COST MINIMIZATION IN MULTI−COMMODITY, MULTI−MODE GENERALIZED NETWORKS WITH TIME WINDOWS

A Dissertation

by

PING-SHUN CHEN

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

December 2005

Major Subject: Industrial Engineering

COST MINIMIZATION IN MULTI−COMMODITY, MULTI−MODE GENERALIZED NETWORKS WITH TIME WINDOWS

A Dissertation

by

PING-SHUN CHEN

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Approved by:

Chair of Committee,	Alberto Garcia-Diaz
Committee Members,	César O. Malavé
	Amarnath Banerjee
	Derya Akleman
Head of Department,	Brett A. Peters

December 2005

Major Subject: Industrial Engineering

ABSTRACT

Cost Minimization in Multi−Commodity, Multi−Mode Generalized

Networks with Time Windows. (December 2005)

Ping-Shun Chen, B.S., National Chiao Tung University, Taiwan;

M.S., National Chiao Tung University, Taiwan

Chair of Advisory Committee: Dr. Alberto Garcia-Diaz

The purpose of this research is to develop a heuristic algorithm to minimize total costs in multi-commodity, multi-mode generalized networks with time windows problems. The proposed mathematical model incorporates features of the congestion of vehicle flows and time restriction of delivering commodities. The heuristic algorithm, **HA**, has two phases. Phase 1 provides lower and upper bounds based on Lagrangian relaxations with subgradient methods. Phase 2 applies two methods, early due date with overdue-date costs and total transportation costs, to search for an improved upper bound.

Two application networks are used to test **HA** for small and medium-scale problems. A different number of commodities and various lengths of planning time periods are generated. Results show that **HA** can provide good feasible solutions within the reasonable range of optimal solutions. If optimal solutions are unknown, the average gap between lower and upper bounds is 0.0239. Minimal and maximal gaps are 0.0007 and 0.3330. If optimal solutions are known, the maximal gap between upper bounds and optimal solutions is less than 10% ranges of optimal solutions.

DEDICATION

To my parents

ACKNOWLEDGMENTS

I would like to express my appreciation and thankfulness to Dr. Alberto Garcia-Diaz, the chair of my committee, for inspiring me to begin this research and giving me a lot of valuable advice during the time I worked on the accomplishment of my dissertation. He encouraged me and guided me to overcome some difficulties I encountered in this research. I also want to thank my committee members, Dr. César O. Malavé, Dr. Amarnath Banerjee, and Dr. Derya Akleman, for their precious opinions about my research. Their input helped me do this research in a more effective manner.

In addition, I want to thank my dear family who supported me during the last five years. I would also like to express my appreciation for my dear friends. Their support and encouragement has been meaningful and important as I participated in the many facets of academic life as an international student.

TABLE OF CONTENTS

CHAPTER Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

CHAPTER I

INTRODUCTION

1.1 Introduction

In multi-commodity, multi-mode generalized networks with time windows problems, commodities can represent personnel, products, vehicles, or a diversity of services, such as communication services provided by phone or internet companies. Multiple modes of the network formulation represent different means of transportation used to deliver commodities from origins to destinations. Companies in the commodity distribution industry consider not only how many products are to be transported, but also how to transport products by means of various transportation methods, i.e., finished goods are transported from plants to retailers by airplane or by truck. In addition, commodities are allowed to be transferred from one mode to another at some certain nodes. Moreover, because commodities may be damaged while transporting, a generalized network is selected to express this feature.

In natural catastrophes or emergent situations, time is very important. For example, before a huge hurricane arrives at some areas, it is critical to quickly evacuate people from their homes to safer shelters. After a hurricane goes away, it is essential to dispatch supply materials and food to disaster areas. Therefore, time factor is also included in this research.

This dissertation follows the format and style of *Transportation Research Part B*.

For example, refinery industries meet all features described in this research. Oil is transported through pipes, trucks, or ships. Due to oil evaporating or leaking, the amount of oil sent on origin may be different than the amount of oil received at its destination. Thus, the generalized network is adequate for modeling losses of oil flows. Time factor has significant impacts on scheduling crews to receive and store oil. When oil is delivered to refineries, the necessary crews must be ready in order to handle receiving. If crews are not ready, oil will not be handled appropriately, which will cause a delay and yield extra costs. Furthermore, because oil transportation involves in huge annual costs, finding the minimal total costs to deliver oil through different methods of transportation within time windows will be very beneficial to refinery industries.

1.2 Motivation

Only a few published articles focus on multiple commodities, multiple modes, commodity losses, and time windows together. Most papers only include multi-commodity, single-mode models, single-commodity, multi-mode models, or multi-commodity, multi-mode models without time windows. Under many situations, these four factors need to be taken into account simultaneously. For instance, a global company takes orders from overseas customers, produces various products at different plants, ships finished goods to distribution centers through trucks or airplanes, and delivers products to customers within due dates. This example includes multiple commodities, multiple modes, losses of commodity flows, and time windows. It is also a typical supply chain problem.

It is desired to incorporate the congestion of vehicle flows into the model. This congestion is one kind of capacity constraint for arcs. For example, a bridge has its capacity for the number of vehicles per hour and its capacity for total weights on it per time unit. Hence, this research includes this characteristic to make the model more realistic.

In general, the entire model involves a lot of resources allocated to the whole system. Therefore, how to find a minimum cost for multi-commodity, multi-mode generalized networks with time windows problems becomes more and more significant for the supply chain management. This research develops a methodology for providing good heuristic feasible solutions within reasonable solving times.

1.3 Numerical Example

An illustrative example in this section is desired to consider a 3-commodity, 2-mode, and 10-planning-time-period problem. A plant is located at node *A*. Customers are located at node *C* and node *D*. Assume a demand of commodity 1 of 5,000 units at node *C* and time 5, a demand of commodity 2 of 7,500 units at node *D* and time 8, and a demand of commodity 3 of 10,000 units at node *D* and time 6. Modes *M* and *L* represent two different transportation methods and have their own routings. Arcs (*A*, *B*), (*B*, *C*), (*B*, *D*) and (*C*, *D*) represent roads among these nodes, as shown in Fig. 1.

