuristic for the Design of Less-than-Truckload Motor Carrier Networks. *Transportation Science*, 20 (4), 246–357.

150. Powell, W.B., Koskosidis, Y.A. (1992). Shipment Routing Algorithms with Tree Constraints. *Transportation Science*, 26 (3), 230–245.

151. Powell, W.B., Sheffi, Y. (1989). Design and Implementation of an Interactive Optimization System for the Network Design in the Motor Carrier Industry. *Operations Research*, 37 (1), 12–29.

152. Rei, W., Cordeau J.F., Gendreau M., Soriano P. (2008). Accelerating Benders Decomposition by Local Branching. *INFORMS Journal on Computing*, 21 (2), 333-345.

153. Rodrigez-Martin, I., Salazar Gonzalez, J.J. (2010). A Local Branching Heuristic for the Capacitated Fixed-Charge Network Design Problem. *Computers & Operations Research*, 37, 575–581.

154. Sa, E.M., Camargo, R.S., Miranda, G. (2013). An Improved Benders Decomposition Algorithm for the Tree of Hubs Location Problem. *European Journal of Operational Research*, 226, 185-202.

155. Saharidis, G.K.D., Boile, M, Theofanis, S. (2011). Initialization of Benders Master Problem Using Valid Inequalities Applied to Fixed-Charge Network Problems. *Expert Systems with Applications*, 38, 6627-6636.

156. Saharidis, G.K.D., Ierapetritou, M.G. (2010). Improving Benders Decomposition Using Maximum Feasible Subsystem (MFS) Cut Generation Strategy. *Computers and Chemical Engineering*, 34, 1237-1245.

157. Saharidis, G.K.D., Minoux, M., Ierapetritou, M.G. (2010). Accelerating Benders Method Using Covering Cut Bundle Generation. *International Transactions in Operational Research*, 17, 221-237.

158. Saharidis, G.K.D., Ierapetritou, M.G. (2013). Speed-up Benders Decomposition Using Maximum Density Cut (MDC) Generation. *Annals of Operations Research*, 210, 101-123.

159. Sahinidis, N.V., Grossman, I.E. (1991). Convergence Properties of Generalized Benders Decomposition. *Computers and Chemical Engineering*, 15 (7), 481-491.

160. Schachtebeck, M., Schöbel, A. (2008). IP-based Techniques for Delay Management with priority Decisions. *ATMOS*, 8002.

161. Schachtebeck, M., Schöbel, A. (2010). To Wait or Not to Wait – And Who Goes First? Delay Management with Priority Decisions. *Transportation Science*, 44 (3), 307-321.

162. Schöbel, A. (2001). A Model for the Delay Management Problem Based on Mixed Integer Programming. *Electronic Notes in Theoretical Computer Science*, 50 (1).

163. Schöbel, A. (2006). Optimization in Public Transportation. Springer, USA.

164. Schöbel, A. (2007). Integer Programming Approaches for Solving the Delay Management Problem. Springer, Berlin Heidelberg.

165. Schöbel, A. (2009). Capacity Constraints in Delay Management. Public Transport, 1 (2), 135-154.

166. Shaw, P. (1998). Using Constraint Programming and Local Search Methods to Solve Vehicle Routing Problems. *Fourth International Conference on Principles and Practice of Constraint Programming, Lecture Notes in Computer Science*, 1520, 417–431.

167. Sirikum, J., Techanitisawad, A., Kachitvichyanukul, V. (2007). A New Efficient GA-Benders Decomposition Method: For Power Generation Expansion Planning with Omission Controls. *IEEE Transactions on Power Systems*, 22 (3), 1092-1100.

168. Sridhar, V., Park, J.S. (2000). Benders-and-cut Algorithm for Fixed-Charge Capacitated Network Design Problem. *European Journal of Operational Research*, 125, 622-632.

169. Shintani, K., Imai, A., Nishimura, E., Papadimitriou, S. (2007). The Container Shipping Network Design Problem with Empty Container Repositioning. *Transportation Research Part E*, 43, 39-59.

170. Sung, C.S., Song, S.H. (2003). Integrated Service Network Design for a Cross-Docking Supply Chain Network. *Journal of the Operational Research Society*, 54, 1283-1295.

171. Şahin, G., Ahuja, R.K. (2009). Lower Bounding Techniques for the Degree-Constrained Network Design Problem. *Networks*, 53 (4), 334-344.

172. Tan, P.Z., Kara, B.Y. (2007). A Hub Covering Model for Cargo Delivery Systems. *Networks*, 49 (1), 28–39.

173. Tang, L., Jiang, W., Saharidis, G.K.D. (2013). An Improved Benders Decomposition Algorithm for the Logistics Facility Location with Capacity Expansions. *Annals of Operations Research*, 210, 165-190.

174. Taşkın, Z.C. (2010). Benders Decomposition. Wiley Encyclopedia of Operations Research and Management Science. John Wiley & Sons, Malden, MA.

175. Van Roy, T.J. (1986). A Cross Decomposition Algorithm for Capacitated Facility Location. *Operations Research*, 34 (1), 145-163.

176. Watkins Jr, D.W., McKinney D.C. (1998). Decomposition Methods for Water Resources Optimization Models With Fixed Costs. *Advances in Water Resources*, 21, 283-295.

177. Wentges, P. (1996). Accelerating Benders Decomposition for the Capacitated Facility Location Problem. *Mathematical Methods of Operations Research*, 44, 267-290.

178. Wieberneit, N. (2008). Service Network Design for Freight Transportation. *OR Spectrum*, 30, 77-112.

179. Wolsey, L. A. (1998). Integer Programming. John Wiley & Sons, USA.

180. Yaghini, M., Foroughi, A., Nadjari, B. (2011). Solving Railroad Blocking Problem Using Ant Colony Optimization Algorithm. *Applied Mathematical Modeling*, 35, 5579-5591.

181. Yaghini, M, Momeni, M., Sarmadi, M. (2012a). A Simplex-Based Simulated Annealing Algorithm for Node-Arc Capacitated Multicommodity Network Design. *Applied Soft Computing*, 12, 2997-3003.

182. Yaghini, M, Akhavan Kazamzadeh, M.R. (2012). A Simulated Annealing Algorithm for Unsplittable Capacitated Network Design. *International Journal of Industrial Engineering & Production Research*, 23 (2), 91-100.

183. Yaghini, M., Seyedabadi, M., Khoshraftar, M.M. (2012b). A Population Based Algorithm for the Railroad Blocking Problem. *Journal of Industrial Engineering International*, 8 (1), 1-11.

184. Yaghini, M., Akhavan, R. (2012). Multicommodity Network Design Problem in Rail Freight Transportation Planning. *Procedia-Social and Behavioral Sciences*, 43, 728-739.

185. Yang, Y., Lee, J.M. (2011). Acceleration of Benders Decomposition for Mixed Integer Linear Programming. *Proceedings of the 4th International Symposium on Advanced Control of Industrial Processes*, China*, May 2011,* 222-227.

186. Yang, Y., Lee, J.M. (2012). A Tighter Cut Generation Strategy for Acceleration of Benders Decomposition. *Computers and Chemical Engineering*, 44, 84-93.

187. Yaman, H., Kara, B.Y., Tansel, B.C. (2007). The Latest Arrival Hub Location Problem for Cargo Delivery Systems with Stopovers. *Transportation Research Part B*, 41 (8), 906-919.

188. Yaman, H., Karasan, O.E., Kara, B.Y. (2012). Release Time Scheduling and Hub Location for Next-Day Delivery. *Operations Research*, 60 (4), 906-917.

189. Yen, J.Y. (1971). Finding k-Shortest Loopless Paths in a Network. *Management Science*, 17 (11), 712-716.

190. Yue, Y., Zhou, L., Yue, Q., Fan, Z. (2011). Multi-Route Railroad Blocking Problem by Improved Model and Ant Colony Algorithm in Real World. *Computers & Industrial Engineering*, 60, 34-42.

191. Zakeri, G, Philpott, A.B., Ryan, D.M. (1998). Inexact Cuts in Benders Decomposition. *SIAM Journal of Optimization*, 10 (3), 643-657.

192. Zaleta, N.C., Alvarez Socarras, A.M. (2004). Tabu Search-Based Algorithm for Capacitated Multicommodity Network Design Problem. *Proceedings of the 14th International Conference on Electronics, Communications and Computers*, Veracruz, Mexico.

193. Zhu, E. (2010). Scheduled Service Network Design for Integrated Planning of Rail Freight Transportation. *Ph.D. Thesis,* University of Montreal, Canada.

194. Zhu, Y., Kuno, T. (2003). Global Optimization of Nonconvex MINLP by a Hybrid Branch-and-Bound and Revised General Benders Decomposition Approach. *Industrial and Engineering Chemistry Research*, 42 (3), 528-539.

APPENDIX – A

TECHNICAL NOTE ON BENDERS DECOMPOSITION

We present Benders decomposition algorithm for mixed integer programming problems having the following form:

$$\begin{aligned}
&\textit{Minimize} \quad cx + f(y) \\
&\textit{subject to} \\
&\qquad Ax + F(y) \geq b \\
&\qquad Dy \geq e \\
&\qquad x \geq 0, \quad y \geq 0 \ \textit{and integer},
\end{aligned}$$

where vectors x and y represent the continuous and integer variables, respectively, and c represents the row vector of the associated costs of x and $f(y)$ represents a scalar function. The constraints of the problem are represented by matrices A, D, vector function $F(y)$, and fixed vectors b and e. All matrices, vectors, and functions are defined with appropriate sizes.

If we fix the integer variables y as $\bar{y} \in Y = \{y \mid Dy \geq e, y \geq 0 \text{ and integer}\}$, we can express this formulation as:

$$\min_{\bar{y} \in Y} \{ f(\bar{y}) + \min_{x \geq 0} \{ cx : Ax \geq b - F(\bar{y}) \} \} \qquad (A.1).$$

The inner minimization problem in $(A.1)$ is the Benders Sub Problem (SP). Since the SP is a linear programming problem, by duality theory we can interchange the primal and the dual formulations and express $(A.1)$ as:

$$\min_{\bar{y} \in Y} \{ f(\bar{y}) + \max_{u \geq 0} \{ u^T (b - F(\bar{y})) : uA \leq c \} \} \qquad (A.2),$$

where u denotes the dual variables associated to constraints $Ax \geq b - F(\bar{y})$.

The inner maximization problem in (A.2) is the Benders Dual Sub Problem (D-SP). Since the link between the variables x and y is broken in D-SP, the feasible space $S = \{u \mid u \geq 0; uA \leq c\}$ of the D-SP becomes independent of the constraints including y variables and D-SP can have either a bounded or an unbounded optimal solution.

- If bounded, then the solution corresponds to one of the extreme points, where Q is the set of index numbers of the extreme points of S.
- If unbounded, then there is a direction λ for which $\lambda^T(b - F(\overline{y})) > 0$.

An unbounded optimal solution for the D-SP results in infeasibility of the inner minimization problem in (A.1) and has to be avoided by eliminating the corresponding $\overline{y}$ variables. For this purpose, the following set of constraints that are called as *extreme ray cuts* have to be added to (A.2):

$$\lambda_w^T(b - F(\overline{y})) \leq 0 \quad w \in W \qquad (A.3),$$

where W denotes the set of index numbers of the extreme rays of S.

Considering (A.3) for $\overline{y}$, the maximum value of the D-SP would equal to one of the extreme points u_q ($q \in Q$):

$$\min_{\overline{y} \in Y}\{f(\overline{y}) + \max\{u_q^T(b - F(\overline{y})) : q = 1, \ldots, Q\}\}$$

subject to

$$\lambda_w^T(b - F(\overline{y})) \leq 0 \qquad w \in W.$$

By associating an auxiliary continuous variable η for the maximum objective function value of the D-SP, the Benders Reformulation can be obtained:

Minimize $f(y) + \eta$

subject to

$$\lambda_w^T(b - F(y)) \leq 0 \qquad w \in W \qquad (A.3)$$

$$\eta \geq u_q^T(b - F(y)) \qquad q \in Q \qquad (A.4)$$

$$y \in Y, \quad \eta \geq 0 \qquad (A.5).$$

Extreme ray cuts defined by (A.3) are also called as the *feasibility cuts*, since they state the necessary conditions for the feasibility of the formulation (A.1). Constraints (A.4) are called as the *optimality cuts*, since they define the optimality conditions of the D-SP.

Benders decomposition algorithm begins with only considering the constraints (A.5), which form the first Master Problem. Then the optimality and feasibility cuts are generated iteratively. Since the initial Master Problem and the following ones include a subset of constraints (A.3) and (A.4), they are called as Relaxed Master Problems (RMP). The objective function of the RMPs would provide a lower (upper) bound to the main minimization (maximization) problem. Each time the RMP is solved a temporary $\bar{y}$ is obtained and the solution of the D-SP with $\bar{y}$ would provide us an upper (lower) bound to the main minimization (maximization) problem. If the D-SP is bounded an optimality cut, otherwise a feasibility cut is added to the RMP. In the Benders decomposition algorithm, the RMP and the D-SP are solved iteratively until the lower bound is equal to the upper bound or the optimality gap is within a predetermined limit, ε.

The main steps of Benders decomposition algorithm for a minimization problem are provided in Figure 63, where UB and LB state the upper and lower bounds respectively, z^*_{D-SP} and z^*_{RMP} state the optimal objective values of the D-SP and the RMP respectively. For more information about Benders decomposition, Benders (1962), Nemhauser and Wolsey (1998), Costa (2005), and Taşkın (2010) can be referenced.

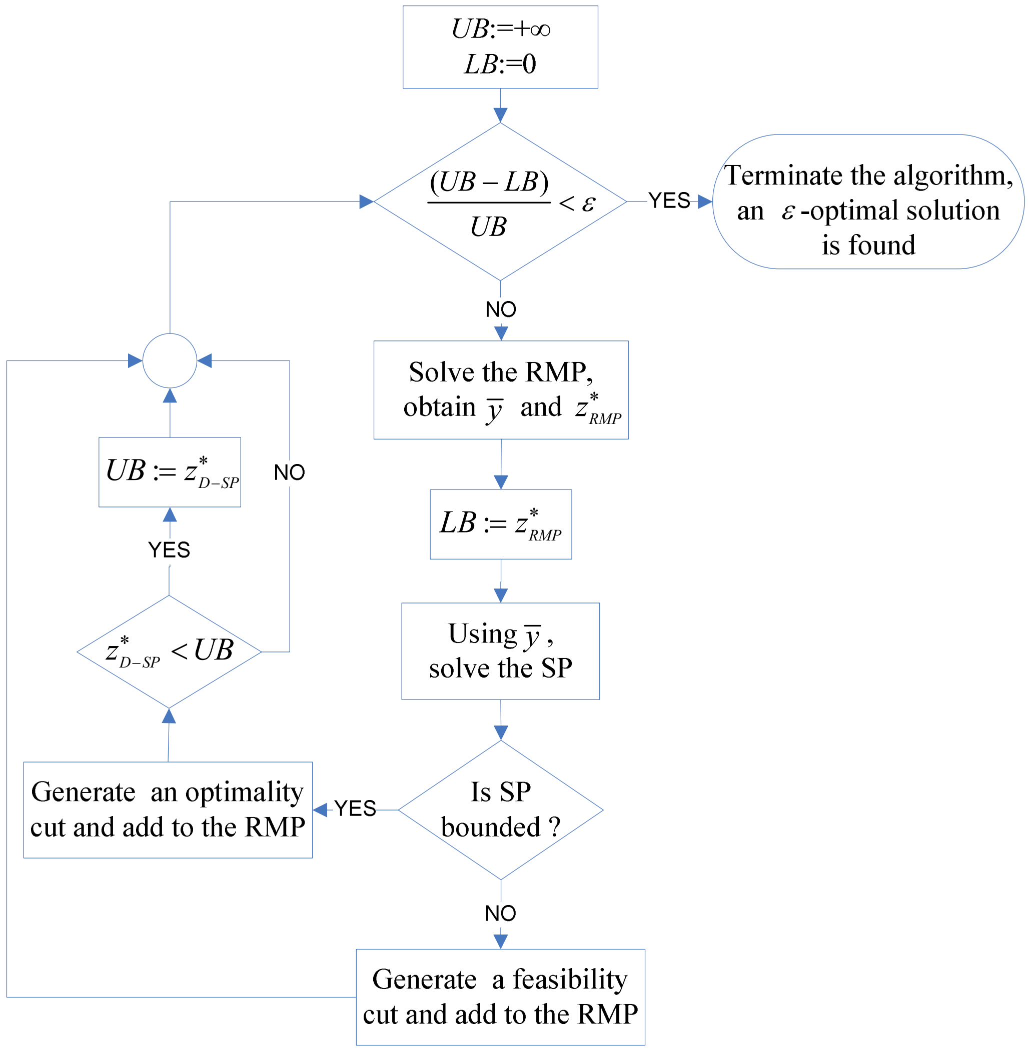


Figure 63. The Flow Chart of Benders Decomposition Algorithm

APPENDIX – B

TECHNICAL NOTE ON GENERALIZED BENDERS DECOMPOSITION

The Projection: We present Generalized Benders Decomposition (GBD) algorithm for NLP problems having the following form:

$$
\begin{aligned}
(P) \quad & \textit{Minimize} \quad f(x, y) \\
& \textit{subject to} \\
& \qquad G(x, y) \le 0 \\
& \qquad x \in X \subseteq \Re^{n_1} \\
& \qquad y \in Y \subseteq \Re^{n_2}.
\end{aligned}
$$

The partitioning of (P) in Generalized Benders decomposition is performed by projecting (P) onto the y-space:

$$
\begin{aligned}
(P') \quad & \textit{Minimize} \quad v(y) \\
& \textit{subject to} \\
& \qquad y \in Y \cap V,
\end{aligned}
$$

where

$$v(y) \equiv \underset{x}{\text{infimum}}\, f(x, y) \ \textit{subject to} \ G(x, y) \le 0, \ x \in X \qquad (A.6)$$

and

$$V \equiv \{y : G(x, y) \le 0 \ \text{ for some } \ x \in X\}.$$

The optimization problem in $(A.6)$ is:

$$
\begin{aligned}
P(y) \quad & \textit{Minimize} \quad f(x, y) \\
& \textit{subject to} \\
& \qquad G(x, y) \le 0 \\
& \qquad x \in X.
\end{aligned}
$$

In this projection scheme,

- (P') is the projection of (*P*) onto the *y*-space,
- $Y \cap V$ denotes the projection of the feasible region of (*P*) onto the *y*-space,
- $v(y)$ is the optimal value of (*P*) for fixed *y*,
- the set *V* contains the feasible *y* values of $P(y)$.

By projecting (*P*) onto the *y*-space, problems (P') and $P(y)$ are obtained, which are easier to solve than (*P*). Using nonlinear duality theory, Generalized Benders Reformulation of (*P*) is derived in a three step procedure: (i) projection, (ii) dual representation of *V*, and (iii) dual representation of $v(y)$.

(i) *Projection:*

(P') is equivalent to (*P*), thus

- (*P*) is infeasible or unbounded if and only if the same holds for (P'),
- if (x^*, y^*) is optimal in (*P*), then y^* has to be optimal in (P') and x^* is the optimizing *x* of $P(y^*)$,
- an ε_1-optimal solution $\bar{y}$ of (P') and an ε_2-optimal solution $\bar{x}$ of $P(\bar{y})$ would be $\varepsilon_1 + \varepsilon_2$-optimal solution $(\bar{x}, \bar{y})$ of (*P*).

(ii) *Dual Representation of V:*

A point $\bar{y} \in Y$ is also in the set *V* if and only if $\bar{y}$ satisfies the infinite system:

$$\underset{x \in X}{\text{infimum}}[\lambda^T G(x, y)] \le 0 \qquad \forall \lambda \in \Lambda \qquad (A.7),$$

where $\Lambda = \{\lambda \in R^m : \lambda \ge 0 \text{ and } \sum_{i=1}^{m} \lambda_i = 1\}$.

$\lambda \in \Lambda$ specifies the convex combination of the constraints that have no solution in *X*.

In order the dual representation of *V* to hold for (*P*), following assumptions have to be satisfied:

Assumption 1: *X* is a nonempty convex set,

Assumption 2: $G(x, y)$ is convex on X for each fixed $\bar{y} \in Y$,

Assumption 3: the set $Z_y = \{z \in R^m : G(x, y) \leq z \text{ for some } x \in X\}$ is closed for each fixed $\bar{y} \in Y$.

(iii) *Dual Representation of* $v(y)$*:*

Problem $P(\bar{y})$ is the subproblem (SP) of the GBD. The SP should be dual adequate in order to derive the $v(y)$ representation, then the optimal value of $P(\bar{y})$ equals to the optimal value of its dual on $Y \cap V$:

$$v(y) = \underset{u \geq 0}{\text{supremum}}[\underset{x \in X}{\text{infimum}}[f(x, y) + u^T G(x, y)]] \quad \forall\, y \in Y \cap V \qquad (A.8),$$

where u is the optimal dual multipliers of $P(\bar{y})$. In order the dual representation of $v(y)$ to hold for (P), following assumptions have to be satisfied:

Assumptions 1-2

Assumption 4: $f(x, y)$ is convex on X for each fixed $\bar{y} \in Y$,

Assumption 5: For each fixed $\bar{y} \in Y \cap V$ at least one of the following conditions must hold:

(a) $v(\bar{y})$ is finite and $P(\bar{y})$ has an optimal multiplier vector $\hat{u}$ ($\hat{u}$ is an optimal multiplier vector, if $\hat{u}$ achieves the supremum in (A.8)),

(b) $v(\bar{y})$ is finite, $G(x, \bar{y})$ and $f(x, \bar{y})$ are continuous on X, X is closed, and for an $\varepsilon \geq 0$ the ε-optimal solution set of $P(\bar{y})$ is nonempty and bounded,

(c) $v(\bar{y}) = -\infty$.

For a problem (P) that satisfies ***Assumptions 1-5***, if the three step procedure described by (i)-(iii) is applied, the following Generalized Benders reformulation is obtained:

$$\begin{aligned}&\textit{Minimize}\ [\underset{u\geq 0}{\text{supremum}}[\underset{x\in X}{\text{infimum}}\,[f(x,y)+u^T G(x,y)]]]\\&\textit{subject to}\\&\qquad (A.7)\\&\qquad y\in Y.\end{aligned}$$

Floudas (1995) present the theoretical derivation of GBD for Mixed Integer Nonlinear Programming (MINLP) problems and present basic GBD variants that could be applied to MINLP problems. For a comprehensive review of MINLP solution techniques Grossman (2002) can be referenced. In addition, for more information on theoretical aspects of GBD, Geoffrion (1972), Lazimy (1986), Floudas et al. (1989), Bagajewicz and Manousiouthakis (1991), Sahinidis and Grossman (1991), Floudas (1995), and Grossman (2002) can be referenced.

<u>Finite Convergence:</u>

Geoffrion (1972) prove the finite convergence for MINLP problems, in which *Y* is a finite discrete set and prove the finite ε-convergence when *Y* is of infinite cardinality in Theorems 2.4 and 2.5 of Geoffrion (1972), respectively.

Finite Convergence (Theorem 2.4 of Geoffrion (1972)):

GBD algorithm terminates in a finite number of steps for any given $\varepsilon > 0$ and also for $\varepsilon = 0$,

- if *Assumptions 1-4* hold,
- if *Assumption 5* hold omitting condition (b), and
- *Y* is a finite discrete set.

Finite ε-Convergence (Theorem 2.5 of Geoffrion (1972)):

GBD algorithm terminates in a finite number of steps for any given $\varepsilon > 0$,

- if *Assumptions 1-5* hold,
- *Y* is a non-empty compact subset of *V*,
- set of optimal multiplier vectors for $v(y)$ is non-empty for $\forall\ y \in Y$ and uniformly bounded in some neighborhood of each such point.

Variants of GBD:

Explicit determination of functions $L^*(y,u)$ and $L_*(y,\lambda)$ is critical for obtaining implementable GBD algorithms and increasing the computational efficiency. Depending on problem characteristics and the partitioning structure, different solution procedures are utilized for this purpose. In this section, the most prominent GBD variants are provided. A comprehensive list of techniques regarding *L* functions is given in Section 3.6. Floudas (1995) presents three GBD variants together with the underlying assumptions:

GBD-v1:

The first variant is the application of Property (P) of Geoffrion (1972) and is applicable to the NLP problems, in which *f* and *G* are linearly separable in *x* and *y*, i.e.

$$f(x,y) \equiv f_1(x) + f_2(y), \quad G(x,y) \equiv G_1(x) + G_2(y).$$

When $P(y)$ is feasible for any $u \geq 0$, it becomes possible to determine $\underset{x\in X}{\text{minimize}}(f(x,y)+u^T G(x,y))$ independently of *y* and the function $L^*(y,u)$ can be obtained explicitly as a function of *y*:

$$L^*(y,u) \equiv \min_{x\in X}(f_1(x)+u^T G_1(x)) + f_2(y) + u^T G_2(y), \quad y \in Y \qquad (A.9).$$

When $P(y)$ is infeasible, for any $\lambda \in \Lambda$, it becomes possible to determine $\underset{x\in X}{\text{minimize}}(\lambda^T G(x,y))$ independently of *y* and the function $L_*(y,\lambda)$ can be obtained explicitly as a function of *y*:

$$L_*(y,\lambda) \equiv \min_{x\in X}(\lambda^T G_1(x)) + \lambda^T G_2(y), \quad y \in Y \qquad (A.10).$$

Due to the strong duality theorem, the optimum solutions of the independent minimization problems in $(A.9)$ and $(A.10)$ are identical to the optimum solutions of feasible and infeasible $P(y)$ problems with respect to *x*, respectively, Floudas (1995).

GBD-v2:

If the projected problem $v(y)$ is convex in *y*, then the optimal primal and dual solutions of $v(\bar{y})$ namely (x^*, u^*) and (x^*, λ^*) can be used to determine the functions $L^*(y,u)$ and $L_*(y,\lambda)$, respectively:

$$L^*(y,u) = L(x^*, y, u^*)$$
$$L_*(y,\lambda) = L(x^*, y, \lambda^*).$$

Floudas (1995) state that when $v(y)$ is convex in *y*, $L(x^*, y, u^*)$ and $L(x^*, y, \lambda^*)$

- represent supporting functions for $v(y)$ at $\bar{y}$, thus
- provide valid GBD cuts.

Variant GBD-v2 can be applied to the problems, in which *f* and *G* functions are not linearly separable in *x* and *y*. Practically, when Property (P) does not hold the determination of functions $L^*(y,u)$ and $L_*(y,\lambda)$ are performed by fixing the (x,u) and (x,λ) to the optimal solutions of the sub problems, Floudas et al. (1989), Sahinidis and Grossman (1991). If $v(y)$ is nonconvex, then the GBD cuts obtained by GBD-v2 does not guarantee validity for the original problem and the GBD algorithm may terminate at a local optimum point, Floudas (1995), Cai et al. (2001).

GBD-3:

This variant is called as Global Optimum Search (GOS) and is proposed by Floudas et al. (1989) for nonconvex NLP and MINLP problems. GBD-v3 uses the same assumptions and algorithm steps of GBD-v2. But GBD-v3 assumes that $f(x,y)$ and $G(x,y)$ are convex in *Y* for each fixed $x \in X$. This additional assumption is required to obtain a special structure in the RMP. In GOS, the *x* and *y* variables are selected so as to satisfy the convexity assumptions and by this way global solutions are ensured to be obtained for the RMP and $P(y)$. For problems with nonconvex $v(y)$, if *f* and *G* are not linearly separable in *x* and *y*, then GBD-v3 provides approximate GBD cuts rather than valid ones.

APPENDIX – C

THE CITIES AND THE REGIONS OF TURKEY

Table 28. The City and License Tag Information of Turkey Regions

Regions	License Tag	City
Aegean Region	3	Afyon
	9	Aydın
	20	Denizli
	35	İzmir
	43	Kütahya
	45	Manisa
	48	Muğla
	64	Uşak
Central Anatolia	6	Ankara
	18	Çankırı
	26	Eskişehir
	38	Kayseri
	40	Kırşehir
	42	Konya
	50	Nevşehir
	51	Niğde
	58	Sivas
	66	Yozgat
	68	Aksaray
	70	Karaman
	71	Kırıkkale
Mediterranean Region	1	Adana
	7	Antalya
	15	Burdur
	31	Hatay
	32	Isparta
	33	İçel
	46	Kahramanmaraş
	80	Osmaniye
Middle Black Sea	5	Amasya
	19	Çorum
	52	Ordu
	55	Samsun
	60	Tokat
Western Black Sea	14	Bolu
	37	Kastamonu
	57	Sinop
	67	Zonguldak
	74	Bartin
	78	Karabük
	81	Düzce

Regions	License Tag	City
Marmara Region	10	Balıkesir
	11	Bilecik
	16	Bursa
	17	Çanakkale
	22	Edirne
	34	İstanbul
	39	Kırklareli
	41	Kocaeli
	54	Sakarya
	59	Tekirdağ
	77	Yalova
South Eastern Anatolia	2	Adıyaman
	21	Diyarbakır
	27	Gaziantep
	47	Mardin
	56	Siirt
	63	Şanlıurfa
	72	Batman
	73	Şırnak
	79	Kilis
Eastern Anatolia	4	Ağrı
	12	Bingöl
	13	Bitlis
	23	Elazığ
	24	Erzincan
	25	Erzurum
	30	Hakkari
	36	Kars
	44	Malatya
	49	Muş
	62	Tunceli
	65	Van
	75	Ardahan
	76	Iğdır
Eastern Black Sea	8	Artvin
	28	Giresun
	29	Gümüşhane
	53	Rize
	61	Trabzon
	69	Bayburt

Figure 64. Map of Turkey

APPENDIX – D

THE TEST NETWORKS

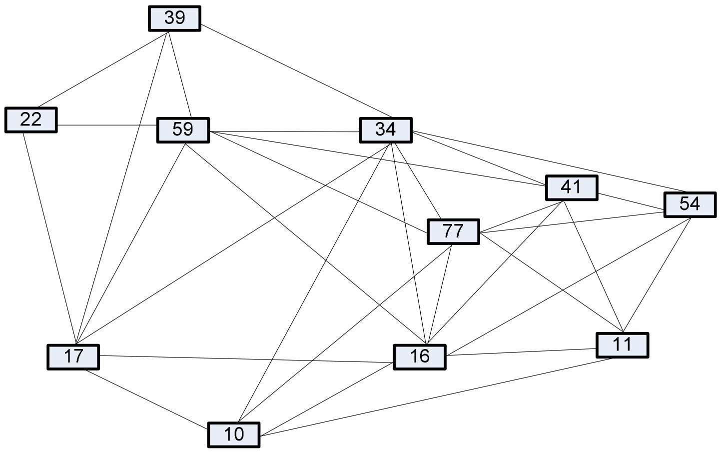


Figure 65. T1-1 Marmara Region Test Network – 11 Cities

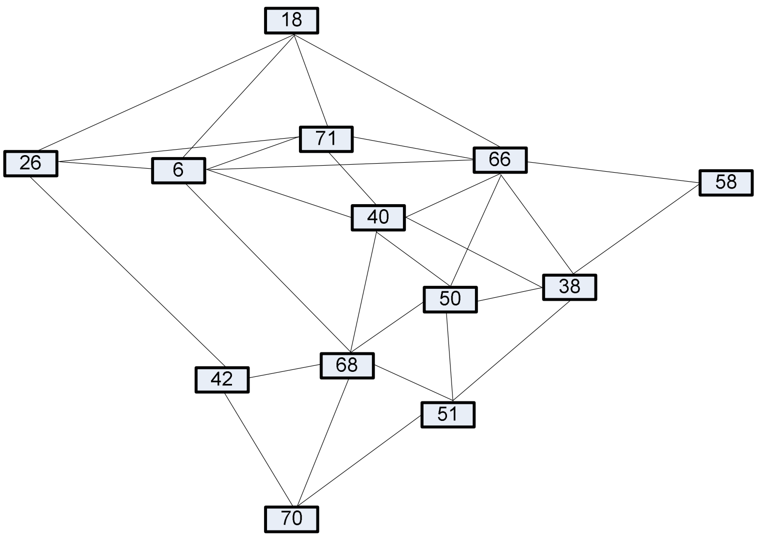


Figure 66. T1-2 Central Anatolia Region Test Network – 13 Cities

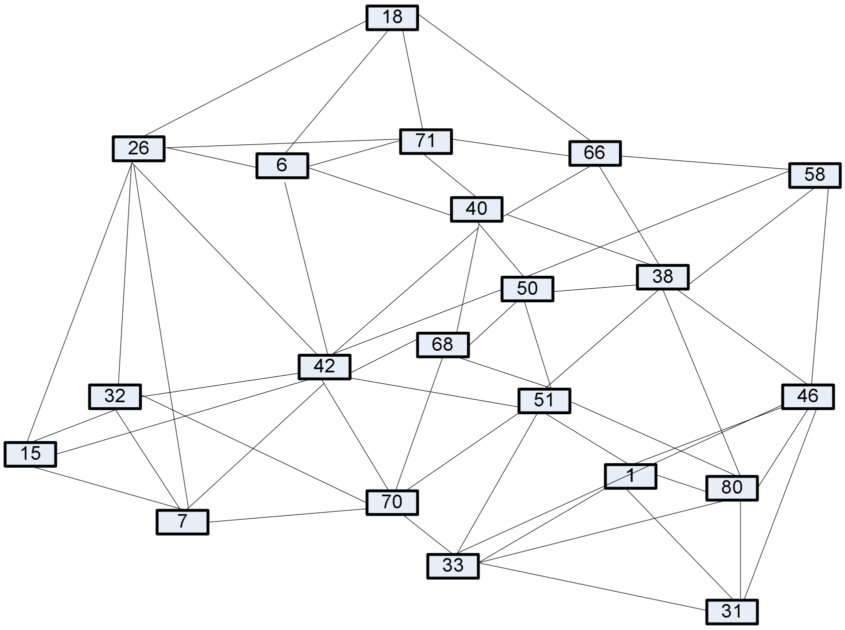


Figure 67. T2-1 Central Anatolia and Mediterranean Regions Test Network – 21 Cities

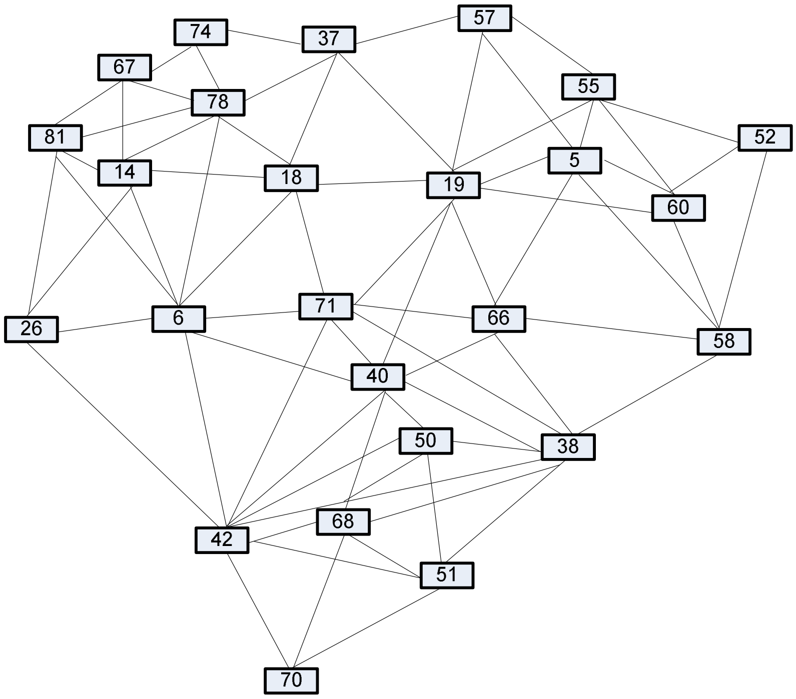


Figure 68. T2-2 Central Anatolia & Western and Middle Black Sea Regions Test Network – 25 Cities

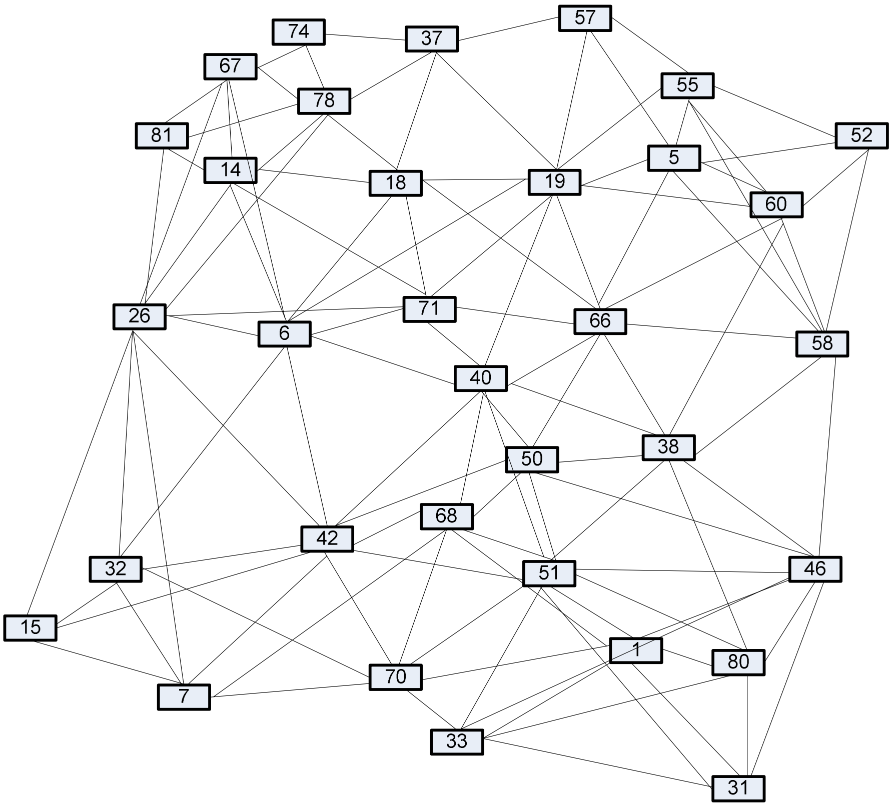


Figure 69. T2-3 Central Anatolia & Western and Black Sea & Mediterranean Regions Test Network – 33 Cities

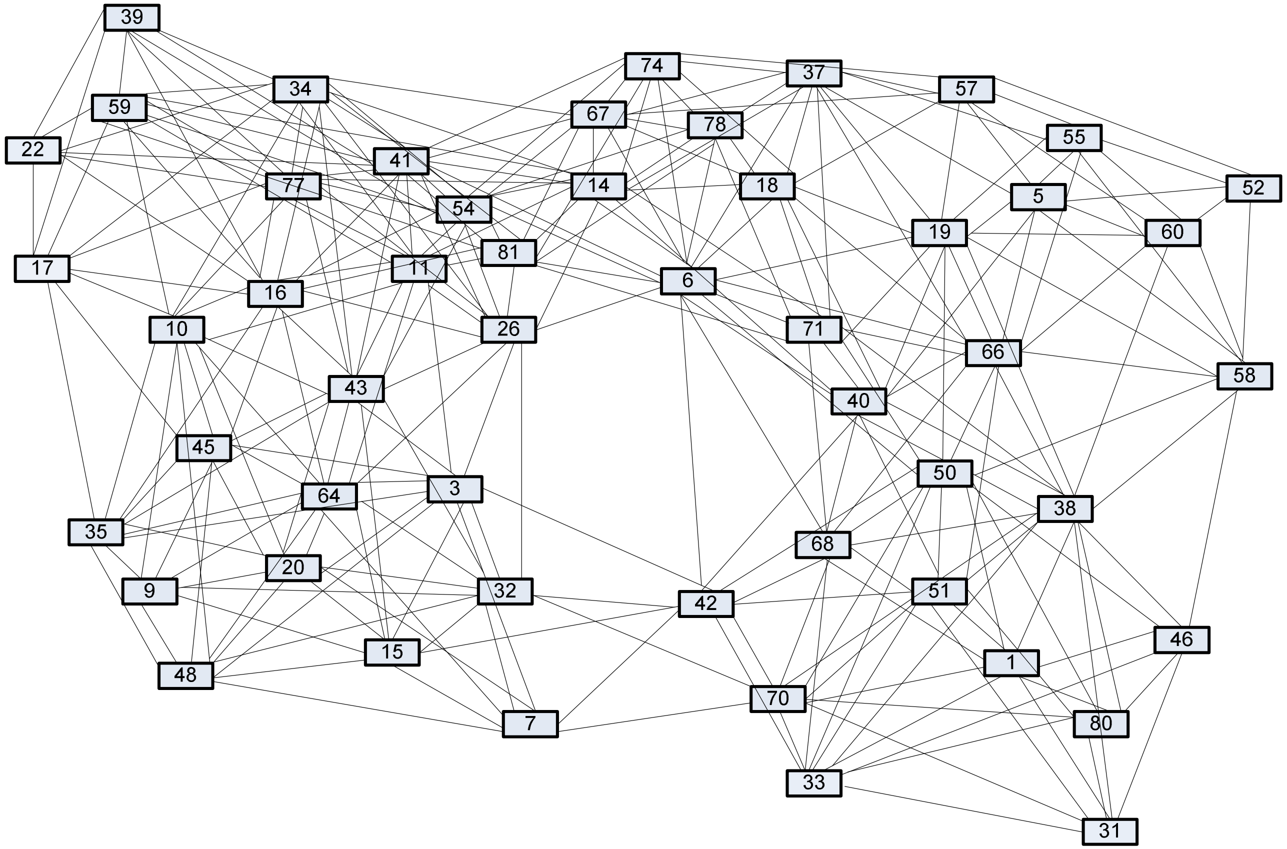


Figure 70. T3-1 Central Anatolia & Western & Middle Black Sea & Mediterranean & Aegean & Marmara Regions Test Network – 52 Cities

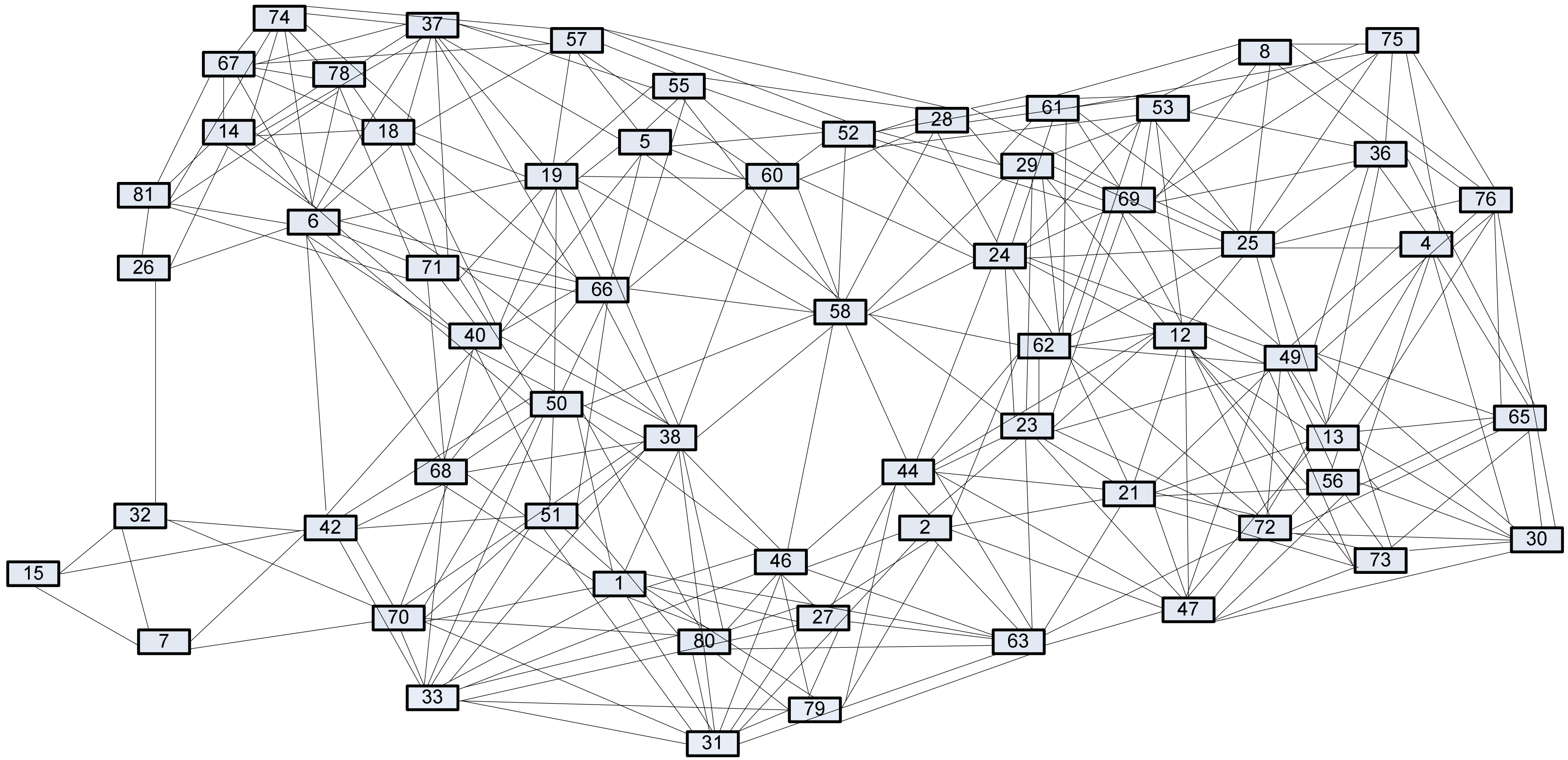


Figure 71. T3-2 Central Anatolia & Black Sea & Mediterranean & Eastern and South Eastern Anatolia Regions Test Network – 62 Cities

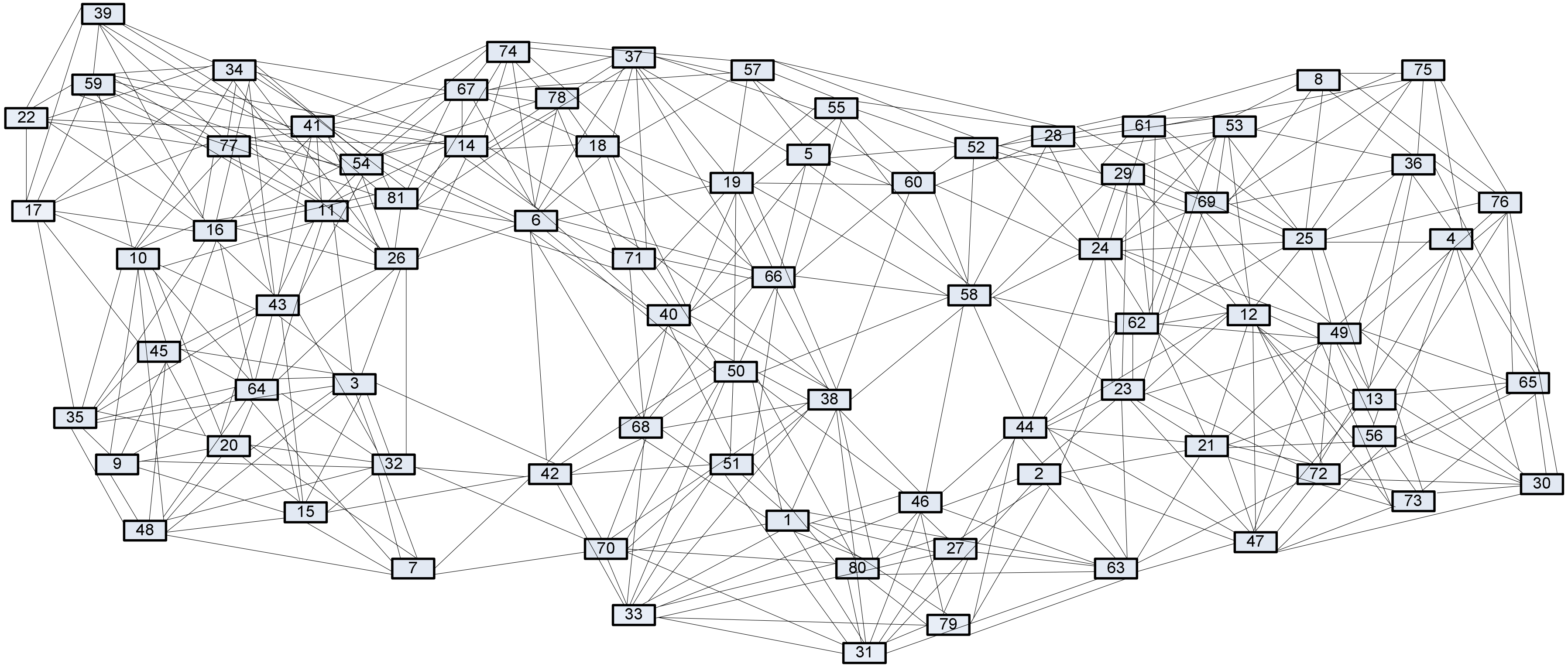


Figure 72. T3-3 Turkey Test Network – 81 Cities

APPENDIX – E

COMPUTATIONAL RESULTS OF GBD-1-1 AND GBD-2-1 ON T1 INSTANCES

Table 29. Computational Results of GBD-1-1 and GBD-2-1 on T1 Instances – Random 1