To simplify the problem, some assumptions are made in this example. First, each vehicle on mode *M* can carry either 100 units of commodity 1, 100 units of commodity 2, or 100 units of commodity 3. Each vehicle on mode *L* can carry either 150 units of

commodity 1, 200 units of commodity 2, or 200 units of commodity 3. Second, each mode has one kind of vehicle. Third, mode *M* at nodes *A*, *B*, and *C* has 100 vehicles at time 0 and mode *L* at nodes *A* and *B* has 100 vehicles at time 0. Fourth, changing modes for commodities takes one time period and there is no damage on commodities while changing modes. Finally, the earliest delivery time of each commodity is three time periods ahead of its due date. The planning time periods are 10 time periods. Data and parameters are displayed in Table 1 and 2.

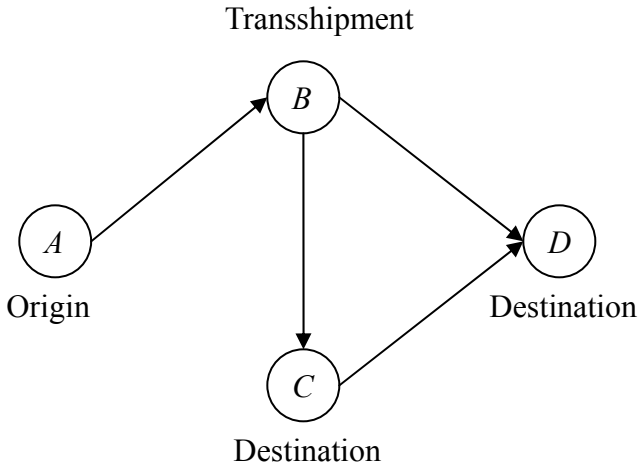


Fig. 1. A physical network.

Table 1
Parameters for each commodity and mode on each arc

Arc	Loss factor		Cost to transport commodity		Cost to transport vehicles	
	Com. 1	Com. 2&3	Com. 1	Com. 2&3	Mode *M*	Mode *L*
(*A*, *B*)	0.9	0.9	10	8	100	120
(*B*, *C*)	1	0.9	5	5	50	∞
(*B*, *D*)	0.85	0.9	5	12	∞	80
(*C*, *D*)	0.9	0.85	12	10	100	∞

Note : "∞" means there is no routing of mode *M* or mode *L* on this arc.

Table 2
Capacity of each vehicle on mode *M* and *L* for each commodity

Mode	Commodity 1	Commodity 2	Commodity 3
M	100	100	100
L	150	200	200

In Fig. 2, lower and upper bounds of vehicles on arc (*A*, *B*) are 0 and 200 vehicles, respectively. Flow of vehicles on mode *M* on arc (*A*, *B*) is 60 vehicles. Cost to transport per vehicle on mode *M* for arc (*A*, *B*) is $100, and travel time for vehicles on mode *M* on arc (*A*, *B*) is 2 time periods. In Fig. 3, (1,555.6, 10, 0.9) represents 1,555.6 units of commodity 1 transported by vehicles on mode *M* at time 0, transporting cost is $10 per commodity 1, and loss factor of commodity 1 on arc (*A*, *B*) is 0.9. Therefore, 1,555.6 * 0.9 = 1,400 units of commodity 1 at node *B* are received. The objective function includes the transporting-commodity cost, transporting-vehicle cost, holding-commodity cost, holding-vehicle cost, overdue-date cost, and transferring-mode cost. An optimal solution is $537,661.28 solved by the CPLEX 9.0 software.

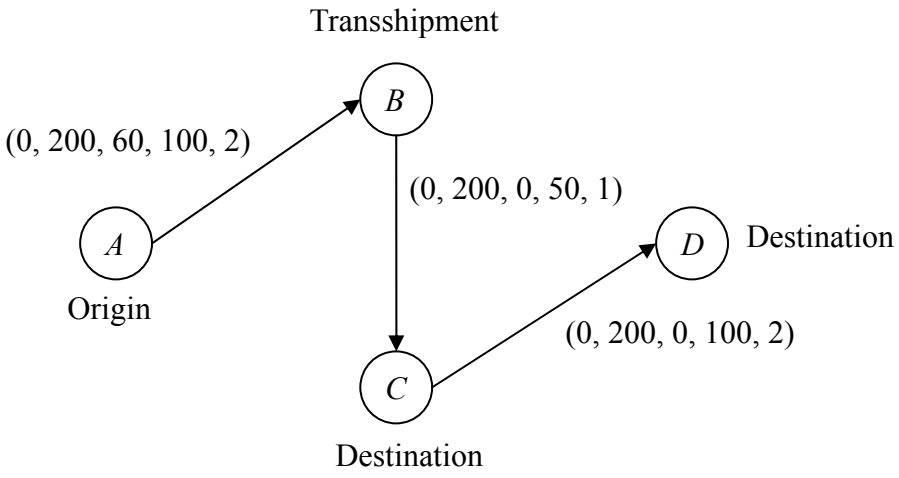


Fig. 2. Flow of vehicles on mode *M* at time 0.

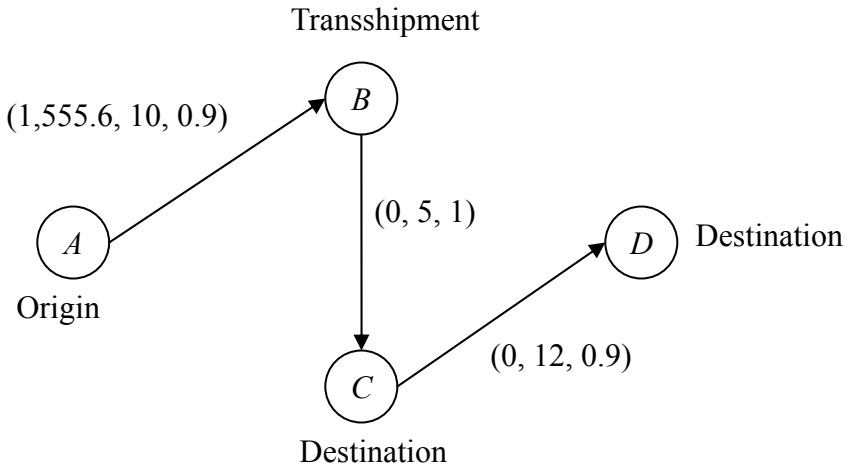


Fig. 3. Flow of commodity 1 delivered by mode *M* at time 0.

1.4 Objectives and Contributions

This research is extended from Haghani and Oh's (1996) model and their pure network is changed to a generalized network. The proposed mathematical model of this research incorporates the congestion of vehicle flows and time restriction of delivering commodities. The main goal of this research is to develop a heuristic algorithm, **HA**, to obtain an approximate optimal solution to minimize total costs incurred during transporting and holding for commodities and vehicles within time windows.

The most significant contributions of this research are as follows: first, it develops a procedure that considers multiple commodities, multiple modes, commodity losses, and time windows. Because only a few papers combine with all four factors simultaneously, this research can provide more insights in this area. Second, the procedure in this research is more computational efficiency than solving original problems directly. This procedure is a heuristic algorithm and provides good heuristic feasible solutions instead

of optimal solutions.

Third, it improves procedures for lower bound generations of subgradient methods. Because the step size in this research is adopted in Proposition 2, this step size is larger than the traditional step size. Larger step sizes will reduce convergence times. Finally, it is an effective procedure for the reduction of upper bounds. Two methods, early due date with overdue-date costs and total transportation costs in Phase 2 of the heuristic algorithm, **HA**, are applied to seek an improved upper bound. Testing runs generated in Chapter V show that most upper bounds are improved in Phase 2.

1.5 Organization of Dissertation

Chapter I introduces the scope, motivations, and an illustrative example of a multi-commodity, multi-mode generalized network with time windows problems. It also points out objectives and contributions of this research. Chapter II reviews and summarizes literature of multi-commodity and multi-mode, and Lagrangian relaxations with subgradient methods. Chapter III describes the problem, assumptions, model, and solution approaches. Chapter IV proposes a methodology, which has two phases. Phase 1 is based on Lagrangian relaxations with subgradient methods. Phase 2 is used to search for an improved upper bound. Different scenarios are generated to test the heuristic algorithm, **HA**, in Chapter V. Data are also summarized and analyzed in Chapter V. Chapter VI gives conclusions of this research and recommends future related research.

CHAPTER II

LITERATURE REVIEW

2.1 Introduction

This research involves solving multi-commodity, multi-mode mixed integers with time windows problems. Therefore, literature reviews include two review papers for multi-commodity problems and several related papers. It also reviews Lagrangian relaxations with subgradient methods, which will be applied in Phase 1 of the heuristic algorithm, **HA**.

2.2 Multi-Commodity and Multi-Mode

During the past decades, researchers have worked on multi-commodity flow problems. Basically, these problems can be categorized into two classes depending on the objective function: linear and nonlinear cost multi-commodity flow problems. Kennington (1978) discussed and proved some integrity properties of multi-commodity flows. Due to side constraints, which include arc capacity restrictions for all commodities sharing the same arc, some integrity properties in a single-commodity flow problem do not apply to a multi-commodity flow problem. Solution techniques for linear cost multi-commodity flow problems can be classified accordingly into three methods: partitioning, price-directive, and resource-directive.

Assad (1978) surveyed both linear and nonlinear cost multi-commodity flow problems, and found that there were three general types of formulations: node-arc,

arc-chain, and arc-circuit. He focused on the optimization of a nonlinear cost multi-commodity flow problem over a convex feasible set defined by linear constraints. He summarized several algorithms using feasible direction methods, which consist of a feasible direction approach, a linear approximation algorithm, and an equilibration approach. He also summarized other methods, such as a reduced gradient method, a gradient projection method, and an alternative solution technique.

Guélat et al. (1990) developed a multi-mode, multiple-product network assignment model for strategic planning of freight flows. In their model, they only included the flow conservation and nonnegative constraints, but excluded capacity constraints of congestions. In order to obtain the minimal transporting and transferring cost, they adopted the Gauss-Seidel-Linear Approximation algorithm (GSLA). The disadvantage of GSLA was the poor rate of convergence in the vicinity of the optimal solution. Drissi-Kaitouni (1991) refined Guélat's model to reduce the computational time and compared the computational efficiency of the GSLA algorithm with other decomposition algorithms.

Rathi et al. (1992) studied a problem which involved allocations of a limited number of transport resources to shipments of cargos and personnel within specified time windows. They developed three different formulations and solved them using the Lift Optimizer module of the Deployment Analysis Prototype (DAP) system. Haghani and Oh (1996) studied large-scale multi-commodity, multi-mode network flow problems with time windows. They used a time-space network formulation. Their model was allowed to transfer commodities between modes on a certain node. They developed

two heuristic algorithms to solve this complex problem. One was based on Lagrangian relaxations and applied LINDO (Linear, Interactive, and Discrete Optimizer) software to solve subproblems. The other was an interactive fix-and-run heuristic algorithm.