Test Problem	|*N*|	*p*	|*K*|	CPLEX *LA*	CPLEX CPU Time (s)	GBD-1-1 *LA*	GBD-1-1 # of Cuts	GBD-1-1 CPU Time (s)	GBD-2-1 *LA*	GBD-2-1 # of Cuts	GBD-2-1 CPU Time (s)
T2-1	11	2	22	566	0.6	566	2	0.5	566	2	0.4
			44	701	0.6	701	3	1.0	701	4	0.9
			66	701	1.5	701	5	2.2	701	4	1.2
			88	No Solution							
			Average		**0.9**		**3**	**1.2**		**3**	**0.8**
		3	22	566	0.5	566	2	0.4	566	2	0.3
			44	701	0.5	701	3	1.0	701	4	0.9
			66	701	1.5	701	4	1.8	701	6	1.7
			88	701	1.2	701	7	3.7	701	7	2.5
			Average		**0.9**		**4**	**1.7**		**5**	**1.3**
		4	22	566	0.5	566	3	0.4	566	2	0.3
			44	701	0.5	701	2	0.7	701	2	0.5
			66	701	1.3	701	3	1.4	701	3	0.9
			88	701	1.9	701	4	1.9	701	4	1.4
			Average		**1.1**		**3**	**1.1**		**3**	**0.8**
T2-2	13	2	31	800	0.7	800	3	0.8	800	3	0.6
			62	906	1.4	906	3	1.5	906	3	1.0
			94	No Solution							
			125	No Solution							
			Average		**1.1**		**3**	**1.2**		**3.0**	**0.8**
		3	31	800	0.7	800	3	0.7	800	3	0.5
			62	906	1.4	906	3	1.4	906	3	1.0
			94	906	2.0	906	3	2.0	906	3	1.4
			125	906	2.1	906	9	5.9	906	7	3.2
			Average		**1.6**		**5**	**2.5**		**4**	**1.5**
		4	31	800	0.6	800	4	0.9	800	4	0.7
			62	906	1.3	906	1	0.6	906	1	0.5
			94	906	2.1	906	2	1.3	906	2	0.9
			125	906	1.7	906	6	4.0	906	6	2.7
			Average		**1.4**		**3**	**1.7**		**3**	**1.2**

Table 30. Computational Results of GBD-1-1 and GBD-2-1 on T1 Instances – Random 2

Test Problem	|*N*|	*p*	|*K*|	CPLEX *LA*	CPLEX CPU Time (s)	GBD-1-1 *LA*	GBD-1-1 # of Cuts	GBD-1-1 CPU Time (s)	GBD-2-1 *LA*	GBD-2-1 # of Cuts	GBD-2-1 CPU Time (s)
T2-1	11	2	22	559	0.6	559	5	1.0	559	5	0,7
			44	575	0.9	575	7	1.8	575	18	3,1
			66	702	1.7	702	5	2.2	702	5	1.6
			88	702	2.2	702	5	2.7	702	4	1.6
			Average		**1.4**		**6**	**1.9**		**8**	**1.7**
		3	22	559	0.6	559	5	0.9	559	5	0,6
			44	575	0.9	575	7	1.8	575	18	3.1
			66	702	1.8	702	5	2.2	702	5	1.6
			88	702	2.7	702	5	2.7	702	5	1.8
			Average		**1.5**		**6**	**1.9**		**8**	**1.8**
		4	22	559	0.6	559	6	1.0	559	6	0,7
			44	575	1.0	575	11	2.6	575	10	1.7
			66	702	1.6	702	5	2.0	702	5	1.4
			88	702	1.9	702	8	3.8	702	8	2.5
			Average		**1.3**		**8**	**2.3**		**7**	**1.6**
T2-2	13	2	31	840	0.5	840	4	0.7	840	4	0.5
			62	885	0.7	885	5	1.7	885	5	1.3
			94	906	2.1	906	9	5.0	906	11	4.0
			125	No Solution							
			Average		**1.1**		**6**	**2.5**		**7**	**1.9**
		3	31	840	0.6	840	4	0.7	840	4	0.5
			62	885	0.7	885	5	1.8	885	5	1.3
			94	906	2.1	906	10	5.5	906	8	3.0
			125	No Solution							
			Average		**1.1**		**6**	**2.7**		**6**	**1.6**
		4	31	840	0.6	840	1	0.3	840	1	0.2
			62	885	0.7	885	1	0.7	885	1	0.5
			94	906	1.9	906	7	3.5	906	6	1.9
			125	906	2.0	906	8	5.3	906	10	4.2
			Average		**1.3**		**4**	**2.5**		**5**	**1.7**

Table 31. Computational Results of GBD-1-1 and GBD-2-1 on T1 Instances – Random 3

Test Problem	|*N*|	*p*	|*K*|	CPLEX *LA*	CPLEX CPU Time (s)	GBD-1-1 *LA*	GBD-1-1 # of Cuts	GBD-1-1 CPU Time (s)	GBD-2-1 *LA*	GBD-2-1 # of Cuts	GBD-2-1 CPU Time (s)
T2-1	11	2	22	559	0.3	559	1	0.3	559	1	0.2
			44	701	0.7	701	3	0.9	701	2	0.4
			66	701	0.9	701	6	2.5	701	9	2.5
			88	835	1.0	835	3	1.8	835	3	1.2
			Average		**0.7**		**3**	**1.3**		**4**	**1.1**
		3	22	559	0.3	559	1	0.3	559	1	0.2
			44	701	0.7	701	3	0.9	701	2	0.4
			66	701	1.0	701	6	2.5	701	9	2.5
			88	824	1.9	824	7	3.9	824	9	3.2
			Average		**1.0**		**4**	**1.9**		**5**	**1.6**
		4	22	559	0.3	559	1	0.3	559	1	0.2
			44	701	0.5	701	2	0.6	701	2	0.5
			66	701	1.0	701	5	2.0	701	5	1.3
			88	809	1.1	809	6	2.9	809	7	2.1
			Average		**0.7**		**4**	**1.5**		**4**	**1.0**
T2-2	13	2	31	885	0.7	885	4	0.9	885	4	0.6
			62	885	1.9	885	15	6.3	885	11	3.3
			94	906	2.0	906	9	5.2	906	11	4.5
			125	1.293	2.3	1.293	5	3.9	1.293	6	3.3
			Average		**1.7**		**8**	**4.0**		**8**	**2.9**
		3	31	885	0.6	885	4	0.9	885	4	0.6
			62	885	1.5	885	16	6.6	885	11	3.3
			94	906	2.9	906	9	5.2	906	11	4.4
			125	906	3.6	906	14	10.3	906	16	7.9
			Average		**2.2**		**11**	**5.7**		**11**	**4.0**
		4	31	885	0.8	885	3	0.6	885	3	0.5
			62	885	1.8	885	16	5.6	885	17	3.9
			94	906	2.4	906	14	7.1	906	14	4.3
			125	906	2.8	906	12	7.8	906	12	5.0
			Average		**2.0**		**11**	**5.3**		**12**	**3.4**

Table 32. Computational Results of GBD-1-1 and GBD-2-1 on T2 Instances – Random 4

Test Problem	|*N*|	*p*	|*K*|	CPLEX *LA*	CPLEX CPU Time (s)	GBD-1-1 *LA*	GBD-1-1 # of Cuts	GBD-1-1 CPU Time (s)	GBD-2-1 *LA*	GBD-2-1 # of Cuts	GBD-2-1 CPU Time (s)
T2-1	11	2	22	566	0.6	566	5	1.0	566	6	0.7
			44	566	1.1	566	6	1.7	566	7	1.4
			66	575	1.6	575	12	4.7	575	12	3.1
			88	No Solution							
			Average		**1.1**		**8**	**2.5**		**8**	**1.7**
		3	22	566	0.6	566	6	1.0	566	6	0.7
			44	566	1.1	566	6	1.8	566	7	1.5
			66	575	1.6	575	12	4.8	575	12	3.1
			88	809	2.1	809	7	3.5	809	10	3.2
			Average		**1.3**		**8**	**2.7**		**9**	**2.1**
		4	22	566	0.5	566	4	0.9	566	4	0.5
			44	566	0.9	566	5	1.3	566	5	0.9
			66	575	1.5	575	5	1.9	575	5	1.3
			88	809	1.8	809	4	2.1	809	4	1.3
			Average		**1.2**		**5**	**1.5**		**5**	**1.0**
T2-2	13	2	31	840	0.7	840	5	1.1	840	5	0.8
			62	840	1.3	840	7	2.9	840	8	2.2
			94	885	2.0	885	6	3.5	885	6	2.4
			125	No Solution							
			Average		**1.3**		**6**	**2.5**		**6**	**1.8**
		3	31	840	0.6	840	5	1.1	840	5	0.8
			62	840	1.1	840	8	3.1	840	8	2.2
			94	870	2.2	870	9	5.2	870	9	3.5
			125	No Solution							
			Average		**1.3**		**7**	**3.1**		**7**	**2.2**
		4	31	840	0.6	840	2	0.5	840	2	0.4
			62	840	1.7	840	3	1.2	840	3	0.8
			94	870	1.6	870	5	2.8	870	5	1.8
			125	906	3.6	906	9	5.7	906	13	5.6
			Average		**1.9**		**5**	**2.5**		**6**	**2.1**

Table 33. Computational Results of GBD-1-1 and GBD-2-1 on T1 Instances – Random 5

Test Problem	|*N*|	*p*	|*K*|	CPLEX *LA*	CPLEX CPU Time (s)	GBD-1-1 *LA*	GBD-1-1 # of Cuts	GBD-1-1 CPU Time (s)	GBD-2-1 *LA*	GBD-2-1 # of Cuts	GBD-2-1 CPU Time (s)
T2-1	11	2	22	701	0.3	701	1	0.3	701	1	0.2
			44	701	0.6	701	2	0.7	701	2	0.5
			66	809	1.3	809	2	1.0	809	2	0.7
			88	809	1.9	809	5	2.6	809	5	1.6
			Average		**1.0**		**3**	**1.1**		**3**	**0.8**
		3	22	701	0.3	701	1	0.3	701	1	0.2
			44	701	0.6	701	2	0.7	701	2	0.6
			66	809	1.1	809	4	1.7	809	7	1.8
			88	809	1.8	809	3	1.8	809	3	1.1
			Average		**0.9**		**3**	**1.1**		**3**	**0.9**
		4	22	701	0.3	701	2	0.4	701	2	0.3
			44	701	0.8	701	3	0.8	701	3	0.6
			66	809	1.2	809	4	1.7	809	4	1.1
			88	809	1.9	809	4	2.1	809	4	1.3
			Average		**1.0**		**3**	**1.3**		**3**	**0.8**
T2-2	13	2	31	870	0.7	870	3	0.7	870	3	0.5
			62	888	1.4	888	3	1.2	888	4	1.0
			94	888	2.3	888	7	4.0	888	7	2.5
			125	No Solution							
			Average		**1.5**		**4**	**1.9**		**5**	**1.3**
		3	31	870	0.7	870	3	0.7	870	3	0.5
			62	888	1.2	888	3	1.2	888	4	1.0
			94	888	1.9	888	10	5.6	888	6	2.3
			125	888	3.1	888	16	11.4	888	13	6.4
			Average		**1.7**		**8**	**4.7**		**7**	**2.5**
		4	31	870	0.6	870	6	1.0	870	5	0.7
			62	888	1.1	888	5	1.6	888	5	1.1
			94	888	2.1	888	10	4.9	888	7	2.1
			125	888	3.6	888	11	6.9	888	12	5.1
			Average		**1.9**		**8**	**3.6**		**7**	**2.3**

258

APPENDIX – F

COMPUTATIONAL RESULTS OF GBD VARIANTS WITH VALID INEQUALITIES ON T1 INSTANCES

Table 34. Computational Results of GBD-1 Variants on T1 Instances – Random 1

Test Problem	|*N*|	|*K*|	GBD-1-1 # of Cuts	GBD-1-1 CPU Time (s)	GBD-1-2 # of Cuts	GBD-1-2 CPU Time (s)	GBD-1-3 # of Cuts	GBD-1-3 CPU Time (s)	GBD-1-4 # of Cuts	GBD-1-4 CPU Time (s)
T1-1	11	22	3	0.4	2	0.4	2	0.4	4	0.6
		44	2	0.7	2	0.7	2	0.7	2	0.7
		66	3	1.3	3	1.4	3	1.4	3	1.4
		88	4	2.1	4	1.9	5	2.8	4	2.3
		Average	**3**	**1.1**	**3**	**1.1**	**3**	**1.3**	**3**	**1.2**
T1-2	13	31	4	0.9	4	0.9	2	0.6	4	1.0
		62	1	0.6	1	0.7	1	0.7	1	0.7
		94	2	1.3	1	1.1	3	2.0	2	1.5
		125	6	4.0	7	5.0	6	4.3	5	3.8
		Average	**3**	**1.7**	**3**	**1.9**	**3**	**1.9**	**3**	**1.7**

Table 35. Computational Results of GBD-1 Variants on T1 Instances – Random 2

Test Problem	|*N*|	|*K*|	GBD-1-1 # of Cuts	GBD-1-1 CPU Time (s)	GBD-1-2 # of Cuts	GBD-1-2 CPU Time (s)	GBD-1-3 # of Cuts	GBD-1-3 CPU Time (s)	GBD-1-4 # of Cuts	GBD-1-4 CPU Time (s)
T1-1	11	22	6	1.0	6	0.9	3	0.4	5	0.7
		44	11	2.6	7	1.8	2	0.7	6	1.6
		66	5	2.0	5	2.1	3	1.4	6	2.3
		88	8	3.8	8	4.2	5	2.9	5	2.7
		Average	**8**	**2.3**	**7**	**2.2**	**3**	**1.3**	**6**	**1.8**
T1-2	13	31	1	0.3	1	0.3	1	0.3	2	0.5
		62	1	0.7	1	0.6	1	0.7	2	0.9
		94	7	3.5	9	5.0	6	3.4	5	2.9
		125	8	5.3	10	6.7	8	5.6	8	5.9
		Average	**4**	**2.5**	**5**	**3.2**	**4**	**2.5**	**4**	**2.6**

Table 36. Computational Results of GBD-1 Variants on T1 Instances – Random 3

Test Problem	|*N*|	|*K*|	GBD-1-1 # of Cuts	GBD-1-1 CPU Time (s)	GBD-1-2 # of Cuts	GBD-1-2 CPU Time (s)	GBD-1-3 # of Cuts	GBD-1-3 CPU Time (s)	GBD-1-4 # of Cuts	GBD-1-4 CPU Time (s)
T1-1	11	22	1	0.3	1	0.3	1	0.3	1	0.3
		44	2	0.6	2	0.7	2	0.7	2	0.6
		66	5	2.0	3	1.3	3	1.3	4	1.7
		88	6	2.9	4	2.4	4	2.4	6	3.2
		Average	**4**	**1.5**	**3**	**1.2**	**3**	**1.2**	**3**	**1.4**
T1-2	13	31	3	0.6	2	0.5	3	0.7	2	0.5
		62	16	5.6	13	4.6	13	4.8	15	5.6
		94	14	7.1	14	7.9	14	7.9	6	3.5
		125	12	7.8	13	8.9	14	9.8	12	8.3
		Average	**11**	**5.3**	**11**	**5.5**	**11**	**5.8**	**9**	**4.5**

Table 37. Computational Results of GBD-1 Variants on T1 Instances – Random 4

Test Problem	|*N*|	|*K*|	GBD-1-1 # of Cuts	GBD-1-1 CPU Time (s)	GBD-1-2 # of Cuts	GBD-1-2 CPU Time (s)	GBD-1-3 # of Cuts	GBD-1-3 CPU Time (s)	GBD-1-4 # of Cuts	GBD-1-4 CPU Time (s)
T1-1	11	22	4	0.9	5	0.9	4	0.6	7	1.0
		44	5	1.3	5	1.4	4	1.2	5	1.3
		66	5	1.9	8	3.0	5	2.1	8	3.0
		88	4	2.1	4	2.3	4	2.4	5	2.7
		Average	**5**	**1.5**	**6**	**1.9**	**4**	**1.6**	**6**	**2.0**
T1-2	13	31	2	0.5	3	0.7	3	0.7	2	0.5
		62	3	1.2	7	2.5	6	2.2	9	3.1
		94	5	2.8	4	2.5	4	2.5	9	5.0
		125	9	5.7	10	7.2	10	7.3	9	6.5
		Average	**5**	**2.5**	**6**	**3.2**	**6**	**3.2**	**7**	**3.8**

Table 38. Computational Results of GBD-1 Variants on T1 Instances – Random 5

Test Problem	|*N*|	|*K*|	GBD-1-1 # of Cuts	GBD-1-1 CPU Time (s)	GBD-1-2 # of Cuts	GBD-1-2 CPU Time (s)	GBD-1-3 # of Cuts	GBD-1-3 CPU Time (s)	GBD-1-4 # of Cuts	GBD-1-4 CPU Time (s)
T1-1	11	22	2	0.4	1	0.3	1	0.3	1	0.3
		44	3	0.8	1	0.5	1	0.5	3	1.0
		66	4	1.7	3	1.4	3	1.3	3	1.3
		88	4	2.1	5	2.9	3	1.9	5	2.9
		Average	**3**	**1.3**	**3**	**1.2**	**2**	**1.0**	**3**	**1.4**
T1-2	13	31	6	1.0	6	1.1	6	1.0	6	1.0
		62	5	1.6	3	1.2	3	1.3	4	1.6
		94	10	4.9	6	3.5	9	4.7	5	2.9
		125	11	6.9	13	8.9	10	6.9	13	9.2
		Average	**8**	**3,6**	**7**	**3.7**	**7**	**3.5**	**7**	**3.7**

Table 39. Computational Results of GBD-2 Variants on T1 Instances – Random 1

Test Problem	|*N*|	|*K*|	GBD-2-1 # of Cuts	GBD-2-1 CPU Time (s)	GBD-2-2 # of Cuts	GBD-2-2 CPU Time (s)	GBD-2-3 # of Cuts	GBD-2-3 CPU Time (s)	GBD-2-4 # of Cuts	GBD-2-4 CPU Time (s)
T1-1	11	22	2	0.3	2	0.4	2	0.4	3	0.5
		44	2	0.5	2	0.5	2	0.5	2	0.6
		66	3	0.9	4	1.0	3	1.0	3	0.9
		88	4	1.4	3	1.3	5	1.9	4	1.6
		Average	**3**	**0.8**	**3**	**0.8**	**3**	**0.9**	**3**	**0.9**
T1-2	13	31	4	0.7	4	0.7	2	0.4	4	0.8
		62	1	0.5	1	0.5	1	0.5	1	0.5
		94	2	0.9	1	0.7	3	1.3	2	1.1
		125	6	2.7	7	3.4	6	3.1	5	2.7
		Average	**3**	**1.2**	**3**	**1.3**	**3**	**1.3**	**3**	**1.3**

Table 40. Computational Results of GBD-2 Variants on T1 Instances – Random 2

Test Problem	|*N*|	|*K*|	GBD-2-1 # of Cuts	GBD-2-1 CPU Time (s)	GBD-2-2 # of Cuts	GBD-2-2 CPU Time (s)	GBD-2-3 # of Cuts	GBD-2-3 CPU Time (s)	GBD-2-4 # of Cuts	GBD-2-4 CPU Time (s)
T1-1	11	22	6	0.7	6	0.7	5	0.6	5	0.6
		44	10	1.7	10	1.8	12	2.0	8	1.5
		66	5	1.4	5	1.5	7	1.9	6	1.7
		88	8	2.5	8	2.7	6	2.1	5	1.9
		Average	**7**	**1.6**	**7**	**1.7**	**8**	**1.7**	**6**	**1.4**
T1-2	13	31	1	0.2	1	0.3	1	0.3	2	0.4
		62	1	0.5	1	0.5	1	0.4	2	0.7
		94	6	1.9	7	2.6	5	2.0	5	2.1
		125	10	4.2	9	4.3	9	4.3	8	4.1
		Average	**5**	**1.7**	**5**	**1.9**	**4**	**1.7**	**4**	**1.8**

Table 41. Computational Results of GBD-2 Variants on T1 Instances – Random 3

Test Problem	|*N*|	|*K*|	GBD-2-1 # of Cuts	GBD-2-1 CPU Time (s)	GBD-2-2 # of Cuts	GBD-2-2 CPU Time (s)	GBD-2-3 # of Cuts	GBD-2-3 CPU Time (s)	GBD-2-4 # of Cuts	GBD-2-4 CPU Time (s)
T1-1	11	22	1	0.2	1	0.2	1	0.2	1	0.3
		44	2	0.5	2	0.5	2	0.5	2	0.5
		66	5	1.3	5	1.4	3	0.9	4	1.1
		88	7	2.1	4	1.5	4	1.5	6	2.2
		Average	**4**	**1.0**	**3**	**0.9**	**3**	**0.8**	**3**	**1.0**
T1-2	13	31	3	0.5	2	0.4	3	0.6	2	0.4
		62	17	3.9	12	3.1	15	3.8	13	3.4
		94	14	4.3	11	4.1	8	3.0	8	3.2
		125	12	5.0	10	4.8	12	5.6	12	5.9
		Average	**12**	**3.4**	**9**	**3.1**	**10**	**3.2**	**9**	**3.2**

Table 42. Computational Results of GBD-2 Variants on T1 Instances – Random 4

Test Problem	|*N*|	|*K*|	GBD-2-1		GBD-2-2		GBD-2-3		GBD-2-4	
			# of Cuts	CPU Time (s)	# of Cuts	CPU Time (s)	# of Cuts	CPU Time (s)	# of Cuts	CPU Time (s)
T1-1	11	22	4	0.5	5	0.6	4	0.5	6	0.7
		44	5	0.9	5	0.9	4	0.8	5	1.0
		66	5	1.3	8	2.0	5	1.4	8	2.1
		88	4	1.3	4	1.6	4	1.5	5	1.9
		Average	**5**	**1.0**	**6**	**1.3**	**4**	**1.0**	**6**	**1.4**
T1-2	13	31	2	0.4	4	0.6	3	0.5	3	0.5
		62	3	0.8	9	2.2	6	1.5	7	1.9
		94	5	1.8	4	1.7	4	1.7	13	5.0
		125	13	5.6	13	6.5	18	8.6	10	4.9
		Average	**6**	**2.1**	**8**	**2.8**	**8**	**3.1**	**8**	**3.1**

Table 43. Computational Results of GBD-2 Variants on T1 Instances – Random 5

Test Problem	|*N*|	|*K*|	GBD-2-1		GBD-2-2		GBD-2-3		GBD-2-4	
			# of Cuts	CPU Time (s)	# of Cuts	CPU Time (s)	# of Cuts	CPU Time (s)	# of Cuts	CPU Time (s)
T1-1	11	22	2	0.3	1	0.2	1	0.2	1	0.3
		44	3	0.6	1	0.4	1	0.4	2	0.6
		66	4	1.1	3	0.9	3	0.9	3	1.0
		88	4	1.3	5	1.8	3	1.1	4	1.6
		Average	**3**	**0.8**	**3**	**0.8**	**2**	**0.6**	**3**	**0.8**
T1-2	13	31	5	0.7	6	0.7	6	0.7	6	0.8
		62	5	1.1	3	0.9	4	1.0	4	1.1
		94	7	2.1	7	2.6	9	3.2	5	2.1
		125	12	5.1	13	5.9	10	4.6	13	6.2
		Average	**7**	**2.3**	**7**	**2.6**	**7**	**2.4**	**7**	**2.6**

APPENDIX – G

COMPUTATIONAL RESULTS OF GBD ALGORITHMS ON T2 INSTANCES

The tables of Appendix G are presented starting on the next page.

Table 44. Computational Results of GBD-1-1 and GBD-1-4 on T2 Instances – Random 1

Test Problem	|*N*|	|*K*|	CPLEX *LA*	CPLEX (UB-LB)/UB	CPLEX CPU Time (s)	GBD-1-1 *LA*	GBD-1-1 # of Cuts	GBD-1-1 % Dev. from CPLEX	GBD-1-1 (UB-LB)/UB	GBD-1-1 CPU Time (s)	GBD-1-4 *LA*	GBD-1-4 # of Cuts	GBD-1-4 % Dev. from CPLEX	GBD-1-4 (UB-LB)/UB	GBD-1-4 CPU Time (s)
T2-1	21	25	1,028	0.0%	3.0	1,028	9	0.0%	0.0%	2.9	1,028	15	0.0%	0.0%	5.7
		50	1,334	0.0%	13.5	1,334	4	0.0%	0.0%	2.5	1,334	3	0.0%	0.0%	2.4
		75	1,382	0.0%	31.6	1,382	17	0.0%	0.0%	14.5	1,382	16	0.0%	0.0%	15.8
		100	1,522	0.0%	33.1	1,522	28	0.0%	0.0%	31.0	1,522	40	0.0%	0.0%	58.5
		125	1,522	0.0%	36.1	1,522	45	0.0%	0.0%	65.6	1,522	71	0.0%	0.0%	131.1
		150	1,522	0.0%	109.6	1,522	113	0.0%	25.1%	267.5	1,522	97	0.0%	0.0%	229.6
		Average		**0.0%**	**37.8**		**36**	**0.0%**	**4.2%**	**64.0**		**40**	**0.0%**	**0.0%**	**73.8**
T2-2	25	25	969	0.0%	1.5	969	4	0.0%	0.0%	1.5	969	6	0.0%	0.0%	2.2
		50	969	0.0%	23.5	969	6	0.0%	0.0%	4.2	969	12	0.0%	0.0%	9.0
		75	969	0.0%	5.6	969	7	0.0%	0.0%	6.8	969	18	0.0%	0.0%	18.1
		100	969	0.0%	36.0	969	120	0.0%	10.7%	211.1	1,000	120	3.2%	28.6%	238.8
		125	1,095	0.0%	80.7	1,095	113	0.0%	62.6%	237.8	1,095	114	0.0%	76.1%	266.6
		150	1,095	0.0%	1,013.8	1,135	109	3.7%	100.0%	255.3	1,134	109	3.6%	100.0%	299.3
		Average		**0.0%**	**193.5**		**60**	**0,6%**	**28.9%**	**119.4**		**63**	**1.1%**	**34.1%**	**139.0**
T2-3	33	25	1,082	0.0%	11.81	1,082	23	0.0%	0.0%	12.8	1,082	19	0.0%	0.0%	11.4
		50	1,177	0.0%	649.7	1,177	87	0.0%	0.0%	109.1	1,177	88	0.0%	0.0%	121.3
		75	1,177	25.5%	283.0	1,221	115	3.7%	100.0%	256.7	1,185	116	0.7%	100.0%	274.9
		100	1,181	26.8%	451.8	1,244	108	5.3%	90.8%	289.9	1,284	108	8.7%	100.0%	324.5
		125	1,177	21.1%	3,140.2	1,284	106	9.1%	100.0%	337.0	1,397	103	18.7%	100.0%	328.2
		150	1,208	26.5%	1,875.6	1,456	96	20.5%	100.0%	389.6	1,382	98	14.4%	100.0%	448.0
		Average		**16.6%**	**1,068.7**		**89**	**6.4%**	**65.1%**	**232.5**		**89**	**7.1%**	**66.7%**	**251.4**

Table 45. Computational Results of GBD-1-1 and GBD-1-4 on T2 Instances – Random 2

Test Problem	|*N*|	|*K*|	CPLEX *LA*	CPLEX (UB-LB) /UB	CPLEX CPU Time (s)	GBD-1-1 *LA*	GBD-1-1 # of Cuts	GBD-1-1 % Dev. from CPLEX	GBD-1-1 (UB-LB) /UB	GBD-1-1 CPU Time (s)	GBD-1-4 *LA*	GBD-1-4 # of Cuts	GBD-1-4 % Dev. from CPLEX	GBD-1-4 (UB-LB) /UB	GBD-1-4 CPU Time (s)
T2-1	21	25	977	0.0%	1.3	977	5	0.0%	0.0%	1.5	977	5	0.0%	0.0%	1.5
		50	977	0.0%	23.4	977	11	0.0%	0.0%	6.4	977	7	0.0%	0.0%	4.5
		75	977	0.0%	25.9	977	111	0.0%	0.0%	127.0	977	17	0.0%	0.0%	15.6
		100	977	0.0%	61.6	983	121	0.6%	35.0%	185.5	983	121	0.6%	25.6%	234.1
		125	1,096	0.0%	55.0	1,127	119	2.8%	43.1%	228.7	1,096	119	0.0%	0.0%	252.2
		150	1,096	0.0%	83.8	1,236	114	12.8%	43.0%	247.1	1,236	113	12.8%	29.7%	265.5
		Average		**0,0%**	**41.8**		**80**	**2.7%**	**20.2%**	**132.7**		**64**	**2.2%**	**9.2%**	**128.9**
T2-2	25	25	862	0.0%	1.6	862	5	0.0%	0.0%	1.7	862	5	0.0%	0.0%	1.9
		50	895	0.0%	17.0	895	69	0.0%	0.0%	54.8	895	41	0.0%	0.0%	32.3
		75	999	0.0%	21.3	999	86	0.0%	0.0%	95.4	999	123	0.0%	0.0%	173.7
		100	999	0.0%	35.3	999	97	0.0%	0.0%	143.5	1,009	120	1.0%	58.3%	222.7
		125	1,095	0.0%	71.4	1,095	113	0.0%	64.7%	239.1	1,095	110	0.0%	0.0%	269.1
		150	1,114	0.0%	74.7	1,114	50	0.0%	0.0%	107.0	1,114	9	0.0%	0.0%	20.7
		Average		**0,0%**	**36.9**		**70**	**0.0%**	**10.8%**	**106.9**		**68**	**0.2%**	**9.7%**	**120.1**
T2-3	33	25	1,112	0.0%	26.7	1,112	37	0.0%	0.0%	20.1	1,112	39	0.0%	0.0%	23.5
		50	1,208	0.0%	231.7	1,234	124	2.2%	60.0%	159.0	1,235	125	2.2%	49.2%	178.1
		75	1,208	26.9%	361.7	1,260	116	4.3%	100.0%	211.9	1,269	116	5.0%	100.0%	240.9
		100	1,208	5.1%	3,476.6	1,352	108	11.9%	100.0%	286.0	1,280	107	6.0%	100.0%	313.8
		125	1,208	19.0%	1,958.0	1,316	106	8.9%	100.0%	341.3	1,326	103	9.8%	45.9%	372.7
		150	1,208	15.0%	3,868.9	1,337	97	10.7%	54.3%	370.3	1,324	98	9.6%	100.0%	426.2
		Average		**11,0%**	**1.653.9**		**98**	**6.3%**	**69.1%**	**231.4**		**98**	**5.4%**	**65.8%**	**259.2**

Table 46. Computational Results of GBD-1-1 and GBD-1-4 on T2 Instances – Random 3

			CPLEX			GBD-1-1					GBD-1-4				
Test Problem	**|*N*|**	**|*K*|**	***LA***	**(UB-LB) /UB**	**CPU Time (s)**	***LA***	**# of Cuts**	**% Dev. from CPLEX**	**(UB-LB) /UB**	**CPU Time (s)**	***LA***	**# of Cuts**	**% Dev. from CPLEX**	**(UB-LB) /UB**	**CPU Time (s)**
T2-1	21	25	948	0.0%	2.6	948	11	0.0%	0.0%	2.9	948	9	0.0%	0.0%	2.8
		50	1,109	0.0%	4.7	1,109	4	0.0%	0.0%	2.5	1,109	5	0.0%	0.0%	3.4
		75	1,198	0.0%	9.0	1,198	6	0.0%	0.0%	5.4	1,198	4	0.0%	0.0%	3.5
		100	1,375	0.0%	33.4	1,375	90	0.0%	0.0%	126.7	1,375	54	0.0%	0.0%	72.6
		125	1,515	0.0%	70.3	1,515	30	0.0%	0.0%	46.0	1,515	38	0.0%	0.0%	65.0
		150	1,515	0.0%	73.3	1,515	113	0.0%	4.7%	257.2	1,515	26	0.0%	0.0%	51.7
		Average		**0.0%**	**32.2**		**42**	**0.0%**	**0.8%**	**73.4**		**23**	**0.0%**	**0.0%**	**33.1**
T2-2	25	25	847	0.0%	1.41	847	3	0.0%	0.0%	1.2	847	3	0.0%	0.0%	1.3
		50	1,002	0.0%	16.7	1,002	35	0.0%	0.0%	23.4	1,002	37	0.0%	0.0%	21.2
		75	1,005	0.0%	69.0	1,005	76	0.0%	0.0%	78.3	1,005	88	0.0%	0.0%	108.6
		100	1,095	0.0%	104.0	1,095	121	0.0%	59.1%	202.2	1,095	121	0.0%	62.6%	222.1
		125	1,095	0.0%	389.1	1,095	113	0.0%	96.6%	235.2	1,095	113	0.0%	96.6%	282.3
		150	1,095	0.0%	162.1	1,145	109	4.6%	100.0%	256.9	1,106	110	1.0%	100.0%	286.8
		Average		**0.0%**	**123.7**		**76**	**0.8%**	**42.6%**	**132.9**		**79**	**0.2%**	**43.2%**	**153.7**
T2-3	33	25	1,043	0.0%	17.5	1,043	101	0.0%	0.0%	63.3	1,043	76	0.0%	0.0%	48.9
		50	1,043	0.0%	212.1	1,069	124	2.5%	94.6%	170.7	1,069	124	2.5%	100.0%	182.0
		75	1,125	14.5%	255.8	1,205	116	7.1%	100.0%	207.9	1,125	116	0.0%	100.0%	237.2
		100	1,125	4.8%	449.1	1,215	108	8.0%	100.0%	273.1	1,224	109	8.8%	100.0%	296.3
		125	1,161	7.8%	749.4	1,342	106	15.6%	100.0%	326.8	1,278	103	10.1%	100.0%	356.9
		150	1,162	7.8%	3.701.4	1,434	98	23.4%	100.0%	355.6	1,309	98	12.7%	100.0%	397.6
		Average		**5.8%**	**897.5**		**109**	**9.4%**	**82.4%**	**232.9**		**104**	**5.7%**	**83.3%**	**253.1**

Table 47. Computational Results of GBD-1-1 and GBD-1-4 on T2 Instances – Random 4

			CPLEX			GBD-1-1					GBD-1-4				
Test Problem	|*N*|	|*K*|	*LA*	(UB-LB)/UB	CPU Time (s)	*LA*	# of Cuts	% Dev. from CPLEX	(UB-LB)/UB	CPU Time (s)	*LA*	# of Cuts	% Dev. from CPLEX	(UB-LB)/UB	CPU Time (s)
T2-1	21	25	873	0.0%	2.3	873	16	0.0%	0.0%	4.1	873	20	0.0%	0.0%	5.7
		50	993	0.0%	26.2	993	13	0.0%	0.0%	7.2	993	15	0.0%	0.0%	9.9
		75	1,005	0.0%	30.5	1,005	34	0.0%	0.0%	29.9	1,005	31	0.0%	0.0%	33.0
		100	1,073	0.0%	80.5	1,073	65	0.0%	0.0%	80.9	1,073	48	0.0%	0.0%	66.7
		125	1,073	0.0%	101.7	1,073	104	0.0%	0.0%	194.5	1,073	95	0.0%	0.0%	180.3
		150	1,073	0.0%	231.6	1,084	112	1.0%	93.7%	254.9	1,073	112	0.0%	92.6%	283.0
		Average		**0,0%**	**78.8**		**57**	**0.2%**	**15.6%**	**95.2**		**54**	**0.0%**	**15.4%**	**96.4**
T2-2	25	25	809	0.0%	6.7	809	17	0.0%	0.0%	5.7	809	10	0.0%	0.0%	3.4
		50	969	0.0%	4.5	969	74	0.0%	0.0%	55.7	969	74	0.0%	0.0%	62.6
		75	969	0.0%	23.3	998	124	3.0%	100.0%	149.1	979	124	1.0%	80.4%	190.2
		100	1,070	0.0%	111.3	1,086	118	1.5%	83.7%	239.4	1,086	119	1.5%	100.0%	229.2
		125	1,070	0.0%	648.3	1,132	112	5.8%	100.0%	256.2	1,120	111	4.7%	100.0%	272.1
		150	1,086	0.0%	3,098.0	1,135	109	4.5%	48.8%	275.8	1,134	109	4.4%	100.0%	318.3
		Average		**0,0%**	**648.7**		**92**	**2.5%**	**55.4%**	**163.7**		**91**	**1.9%**	**63.4%**	**179.3**
T2-3	33	25	1,052	0.0%	16.6	1,052	63	0.0%	0.0%	33.4	1,052	51	0.0%	0.0%	31.2
		50	1,166	0.0%	575.4	1,173	124	0.6%	100.0%	163.7	1,173	123	0.6%	100.0%	185.5
		75	1,166	13.4%	373.4	1,262	115	8.2%	100.0%	247.6	1,280	116	9.8%	100.0%	268.6
		100	1,166	2.2%	975.6	1,303	109	11.7%	100.0%	280.3	1,337	109	14.7%	100.0%	310.1
		125	1,166	4.6%	2,361.6	1,291	106	10.7%	100.0%	331.2	1,291	102	10.7%	100.0%	348.7
		150	1,497	44.2%	157.7	1,327	98	-11.4%	100.0%	356.2	1,363	97	-9.0%	100.0%	402.0
		Average		**10.7%**	**743.4**		**103**	**3.3%**	**83.3%**	**235.4**		**100**	**4.5%**	**83.3%**	**257.7**

Table 48. Computational Results of GBD-1-1 and GBD-1-4 on T2 Instances – Random 5

			CPLEX			GBD-1-1					GBD-1-4				
Test Problem	**|*N*|**	**|*K*|**	***LA***	**(UB-LB)/UB**	**CPU Time (s)**	***LA***	**# of Cuts**	**% Dev. from CPLEX**	**(UB-LB)/UB**	**CPU Time (s)**	***LA***	**# of Cuts**	**% Dev. from CPLEX**	**(UB-LB)/UB**	**CPU Time (s)**
T2-1	21	25	1,153	0.0%	5.5	1,153	28	0.0%	0.0%	7.5	1,153	28	0.0%	0.0%	8.3
		50	1,153	0.0%	27.7	1,153	64	0.0%	0.0%	37.4	1,153	49	0.0%	0.0%	32.1
		75	1,154	0.0%	37.6	1,154	74	0.0%	0.0%	70.8	1,154	81	0.0%	0.0%	90.0
		100	1,154	0.0%	73.4	1,154	121	0.0%	52.8%	197.8	1,206	121	4.5%	23.6%	205.1
		125	1,515	0.0%	169.5	1,515	120	0.0%	24.2%	208.0	1,515	92	0.0%	0.0%	169.9
		150	1,515	0.0%	323.7	1,523	113	0.5%	29.2%	253.2	1,515	113	0.0%	24.2%	280.8
		Average		**0.0%**	**106.2**		**87**	**0.0%**	**17.7%**	**129.1**		**81**	**0.0%**	**8.0%**	**131.0**
T2-2	25	25	830	0.0%	10.7	830	26	0.0%	0.0%	9.0	830	33	0.0%	0.0%	14.4
		50	1,095	0.0%	27.4	1,095	50	0.0%	0.0%	31.4	1,095	46	0.0%	0.0%	33.5
		75	1,095	0.0%	44.6	1,095	116	0.0%	0.0%	124.8	1,095	77	0.0%	0.0%	92.6
		100	1,095	0.0%	89.2	1,095	108	0.0%	0.0%	170.5	1,095	120	0.0%	62.6%	221.8
		125	1,095	0.0%	116.0	1,106	113	1.0%	100.0%	239.3	1,106	112	1.0%	100.0%	258.9
		150	1,095	0.0%	549.4	1,142	111	4.3%	100.0%	280.6	1,168	110	6.7%	100.0%	306.9
		Average		**0.0%**	**139.5**		**87**	**0.9%**	**33.3%**	**142.6**		**83**	**1.3%**	**43.8%**	**154.7**
T2-3	33	25	978	0.0%	20.3	978	54	0.0%	0.0%	35.0	978	55	0.0%	0.0%	42.0
		50	1,009	0.0%	126.3	1,010	123	0.1%	59.4%	237.8	1,035	123	2.6%	54.7%	274.2
		75	1,035	0.0%	146.5	1,142	115	10.3%	100.0%	227.8	1,140	116	10.1%	100.0%	256.3
		100	1,123	0.0%	68.8	1,185	108	5.5%	100.0%	287.4	1,198	109	6.7%	100.0%	321.6
		125	1,140	1.5%	365.1	1,308	105	14.7%	100.0%	328.1	1,317	103	15.5%	100.0%	375.1
		150	1,368	39.6%	149.5	1,366	98	-0.1%	100.0%	354.0	1,382	97	1.0%	100.0%	411.6
		Average		**6,9%**	**146.1**		**101**	**5.1%**	**76.6%**	**245.0**		**101**	**6.0%**	**75.8%**	**280.1**

Table 49. Computational Results of GBD-2-1 and GBD-2-4 on T2 Instances – Random 1

			CPLEX			GBD-2-1					GBD-2-4				
Test Problem	**|*N*|**	**|*K*|**	***LA***	**(UB-LB)/UB**	**CPU Time (s)**	***LA***	**# of Cuts**	**% Dev. from CPLEX**	**(UB-LB)/UB**	**CPU Time (s)**	***LA***	**# of Cuts**	**% Dev. from CPLEX**	**(UB-LB)/UB**	**CPU Time (s)**
T2-1	21	25	1,028	0.0%	3.0	1,028	9	0.0%	0.0%	1.8	1,028	15	0.0%	0.0%	4.0
		50	1,334	0.0%	13.5	1,334	5	0.0%	0.0%	2.0	1,334	3	0.0%	0.0%	1.8
		75	1,382	0.0%	31.6	1,382	14	0.0%	0.0%	7.9	1,382	19	0.0%	0.0%	12.8
		100	1,522	0.0%	33.1	1,522	33	0.0%	0.0%	24.2	1,522	48	0.0%	0.0%	46.5
		125	1,522	0.0%	36.1	1,522	38	0.0%	0.0%	35.9	1,522	65	0.0%	0.0%	78.8
		150	1,522	0.0%	109.6	1,522	171	0.0%	100.0%	249.5	1,522	61	0.0%	0.0%	90.1
		Average		**0.0%**	**37.8**		**45**	**0,0%**	**16.7%**	**53.5**		**35**	**0.0%**	**0.0%**	**39.0**
T2-2	25	25	969	0.0%	1.5	969	4	0.0%	0.0%	1.1	969	5	0.0%	0.0%	1.6
		50	969	0.0%	23.5	969	6	0.0%	0.0%	2.9	969	10	0.0%	0.0%	5.4
		75	969	0.0%	5.6	969	7	0.0%	0.0%	4.6	969	25	0.0%	0.0%	19.6
		100	969	0.0%	36.0	969	159	0.0%	0.0%	168.8	969	180	0.0%	100.0%	233.5
		125	1,095	0.0%	80.7	1,095	174	0.0%	100.0%	232.1	1,095	173	0.0%	100.0%	270.3
		150	1,095	0.0%	1,013.8	1,135	168	3.7%	100.0%	287.4	1,134	166	3.6%	100.0%	339.3
		Average		**0.0%**	**193.5**		**86**	**0,6%**	**33.3%**	**116.1**		**93**	**0.6%**	**50.0%**	**144.9**
T2-3	33	25	1,082	0.0%	11.81	1,082	22	0.0%	0.0%	8.2	1,082	20	0.0%	0.0%	8.5
		50	1,177	0.0%	649.7	1,177	68	0.0%	0.0%	53.4	1,177	96	0.0%	0.0%	88.3
		75	1,177	25.5%	283.0	1,221	176	3.7%	100.0%	240.3	1,186	173	0.8%	100.0%	282.8
		100	1,181	26.8%	451.8	1,231	167	4.2%	100.0%	326.7	1,264	164	7.0%	100.0%	390.9
		125	1,177	21.1%	3,140.2	1,315	160	11.7%	100.0%	379.5	1,375	153	16.8%	100.0%	450.3
		150	1,208	26.5%	1,875.6	1,374	150	13.7%	100.0%	456.2	1,382	149	14.4%	100.0%	551.3
		Average		**16.6%**	**1.068.7**		**124**	**5,6%**	**66.7%**	**244.0**		**126**	**6.5%**	**66.7%**	**295.3**

Table 50. Computational Results of GBD-2-1 and GBD-2-4 on T2 Instances – Random 2

Test Problem	|*N*|	|*K*|	CPLEX *LA*	CPLEX (UB-LB)/UB	CPLEX CPU Time (s)	GBD-2-1 *LA*	GBD-2-1 # of Cuts	GBD-2-1 % Dev. from CPLEX	GBD-2-1 (UB-LB)/UB	GBD-2-1 CPU Time (s)	GBD-2-4 *LA*	GBD-2-4 # of Cuts	GBD-2-4 % Dev. from CPLEX	GBD-2-4 (UB-LB)/UB	GBD-2-4 CPU Time (s)
T2-1	21	25	977	0.0%	1.3	977	5	0.0%	0.0%	1.0	977	5	0.0%	0.0%	1.1
		50	977	0.0%	23.4	977	16	0.0%	0.0%	6.2	977	7	0.0%	0.0%	3.1
		75	977	0.0%	25.9	977	87	0.0%	0.0%	52.0	977	20	0.0%	0.0%	12.5
		100	977	0.0%	61.6	977	168	0.0%	0.0%	155.0	998	180	2.1%	100.0%	204.5
		125	1,096	0.0%	55.0	1,096	178	0.0%	100.0%	212.2	1,112	178	1.5%	100.0%	248.4
		150	1,096	0.0%	83.8	1,236	171	12.8%	100.0%	246.2	1,119	171	2.1%	100.0%	300.6
		Average		**0.0%**	**41.8**		**104**	**2.1%**	**33.3%**	**112.1**		**94**	**1.0%**	**50.0%**	**128.4**
T2-2	25	25	862	0.0%	1.6	862	5	0.0%	0.0%	1.2	862	6	0.0%	0.0%	1.7
		50	895	0.0%	17.0	895	50	0.0%	0.0%	22.6	895	91	0.0%	0.0%	52.4
		75	999	0.0%	21.3	999	190	0.0%	100.0%	145.8	999	102	0.0%	0.0%	85.1
		100	999	0.0%	35.3	1,044	180	4.5%	100.0%	192.0	1,044	181	4.5%	100.0%	222.2
		125	1,095	0.0%	71.4	1,095	174	0.0%	100.0%	250.4	1,095	128	0.0%	0.0%	200.6
		150	1,114	0.0%	74.7	1,124	167	0.9%	100.0%	278.9	1,114	9	0.0%	0.0%	14.7
		Average		**0.0%**	**36.9**		**128**	**0.9%**	**66.7%**	**148.5**		**86**	**0.8%**	**16.7%**	**96.1**
T2-3	33	25	1,112	0.0%	26.7	1,112	46	0.0%	0.0%	16.2	1,112	42	0.0%	0.0%	17.4
		50	1,208	0.0%	231.7	1,208	190	0.0%	100.0%	139.3	1,208	186	0.0%	100.0%	165.5
		75	1,208	26.9%	361.7	1,259	175	4.2%	100.0%	226.2	1,269	174	5.0%	100.0%	248.8
		100	1,208	5.1%	3,476.6	1,320	167	9.3%	100.0%	323.0	1,291	165	6.9%	100.0%	369.4
		125	1,208	19.0%	1,958.0	1,341	159	11.0%	100.0%	377.8	1,313	153	8.7%	100.0%	414.8
		150	1,208	15.0%	3,868.9	1,336	152	10.6%	100.0%	422.7	1,353	148	12.0%	100.0%	497.0
		Average		**11.0%**	**1,653.9**		**148**	**5.8%**	**83.3%**	**250.9**		**145**	**5.4%**	**83.3%**	**285.5**

Table 51. Computational Results of GBD-2-1 and GBD-2-4 on T2 Instances – Random 3

Test Problem	|*N*|	|*K*|	CPLEX *LA*	CPLEX (UB-LB)/UB	CPLEX CPU Time (s)	GBD-2-1 *LA*	GBD-2-1 # of Cuts	GBD-2-1 % Dev. from CPLEX	GBD-2-1 (UB-LB) /UB	GBD-2-1 CPU Time (s)	GBD-2-4 *LA*	GBD-2-4 # of Cuts	GBD-2-4 % Dev. from CPLEX	GBD-2-4 (UB-LB) /UB	GBD-2-4 CPU Time (s)
T2-1	21	25	948	0.0%	2.6	948	11	0.0%	0.0%	2.1	948	9	0.0%	0.0%	2.0
		50	1,109	0.0%	4.7	1,109	3	0.0%	0.0%	1.3	1,109	7	0.0%	0.0%	3.4
		75	1,198	0.0%	9.0	1,198	6	0.0%	0.0%	3.5	1,198	5	0.0%	0.0%	3.6
		100	1,375	0.0%	33.4	1,375	181	0.0%	0.0%	166.6	1,375	50	0.0%	0.0%	46.3
		125	1,515	0.0%	70.3	1,515	82	0.0%	0.0%	86.8	1,515	40	0.0%	0.0%	48.7
		150	1,515	0.0%	73.3	1,515	149	0.0%	0.0%	219.2	1,515	29	0.0%	0.0%	41.4
		Average		**0.0%**	**32.2**		**72**	**0.0%**	**0.0%**	**79.9**		**23**	**0.0%**	**0.0%**	**24.2**
T2-2	25	25	847	0.0%	1.41	847	3	0.0%	0.0%	0.9	847	3	0.0%	0.0%	0.9
		50	1,002	0.0%	16.7	1,002	40	0.0%	0.0%	17.9	1,002	44	0.0%	0.0%	22.8
		75	1,005	0.0%	69.0	1,005	99	0.0%	0.0%	64.4	1,005	75	0.0%	0.0%	59.9
		100	1,095	0.0%	104.0	1,095	181	0.0%	100.0%	181.4	1,095	181	0.0%	100.0%	205.2
		125	1,095	0.0%	389.1	1,095	174	0.0%	100.0%	230.2	1,106	172	1.0%	100.0%	272.8
		150	1,095	0.0%	162.1	1,125	169	2.7%	100.0%	266.9	1,110	167	1.4%	100.0%	473.0
		Average		**0.0%**	**123.7**		**111**	**0.5%**	**50.0%**	**127.0**		**107**	**0.4%**	**50.0%**	**172.4**
T2-3	33	25	1,043	0.0%	17.5	1,043	106	0.0%	0.0%	43.9	1,043	105	0.0%	0.0%	46.3
		50	1,043	0.0%	212.1	1,043	187	0.0%	100.0%	136.9	1,069	185	2.5%	100.0%	177.3
		75	1,125	14.5%	255.8	1,125	174	0.0%	100.0%	212.1	1,125	173	0.0%	100.0%	252.5
		100	1,125	4.8%	449.1	1,215	167	8.0%	100.0%	283.5	1,224	165	8.8%	100.0%	339.5
		125	1,161	7.8%	749.4	1,339	161	15.3%	100.0%	370.4	1,263	155	8.8%	100.0%	406.1
		150	1,162	7.8%	3,701.4	1,346	152	15.8%	100.0%	410.2	1,314	147	13.1%	100.0%	473.0
		Average		**5.8%**	**897.5**		**158**	**6.5%**	**83.3%**	**242.8**		**155**	**5.5%**	**83.3%**	**282.4**