2.3 Lagrangian Relaxations with Subgradient Methods

Since complex problems are difficult to solve, Lagrangian relaxations propose to relax one or several of complicated constraints in the original problem and to reformulate them into the **Lagrangian Dual (LD)** problem. Therefore, it will be easier to solve the **LD** problem than to solve the original problem. The solution of the **LD** problem provides a lower bound for the original minimal problem (see Fisher, 1985). For example, let $f(x) = cx$ be a convex function subject to constraints of $Ax \leq b$ and $Cx \leq d$ in equation (1), where x is nonnegative and integral. The objective of $f(x)$ is to minimize cx.

$$\begin{aligned} &\textbf{min} \quad f(x) = cx \\ &\textbf{s.t.} \\ &\qquad Ax \leq b, \\ &\qquad Cx \leq d, \\ &\qquad x \geq 0 \ \text{and integer}. \end{aligned} \tag{1}$$

If $A x \leq b$ is relaxed, multiplied by nonnegative μ, and this item, $\mu(Ax - b)$, is added to its objective function, $\theta(\mu)$ becomes a new objective function in equation (2). If $(Ax - b) > 0$, the minimum of $\theta(\mu)$ is cx when μ is 0. If $(Ax - b) \leq 0$, the minimum of $\theta(\mu) \leq cx$. In both cases, $\theta(\mu) \leq f(x)$. Therefore, for each fixed μ, the solution of $\theta(\mu)$ provides a

lower bound for $f(x)$. The problem becomes how to find a maximal value of $\theta(\mu)$ in order to obtain an optimal or approximate optimal solution for $f(x)$. Moreover, $\theta(\mu)$ is a concave function. This property has already been proven (see Fisher, 1981).

$$\begin{aligned} &\textbf{min} \quad \theta(\mu) = cx + \mu(Ax - b) \\ &\textbf{s.t.} \\ &\qquad Cx \le d, \\ &\qquad x \ge 0 \text{ and integer}, \\ &\qquad \mu \ge 0. \end{aligned} \tag{2}$$

Since Lagrangian relaxations only provide lower bounds for original problems, it usually combines with branch-and-bound search methods to obtain an optimal or approximate optimal solution. How to choose different values of μ in equation (2) is an issue. However, many papers select subgradient methods and apply different step sizes to update values of μ.

If $\theta(.)$ satisfies equation (3), g is called a subgradient of $\theta(.)$ at v. Here, $\theta(.)$ is a concave function. If we let $\theta(v) = cx$ and $g = (Ax - b)$ and $v = 0$, equation (3) will become equation (4). Equation (5) is used to update values of μ for equation (4), where λ is the positive step size, and s is the number of iterations.

$$\theta(\mu) \le \theta(v) + g(\mu - v) \qquad \text{for all } \mu \tag{3}$$

$$\theta(\mu) \le cx + \mu(Ax - b) \qquad \text{for all } \mu \tag{4}$$

$$\mu^{(s+1)} = \mu^{s} + \lambda^{(s)}(Ax^{(s)} - b). \tag{5}$$

There are many rules in choosing step sizes in general subgradient methods (see Goffin, 1977). For example, if the step size, λ^k, satisfies equation (6), subgradient methods will guarantee to obtain optimal solutions for original problems (see Correa, 1993; Konnov, 2003).

$$\lim_{k\to\infty} \lambda^k = 0, \quad \sum_{k=0}^{\infty} \lambda^k = \infty, \quad \sum_{k=0}^{\infty} (\lambda^k)^2 < \infty. \tag{6}$$

Proposition 1. If the step size, λ^k, satisfies $0 < \lambda^k < \dfrac{2(\theta(\mu^*) - \theta(\mu^k))}{\|g^k\|^2}$,

then, $\| \mu^{k+1} - \mu^* \| < \| \mu^k - \mu^* \|$, where g^k is the subgradient.

Proof. See Proposition 6.3.1 in p. 610 of Bertsekas (1999).

The step size, λ^k in Proposition 1, is the most common step size adopted for subgradient methods. Let $\lambda^k = \sigma(\theta(\mu^*) - \theta(\mu^k)) / ||g^k||^2$, where σ is a scalar and $0 < \sigma < 2$, and $\theta(\mu^*)$ is an optimal solution of $\theta(\mu)$. If $\theta(\mu^*)$ is unknown, it is usually replaced by the current best upper bound.

Proposition 2. **If the step size,** λ^k**, satisfies** $0 < \lambda^k < \dfrac{2(\theta(\mu^*) - \theta(\mu^k))}{\left\| \hat{g}^k \right\|^2}$,

then, $\| \mu^{k+1} - \mu^* \| < \| \mu^k - \mu^* \|$, **where** g^k **is the subgradient.**

Here,

$$\hat{g}_i^k = \begin{cases} g_i^k, \text{ if } i \notin I^k \text{ where } I^k = \{ i \mid g_i^k \le 0 \text{ and } \mu_i^k = 0 \}. \\ 0, \quad \text{otherwise.} \end{cases}$$

Proof. See Proposition 1 of Section 3 of Wang (2003).

For a special case, if all $\hat{g}_i^k = 0$, then $0 < \lambda < \infty$. Wang (2003) recommended a new step size in Proposition 2 for subgradient methods and proved that μ would converge to its optimal solution. He also gave an example to demonstrate that it worked. His step size is larger than the common popular one in Proposition 1. Thus, the convergence rate of his step size is quicker than that of the traditional step size.

Although subgradient methods lead themselves to an easy implementation in the form of a computer program, the resulting computerized procedure may suffer from the slow speed of convergence in the neighborhood of the optimal solution.

CHAPTER III

MODEL FORMULATION AND SOLUTION APPROACH

3.1 Introduction

This chapter introduces the problem definition, model formulation, and description of a solution approach. The following sections are: Section 3.2 describes the scope of multi-commodity, multi-mode generalized networks with time windows problems. Section 3.3 defines necessary assumptions and required input data. **Problem (P)** is formulated in Section 3.4. Section 3.5 proposes a heuristic procedure, **HA**, for solving **Problem (P)**. A summary of this chapter is given in Section 3.6.

3.2 Problem Definition

In multi-commodity, multi-mode generalized networks with time windows problems, the term "multi-commodity" means either a variety of commodities or a single commodity with different due dates. Additionally, the term "multi-mode" represents various means of transportation used to deliver commodities from origins to destinations. At certain nodes, commodities may be transferred from one mode to another. Each mode has its own capacity and routing. Moreover, because commodities may be damaged while transporting, a generalized network formulation is selected to model losses of commodity flows.