Table 52. Computational Results of GBD-2-1 and GBD-2-4 on T2 Instances – Random 4

			CPLEX			GBD-2-1					GBD-2-4				
Test Problem	**|*N*|**	**|*K*|**	***LA***	**(UB-LB)/UB**	**CPU Time (s)**	***LA***	**# of Cuts**	**% Dev. from CPLEX**	**(UB-LB)/UB**	**CPU Time (s)**	***LA***	**# of Cuts**	**% Dev. from CPLEX**	**(UB-LB)/UB**	**CPU Time (s)**
T2-1	21	25	873	0.0%	2.3	873	18	0.0%	0.0%	3.3	873	19	0.0%	0.0%	4.1
		50	993	0.0%	26.2	993	13	0.0%	0.0%	4.9	993	14	0.0%	0.0%	6.5
		75	1,005	0.0%	30.5	1,005	39	0.0%	0.0%	21.5	1,005	20	0.0%	0.0%	14.0
		100	1,073	0.0%	80.5	1,073	59	0.0%	0.0%	46.2	1,073	39	0.0%	0.0%	35.7
		125	1,073	0.0%	101.7	1,073	112	0.0%	0.0%	117.2	1,073	100	0.0%	0.0%	129.3
		150	1,073	0.0%	231.6	1,088	173	1.4%	100.0%	251.6	1,119	171	4.3%	100.0%	286.1
		Average		**0.0%**	**78.8**		**69**	**0.2%**	**16.7%**	**74.1**		**61**	**0.7%**	**16.7%**	**79.3**
T2-2	25	25	809	0.0%	6.7	809	18	0.0%	0.0%	3.7	809	10	0.0%	0.0%	2.6
		50	969	0.0%	4.5	969	53	0.0%	0.0%	23.0	969	29	0.0%	0.0%	15.4
		75	969	0.0%	23.3	994	189	2.6%	100.0%	157.0	979	186	1.0%	100.0%	181.3
		100	1,070	0.0%	111.3	1,106	180	3.4%	100.0%	201.1	1,095	179	2.3%	100.0%	237.9
		125	1,070	0.0%	648.3	1,117	173	4.4%	100.0%	270.1	1,124	171	5.0%	100.0%	292.7
		150	1,086	0.0%	3,098.0	1,132	167	4.2%	100.0%	307.7	1,134	165	0.2%	100.0%	351.3
		Average		**0.0%**	**648.7**		**130**	**2.4%**	**66.7%**	**160.4**		**123**	**1.4%**	**66.7%**	**180.2**
T2-3	33	25	1,052	0.0%	16.6	1,052	65	0.0%	0.0%	23.3	1,052	50	0.0%	0.0%	21.9
		50	1,166	0.0%	575.4	1,166	132	0.0%	0.0%	110.4	1,166	181	0.0%	0.0%	187.6
		75	1,166	13.4%	373.4	1,268	173	8.7%	100.0%	253.1	1,259	173	8.0%	100.0%	275.8
		100	1,166	2.2%	975.6	1,291	167	10.7%	100.0%	319.1	1,346	165	15.4%	100.0%	363.2
		125	1,166	4.6%	2,361.6	1,291	161	10.7%	100.0%	371.3	1,378	156	18.2%	100.0%	409.9
		150	1,497	44.2%	157.7	1,332	152	-11.0%	100.0%	409.8	1,365	147	-8.8%	100.0%	471.5
		Average		**10.7%**	**743.4**		**142**	**3.2%**	**66.7%**	**247.8**		**145**	**5.5%**	**66.7%**	**288.3**

Table 53. Computational Results of GBD-2-1 and GBD-2-4 on T2 Instances – Random 5

Test Problem	|*N*|	|*K*|	CPLEX *LA*	CPLEX (UB-LB)/UB	CPLEX CPU Time (s)	GBD-2-1 *LA*	GBD-2-1 # of Cuts	GBD-2-1 % Dev. from CPLEX	GBD-2-1 (UB-LB)/UB	GBD-2-1 CPU Time (s)	GBD-2-4 *LA*	GBD-2-4 # of Cuts	GBD-2-4 % Dev. from CPLEX	GBD-2-4 (UB-LB)/UB	GBD-2-4 CPU Time (s)
T2-1	21	25	1,153	0.0%	5.5	1,153	28	0.0%	0.0%	4.8	1,153	28	0.0%	0.0%	5.7
		50	1,153	0.0%	27.7	1,153	58	0.0%	0.0%	20.9	1,153	69	0.0%	0.0%	29.6
		75	1,154	0.0%	37.6	1,154	78	0.0%	0.0%	46.3	1,154	84	0.0%	0.0%	58.5
		100	1,154	0.0%	73.4	1,154	185	0.0%	100.0%	177.2	1,160	180	0.5%	100.0%	199.9
		125	1,515	0.0%	169.5	1,515	180	0.0%	100.0%	204.0	1,515	102	0.0%	0.0%	122.5
		150	1,515	0.0%	323.7	1,524	172	0.6%	100.0%	259.4	1,515	70	0.0%	0.0%	90.1
		Average		**0.0%**	**106.2**		**117**	**0.1%**	**50.0%**	**118.8**		**89**	**0.0%**	**16.7%**	**84.4**
T2-2	25	25	830	0.0%	10.7	830	23	0.0%	0.0%	5.5	830	30	0.0%	0.0%	8.8
		50	1,095	0.0%	27.4	1,095	57	0.0%	0.0%	22.9	1,095	53	0.0%	0.0%	27.2
		75	1,095	0.0%	44.6	1,095	117	0.0%	0.0%	71.2	1,095	103	0.0%	0.0%	85.5
		100	1,095	0.0%	89.2	1,095	72	0.0%	0.0%	61.6	1,095	178	0.0%	0.0%	233.6
		125	1,095	0.0%	116.0	1,106	173	1.0%	100.0%	232.5	1,106	172	1.0%	100.0%	281.7
		150	1,095	0.0%	549.4	1,145	167	4.6%	100.0%	284.3	1,154	165	5.4%	100.0%	345.7
		Average		**0.0%**	**139.5**		**102**	**0.9%**	**33.3%**	**113.0**		**117**	**1.1%**	**33.3%**	**163.7**
T2-3	33	25	978	0.0%	20.3	978	50	0.0%	0.0%	18.3	978	63	0.0%	0.0%	30.0
		50	1,009	0.0%	126.3	1,047	187	3.8%	100.0%	185.8	1,047	186	3.8%	100.0%	194.9
		75	1,035	0.0%	146.5	1,140	173	10.1%	100.0%	233.3	1,169	173	12.9%	100.0%	295.2
		100	1,123	0.0%	68.8	1,198	165	6.7%	100.0%	303.7	1,179	165	5.0%	100.0%	361.3
		125	1,140	1.5%	365.1	1,308	160	14.7%	100.0%	361.8	1,317	155	15.5%	100.0%	426.2
		150	1,368	39.6%	149.5	1,355	151	-1.0%	100.0%	412.2	1,354	145	-1.0%	100.0%	472.2
		Average		**6.9%**	**146.1**		**148**	**5.7%**	**83.3%**	**252.5**		**148**	**6.0%**	**83.3%**	**296.6**

APPENDIX – H

COMPUTATIONAL RESULTS OF GBD ALGORITHMS ON T3 INSTANCES

The tables of Appendix H are presented starting on the next page.

Table 54. Computational Results of GBD-1-1 and GBD-1-4 on T3 Instances – Random 1

Test Problem	|*N*|	|*K*|	Max. SP Length	CPLEX *LA*	CPLEX (UB-LB)/UB	CPLEX CPU Time (s)	GBD-1-1 *LA*	GBD-1-1 # of Cuts	GBD-1-1 % Dev. from CPLEX	GBD-1-1 % Dev. from Max. SP Length	GBD-1-1 (UB-LB)/UB	GBD-1-1 CPU Time (s)	GBD-1-4 *LA*	GBD-1-4 # of Cuts	GBD-1-4 %Dev. from CPLEX	GBD-1-4 % Dev. from Max. SP Length	GBD-1-4 (UB-LB)/UB	GBD-1-4 CPU Time (s)
T3-1	52	50	1,216	1,235	48.3%	749	1.246	102	0.9%	2.5%	80.6%	416	1,246	101	0.9%	2.5%	100,0%	424
		100	1,301	1,571	54.0%	868	1.571	70	0.0%	20.8%	84.1%	615	1,571	71	0.0%	20.8%	100.0%	632
		150	1,316	1,571	55,3%	5,846	1.758	43	11.9%	33.6%	90.7%	534	1,760	48	12.0%	33.7%	94.1%	683
		200	1,316	Cannot find a soln.		1,401	1,921	44	-	46.0%	100.0%	749	1,811	42	-	37.6%	94.3%	781
		Average			**52.6%**	**2,216**		**65**	**4.3%**	**25.7%**	**88.8%**	**578**		**66**	**4.3%**	**23.6%**	**97.1%**	**630**
T3-2	62	50	1,711	1,816	59,8%	615	1.830	90	0.8%	7.0%	100,0%	453	1,837	91	1.2%	7.4%	100.0%	477
		100	1,871	Cannot find a soln.		2,279	2,007	55		7.3%	100.0%	551	1,980	58	-	5.8%	100.0%	669
		150	1,871	Cannot find a soln.		8,440	2,072	49	-	10.7%	100.0%	756	2,070	51	-	10.6%	100.0%	884
		200	1,871	Cannot find a soln.		125	2,215	29	-	18.4%	100.0%	625	2,274	31	-	21.5%	100.0%	731
		Average			**59.8%**	**2,865**		**56**	**0.8%**	**10.8%**	**100.0%**	**596**		**58**	**1.2%**	**11.3%**	**100.0%**	**690**
T3-3	81	50	2,070	2,070	61.6%	731	2.222	78	7.3%	7.3%	100,0%	558	2,228	79	7.6%	7.6%	100,0%	635
		100	2,070	Cannot find a soln.		446	2,382	51	-	15.1%	100.0%	755	2,427	53	-	17.2%	100.0%	881
		150	2,070	Cannot find a soln.		394	2,524	27	-	21.9%	97.7%	621	2,602	29	-	25.7%	100.0%	787
		200	2,090	Cannot find a soln.		268	2,783	21	-	33.2%	94.5%	692	2,665	21	-	27.5%	100.0%	792
		Average			**61.6%**	**460**		**44**	**7.3%**	**19.4%**	**98.1%**	**657**		**46**	**7.6%**	**19.5%**	**100.0%**	**774**

Table 55. Computational Results of GBD-1-1 and GBD-1-4 on T3 Instances – Random 2

				CPLEX			**GBD-1-1**						**GBD-1-4**					
Test Problem	**|*N*|**	**|*K*|**	**Max. SP Length**	***LA***	**(UB-LB) /UB**	**CPU Time (s)**	***LA***	**# of Cuts**	**% Dev. from CPLEX**	**% Dev. from Max. SP Length**	**(UB-LB) /UB**	**CPU Time (s)**	***LA***	**# of Cuts**	**% Dev. from CPLEX**	**% Dev. from Max. SP Length**	**(UB-LB) /UB**	**CPU Time (s)**
T3-1	52	50	1,297	1,297	52.3%	612	1,358	103	4.7%	4.7%	100.0%	365	1,319	101	1.7%	1.7%	100.0%	442
		100	1,297	1,319	30.8%	722	1,374	71	4.2%	5.9%	100.0%	582	1,505	70	14.1%	16.0%	100.0%	624
		150	1,316	Cannot find a soln.		2,588	1,880	41	-	42.9%	100.0%	591	1,803	46	-	37.0%	100.0%	669
		200	1,316	Cannot find a soln.		303	1,970	45	-	49.7%	69.0%	863	1,995	42	-	51.6%	84.7%	787
		Average			**41.5%**	**1,056**		**65**	**4.4%**	**25.8%**	**92.2%**	**600**		**65**	**7.9%**	**26.6%**	**96.2%**	**631**
T3-2	62	50	1,779	1,779	53.0%	330	1,787	89	0.4%	0.4%	100.0%	417	1,782	90	0.2%	0.2%	100.0%	492
		100	1,779	1,782	35.2%	813	1,826	56	2.5%	2.6%	100.0%	535	1,793	58	0.6%	0.8%	100.0%	608
		150	1,779	Cannot find a soln.		277	2,004	49	-	12.6%	100.0%	712	2,034	51	-	14.3%	100.0%	833
		200	1,779	Cannot find a soln.		510	2,182	30	-	22.7%	100.0%	612	2,050	30	-	15.2%	100.0%	682
		Average			**44.1%**	**483**		**56**	**1.5%**	**9.6%**	**100.0%**	**569**		**57**	**0.4%**	**7.6%**	**100.0%**	**654**
T3-3	81	50	2,211	2,723	83.3%	346	2,334	78	-14.3%	5.6%	100.0%	514	2,345	79	-13.9%	6.1%	100.0%	576
		100	2,211	Cannot find a soln.		264	2,383	52	-	7.8%	100.0%	761	2,429	53	-	9.9%	100.0%	864
		150	2,211	Cannot find a soln.		101	2,534	28	-	14.6%	100.0%	641	2,548	29	-	15.2%	100.0%	737
		200	2,211	Cannot find a soln.		38	2,606	22	-	17.9%	100.0%	698	2,611	21	-	18.1%	100.0%	757
		Average			**83.3%**	**187**		**45**	**-14.3%**	**11.5%**	**100.0%**	**653**		**46**	**-13.9%**	**12.3%**	**100.0%**	**733**

Table 56. Computational Results of GBD-1-1 and GBD-1-4 on T3 Instances – Random 3

				CPLEX			GBD-1-1						GBD-1-4					
Test Problem	**|*N*|**	**|*K*|**	**Max. SP Length**	***LA***	**(UB-LB) /UB**	**CPU Time (s)**	***LA***	**# of Cuts**	**% Dev. from CPLEX**	**% Dev. from Max. SP Length**	**(UB-LB) /UB**	**CPU Time (s)**	***LA***	**# of Cuts**	**% Dev. from CPLEX**	**% Dev. from Max. SP Length**	**(UB-LB) /UB**	**CPU Time (s)**
T3-1	52	50	1,518	1,518	59.6%	357	1,541	100	1.5%	1.5%	100.0%	420	1,548	100	2.0%	2.0%	100.0%	440
		100	1,518	1,534	40.5%	743	1,563	72	1.9%	3.0%	100.0%	579	1,568	72	2.2%	3.3%	100.0%	634
		150	1,518	Cannot find a soln.		306	2,134	46	-	40.6%	100.0%	629	2,088	45	-	37.5%	100.0%	659
		200	1,518	Cannot find a soln.		321	2,088	45	-	37.5%	48.9%	871	2,006	43	-	32.1%	100.0%	809
		Average			**50.0%**	**432**		**66**	**1.7%**	**20.7%**	**87.2%**	**625**		**65**	**2.1%**	**18.7%**	**100.0%**	**635**
T3-2	62	50	1,745	1,786	41.8%	850	1,856	88	3.9%	6.4%	100.0%	416	1,845	90	3.3%	5.7%	100.0%	491
		100	1,779	Cannot find a soln.		1.309	1,977	52	-	11.1%	100.0%	513	1,975	58	-	11.0%	100.0%	638
		150	1,779	Cannot find a soln.		5.841	2,092	49	-	17.6%	100.0%	759	2,012	50	-	13.1%	100.0%	839
		200	1,779	Cannot find a soln.		499	2,325	29	-	30.7%	100.0%	629	2,066	31	-	16.1%	100.0%	721
		Average			**41.8%**	**2.125**		**55**	**3.9%**	**16.4%**	**100.0%**	**579**		**57**	**3.3%**	**11.5%**	**100.0%**	**672**
T3-3	81	50	2,039	2,129	75.5%	173	2,061	77	-3.2%	1.1%	100.0%	541	2,086	79	-2.0%	2.3%	100.0%	638
		100	2,039	Cannot find a soln.		3.591	2,478	52	-	21.5%	100.0%	772	2,451	53	-	20.2%	100.0%	930
		150	2,039	Cannot find a soln.		71	2,577	28	-	26.4%	100.0%	659	2,530	28	-	24.1%	100.0%	741
		200	2,039	Cannot find a soln.		41	2,693	21	-	32.1%	100.0%	690	2,633	20	-	29.1%	100.0%	742
		Average			**75.5%**	**969**		**45**	**-3.2%**	**20.3%**	**100.0%**	**666**		**45**	**-2.0%**	**18.9%**	**100.0%**	**763**

Table 57. Computational Results of GBD-1-1 and GBD-1-4 on T3 Instances – Random 4

Test Problem	|*N*|	|*K*|	Max. SP Length	CPLEX *LA*	CPLEX (UB-LB) /UB	CPLEX CPU Time (s)	GBD-1-1 *LA*	GBD-1-1 # of Cuts	GBD-1-1 % Dev. from CPLEX	GBD-1-1 % Dev. from Max. SP Length	GBD-1-1 (UB-LB) /UB	GBD-1-1 CPU Time (s)	GBD-1-4 *LA*	GBD-1-4 # of Cuts	GBD-1-4 % Dev. from CPLEX	GBD-1-4 % Dev. from Max. SP Length	GBD-1-4 (UB-LB) /UB	GBD-1-4 CPU Time (s)
T3-1	52	50	1,361	1,361	71.1%	351	1,361	101	0.0%	0.0%	100.0%	408	1,361	99	0.0%	0.0%	100.0%	421
		100	1,362	1,965	82.3%	144	1,571	72	-20.1%	15.3%	100.0%	580	1,607	72	-18.2%	18.0%	100.0%	630
		150	1,381	1,602	72.4%	1,515	1,740	45	8.6%	26.0%	100.0%	590	1,726	48	7.7%	25.0%	85.7%	640
		200	1,566	Cannot find a soln,		477	2,251	45	-	43.7%	77.3%	749	2,242	43	-	43.2%	79.8%	809
		Average			**75.2%**	**622**		**66**	**-3.8%**	**21.3%**	**94.3%**	**582**		**66**	**-3.5%**	**21.5%**	**91.4%**	**625**
T3-2	62	50	1,871	1,871	69.5%	119	1,871	89	0.0%	0.0%	100.0%	434	1,887	91	0.9%	0.9%	100.0%	469
		100	1,871	1,919	69.7%	572	1,942	56	1.2%	3.8%	100.0%	552	1,931	58	0.6%	3.2%	100.0%	610
		150	1,871	Cannot find a soln,		313	2,042	49	-	9.1%	100.0%	740	1,997	50	-	6.7%	100.0%	825
		200	1,871	Cannot find a soln,		127	2,304	28	-	23.1%	100.0%	589	2,136	31	-	14.2%	100.0%	707
		Average			**69.6%**	**283**		**56**	**0.6%**	**9.0%**	**100.0%**	**579**		**58**	**0.7%**	**6.2%**	**100.0%**	**653**
T3-3	81	50	2,036	2,356	82.3%	134	2,138	79	-9.3%	5.0%	100.0%	518	2,114	79	-10.3%	3.8%	100.0%	582
		100	2,036	Cannot find a soln.		3,047	2,264	51	-	11.2%	100.0%	777	2,294	52	-	12.7%	100.0%	862
		150	2,036	Cannot find a soln.		56	2,333	27	-	14.6%	100.0%	648	2,513	28	-	23.4%	100.0%	753
		200	2,101	Cannot find a soln.		49	2,602	21	-	23.8%	100.0%	704	2,632	20	-	25.3%	100.0%	745
		Average			**82.3%**	**822**		**45**	**-9.3%**	**13.7%**	**100.0%**	**662**		**45**	**-10.3%**	**16.3%**	**100.0%**	**736**

Table 58. Computational Results of GBD-1-1 and GBD-1-4 on T3 Instances – Random 5

Test Problem	|*N*|	|*K*|	Max. SP Length	CPLEX			GBD-1-1						GBD-1-4					
				LA	**(UB-LB) /UB**	**CPU Time (s)**	***LA***	**# of Cuts**	**% Dev. from CPLEX**	**% Dev. from Max. SP Length**	**(UB-LB) /UB**	**CPU Time (s)**	***LA***	**# of Cuts**	**% Dev. from CPLEX**	**% Dev. from Max. SP Length**	**(UB-LB) /UB**	**CPU Time (s)**
T3-1	52	50	1,313	1,313	45.2%	393	1,364	100	3.9%	3.9%	77.6%	468	1,341	99	2.1%	2.1%	39.5%	475
		100	1,554	1,629	64.3%	226	1,572	71	-3.5%	1.2%	100.0%	574	1,579	73	-3.1%	1.6%	100.0%	638
		150	1,554	1,554	53.7%	1,684	1,683	43	8.3%	8.3%	100.0%	527	1,677	48	7.9%	7.9%	100.0%	647
		200	1,554	Cannot find a soln,		274	1,817	44	-	16.9%	100.0%	737	1.738	44	-	11.8%	100.0%	813
		Average			**54.4%**	**644**		**65**	**2.9%**	**7.6%**	**94.4%**	**576**		**66**	**2.3%**	**5.9%**	**84.9%**	**643**
T3-2	62	50	1,777	1,777	62.8%	534	1,825	88	2.7%	2.7%	100.0%	448	1,821	90	2.5%	2.5%	100.0%	468
		100	1,777	1,793	63.1%	6,036	1,873	55	4.5%	5.4%	100,0%	532	1,884	58	5.1%	6.0%	100.0%	621
		150	1,777	Cannot find a soln,		336	2,060	49	-	15.9%	100.0%	734	2,059	51	-	15.9%	100.0%	842
		200	1,791	Cannot find a soln,		773	2,288	29	-	27.7%	100.0%	588	2,096	31	-	17.0%	100.0%	689
		Average			**63.0%**	**1,920**		**55**	**3.6%**	**12.9%**	**100.0%**	**576**		**58**	**3.8%**	**10.3%**	**100.0%**	**655**
T3-3	81	50	1,918	1,923	69.9%	5,335	2,004	80	4.2%	4.5%	100,0%	484	2,032	78	5.7%	5.9%	100.0%	548
		100	2,148	Cannot find a soln.		162	2,368	51	-	10.2%	100.0%	727	2,347	53	-	9.3%	100.0%	826
		150	2,148	Cannot find a soln.		138	2,498	27	-	16.3%	100.0%	631	2,470	28	-	15.0%	100.0%	705
		200	2,148	Cannot find a soln.		40	2,621	22	-	22.0%	100.0%	709	2,644	20	-	23.1%	100.0%	715
		Average			**69.9%**	**1,419**		**45**	**4.2%**	**13.3%**	**100.0%**	**638**		**45**	**5.7%**	**13.3%**	**100.0%**	**698**

Table 59. Computational Results of GBD-2-1 and GBD-2-4 on T3 Instances – Random 1