Each commodity has its time windows, (t_1 - t_2). Time windows, (t_1 - t_2), mean that a commodity is not allowed to deliver to its destination before t_1. The latest time to deliver

a commodity to its destination without penalty is t_2. If the time of delivering is later than t_2, it will cause a penalty for the overdue date. The objective function is to minimize sums of transporting-commodity cost, holding-commodity cost, overdue-date cost, transferring-mode cost, transporting-vehicle cost, and holding-vehicle cost.

3.3 Model Assumption and Input Data

This model assumes that (a) holding and transporting costs for commodities are proportional to the amount of commodities, (b) holding and transporting costs for vehicles are proportional to the amount of vehicles, and (c) overdue-date penalty per time period is proportional to the amount of commodities.

The following information is necessary for constructing the model: (a) origin-destination matrix for each commodity, (b) known number of vehicles on each mode at time 0, (c) loss factor for each commodity on each arc, and (d) the length of planning time periods.

3.4 Model Formulation

In the mathematical model formulation, assume that $\underline{\boldsymbol{A}}$ and $\underline{\boldsymbol{A}}'$ are $m \times n$ matrices; $\underline{\boldsymbol{X}}$ and $\underline{\boldsymbol{Y}}$ are $n \times 1$ matrices; and $\underline{\boldsymbol{b}}$, $\underline{\boldsymbol{r}}$, and $\underline{\boldsymbol{U}}$ are $m \times 1$ matrices. This model includes three decision variables. The first decision variable, $x_{ijtt'}^{km}$, denotes flow of k delivered by m from i to j during the [t, t'] period, where $x_{ijtt'}^{km} \in \underline{\boldsymbol{X}}$ and $x_{ijtt'}^{km}$ is nonnegative. The second decision variable, $y_{ijtt'}^{m}$, denotes flow of vehicles on m from i to j during the [t, t']

period, where $y^{m}_{ijtt'} \in \underline{\boldsymbol{Y}}$ and $y^{m}_{ijtt'}$ is a nonnegative integer. The third decision variable, $\boldsymbol{u}^{m}_{ijtt'}$, denotes flow of unused vehicles on *m* from *i* to *j* during the [*t*, *t'*] period, where $\boldsymbol{u}^{m}_{ijtt'} \in \underline{\boldsymbol{U}}$ and $\boldsymbol{u}^{m}_{ijtt'}$ is a nonnegative integer.

Notation:

i, j	=	index of node.
k	=	index of commodity.
m, m'	=	index of vehicle on mode.
t, t'	=	index of time period.
$\boldsymbol{TCC}^{k}_{ij}$	=	transporting cost for *k* on (*i*, *j*).
$\boldsymbol{ODC}^{k}_{t}$	=	overdue-date cost for *k* at *t*, where *t* is greater than its due date.
$\boldsymbol{TMC}^{k}_{mm'}$	=	transferring-mode cost of *k* from *m* to *m'*.
$\boldsymbol{HCC}^{k}_{i}$	=	holding cost for *k* on *i*.
$\boldsymbol{TVC}^{m}_{ij}$	=	transporting cost for each vehicle on *m* on (*i*, *j*).
$\boldsymbol{HVC}^{m}_{i}$	=	holding cost for each vehicle on *m* on *i*.
$\boldsymbol{V}^{km}$	=	vehicle capacity on *m* for *k*.
$\boldsymbol{W}^{m}$	=	weight of vehicle on *m*.
$\boldsymbol{AC}_{ijt}$	=	capacity of (*i, j*) during the [*t*-1, *t*] period.
$\boldsymbol{CA}^{m}_{ijtt'}$	=	capacity of vehicles on *m* for (*i, j*) during the [*t*, *t'*] period.

$L^{m}_{ijtt'}$ = lower bound of vehicles on *m* for (*i, j*) during the [*t, t'*] period.

$U^{m}_{ijtt'}$ = upper bound of vehicles on *m* for (*i, j*) during the [*t, t'*] period.

ED^{k} = the earliest day for *k* being able to be delivered to its destination.

T = planning time periods.

The generic model, **Problem (P)**, is formulated as follows:

Problem (P)

Minimize

$$\sum_{x^{km}_{ijtt'} \in \underline{X}} (TCC^{k}_{ij} + ODC^{k}_{t} + TMC^{k}_{mm'} + HCC^{k}_{i})\ x^{km}_{ijtt'} + \sum_{y^{m}_{ijtt'} \in \underline{Y}} (TVC^{m}_{ij} + HVC^{m}_{i})\ y^{m}_{ijtt'} \quad (7)$$

Subject to

$$\underline{A}\ \underline{X} = \underline{b} \quad (8)$$

$$\underline{A}'\ \underline{Y} = \underline{r} + \underline{U} \quad (9)$$

$$\sum_{k} (x^{km}_{ijtt'} / V^{km}) - y^{m}_{ijtt'} \le 0 \quad \text{for all } m, i, j, t, t' \quad (10)$$

$$\sum_{m} \sum_{t_1 \in T, t_1 \le t \le t_2} W^{m}\ y^{m}_{ijt_1t_2} \le AC_{ijt} \quad \text{for all } i, j, t \quad (11)$$

$$y^{m}_{ijtt'} \le CA^{m}_{ijtt'} \quad \text{for all } m, i, j, t, t' \quad (12)$$

$$L^{m}_{ijtt'} \le y^{m}_{ijtt'} \le U^{m}_{ijtt'} \quad \text{for all } m, i, j, t, t' \quad (13)$$

$$\sum_{m,\ t' < ED^k} x^{km}_{ijtt'} = 0 \qquad \text{for all } k, i, j, t \qquad (14)$$

$$x^{km}_{ijtt'} \geq 0 \qquad \text{for all } k, m, i, j, t, t' \qquad (15)$$

$$y^{m}_{ijtt'}, u^{m}_{ijtt'} \geq 0 \text{ and integer} \qquad \text{for all } m, i, j, t, t' \qquad (16)$$

Constraint (7) is the objective function. Constraint (8) and (9) are the flow conservation of commodities and vehicles, respectively. Constraint (10) is that required vehicles must be less than or equal to used vehicles on each mode, arc, and time period. Constraint (11) represents the congestion of vehicle flows, which means vehicles on all modes must be less than or equal to the arc capacity for each time period, where $t_2 = t_1 + t_{ij}$ and t_{ij} is the traveling time from i to j. Constraint (12) is the capacity of vehicles on each mode, arc, and time period. Constraint (13) is the lower and upper bound of vehicles on each mode, arc, and time period. Constraint (14) is the earliest time for each commodity being able to be delivered to its destination. Constraint (15) is the nonnegative flow of each commodity transported by vehicles on each mode, arc, and time period. Constraint (16) is the nonnegative and integral flows of vehicles and unused vehicles on each mode, arc, and time period.

3.5 Solution Procedures of HA

Before applying Lagrangian relaxations, the original physical network needs to be transformed into the transformed network. **Problem (P)** is formulated according to the transformed time-space network. Constraint (10) of **Problem (P)** in Phase 1 is relaxed

by Lagrangian relaxations and the new problem becomes the **Lagrangian Dual** (**LD**) problem. The **LD** problem is separated into two subproblems. One subproblem consists of variables of commodities and related constraints; the other includes variables of vehicles and related constraints. The former is a multi-commodity generalized flow problem, which is a linear programming problem. Therefore, it can easily be solved by the simplex method. Then, values of commodities in **Problem (P)** are substituted by those of commodities from the simplex method. After solving **Problem (P)**, whether or not the upper bound needs to be updated is checked. The latter is a multi-vehicle integer problem and can be solved by the Mixed Integer Problem (MIP) module in the CPLEX software. Similar to the former, values of vehicles in **Problem (P)** are substituted by those of vehicles from the Mixed Integer Problem module. After solving **Problem (P)**, whether or not the upper bound needs to be updated is checked.

If the sum of objective values of two subproblems is greater than the lower bound, the new lower bound is set as the sum. It checks the stop criteria, $GAP_1 = (LB - UB) / LB \leq$ threshold 1 (ε_1). If it satisfies the stop criteria, the procedure is terminated. Otherwise, the number of iterations is checked.

If it is less than max iterations, the step size, λ, needs to be updated and μ needs to be recomputed by the new value of λ. Moreover, σ is divided by a factor of 2 if the lower bound does not improve in five consecutive iterations. The number of iterations is to increase one. After updating λ and μ, a new **LD** problem is formulated and the above procedures are repeated. Otherwise, if $GAP_1 >$ threshold 2 (ε_2), then enter Phase 2. Otherwise, the procedure is terminated. Here, ε_1 is less than ε_2.

Two heuristic methods in Phase 2 are proposed to search for improved feasible solutions of upper bounds. The concept of Phase 2 is to decompose a multi-commodity, multi-mode problem into multiple single-commodity, single-mode subproblems. **Problem (P)** is a cost minimization problem and its objective function is related to transportation, holding, and overdue-date costs. Therefore, one method is based on early due date (EDD) with overdue-date costs to sort commodities. The other is based on total transportation costs to sort commodities by high costs first. Hence, two methods provide two sorting sequences.

According to each sorting sequence, the first commodity is assigned to the cheapest available mode and this subproblem is solved by the CPLEX software. After solving a single-commodity, single-mode problem, remaining vehicles on each arc are calculated. The second commodity is assigned to the cheapest available mode and the above procedures are repeated until all remaining commodities are assigned. The reason for computing remaining resources is to avoid over-the-limit of vehicles on any arc and to guarantee obtaining a feasible solution.

From each single-commodity, single-mode problem, optimal values of commodities are obtained and substituted into **Problem (P)**. Due to two methods, two optimal solutions of **Problem (P)** are found. If the minimum of two optimal solutions is less than the upper bound, it is set as the new upper bound and the procedure is terminated. Otherwise, there is no improvement on the upper bound in Phase 2 and the procedure is also terminated. Fig. 4 is the flowchart of **HA**.