Test Problem	|*N*|	|*K*|	Max. SP Length	CPLEX			GBD-2-1						GBD-2-4					
				LA	(UB-LB) /UB	CPU Time (s)	*LA*	# of Cuts	% Dev. from CPLEX	% Dev. from Max. SP Length	(UB-LB) /UB	CPU Time (s)	*LA*	# of Cuts	% Dev. from CPLEX	% Dev. from Max. SP Length	(UB-LB) /UB	CPU Time (s)
T3-1	52	50	1,216	1,235	48.3%	749	1,246	157	0.9%	2.5%	100%	444	1,234	157	-0.1%	1.5%	100%	499
		100	1,301	1,571	54.0%	868	1,571	120	0.0%	20.8%	100%	747	1,571	115	0.0%	20.8%	100%	829
		150	1,316	1,571	55.3%	5.846	1,772	72	11.3%	34.7%	100%	716	1,776	79	13.0%	35,0%	100%	926
		200	1,316	Cannot find a soln.		1.401	1,803	82	-	37.0%	100%	1,130	1,811	74	-	37,6%	100%	1,153
		Average			**52.6%**	**2.216**		**108**	**4.1%**	**23.7%**	**100%**	**759**		**106**	**4.3%**	**23.7%**	**100%**	**852**
T3-2	62	50	1,711	1,816	59.8%	615	1,830	140	0.8%	7.0%	100%	518	1,785	76	-1.7%	4.3%	100%	580
		100	1,871	Cannot find a soln.		2.279	1,983	94	-	6.0%	100%	773	1,990	95	-	6.4%	100%	877
		150	1,871	Cannot find a soln.		8.440	2,107	88	-	12.6%	100%	1,097	2,043	88	-	9.2%	100%	1,245
		200	1,871	Cannot find a soln.		125	2,244	60	-	19.9%	100%	1,052	2,135	58	-	14.1%	100%	1,136
		Average			**59.8%**	**2.865**		**96**	**0.8%**	**11.4%**	**100%**	**860**		**79**	**-1,7%**	**8.5%**	**100%**	**959**
T3-3	81	50	2,070	2,070	61.6%	731	2,192	124	5.9%	5.9%	100%	707	2,158	127	4.3%	4.3%	100%	839
		100	2,070	Cannot find a soln.		446	2,292	90	-	10.7%	100%	1,088	2,404	89	-	16.1%	100%	1,236
		150	2,070	Cannot find a soln.		394	2,476	56	-	19.6%	100%	1,052	2,571	56	-	24.2%	100%	1,299
		200	2,090	Cannot find a soln.		268	2,736	47	-	30.9%	100%	1,311	2,681	43	-	28.3%	100%	1,376
		Average			**61.6%**	**460**		**79**	**5.9%**	**16.8%**	**100%**	**1,039**		**79**	**4.3%**	**18.2%**	**100%**	**1,187**

Table 60. Computational Results of GBD-2-1 and GBD-2-4 on T3 Instances – Random 2

				CPLEX			GBD-2-1						GBD-2-4					
Test Problem	|*N*|	|*K*|	Max. SP Length	*LA*	(UB-LB) /UB	CPU Time (s)	*LA*	# of Cuts	% Dev. from CPLEX	% Dev. from Max. SP Length	(UB-LB) /UB	CPU Time (s)	*LA*	# of Cuts	% Dev. from CPLEX	% Dev. from Max. SP Length	(UB-LB) /UB	CPU Time (s)
T3-1	52	50	1,297	1,297	52.3%	612	1,361	161	4.9%	4.9%	100%	423	1,336	155	3.0%	3.0%	100%	528
		100	1,297	1,319	30.8%	722	1,560	121	18.3%	20.3%	100%	773	1,555	114	17.9%	19.9%	100%	806
		150	1,316	Cannot find a soln.		2.588	1,811	76	-	37.6%	100%	803	1,811	79	-	37.6%	100%	901
		200	1,316	Cannot find a soln.		303	2,024	83	-	53.8%	100%	1,175	1,906	75	-	44.8%	100%	1,141
		Average			**41.5%**	**1.056**		**110**	**11.6%**	**29.2%**	**100%**	**794**		**106**	**10.4%**	**26.3%**	**100%**	**844**
T3-2	62	50	1,779	1,779	53.0%	330	1,779	140	0.0%	0.0%	100%	505	1,779	140	0.0%	0.0%	100%	605
		100	1,779	1,782	35.2%	813	1,798	96	0.9%	1.1%	100%	717	1,806	97	1.3%	1,5%	100%	857
		150	1,779	Cannot find a soln.		277	2,004	89	-	12.6%	100%	1,060	1,961	88	-	10.2%	100%	1,190
		200	1,779	Cannot find a soln.		510	1,977	57	-	11.1%	100%	1,015	2,050	59	-	15.2%	100%	1,129
		Average			**44.1%**	**483**		**96**	**0.4%**	**6.2%**	**100%**	**824**		**96**	**0.7%**	**6.7%**	**100%**	**945**
T3-3	81	50	2,211	2,723	83.3%	346	2,332	125	-14.4%	5.5%	100%	645	2,303	126	-15.4%	4.2%	100%	745
		100	2,211	Cannot find a soln.		264	2,413	89	-	9.1%	100%	1,023	2,398	88	-	8.5%	100%	1,186
		150	2,211	Cannot find a soln.		101	2,574	58	-	16.4%	100%	1,091	2,548	57	-	15.2%	100%	1,214
		200	2,211	Cannot find a soln.		38	2,606	51	-	17.9%	100%	1,341	2,611	45	-	18.1%	100%	1,303
		Average			**83.3%**	**187**		**81**	**-14.4%**	**12.2%**	**100%**	**1,025**		**79**	**-15.4%**	**11.5%**	**100%**	**1,112**

Table 61. Computational Results of GBD-2-1 and GBD-2-4 on T3 Instances – Random 3

Test Problem	|*N*|	|*K*|	Max. SP Length	CPLEX			GBD-2-1						GBD-2-4					
				LA	(UB-LB) /UB	CPU Time (s)	*LA*	# of Cuts	% Dev. from CPLEX	% Dev. from Max. SP Length	(UB-LB) /UB	CPU Time (s)	*LA*	# of Cuts	% Dev. from CPLEX	% Dev. from Max. SP Length	(UB-LB) /UB	CPU Time (s)
T3-1	52	50	1,518	1,518	59.6%	357	1,553	157	2.3%	2.3%	100%	516	1,518	155	0.0%	0.0%	100%	540
		100	1,518	1,534	40.5%	743	1,590	119	3.7%	4.7%	100%	761	1,589	112	3.6%	4.7%	100%	809
		150	1,518	Cannot find a soln.		306	1,807	80	-	19.0%	100%	853	1,811	80	-	19.3%	100%	915
		200	1,518	Cannot find a soln.		321	2,028	82	-	33.6%	100%	1,221	2,229	75	-	46.8%	100%	1,139
		Average			**50.0%**	**432**		**110**	**3.0%**	**14.9%**	**100%**	**838**		**106**	**1.8%**	**17.7%**	**100%**	**851**
T3-2	62	50	1,745	1,786	41.8%	850	1,845	141	3.3%	5.7%	100%	465	1,780	139	-0.3%	2.0%	100%	592
		100	1,779	Cannot find a soln.		1.309	2,004	88	-	12.6%	100%	702	1,970	96	-	10.7%	100%	885
		150	1,779	Cannot find a soln.		5.841	2,044	88	-	14.9%	100%	1,141	2,018	88	-	13.4%	100%	1,242
		200	1,779	Cannot find a soln.		499	2,219	58	-	24.7%	100%	1,042	2,086	58	-	17.3%	100%	1,124
		Average			**41.8%**	**2.125**		**94**	**3.3%**	**14.5%**	**100%**	**837**		**95**	**-0.3%**	**10.9%**	**100%**	**961**
T3-3	81	50	2,039	2,129	75.5%	173	2,092	124	-1.7%	2.6%	100%	693	2,078	127	-2.4%	1.9%	100%	848
		100	2,039	Cannot find a soln.		3.591	2,384	90	-	16.9%	100%	1,088	2,353	88	-	15.4%	100%	1,310
		150	2,039	Cannot find a soln.		71	2,571	55	-	26.1%	100%	1,068	2,497	56	-	22.5%	100%	1,253
		200	2,039	Cannot find a soln.		41	2,646	47	-	29.8%	100%	1,269	2,625	43	-	28.7%	100%	1,349
		Average			**75.5%**	**969**		**79**	**-1.7%**	**18.8%**	**100%**	**1,030**		**79**	**-2.4%**	**17.1%**	**100%**	**1,190**

Table 62. Computational Results of GBD-2-1 and GBD-2-4 on T3 Instances – Random 4

Test Problem	|*N*|	|*K*|	Max. SP Length	CPLEX			GBD-2-1						GBD-2-4					
				LA	(UB-LB) /UB	CPU Time (s)	*LA*	# of Cuts	% Dev. from CPLEX	% Dev. from Max. SP Length	(UB-LB) /UB	CPU Time (s)	*LA*	# of Cuts	% Dev. from CPLEX	% Dev. from Max. SP Length	(UB-LB) /UB	CPU Time (s)
T3-1	52	50	1,361	1,361	71.1%	351	1,361	157	0.0%	0.0%	100%	470	1,361	155	0.0%	0.0%	100%	521
		100	1,362	1,965	82.3%	144	1,608	120	-18.2%	18.1%	100%	771	1,607	115	-18.2%	18.0%	100%	815
		150	1,381	1,602	72.4%	1,515	1,740	79	8.6%	26.0%	100%	834	1,746	81	9.0%	26.4%	100%	890
		200	1,566	Cannot find a soln.		477	2,265	82	-	44.6%	100%	1,153	2,217	75	-	41.6%	100%	1,149
		Average			**75.2%**	**622**		**110**	**-3.2%**	**22.2%**	**100%**	**807**		**107**	**-3.1%**	**21.5%**	**100%**	**844**
T3-2	62	50	1,871	1,871	69.5%	119	1,880	139	0.5%	0.5%	100%	529	1,871	139	0.0%	0,0%	100%	563
		100	1,871	1,919	69.7%	572	1,892	95	-1.4%	1.1%	100%	750	1,948	96	1.5%	4,1%	100%	824
		150	1,871	Cannot find a soln.		313	2,052	87	-	9.7%	100%	1,085	2,046	87	-	9.4%	100%	1,231
		200	1,871	Cannot find a soln.		127	2,130	58	-	13.8%	100%	1,019	2,197	58	-	17.4%	100%	1,085
		Average			**69.6%**	**283**		**95**	**-0.5%**	**6.3%**	**100%**	**846**		**95**	**0.8%**	**7.7%**	**100%**	**926**
T3-3	81	50	2,036	2,356	82.3%	134	2,111	130	-10.4%	3.7%	100%	653	2,099	128	-10.9%	3.1%	100%	782
		100	2,036	Cannot find a soln.		3,047	2,296	90	-	12.8%	100%	1,110	2,217	87	-	8.9%	100%	1,201
		150	2,036	Cannot find a soln.		56	2,389	55	-	17.3%	100%	1,126	2,463	55	-	21.0%	100%	1,256
		200	2,101	Cannot find a soln.		49	2,602	49	-	23.8%	100%	1,392	2,632	44	-	25.3%	100%	1,485
		Average			**82.3%**	**822**		**81**	**-10.4%**	**14.4%**	**100%**	**1,070**		**79**	**-10.9%**	**14.6%**	**100%**	**1,181**

Table 63. Computational Results of GBD-2-1 and GBD-2-4 on T3 Instances – Random 5

				CPLEX			GBD-2-1						GBD-2-4					
Test Problem	**|*N*|**	**|*K*|**	**Max. SP Length**	***LA***	**(UB-LB) /UB**	**CPU Time (s)**	***LA***	**# of Cuts**	**% Dev. from CPLEX**	**% Dev. from Max. SP Length**	**(UB-LB) /UB**	**CPU Time (s)**	***LA***	**# of Cuts**	**% Dev. from CPLEX**	**% Dev. from Max. SP Length**	**(UB-LB) /UB**	**CPU Time (s)**
T3-1	52	50	1,313	1,313	45.2%	393	1,335	157	1.7%	1.7%	100%	487	1,313	154	0.0%	0.0%	100%	506
		100	1,554	1,629	64.3%	226	1,579	119	-3.1%	1.6%	100%	733	1,597	115	-2.0%	2.8%	100%	808
		150	1,554	1,554	53.7%	1,684	1,649	78	6.1%	6.1%	100%	751	1,677	82	7.9%	7.9%	100%	896
		200	1,554	Cannot find a soln.		274	1,756	82	-	13.0%	100%	1,120	1,733	76	-	11.5%	100%	1,148
		Average			**54.4%**	**644**		**109**	**1.6%**	**5.6%**	**100%**	**773**		**107**	**2.0%**	**5.6%**	**100%**	**840**
T3-2	62	50	1,777	1,777	62.8%	534	1,837	139	3.4%	3.4%	100%	514	1,821	140	2.5%	2.5%	100%	574
		100	1,777	1,793	63.1%	6,036	1,805	93	0.7%	1.6%	100%	727	1,889	95	5.4%	6.3%	100%	818
		150	1,777	Cannot find a soln.		336	2,090	88	-	17.6%	100%	1,063	2,010	88	-	13.1%	100%	1,197
		200	1,791	Cannot find a soln.		773	2,285	58	-	27.6%	100%	958	2,096	58	-	17.0%	100%	1,053
		Average			**63.0%**	**1,920**		**95**	**2.0%**	**12.5%**	**100%**	**816**		**95**	**3.9%**	**9.7%**	**100%**	**910**
T3-3	81	50	1,918	1,923	69.9%	5,335	2,002	127	4.1%	4.4%	100%	598	2,003	126	4.2%	4.4%	100%	829
		100	2,148	Cannot find a soln.		162	2,352	89	-	9.5%	100%	1,030	2,339	90	-	8.9%	100%	1,154
		150	2,148	Cannot find a soln.		138	2,498	57	-	16.3%	100%	1,108	2,470	56	-	15.0%	100%	1,182
		200	2,148	Cannot find a soln.		40	2,605	48	-	21.3%	100%	1,306	2,644	43	-	23.1%	100%	1,269
		Average			**69.9%**	**1,419**		**80**	**4.1%**	**12.9%**	**100%**	**1,010**		**79**	**4.2%**	**12.9%**	**100%**	**1,109**

APPENDIX – I

THE PSEUDO-CODES OF THE SUBROUTINES IN LNS-LA ALGORITHM

subroutine Initial Solution Formation
begin
  input: the physical network and all the parameters of model LA-CMNDP;
  $\mathbb{S} := \{\}$;
  for $k \in K$
    generate $\mathbb{P}^k = \{P^1, P^2, ..., P^{\mathbb{k}}\}$ by $\mathbb{k}$-shortest path algorithm;
  end;
  for $i = 1 : I$
    $S^i := \{\}$;
    for $k = 1 : K$
      $\mathscr{P}^k :=$ a randomly chosen path from set $\mathbb{P}^k$; $S^i := S^i \cup \{\mathscr{P}^k\}$;
    end;
    $\mathbb{S} := \mathbb{S} \cup S^i$;
  end;
  for $i = 1 : I$
    if S^i is not feasible with respect to constraints (4.8)-(4.9) of LA-CMNDP **then**
      $\mathbb{S} := \mathbb{S} \setminus S^i$;
    else
      calculate DA, DT, LA values and check feasibility with respect to constraints (4.1)-(4.3) & (4.5);
      if S^i is not feasible with respect to constraints (4.1)-(4.3) & (4.5) **then** $\mathbb{S} := \mathbb{S} \setminus S^i$;
    end;
  end;
  if $\mathbb{S}$ is not empty **then**
    $S^{best} := \min(LA(\mathbb{S}))$;
    return S^{best} ;
  else
    return "*no feasible solution is found*";
  end;
end;

Figure 73. The Pseudo-Code of Initial Solution Generation Subroutine

```
subroutine ready;
begin
    input: x, A^AC, N^AC;
    for i ∈ N^AC
        for (i, j) ∈ A^AC
            if Σ_{(m,i) ∈ A^AC} y_mi = 0 then
                ready(i, j) := 0;
            else
                for (i, j) ∈ A^AC
                    ready(i, j) := 0;
                    for (m, i) ∈ A^AC
                        if there are k ∈ K with x^k_mi = x^k_ij = 1 then
                            ready(i, j) := 1;
                        end;
                    end;
                end;
            end;
        end;
    end;
    return ready(i, j) for all (i, j) ∈ A^AC;
end;
```

Figure 74. The Pseudo-Code of Ready Subroutine

subroutine *Latest Arrival*;
begin

  input: x ; $A^{AC} := \{(i,j) \in A : x_{ij}^{k}=1\}$; $N^{AC} := \{\}$;

  for all $(i,j) \in A^{AC}$ calculate *ready* values and determine feasibility;

  if feasibility is not violated

    for $i = O(k)$, $k \in K$

      if $(i,j) \in A^{AC}$ and *ready(i,j)*=1 **then**

        $DT_{ij} := \delta_i$; $DA_j^k := DT_{ij} + t_{ij}$ for k with $x_{ij}^k = 1$;

        drop (i,j) from A^{AC} ; add node *j* to N^{AC} ;

      end;

    end;

  end;

  for $i \in N^{AC}$

      for $(i,j) \in A^{AC}$ update *ready* values and determine feasibility;

  end;

  while $N^{AC} \neq \{\}$ **and** feasibility is not violated **do**

    for $i \in N^{AC}$

      for $(i,j) \in A^{AC}$

        if *ready(i,j)*=1 **then**

          $DT_{ij} := \max_{k \in K}(DA_i^k) + \delta_i$;

          for $k \in K$ with $x_{ij}^k = 1$

            $DA_j^k := DT_{ij} + t_{ij}$;

          end;

          drop arc (i,j) from A^{AC} ; add node *j* to N^{AC} ;

        end;

      end;

      if there is no emanating arc $(i,j) \in A^{AC}$ from node *i* **then** drop node *i* from N^{AC} ;

    end;

    for $(i,j) \in A^{AC}$ update *ready* values and determine feasibility;

  end;

  if $N^{AC} = \{\}$ **then**

    x is feasible; $LA := \max_{k \in K} DA_{D(k)}^k$;

  else *x* is infeasible;

  end;

  return feasibility of *x*;

  if *x* is feasible **then return** LA , DA , DT ;

  end;

end;

Figure 75. The Pseudo-Code of Latest Arrival Subroutine

subroutine *Find LA-Path;*
begin
  input: x , LA , DA , DT ;
  arrival := LA;
  $A^{LA} := \{\}$; $X^{LA} := \{\}$; $K^{LA} := \{\}$; $K^{CANDIDATE} := \{\}$;
  $NEXT$:= 0; i=1;
  while $NEXT$ = 0 **do**
    if $\max_k DA_i^k$ = arrival **then**
      $NEXT$:= i;
    else
      i := i+1;
    end;
  end;
  for $i \in N$
    if $DT_{i\,NEXT}$ = arrival $- t_{i\,NEXT}$ **then**
      arrival $:= DT_{i\,NEXT}$;
      $A^{LA} := A^{LA} \cup (i, NEXT)$;
      for $x_{i\,NEXT}^k = 1$
        $X^{LA} := X^{LA} \cup x_{i\,NEXT}^k$;
        $K^{LA} := K^{LA} \cup k$; $K^{CANDIDATE} := K^{CANDIDATE} \cup k$;
      end;
      $NEXT$:= i ;
    end;
  end;
  while arrival > 0
    for $i \in N$
      if $DT_{i\,NEXT}$ = arrival $- t_{i\,NEXT}$ **then**
        if there is at least one $k \in K^{CANDIDATE}$ with $x_{i\,NEXT}^k = 1$ **then**
          arrival $:= DT_{i\,NEXT}$;
          $A^{LA} := A^{LA} \cup (i, NEXT)$; $K^{CANDIDATE} := \{\}$;
          for $x_{i\,NEXT}^k = 1$
            $X^{LA} := X^{LA} \cup x_{i\,NEXT}^k$;
            $K^{LA} := K^{LA} \cup k$; $K^{CANDIDATE} := K^{CANDIDATE} \cup k$;
          end;
          $NEXT$:= i ;
        end;
      end;
    end;
  end;
  return A^{LA} , X^{LA} , K^{LA} ;
end;

Figure 76. The Pseudo-Code of Find LA-Path Subroutine

subroutine *Path Change*;

begin

input: p^C and p^N of commodity k;

$A^{REM} := \{\}$; $N^{REM} := \{\}$;

$A^{ADD} := \{\}$; $N^{ADD} := \{\}$;

$i := O(k)$;

repeat until $i \notin p^N$

i_next := the successor node of i on p^C;

j_next := the successor node of i on p^N;

if $i\_next = j\_next$ **then**

$i := i\_next$;

else

$diff := i$;

end;

end;

put all forward arcs of p^C that emanates from node *diff* to A^{REM};

put *diff* and all forward nodes that succeeds *diff* on p^C to N^{REM};

put all forward arcs of p^N that emanates from node *diff* to A^{ADD};

put *diff* and all forward nodes that succeeds *diff* on p^N to N^{ADD};

return A^{REM}, N^{REM}, A^{ADD}, N^{ADD};

end;

Figure 77. The Pseudo-Code of Path Change Subroutine

subroutine *Remove Path*;
begin
input: x, LA, DA, DT, and A^{REM}, N^{REM} for commodity k;
$A^{UP} := \{\}$; $N^{UP} := \{\}$; $K^{UP} := \{\}$;
for $(i,j) \in A^{REM}$ $x_{ij}^{k} := 0$;
for $(i,j) \in A^{REM}$
if $DA_i^k > DA_i^p$ for all $p \in K$ with $x_{ij}^p = 1$ **then**
add (i,j) to A^{UP}; add nodes i and j to N^{UP};
add p with $x_{ij}^p = 1$ to K^{UP};
end;
end;
for $j \in N^{REM}$ **if** $j \neq diff$ **then** $DA_j^k := 0$;
while there is an emanating arc from $i \in N^{UP}$ not analyzed yet **do**
for $m \in N \setminus N^{UP}$
if $x_{im}^p = 1$ and $p \in K^{UP}$
add node m to N^{UP}; add arc (i,m) to A^{UP};
add all commodities p flowing on arc (i,m) to K^{UP};
end;
end;
end;
for $(i,j) \in A^{UP}$ update *ready* values;
$N^{UP} := \{\}$; add all nodes $i \in N$ that have an emanating arc $(i,j) \in A^{UP}$ with *ready*(i,j)=1 to N^{UP};
while $N^{UP} \neq \{\}$ **do**
for $i \in N^{UP}$
for $(i,j) \in A^{UP}$
if *ready*(i,j)=1
$DT_{ij} := \max_k (DA_i^k) + \delta_i$;
for $x_{ij}^p = 1$ $DA_j^p := DT_{ij} + t_{ij}$;
drop arc (i,j) from A^{UP}; add node j to N^{UP};
end;
if there is no emanating arc $(i,j) \in A^{UP}$ from node i **then** drop node i from N^{UP};
end;
end;
for $(i,j) \in A^{UPDATE}$ update *ready* values;
end;
return updated x, DA, DT;
end;

Figure 78. The Pseudo-Code of Remove Path Subroutine

subroutine *Add Path*;
begin
    input: updated x , DA , DT and A^{ADD} , N^{ADD} for commodity k ;
    for $(i, j) \in A^{ADD}$ $x_{ij}^{k} := 1$; $A^{UP} := A^{ADD}$; $N^{UP} := N^{ADD}$;
    put all commodities flowing on $(i, j) \in A^{ADD}$ to K^{UP} ;
    while there is an emanating arc from $i \in N^{UP}$ not analyzed yet **do**
        for $m \in N \setminus N^{UP}$
            if $x_{im}^{p} = 1$ and $p \in K^{UP}$
                add node m to N^{UP} ; add arc (i, m) to A^{UP} ;
                add all commodities p flowing on arc (i, m) to K^{UP} ;
            end;
        end;
    end;
    for $(i, j) \in A^{UP}$ update *ready* values and determine feasibility;
    $N^{UP} := \{\}$;add all nodes $i \in N$ that have an emanating arc $(i, j) \in A^{UP}$ with *ready*(i,j)=1 to N^{UP} ;
    while $N^{UP} \neq \{\}$ and feasibility is not violated **do**
        for $i \in N^{UP}$
            for $(i, j) \in A^{UP}$
                if *ready*(i,j)=1 **then**
                    $DT_{ij} := \max_{k}(DA_{i}^{k}) + \delta_{i}$;
                    for $x_{ij}^{p} = 1$ $DA_{j}^{p} := DT_{ij} + t_{ij}$;
                    drop arc (i, j) from A^{UP}; add node j to N^{UP} ;
                end;
            end;
            if there is no emanating arc $(i, j) \in A^{UP}$ from node i **then** drop node i from N^{UP} ;
        end;
        for $(i, j) \in A^{UP}$ update *ready* values and determine feasibility;
    end;
    if $A^{UP} = \{\}$ **then**
        updated x is feasible;
        $LA := \max_{k \in K} DA_{D(k)}^{k}$;
    else
        updated x is infeasible;
    end;
    return feasibility of updated x;
    if updated x is feasible **then return** updated x, , LA, DA , DT ;
    end;
end;

Figure 79. The Pseudo-Code of Add Path Subroutine

APPENDIX – J

COMPARISON OF LNS-LA WITH GBD-2-4 ALGORITHM

The tables of Appendix J are presented starting on the next page.