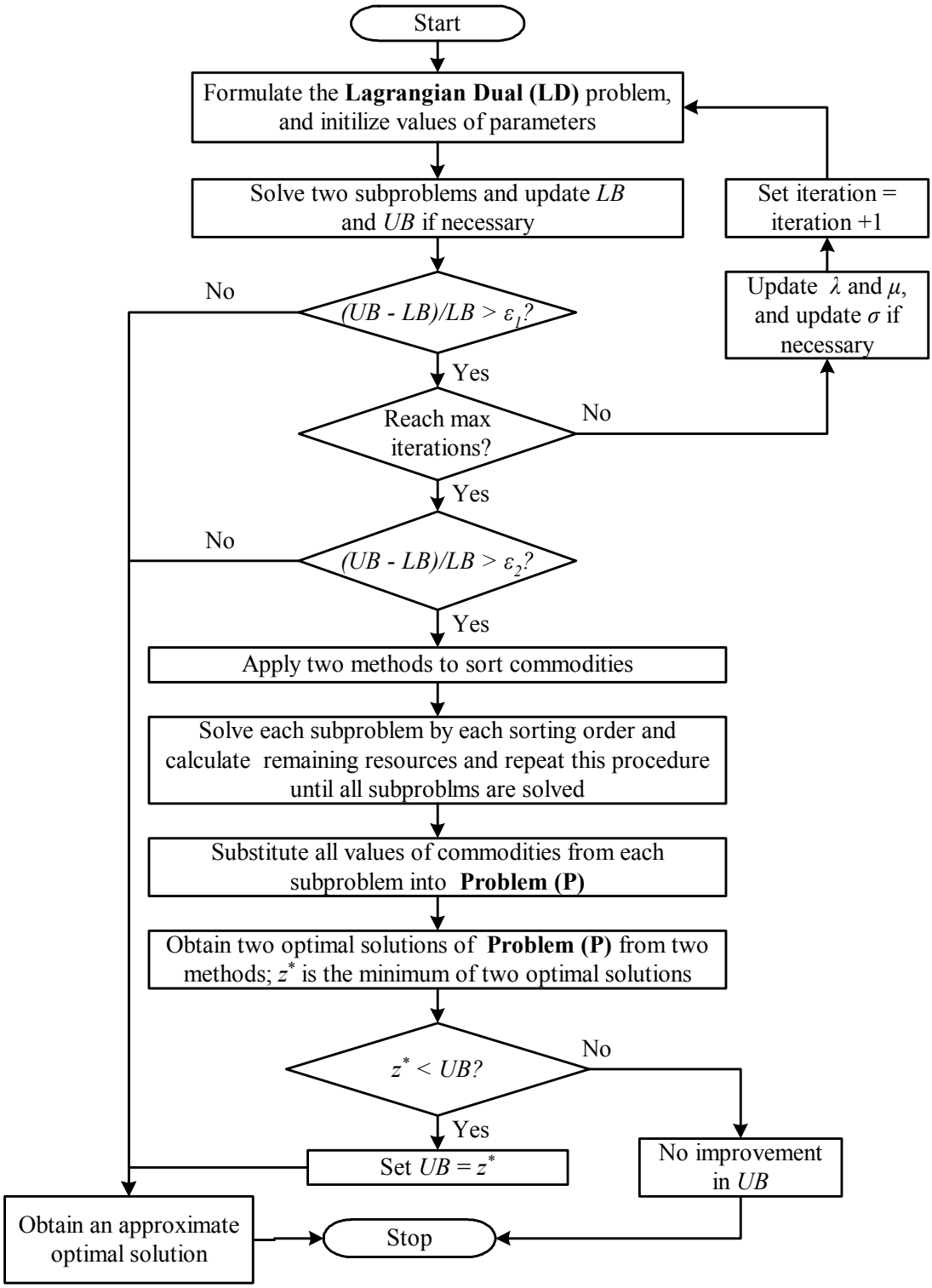


Fig. 4. Flowchart of **HA**.

3.6 Summary

This chapter contains the scope of problems, assumptions, and formulations. It also proposes a solution approach and its flowchart figure helps understand fundamental procedures. The solution approach in Phase 1 can provide lower and upper bounds. In addition, if the lower bound is a feasible solution of **Problem (P)**, it must be an optimal solution of **Problem (P)**. Moreover, the upper bound must be a feasible solution if it exists.

Phase 2 provides two opportunities to improve the upper bound in order to enhance the quality of the solution of **HA**. Detailed solution procedures are described in Chapter IV. An example for executing an iteration of Phase 1 and 2 of **HA** is also shown in Chapter IV.

CHAPTER IV

DEVELOPMENT OF SOLUTION PROCEDURES OF HA

4.1 Introduction

The purpose of this chapter is to develop a heuristic algorithm, **HA**, to solve multi-commodity, multi-mode generalized networks with time windows problems. **HA** is a two-phase procedure. Phase 1 is based on Lagrangian relaxations with subgradient methods and provides lower and upper bounds for **Problem (P)**. The upper bound is also a feasible solution for **Problem (P)**. There is a threshold (ε_2) between Phase 1 and 2. If $GAP_1 > \varepsilon_2$, then it enters Phase 2. Otherwise, the procedure is terminated. Two heuristic methods, early due date with overdue-date costs and total transportation costs, are used to search for improved upper bounds. Fig. 5 is the conceptual flowchart of **HA**.

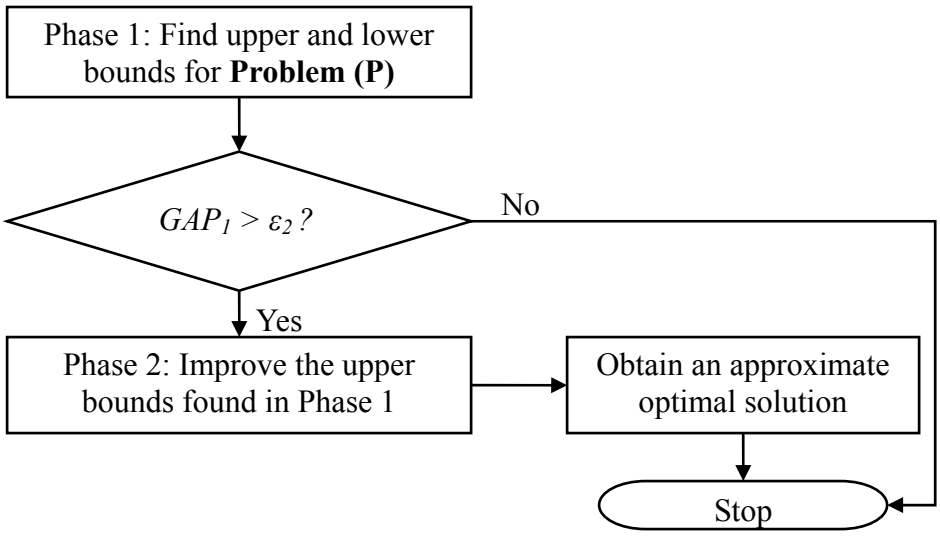


Fig. 5. Conceptual flowchart of **HA**.

There are two network applications for **HA** in this research. One is a 1-origin, 2-destination, and 3-mode network seen in Fig. 6; the other is a 2-origin, 3-destination, and 4-mode network shown in Fig. 7.