Table 64. The Comparison of LNS-LA with GBD-2-4 on T2 Instances – Random 1

			CPLEX			GBD-2-4					LNS-LA			
Test Problem	**|*N*|**	**|*K*|**	***LA***	**(UB-LB)/ UB**	**CPU Time (s)**	***LA***	**# of Cuts**	**% Dev. from CPLEX**	**(UB-LB)/ UB**	**CPU Time (s)**	***LA***	**% Dev. from CPLEX**	**% Dev. From GBD-2-4**	**CPU Time (s)**
T2-1	21	25	1,028	0.0%	3.0	1,028	15	0.0%	0.0%	4.0	1,028	0.0%	0.0%	0.8
		50	1,334	0.0%	13.5	1,334	3	0.0%	0.0%	1.8	1,334	0.0%	0.0%	1.4
		75	1,382	0.0%	31.6	1,382	19	0.0%	0.0%	12.8	1,382	0.0%	0.0%	2.4
		100	1,522	0.0%	33.1	1,522	48	0.0%	0.0%	46.5	1,522	0.0%	0.0%	3.5
		125	1,522	0.0%	36.1	1,522	65	0.0%	0.0%	78.8	1,522	0.0%	0.0%	6.7
		150	1,522	0.0%	109.6	1,522	61	0.0%	0.0%	90.1	1,522	0.0%	0.0%	7.1
		Average		**0.0%**	**37.8**		**35**	**0.0%**	**0.0%**	**39.0**		**0.0%**	**0.0%**	**3.6**
T2-2	25	25	969	0.0%	1.5	969	5	0.0%	0.0%	1.6	969	0.0%	0.0%	0.6
		50	969	0.0%	23.5	969	10	0.0%	0.0%	5.4	969	0.0%	0.0%	1.9
		75	969	0.0%	5.6	969	25	0.0%	0.0%	19.6	969	0.0%	0.0%	2.6
		100	969	0.0%	36.0	969	180	0.0%	100.0%	233.5	969	0.0%	0.0%	3.7
		125	1,095	0.0%	80.7	1,095	173	0.0%	100.0%	270.3	1,095	0.0%	0.0%	5.0
		150	1,095	0.0%	1,013.8	1,134	166	3.6%	100.0%	339.3	1,095	0.0%	-3.4%	5.5
		Average		**0.0%**	**193.5**		**93**	**0.6%**	**50.0%**	**144.9**		**0.0%**	**-0.6%**	**3.2**
T2-3	33	25	1,082	0.0%	11.81	1,082	20	0.0%	0.0%	8.5	1,082	0.0%	0.0%	2.3
		50	1,177	0.0%	649.7	1,177	96	0.0%	0.0%	88.3	1,177	0.0%	0.0%	4.4
		75	1,177	25.5%	283.0	1,186	173	0.8%	100.0%	282.8	1,177	0.0%	-0.8%	6.6
		100	1,181	26.8%	451.8	1,264	164	7.0%	100.0%	390.9	1,177	-0.3%	-6.9%	9.0
		125	1,177	21.1%	3,40.2	1,375	153	16.8%	100.0%	450.3	1,177	0.0%	-14.4%	12.4
		150	1,208	26.5%	1,875.6	1,382	149	14.4%	100.0%	551.3	1,208	0.0%	-12.6%	16.2
		Average		**16.6%**	**1,068.7**		**126**	**6.5%**	**66.7%**	**295.3**		**-0.1%**	**-5.8%**	**8.5**

Table 65. The Comparison of LNS-LA with GBD-2-4 on T2 Instances – Random 2

			CPLEX			GBD-2-4					LNS-LA			
Test Problem	**|*N*|**	**|*K*|**	***LA***	**(UB-LB)/ UB**	**CPU Time (s)**	***LA***	**# of Cuts**	**% Dev. from CPLEX**	**(UB-LB)/ UB**	**CPU Time (s)**	***LA***	**% Dev. from CPLEX**	**% Dev. From GBD-2-4**	**CPU Time (s)**
T2-1	21	25	977	0.0%	1.3	977	5	0.0%	0.0%	1.1	977	0.0%	0.0%	0.7
		50	977	0.0%	23.4	977	7	0.0%	0.0%	3.1	977	0.0%	0.0%	1.4
		75	977	0.0%	25.9	977	20	0.0%	0.0%	12.5	977	0.0%	0.0%	2.0
		100	977	0.0%	61.6	998	180	2.1%	100.0%	204.5	977	0.0%	-2.1%	3.1
		125	1,096	0.0%	55.0	1,112	178	1.5%	100.0%	248.4	1,096	0.0%	-1.4%	4.3
		150	1,096	0.0%	83.8	1,119	171	2.1%	100.0%	300.6	1,096	0.0%	-2.1%	6.0
		Average		**0.0%**	**41.8**		**94**	**1.0%**	**50.0%**	**128.4**		**0.0%**	**-0.9%**	**2.9**
T2-2	25	25	862	0.0%	1.6	862	6	0.0%	0.0%	1.7	862	0.0%	0.0%	1.0
		50	895	0.0%	17.0	895	91	0.0%	0.0%	52.4	895	0.0%	0.0%	2.1
		75	999	0.0%	21.3	999	102	0.0%	0.0%	85.1	999	0.0%	0.0%	3.4
		100	999	0.0%	35.3	1,044	181	4.5%	100.0%	222.2	999	0.0%	-4.3%	4.2
		125	1,095	0.0%	71.4	1,095	128	0.0%	0.0%	200.6	1,095	0.0%	0.0%	5.6
		150	1,114	0.0%	74.7	1,114	9	0.0%	0.0%	14.7	1,114	0.0%	0.0%	6.2
		Average		**0.0%**	**36.9**		**86**	**0.8%**	**16.7%**	**96.1**		**0.0%**	**-0.7%**	**3.7**
T2-3	33	25	1,112	0.0%	26.7	1,112	42	0.0%	0.0%	17.4	1,112	0.0%	0.0%	2.7
		50	1,208	0.0%	231.7	1,208	186	0.0%	100.0%	165.5	1,208	0.0%	0.0%	4.3
		75	1,208	26.9%	361.7	1,269	174	5.0%	100.0%	248.8	1,208	0.0%	-4.8%	8.1
		100	1,208	5.1%	3,476.6	1,291	165	6.9%	100.0%	369.4	1,208	0.0%	-6.4%	9.1
		125	1,208	19.0%	1,958.0	1,313	153	8.7%	100.0%	414.8	1,208	0.0%	-8.0%	14.5
		150	1,208	15.0%	3,868.9	1,353	148	12.0%	100.0%	497.0	1,208	0.0%	-10.7%	14.5
		Average		**11.0%**	**1,653.9**		**145**	**5.4%**	**83.3%**	**285.5**		**0.0%**	**-5.0%**	**8.8**

Table 66. The Comparison of LNS-LA with GBD-2-4 on T2 Instances – Random 3

			CPLEX			GBD-2-4					LNS-LA			
Test Problem	**|*N*|**	**|*K*|**	***LA***	**(UB-LB)/ UB**	**CPU Time (s)**	***LA***	**# of Cuts**	**% Dev. from CPLEX**	**(UB-LB)/ UB**	**CPU Time (s)**	***LA***	**% Dev. from CPLEX**	**% Dev. From GBD-2-4**	**CPU Time (s)**
T2-1	21	25	948	0.0%	2.6	948	9	0.0%	0.0%	2.0	948	0.0%	0.0%	0.9
		50	1,109	0.0%	4.7	1,109	7	0.0%	0.0%	3.4	1,109	0.0%	0.0%	1.4
		75	1,198	0.0%	9.0	1,198	5	0.0%	0.0%	3.6	1,198	0.0%	0.0%	2.0
		100	1,375	0.0%	33.4	1,375	50	0.0%	0.0%	46.3	1,375	0.0%	0.0%	2.7
		125	1,515	0.0%	70.3	1,515	40	0.0%	0.0%	48.7	1,515	0.0%	0.0%	4.0
		150	1,515	0.0%	73.3	1,515	29	0.0%	0.0%	41.4	1,515	0.0%	0.0%	6.0
		Average		**0.0%**	**32.2**		**23**	**0.0%**	**0.0%**	**24.2**		**0.0%**	**0.0%**	**2.8**
T2-2	25	25	847	0.0%	1.41	847	3	0.0%	0.0%	0.9	847	0.0%	0.0%	1.1
		50	1,002	0.0%	16.7	1,002	44	0.0%	0.0%	22.8	1,002	0.0%	0.0%	2.1
		75	1,005	0.0%	69.0	1,005	75	0.0%	0.0%	59.9	1,005	0.0%	0.0%	3.3
		100	1,095	0.0%	104.0	1,095	181	0.0%	100.0%	205.2	1,095	0.0%	0.0%	4.5
		125	1,095	0.0%	389.1	1,106	172	1.0%	100.0%	272.8	1,095	0.0%	-1.0%	5.6
		150	1,095	0.0%	162.1	1,110	167	1.4%	100.0%	473.0	1,095	0.0%	-1.4%	6.8
		Average		**0.0%**	**123.7**		**107**	**0.4%**	**50.0%**	**172.4**		**0.0%**	**-0.4%**	**3.9**
T2-3	33	25	1,043	0.0%	17.5	1,043	105	0.0%	0.0%	46.3	1,043	0.0%	0.0%	3.3
		50	1,043	0.0%	212.1	1,069	185	2.5%	100.0%	177.3	1,043	0.0%	-2.4%	5.0
		75	1,125	14.5%	255.8	1,125	173	0.0%	100.0%	252.5	1,125	0.0%	0.0%	6.1
		100	1,125	4.8%	449.1	1,224	165	8.8%	100.0%	339.5	1,125	0.0%	-8.1%	7.6
		125	1,161	7.8%	749.4	1,263	155	8.8%	100.0%	406.1	1,161	0.0%	-8.1%	14.2
		150	1,162	7.8%	3,701.4	1,314	147	13.1%	100.0%	473.0	1,162	0.0%	-11.6%	14.0
		Average		**5.8%**	**897.5**		**155**	**5.5%**	**83.3%**	**282.4**		**0.0%**	**-5.0%**	**8.3**

Table 67. The Comparison of LNS-LA with GBD-2-4 on T2 Instances – Random 4

			CPLEX			GBD-2-4					LNS-LA			
Test Problem	|*N*|	|*K*|	*LA*	(UB-LB)/ UB	CPU Time (s)	*LA*	# of Cuts	% Dev. from CPLEX	(UB-LB)/ UB	CPU Time (s)	*LA*	% Dev. from CPLEX	% Dev. From GBD-2-4	CPU Time (s)
T2-1	21	25	873	0.0%	2.3	873	19	0.0%	0.0%	4.1	873	0.0%	0.0%	1.0
		50	993	0.0%	26.2	993	14	0.0%	0.0%	6.5	993	0.0%	0.0%	1.8
		75	1,005	0.0%	30.5	1,005	20	0.0%	0.0%	14.0	1,005	0.0%	0.0%	2.3
		100	1,073	0.0%	80.5	1,073	39	0.0%	0.0%	35.7	1,073	0.0%	0.0%	2.8
		125	1,073	0.0%	101.7	1,073	100	0.0%	0.0%	129.3	1,073	0.0%	0.0%	3.6
		150	1,073	0.0%	231.6	1,119	171	4.3%	100.0%	286.1	1,073	0.0%	-4.1%	4.4
		Average		**0.0%**	**78.8**		**61**	**0.7%**	**16.7%**	**79.3**		**0.0%**	**-0.7%**	**2.7**
T2-2	25	25	809	0.0%	6.7	809	10	0.0%	0.0%	2.6	809	0.0%	0.0%	1.4
		50	969	0.0%	4.5	969	29	0.0%	0.0%	15.4	969	0.0%	0.0%	2.4
		75	969	0.0%	23.3	979	186	1.0%	100.0%	181.3	969	0.0%	-1.0%	3.3
		100	1,070	0.0%	111.3	1,095	179	2.3%	100.0%	237.9	1,070	0.0%	-2.3%	5.2
		125	1,070	0.0%	648.3	1,124	171	5.0%	100.0%	292.7	1,070	0.0%	-4.8%	7.8
		150	1,086	0.0%	3,098.0	1,134	165	4.4%	100.0%	351.3	1,086	0.0%	-4.2%	8.2
		Average		**0.0%**	**648.7**		**123**	**2.1%**	**66.7%**	**180.2**		**0.0%**	**-2.1%**	**4.7**
T2-3	33	25	1,052	0.0%	16.6	1,052	50	0.0%	0.0%	21.9	1,052	0.0%	0.0%	2.9
		50	1,166	0.0%	575.4	1,166	181	0.0%	0.0%	187.6	1,166	0.0%	0.0%	4.3
		75	1,166	13.4%	373.4	1,259	173	8.0%	100.0%	275.8	1,166	0.0%	-7.4%	6.6
		100	1,166	2.2%	975.6	1,346	165	15.4%	100.0%	363.2	1,166	0.0%	-13.4%	11.8
		125	1,166	4.6%	2,361.6	1,378	156	18.2%	100.0%	409.9	1,166	0.0%	-15.4%	15.6
		150	1,497	44.2%	157.7	1,365	147	-8.8%	100.0%	471.5	1,208	-19.3%	-11.5%	16.3
		Average		**10.7%**	**743.4**		**145**	**5.5%**	**66.7%**	**288.3**		**-3.2%**	**-7.9%**	**9.6**

Table 68. The Comparison of LNS-LA with GBD-2-4 on T2 Instances – Random 5

			CPLEX			GBD-2-4					LNS-LA			
Test Problem	**|*N*|**	**|*K*|**	***LA***	**(UB-LB)/ UB**	**CPU Time (s)**	***LA***	**# of Cuts**	**% Dev. from CPLEX**	**(UB-LB)/ UB**	**CPU Time (s)**	***LA***	**% Dev. from CPLEX**	**% Dev. From GBD-2-4**	**CPU Time (s)**
T2-1	21	25	1,153	0.0%	5.5	1,153	28	0.0%	0.0%	5.7	1,153	0.0%	0.0%	1.1
		50	1,153	0.0%	27.7	1,153	69	0.0%	0.0%	29.6	1,153	0.0%	0.0%	1.7
		75	1,154	0.0%	37.6	1,154	84	0.0%	0.0%	58.5	1,154	0.0%	0.0%	2.5
		100	1,154	0.0%	73.4	1,160	180	0.5%	100.0%	199.9	1,154	0.0%	-0.5%	4.3
		125	1,515	0.0%	169.5	1,515	102	0.0%	0.0%	122.5	1,515	0.0%	0.0%	5.6
		150	1,515	0.0%	323.7	1,515	70	0.0%	0.0%	90.1	1,515	0.0%	0.0%	6.0
		Average		**0.0%**	**106.2**		**89**	**0.1%**	**16.7%**	**84.4**		**0.0%**	**-0.1%**	**3.5**
T2-2	25	25	830	0.0%	10.7	830	30	0.0%	0.0%	8.8	830	0.0%	0.0%	1.2
		50	1,095	0.0%	27.4	1,095	53	0.0%	0.0%	27.2	1,095	0.0%	0.0%	2.6
		75	1,095	0.0%	44.6	1,095	103	0.0%	0.0%	85.5	1,095	0.0%	0.0%	3.5
		100	1,095	0.0%	89.2	1,095	178	0.0%	0.0%	233.6	1,095	0.0%	0.0%	4.5
		125	1,095	0.0%	116.0	1,106	172	1.0%	100.0%	281.7	1,095	0.0%	-1.0%	6.0
		150	1,095	0.0%	549.4	1,154	165	5.4%	100.0%	345.7	1,095	0.0%	-5.1%	7.7
		Average		**0.0%**	**139.5**		**117**	**1.1%**	**33.3%**	**163.7**		**0.0%**	**-1.0%**	**4.2**
T2-3	33	25	978	0.0%	20.3	978	63	0.0%	0.0%	30.0	978	0.0%	0.0%	2.3
		50	1,009	0.0%	126.3	1,047	186	3.8%	100.0%	194.9	1,009	0.0%	-3.6%	4.2
		75	1,035	0.0%	146.5	1,169	173	12.9%	100.0%	295.2	1,035	0.0%	-11.5%	6.1
		100	1,123	0.0%	68.8	1,179	165	5.0%	100.0%	361.3	1,123	0.0%	-4.7%	7.0
		125	1,140	1.5%	365.1	1,317	155	15.5%	100.0%	426.2	1,140	0.0%	-13.4%	9.0
		150	1,368	39.6%	149.5	1,354	145	-1.0%	100.0%	472.2	1,198	-12.4%	-11.5%	12.0
		Average		**6.9%**	**146.1**		**148**	**6.0%**	**83.3%**	**296.6**		**-2.1%**	**-7.5%**	**6.8**

Table 69. The Comparison of LNS-LA with GBD-2-4 on T3 Instances – Random 1

Test Problem	$\lvert N\rvert$	$\lvert K\rvert$	Max. SP Length	CPLEX *LA*	CPLEX (UB-LB) /UB	CPLEX CPU Time (s)	GBD-2-4 *LA*	GBD-2-4 # of Cuts	GBD-2-4 % Dev. from CPLEX	GBD-2-4 % Dev. from Max.SP Length	GBD-2-4 (UB-LB) /UB	GBD-2-4 CPU Time (s)	LNS-LA *LA*	LNS-LA % Dev. from CPLEX	LNS-LA % Dev. From GBD-2-4	LNS-LA % Dev. from Max.SP Length	LNS-LA CPU Time (s)
T3-1	52	50	1,216	1,235	48.3%	749	1,234	157	-0.1%	1.5%	100%	499	1,234	-0.1%	0.0%	1.5%	10
		100	1,301	1,571	54.0%	868	1,571	115	0.0%	20.8%	100%	829	1,571	0.0%	0.0%	20.8%	18
		150	1,316	1,571	55.3%	5,846	1,776	79	13.0%	35.0%	100%	926	1,571	0.0%	-11.5%	19.4%	27
		200	1,316	Cannot find a soln.		1,401	1,811	74	-	37.6%	100%	1,153	1,597	-	-11.8%	21.4%	35
		Average			**52.6%**	**2,216**		**106**	**4.3%**	**23.7%**	**100%**	**852**		**0.0%**	**-5.8%**	**15.7%**	**23**
T3-2	62	50	1,711	1,816	59.8%	615	1,785	76	-1.7%	4.3%	100%	580	1,744	-4.0%	-2.3%	1.9%	17
		100	1,871	Cannot find a soln.		2,279	1,990	95	-	6.4%	100%	877	1,871	-	-6.0%	0.0%	29
		150	1,871	Cannot find a soln.		8,440	2,043	88	-	9.2%	100%	1,245	1,871	-	-8.4%	0.0%	41
		200	1,871	Cannot find a soln.		125	2,135	58	-	14.1%	100%	1,136	1,871	-	-12.4%	0.0%	58
		Average			**59.8%**	**2,865**		**79**	**-1.7%**	**8.5%**	**100%**	**959**		**-4.0%**	**-7.3%**	**0.5%**	**37**
T3-3	81	50	2,070	2,070	61.6%	731	2,158	127	4.3%	4.3%	100%	839	2,070	0.0%	-4.1%	0.0%	30
		100	2,070	Cannot find a soln.		446	2,404	89	-	16.1%	100%	1,236	2,070	-	-13.9%	0.0%	60
		150	2,070	Cannot find a soln.		394	2,571	56	-	24.2%	100%	1,299	2,156	-	-16.1%	4.2%	92
		200	2,090	Cannot find a soln.		268	2,681	43	-	28.3%	100%	1,376	2,216	-	-17.3%	6.0%	110
		Average			**61.6%**	**460**		**79**	**4.3%**	**18.2%**	**100%**	**1,187**		**0.0%**	**-12.9%**	**2.5%**	**73**

Table 70. The Comparison of LNS-LA with GBD-2-4 on T3 Instances – Random 2

				CPLEX			GBD-2-4						LNS-LA				
Test Problem	$\|N\|$	$\|K\|$	Max. SP Length	*LA*	(UB-LB) /UB	CPU Time (s)	*LA*	# of Cuts	% Dev. from CPLEX	% Dev. from Max.SP Length	(UB-LB) /UB	CPU Time (s)	*LA*	% Dev. from CPLEX	% Dev. From GBD-2-4	% Dev. from Max.SP Length	CPU Time (s)
T3-1	52	50	1,297	1,297	52.3%	612	1,336	155	3.0%	3.0%	100%	528	1,297	0.0%	-2.9%	0.0%	14
		100	1,297	1,319	30.8%	722	1,555	114	17.9%	19.9%	100%	806	1,297	-1.7%	-16.6%	0.0%	23
		150	1,316	Cannot find a soln.		2,588	1,811	79	-	37.6%	100%	901	1,593	-	-12.0%	21.0%	27
		200	1,316	Cannot find a soln.		303	1,906	75	-	44.8%	100%	1,141	1,658	-	-13.0%	26.0%	39
		Average			**41.5%**	**1,056**		**106**	**10.4%**	**26.3%**	**100%**	**844**		**-0.8%**	**-11.1%**	**11.8%**	**26**
T3-2	62	50	1,779	1,779	53.0%	330	1,779	140	0.0%	0.0%	100%	605	1,779	0.0%	0.0%	0.0%	12
		100	1,779	1,782	35.2%	813	1,806	97	1.3%	1.5%	100%	857	1,779	-0.2%	-1.5%	0.0%	24
		150	1,779	Cannot find a soln.		277	1,961	88	-	10.2%	100%	1,190	1,779	-	-9.3%	0.0%	38
		200	1,779	Cannot find a soln.		510	2,050	59	-	15.2%	100%	1,129	1,779	-	-13.2%	0.0%	49
		Average			**44.1%**	**483**		**96**	**0.7%**	**6.7%**	**100%**	**945**		**-0.1%**	**-6.0%**	**0.0%**	**31**
T3-3	81	50	2,211	2,723	83.3%	346	2,303	126	-15.4%	4.2%	100%	745	2,211	-18.8%	-4.0%	0.0%	26
		100	2,211	Cannot find a soln.		264	2,398	88	-	8.5%	100%	1,186	2,211	-	-7.8%	0.0%	56
		150	2,211	Cannot find a soln.		101	2,548	57	-	15.2%	100%	1,214	2,339	-	-8.2%	5.8%	102
		200	2,211	Cannot find a soln.		38	2,611	45	-	18.1%	100%	1,303	2,394	-	-8.3%	8.3%	114
		Average			**83.3%**	**187**		**79**	**-15.4%**	**11.5%**	**100%**	**1,112**		**-18.8%**	**-7.1%**	**3.5%**	**74**

Table 71. The Comparison of LNS-LA on GBD-2-4 Instances – Random 3

				CPLEX			GBD-2-4						LNS-LA				
Test Problem	**|*N*|**	**|*K*|**	**Max. SP Length**	***LA***	**(UB-LB) /UB**	**CPU Time (s)**	***LA***	**# of Cuts**	**% Dev. from CPLEX**	**% Dev. from Max.SP Length**	**(UB-LB) /UB**	**CPU Time (s)**	***LA***	**% Dev. from CPLEX**	**% Dev. From GBD-2-4**	**% Dev. from Max.SP Length**	**CPU Time (s)**
T3-1	52	50	1,518	1,518	59.6%	357	1,518	155	0.0%	0.0%	100%	540	1,518	0.0%	0.0%	0.0%	24
		100	1,518	1,534	40.5%	743	1,589	112	3.6%	4.7%	100%	809	1,518	-1.0%	-4.5%	0.0%	20
		150	1,518	Cannot find a soln.		306	1,811	80	-	19.3%	100%	915	1,593	-	-12.0%	4.9%	26
		200	1,518	Cannot find a soln.		321	2,229	75	-	46.8%	100%	1,139	1,658	-	-25.6%	9.2%	40
		Average			**50.0%**	**432**		**106**	**1.8%**	**17.7%**	**100%**	**851**		**-0.5%**	**-10.5%**	**3.5%**	**27**
T3-2	62	50	1,745	1,786	41.8%	850	1,780	139	-0.3%	2.0%	100%	592	1,745	-2.3%	-2.0%	0.0%	14
		100	1,779	Cannot find a soln.		1,309	1,970	96	-	10.7%	100%	885	1,782	-	-9.5%	0.2%	33
		150	1,779	Cannot find a soln.		5,841	2,018	88	-	13.4%	100%	1,242	1,845	-	-8.6%	3.7%	48
		200	1,779	Cannot find a soln.		499	2,086	58	-	17.3%	100%	1,124	1,893	-	-9.3%	6.4%	75
		Average			**41.8%**	**2,125**		**95**	**-0.3%**	**10.9%**	**100%**	**961**		**-2.3%**	**-7.3%**	**2.6%**	**43**
T3-3	81	50	2,039	2,129	75.5%	173	2,078	127	-2.4%	1.9%	100%	848	2,039	-4.2%	-1.9%	0.0%	27
		100	2,039	Cannot find a soln.		3,591	2,353	88	-	15.4%	100%	1,310	2,156	-	-8.4%	5.7%	65
		150	2,039	Cannot find a soln.		71	2,497	56	-	22.5%	100%	1,253	2,156	-	-13.7%	5.7%	93
		200	2,039	Cannot find a soln.		41	2,625	43	-	28.7%	100%	1,349	2,323	-	-11.5%	13.9%	129
		Average			**75.5%**	**969**		**79**	**-2.4%**	**17.1%**	**100%**	**1,190**		**-4.2%**	**-8.9%**	**6.4%**	**78**

Table 72. The Comparison of LNS-LA with GBD-2-4 on T3 Instances – Random 4

				CPLEX			GBD-2-4						LNS-LA				
Test Problem	**|*N*|**	**|*K*|**	**Max. SP Length**	***LA***	**(UB-LB) /UB**	**CPU Time (s)**	***LA***	**# of Cuts**	**% Dev. from CPLEX**	**% Dev. from Max.SP Length**	**(UB-LB) /UB**	**CPU Time (s)**	***LA***	**% Dev. from CPLEX**	**% Dev. From GBD-2-4**	**% Dev. from Max.SP Length**	**CPU Time (s)**
T3-1	52	50	1,361	1,361	71.1%	351	1,361	155	0.0%	0,0%	100%	521	1,361	0.0%	0.0%	0.0%	12
		100	1,362	1,965	82.3%	144	1,607	115	-18.2%	18.0%	100%	815	1,379	-29.8%	-14.2%	1.2%	24
		150	1,381	1,602	72.4%	1,515	1,746	81	9.0%	26.4%	100%	890	1,409	-12.0%	-19.3%	2.0%	37
		200	1,566	Cannot find a soln.		477	2,217	75	-	41.6%	100%	1,149	1,597	-	-28.0%	2.0%	44
		Average			**75,2%**	**622**		**107**	**-3.1%**	**21.5%**	**100%**	**844**		**-14.0%**	**-15.4%**	**1.3%**	**29**
T3-2	62	50	1,871	1,871	69.5%	119	1,871	139	0.0%	0.0%	100%	563	1,871	0.0%	0.0%	0.0%	14
		100	1,871	1,919	69.7%	572	1,948	96	1.5%	4.1%	100%	824	1,874	-2.3%	-3.8%	0.2%	27
		150	1,871	Cannot find a soln.		313	2,046	87	-	9.4%	100%	1,231	1,874	-	-8.4%	0.2%	39
		200	1,871	Cannot find a soln.		127	2,197	58	-	17.4%	100%	1,085	1,893	-	-13.8%	1.2%	52
		Average			**69.6%**	**283**		**95**	**0.8%**	**7.7%**	**100%**	**926**		**-1.2%**	**-6.5%**	**0.4%**	**33**
T3-3	81	50	2,036	2,356	82.3%	134	2,099	128	-10.9%	3,1%	100%	782	2,080	-11.7%	-0.9%	2,2%	24
		100	2,036	Cannot find a soln.		3,047	2,217	87	-	8.9%	100%	1,201	2,080	-	-6.2%	2.2%	50
		150	2,036	Cannot find a soln.		56	2,463	55	-	21.0%	100%	1,256	2,080	-	-15.6%	2.2%	76
		200	2,101	Cannot find a soln.		49	2,632	44	-	25.3%	100%	1,485	2,238	-	-15.0%	6.5%	115
		Average			**82.3%**	**822**		**79**	**-10.9%**	**14.6%**	**100%**	**1,181**		**-11.7%**	**-9.4%**	**3.3%**	**66**

Table 73. The Comparison of LNS-LA with GBD-2-4 on T3 Instances – Random 5

Test Problem	|*N*|	|*K*|	Max. SP Length	CPLEX			GBD-2-4						LNS-LA				
				LA	(UB-LB) /UB	CPU Time (s)	*LA*	# of Cuts	% Dev. from CPLEX	% Dev. from Max.SP Length	(UB-LB) /UB	CPU Time (s)	*LA*	% Dev. from CPLEX	% Dev. From GBD-2-4	% Dev. from Max.SP Length	CPU Time (s)
T3-1	52	50	1,313	1,313	45.2%	393	1,313	154	0.0%	0.0%	100%	506	1,313	0.0%	0.0%	0.0%	10
		100	1,554	1,629	64.3%	226	1,597	115	-2.0%	2.8%	100%	808	1,554	-4.6%	-2.7%	0.0%	18
		150	1,554	1,554	53.7%	1,684	1,677	82	7.9%	7.9%	100%	896	1,554	0.0%	-7.3%	0.0%	27
		200	1,554	Cannot find a soln.		274	1,733	76	-	11.5%	100%	1,148	1,554	-	-10.3%	0.0%	40
		Average			**54,4%**	**644**		**107**	**2.0%**	**5.6%**	**100%**	**840**		**-1.5%**	**-5.1%**	**0.0%**	**24**
T3-2	62	50	1,777	1,777	62.8%	534	1,821	140	2.5%	2.5%	100%	574	1,777	0.0%	-2.4%	0.0%	15
		100	1,777	1,793	63.1%	6,036	1,889	95	5.4%	6.3%	100%	818	1,786	-0.4%	-5.5%	0,5%	30
		150	1,777	Cannot find a soln.		336	2,010	88	-	13.1%	100%	1,197	1,786	-	-11.1%	0.5%	41
		200	1,791	Cannot find a soln.		773	2,096	58	-	17.0%	100%	1,053	1,804	-	-13.9%	0.7%	56
		Average			**63.0%**	**1.920**		**95**	**3.9%**	**9.7%**	**100%**	**910**		**-0.2%**	**-8.2%**	**0.4%**	**36**
T3-3	81	50	1,918	1,923	69.9%	5,335	2,003	126	4.2%	4.4%	100%	829	1,923	0.0%	-4.0%	0.3%	32
		100	2,148	Cannot find a soln.		162	2,339	90	-	8.9%	100%	1,154	2,148	-	-8.2%	0.0%	58
		150	2,148	Cannot find a soln.		138	2,470	56	-	15.0%	100%	1,182	2,148	-	-13.0%	0.0%	86
		200	2,148	Cannot find a soln.		40	2,644	43	-	23.1%	100%	1,269	2,162	-	-18.2%	0.7%	114
		Average			**69.9%**	**1,419**		**79**	**4.2%**	**12.9%**	**100%**	**1,109**		**0.0%**	**-10.9%**	**0.2%**	**72**

APPENDIX – K

COMPUTATIONAL RESULTS OF LNS-LA ON T2 INSTANCES

The tables of Appendix K are presented starting on the next page.

Table 74. Computational Results of LNS-LA on T2 Instances – Random 1

Test Problem	|N|	p	Commodity Density	|K|	Maximum Shortest Path Length	LNS-LA Results				
						Initial LA	Final LA	% Improvement over Initial LA	% Deviation from Max. SP Length	CPU Time (s)
T2-1	21	4	35%	147	1,140	1,721	1,149	33.2%	0.8%	9.4
			65%	273	1,153	3,037	1,209	60.2%	4.9%	24.6
		5	35%	147	1,140	1,760	1,142	35.1%	0.2%	11.2
			65%	273	1,153	1,975	1,209	38.8%	4.9%	21.1
		6	35%	147	1,140	1,790	1,140	36.3%	0.%	9.9
			65%	273	1,153	1,880	1,193	36.5%	3.5%	17.9
T2-2	25	5	35%	210	969	1,835	1,075	41.4%	10.9%	20.9
			65%	390	969	No Solution				
		6	35%	210	969	1,562	1,043	33.2%	7.6%	21.2
			65%	390	969	2,302	1,125	51.1%	16.1%	35.3
		7	35%	210	969	1,562	1,038	33.6%	7.1%	20.5
			65%	390	969	1,567	1,095	30.1%	13.0%	36.8
T2-3	33	7	35%	370	1,161	2,151	1,297	39.7%	11.7%	57.4
			65%	686	1,299	No Solution				
		8	35%	370	1,161	2,151	1,297	39.7%	11.7%	45.3
			65%	686	1,299	4,174	1,332	68.1%	2.5%	174.9
		9	35%	370	1,161	2,151	1,297	39.7%	11.7%	54.2
			65%	686	1,299	4,174	1,321	68.4%	1.7%	166.6

Table 75. Computational Results of LNS-LA on T2 Instances – Random 2

Test Problem	|N|	p	Commodity Density	|K|	Maximum Shortest Path Length	LNS-LA Results				
						Initial LA	Final LA	% Improvement over Initial LA	% Deviation from Max. SP Length	CPU Time (s)
T2-1	21	4	35%	147	1,014	1,721	1,096	36.3%	8.1%	8.6
			65%	273	1,153	2,119	1,234	41.8%	7.0%	19.3
		5	35%	147	1,014	1,720	1,096	36.3%	8.1%	11.0
			65%	273	1,153	1,997	1,231	38.4%	6.8%	20.5
		6	35%	147	1,014	1,707	1,096	35.8%	8.1%	9.8
			65%	273	1,153	1,892	1,183	37.5%	2.6%	16.9
T2-2	25	5	35%	210	949	1,595	1,095	31.3%	15.4%	19.2
			65%	390	969	No Solution				
		6	35%	210	949	1,595	1,095	31.3%	15.4%	22.2
			65%	390	969	1,595	1,095	31.3%	13.0%	45.4
		7	35%	210	949	1,595	1,095	31.3%	15.4%	19.9
			65%	390	969	1,595	1,095	31.3%	13.0%	44.5
T2-3	33	7	35%	370	1,299	2,043	1,332	34.8%	2.5%	58.4
			65%	686	1,299	No Solution				
		8	35%	370	1,299	2,043	1,318	35.5%	1.5%	61.6
			65%	686	1,299	2,483	1,320	46.8%	1.6%	157.7
		9	35%	370	1,299	2,043	1,318	35.5%	1.5%	67.7
			65%	686	1,299	2,483	1,321	46.8%	1.7%	190.4

Table 76. Computational Results of LNS-LA on T2 Instances – Random 3

Test Problem	|N|	p	Commodity Density	|K|	Maximum Shortest Path Length	LNS-LA Results				
						Initial LA	Final LA	% Improvement over Initial LA	% Deviation from Max. SP Length	CPU Time (s)
T2-1	21	4	35%	147	1,005	1,794	1,056	41.1%	5.1%	11.4
			65%	273	1,153	3,261	1,212	62.8%	5.1%	22.0
		5	35%	147	1,005	1,707	1,056	38.1%	5.1%	13.1
			65%	273	1,153	2,187	1,209	44.7%	4.9%	17.2
		6	35%	147	1,005	1,801	1,056	41.4%	5.1%	10.7
			65%	273	1,153	1,892	1,201	36.5%	4.2%	19.1
T2-2	25	5	35%	210	949	1,595	1,095	31.3%	15.4%	16.5
			65%	390	969	No Solution				
		6	35%	210	949	1,595	1,095	31.3%	15.4%	18.1
			65%	390	969	1,595	1,095	31.3%	13.0%	43.1
		7	35%	210	949	1,595	1,095	31.3%	15.4%	14.9
			65%	390	969	1,595	1,095	31.3%	13.0%	50.0
T2-3	33	7	35%	370	1,116	1,916	1,308	31.7%	17.2%	47.5
			65%	686	1,291	No Solution				
		8	35%	370	1,116	1,916	1,308	31.7%	17.2%	47.2
			65%	686	1,291	3,030	1,325	56.3%	2.6%	164.6
		9	35%	370	1,116	1,916	1,308	31.7%	17.2%	46.3
			65%	686	1,291	3,030	1,325	56.3%	2.6%	134.9

Table 77. Computational Results of LNS-LA on T2 Instances – Random 4

Test Problem	|N|	p	Commodity Density	|K|	Maximum Shortest Path Length	LNS-LA Results				
						Initial LA	Final LA	% Improvement over Initial LA	% Deviation from Max. SP Length	CPU Time (s)
T2-1	21	4	35%	147	1,005	1,326	1,028	22.5%	2.3%	7.3
			65%	273	1,140	1,975	1,209	38.8%	6.1%	15.3
		5	35%	147	1,005	1,326	1,028	22.5%	2.3%	8.1
			65%	273	1,140	1,871	1,209	35.4%	6.1%	22.8
		6	35%	147	1,005	1,260	1,028	18.4%	2.3%	8.4
			65%	273	1,140	1,577	1,193	24.4%	4.6%	20.1
T2-2	25	5	35%	210	1,095	1,479	1,095	26.0%	0.0%	18.7
			65%	390	1,095	No Solution				
		6	35%	210	1,095	1,479	1,095	26.0%	0.0%	20.4
			65%	390	1,095	1,595	1,095	31.3%	0.0%	36.0
		7	35%	210	1,095	1,479	1,095	26.0%	0.0%	18.4
			65%	390	1,095	1,595	1,095	31.3%	0.0%	37.6
T2-3	33	7	35%	370	1,208	2,669	1,332	50.1%	10.3%	65.2
			65%	686	1,291	No Solution				
		8	35%	370	1,208	2,475	1,317	46.8%	9.0%	49.5
			65%	686	1,291	4,174	1,332	68.1%	3.2%	180.7
		9	35%	370	1,208	2,333	1,297	44.4%	7.4%	64.9
			65%	686	1,291	4,174	1,332	68.1%	3.2%	168.9

Table 78. Computational Results of LNS-LA on T2 Instances – Random 5

Test Problem	|N|	p	Commodity Density	|K|	Maximum Shortest Path Length	LNS-LA Results				
						Initial LA	Final LA	% Improvement over Initial LA	% Deviation from Max. SP Length	CPU Time (s)
T2-1	21	4	35%	147	1,153	1,445	1,193	17.4%	3.5%	13.9
			65%	273	1,153	3,292	1,223	62.8%	6.1%	19.7
		5	35%	147	1,153	1,728	1,193	31.0%	3.5%	10.2
			65%	273	1,153	2,096	1,209	42.3%	4.9%	20.6
		6	35%	147	1,153	1,801	1,173	34.9%	1.7%	13.3
			65%	273	1,153	1,892	1,181	37.6%	2.4%	19.2
T2-2	25	5	35%	210	961	1,579	1,038	34.3%	8.0%	17.1
			65%	390	1,095	No Solution				
		6	35%	210	961	1,474	1,002	32.0%	4.3%	14.2
			65%	390	1,095	2,612	1,095	58.1%	0.0%	33.9
		7	35%	210	961	1,474	999	32.2%	4.0%	14.6
			65%	390	1,095	1,562	1,095	29.9%	0.0%	37.3
T2-3	33	7	35%	370	1,291	2,833	1,326	53.2%	2.7%	49.0
			65%	686	1,299	No Solution				
		8	35%	370	1,291	2,833	1,309	53.8%	1.4%	54.5
			65%	686	1,299	3,794	1,332	64.9%	2.5%	132.8
		9	35%	370	1,291	2,833	1,309	53.8%	1.4%	50.6
			65%	686	1,299	3,794	1,332	64.9%	2.5%	126.0

Table 79. Computational Results of LNS-LA on T2 Instances for the Full-Cross-Traffic Case

Test Problem	|N|	Commodity Density	|K|	p	Maximum Shortest Path Length	LNS-LA Results				
						Initial LA	Final LA	% Improvement over Initial LA	% Deviation from Max. SP Length	CPU Time (s)
T2-1	**21**	100%	420	**4**	1,153	No Solution				
			420	**5**	1,153	2,528	1,209	52.2%	4.9%	57.4
			420	**6**	1,153	1,899	1,209	36.3%	4.9%	41.6
T2-2	**25**	100%	600	**5**	1,095	No Solution				
			600	**6**	1,095					
			600	**7**	1,095	1,595	1,114	30.2%	1.7%	78.5
T2-3	**33**	100%	1,056	**7**	1,299	No Solution				
			1,056	**8**	1,299					
			1,056	**9**	1,299	4,174	1,353	67.6%	4.2%	184.3

APPENDIX – L

COMPUTATIONAL RESULTS OF LNS-LA ON T3 INSTANCES

The tables of Appendix L are presented starting on the next page.

Table 80. Computational Results of LNS-LA on T3 Instances – Random 1

Test Problem	|N|	Commodity Density	|K|	Maximum Shortest Path Length	LNS-LA Results				
					Initial LA	Final LA	% Improvement over Initial LA	% Deviation from Max. SP Length	CPU Time (s)
T3-1	**52**	25%	663	1,518	2,017	1,634	19%	8%	203
		50%	1,326	1,531	3,112	1,717	45%	12%	563
		75%	1,989	1,531	4,198	1,799	57%	18%	695
T3-2	**62**	25%	946	1,871	2,756	1,989	28%	6%	579
		50%	1,891	1,893	3,165	2,464	22%	30%	831
		75%	2,837	1,893	3,623	2,676	26%	41%	1,180
T3-3	**81**	25%	1,620	2,231	2,866	2,634	8%	18%	1,173
		50%	3,240	2,231	4,760	2,730	43%	22%	2,217
		75%	4,860	2,268	6,034	2,756	54%	22%	3,580

Table 81. Computational Results of LNS-LA on T3 Instances – Random 2

Test Problem	|N|	Commodity Density	|K|	Maximum Shortest Path Length	LNS-LA Results				
					Initial LA	Final LA	% Improvement over Initial LA	% Deviation from Max. SP Length	CPU Time (s)
T3-1	52	25%	663	1,478	2,571	1,620	37%	10%	217
		50%	1,326	1,478	4,198	1,701	59%	15%	475
		75%	1,989	1,531	4,198	1,903	55%	24%	599
T3-2	62	25%	946	1,871	4,641	2,038	56%	9%	527
		50%	1,891	1,871	5,251	2,420	54%	29%	901
		75%	2,837	1,871	5,257	2,601	51%	39%	1,327
T3-3	81	25%	1,620	2,211	4,324	2,663	38%	20%	1,138
		50%	3,240	2,231	5,435	2,742	50%	23%	2,580
		75%	4,860	2,268	6,034	2,768	54%	22%	3,281

Table 82. Computational Results of LNS-LA on T3 Instances – Random 3

Test Problem	|N|	Commodity Density	|K|	Maximum Shortest Path Length	LNS-LA Results				
					Initial LA	Final LA	% Improvement over Initial LA	% Deviation from Max. SP Length	CPU Time (s)
T3-1	**52**	25%	663	1,518	2,571	1,638	36%	8%	219
		50%	1,326	1,530	4,198	1,774	58%	16%	558
		75%	1,989	1,554	4,198	1,903	55%	22%	720
T3-2	**62**	25%	946	1,833	4,601	2,004	56%	9%	494
		50%	1,891	1,833	5,232	2,464	53%	34%	861
		75%	2,837	1,871	5,314	2,734	49%	46%	1,256
T3-3	**81**	25%	1,620	2,268	4,759	2,680	44%	18%	1,275
		50%	3,240	2,268	5,360	2,748	49%	21%	3,082
		75%	4,860	2,268	5,392	2,758	49%	22%	3,727

Table 83. Computational Results of LNS-LA on T3 Instances– Random 4

Test Problem	|N|	Commodity Density	|K|	Maximum Shortest Path Length	LNS-LA Results				
					Initial LA	Final LA	% Improvement over Initial LA	% Deviation from Max. SP Length	CPU Time (s)
T3-1	52	25%	663	1,566	2,477	1,658	33%	6%	271
		50%	1,326	1,566	3,965	1,789	55%	14%	630
		75%	1,989	1,566	4,129	1,903	54%	22%	571
T3-2	62	25%	946	1,893	2,734	1,987	27%	5%	523
		50%	1,891	1,893	4,721	2,420	49%	28%	869
		75%	2,837	1,893	5,258	2,719	48%	44%	1,382
T3-3	81	25%	1,620	2,211	4,571	2,667	42%	21%	1,233
		50%	3,240	2,216	4,774	2,698	43%	22%	2,270
		75%	4,860	2,231	5,073	2,757	46%	24%	3,591

Table 84. Computational Results of LNS-LA on T3 Instances – Random 5

Test Problem	|N|	Commodity Density	|K|	Maximum Shortest Path Length	LNS-LA Results				
					Initial LA	Final LA	% Improvement over Initial LA	% Deviation from Max. SP Length	CPU Time (s)
T3-1	52	25%	663	1,554	2,981	1,624	46%	5%	227
		50%	1,326	1,566	3,201	1,718	46%	10%	540
		75%	1,989	1,566	4,342	1,892	56%	21%	647
T3-2	62	25%	946	1,833	2,803	1,987	29%	8%	426
		50%	1,891	1,893	4,647	2,464	47%	30%	854
		75%	2,837	1,893	4,998	2,564	49%	35%	1,453
T3-3	81	25%	1,620	2,211	4,604	2,690	42%	22%	1,344
		50%	3,240	2,268	4,618	2,700	42%	19%	2,083
		75%	4,860	2,268	5,073	2,782	45%	23%	3,561

CURRICULUM VITAE

PERSONAL INFORMATION

Surname, Name: Yiğit, Aybeniz
Nationality: Turkish (TC)
Date and Place of Birth: 5 July 1979, İzmir
Phone: +90 312 592 2507
Fax: +90 312 592 3030
email: ayigit@aselsan.com.tr

EDUCATION

Degree	Institution	Year of Graduation
MS	METU Industrial Engineering	2005
BS	METU Industrial Engineering	2002

WORK EXPERIENCE

Year	Place	Enrollment
2002-Present	ASELSAN Inc.	Chief Engineer

FOREIGN LANGUAGES

Advanced English, Basic German

PUBLICATIONS

Conference Proceedings

1. Yiğit A., Genetik Algoritma Kullanarak Kümelendirilmiş Lojistik Ağlarında Ana Dağıtım Üsssü Seçimi (A Genetic Algorithm for the Hub Location Problem in Clustered Logistic Networks), SAVTEK (Defense Technologies Conference), Ankara, Turkey, 2010.
2. Akesson B., Horne G., Masseli K., Narayanan F., Pakkanen M., Shuockley J., Upton S., Yıldırım Z., Yiğit A. "MSG-088 Data Farming Case Study on Humanitarian Assistance / Disaster Relief", Scyhte: Proceedings and Bulletin of the International Data Farming Community, Issue 11, October, 2011.

Conference Presentations

Yiğit A., Saatçioğlu Ö., Sepil C., Serin. Y. "A Methodology for Determining the Cluster of A New Project When There are Preformed Project Clusters", INFORMS Annual Coference, Seattle, Washington, USA, 2007.

TRAINING and CERTIFICATES

1. Project Management Processes, Makro Consulting (Şekip Karahan), 26-28 March 2003, Ankara.
2. Using MS PROJECT Software, Makro Consulting (Demir Özkaya), 07-09 April 2003, Ankara.
3. Statistical Process Control and Six Sigma Applications, SATEM, 25 April 2003, Ankara.
4. Process Management and Improvement, KALDER, 20-21 October 2003, Ankara.
5. Using DOORS for Requirements Management, Telelogic (Benjamin Blersch), 9-10 June 2004, Ankara.
6. Requirements Methodology, Telelogic (Ian Alexander), 21-22 July 2004, Ankara.
7. The Future of Innovation: Evolving Beyond the "Customer Driven" Paradigm Workshop, Management Roundtable (Tony Ulwick), 28 September 2004, Chicago, USA.
8. Co-Development Metrics – Best Practices Workshop, Management Roundtable (Wayne Mackey), 28 September 2004, Chicago, USA.
9. Written Communication and Reporting (in Turkish), Strata (Kaan Eran), 25 November 2004, Ankara.
10. Improvement of Contemporary Management Skills, Çözüm Human Resources and Management Consultancy Center (Prof. Dr. Tanıl Kılınç), 12-14 April 2007, Ankara.
11. Introduction to Capability Maturity Model Integration v1.2, (Wayne Littlefield), 10-12 September 2007, Ankara (Certificate from Carnegie Mellon University).
12. WAID 2007 – Artificial Intelligence and Data Mining Workshop, 3 November 2007, Seattle Washington, USA.
13. Measurement and Analysis Workshop, Summit Process Engineering Knowledge and Technology Group (Wayne Littlefield), 29-31 January 2008, Ankara.
14. Budgeting and Budget Control Techniques, MBH Group (Prof. Dr. Sudi Apak), 19-20 June 2008, Ankara.

15. Robust Product and Process Design, SATEM, 2 January 2009, Ankara.
16. Effective Presentation Techniques (in Turkish), Derin Education, 27 February 2009, Ankara.
17. Quality Costs, KALDER, 29 December 2009, Ankara.
18. Business Writing, Door Training & Consulting (Prof. Dr. İsmet Barutçugil), 26-27 April 2010, Ankara.
19. Strategic Planning and Management, Rönesans Change and Management Science Institute (Canip Altay), 25-26 November 2010, Ankara.
20. Patent Training, Ankara Patent (Kaan Dericioğlu), 13 November 2011, Ankara.
21. Strategic Management of Research and Development, Global Education (Prof. Dr. C. Ruhi Kaykayoğlu), 18-19 December 2012, Ankara.
22. Lean Management, Lean Institute (Cevdet Özdoğan), 25 April 2014, Ankara.

PARTICIPATED INTERNATIONAL WORK GROUPS

NATO Research and Technology Organization Modeling and Simulation Group – NMSG 088 Data Farming in Support of NATO (2010-2012) – Participated in Design of Experiments Subgroup.

AWARDS

Granted the Scientific Achievement Award 2014 by the NATO Science and Technology Organization (STO) in recognition of her contributions to the NATO Work Group "NMSG 088 Data Farming in Support of NATO", which she is member of.