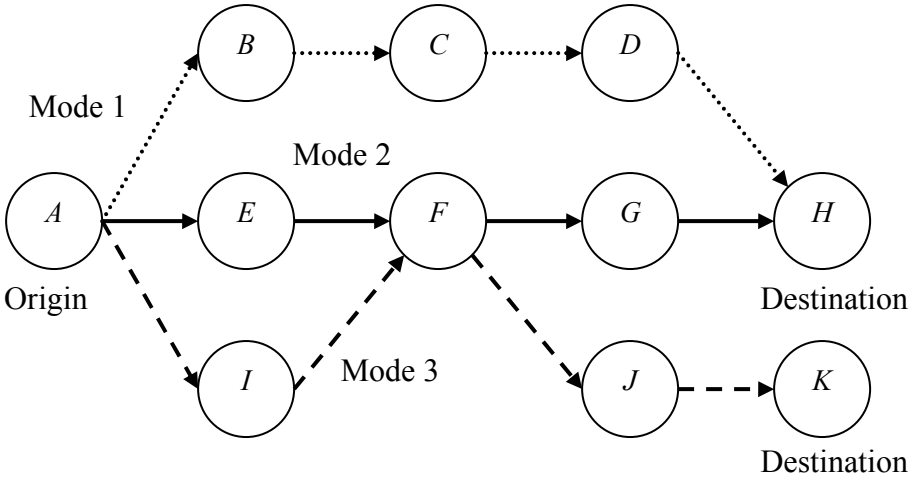


Fig. 6. 1-origin, 2-destination, and 3-mode network.

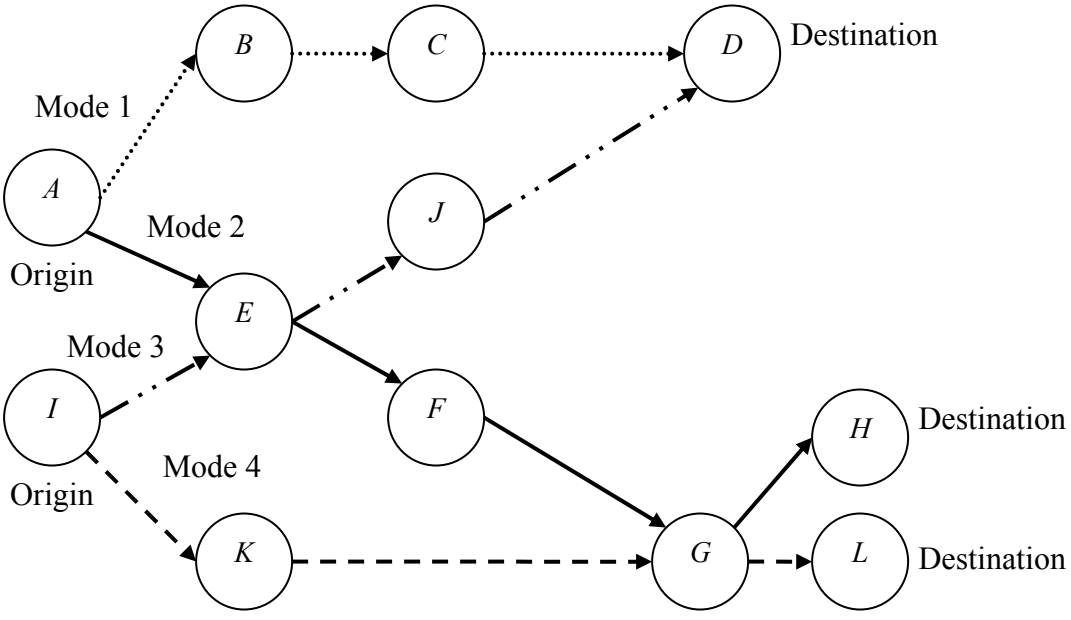


Fig. 7. 2-origin, 3-destination, and 4-mode network.

4.2 Procedures of HA

Problem (P) is reformulated in Phase 1 into the **Lagrangian Dual (LD)** problem. The **LD** problem can be separated into two subproblems. The first subproblem contains variables of commodities and related constraints of commodities. Since variables of commodities are nonnegative, this subproblem is a linear programming problem, which can easily be solved by the simplex method. The second subproblem includes variables of vehicles and related constraints of vehicles. Since variables of vehicles are required to be integers, this subproblem is an integer problem, which can be solved by the Mixed Integer Problem module (MIP) in the CPLEX software. Detailed procedures of **HA** are as follows:

Phase 1

Step 1. Set initial values of s, μ^s, UB, LB, α, ε_1, ε_2, and σ, where s means the number of iteration, μ^s is a Lagrangian multiple, UB and LB are the current upper and lower bound, α is the max iteration, ε_1 and ε_2 are the given thresholds, and σ is a scale and $0 < \sigma < 2$.

Step 2. Relax Constraint (10) in **Problem (P)**, multiply by μ^s, and add this item to the objective function. It becomes the **LD** problem.

Step 3. Separate the **LD** problem into two subproblems. The first subproblem contains variables of commodities and related constraints of commodities, including Constraints (8), (14), and (15). The second subproblem includes variables of vehicles and related constraints of vehicles,

including Constraints (9), (11), (12), (13), and (16).

Step 4. Add a new constraint, (17), to the first subproblem and solve it by the simplex method in CPLEX.

Step 5. Solve **Problem (P)** with fixed values of commodities obtained from Step 4. If it is optimal and less than *UB*, set it as a new *UB*.

Step 6. Add two new constraints, (18) and (19), to the second subproblem and solve it by the Mixed Integer Problem module (MIP) in CPLEX.

Step 7. Solve **Problem (P)** with fixed values of vehicles obtained from Step 6. If it is optimal and less than *UB*, set it as a new *UB*.

Step 8. Sum up optimal solutions from Step 4 and Step 6. If the sum is greater than *LB*, set it as a new *LB*.

Step 9. Check stop criteria. If $(UB - LB) / LB \leq \varepsilon_1$ for a small $\varepsilon_1 > 0$, an approximate optimal solution is found and the procedure is terminated. Otherwise, go to Step 10.

Step 10. If $s <$ max iterations (α), then calculate λ^s, update μ^{s+1} by $\mu^{s+1} = \max(0, \mu^s + \lambda^s)$, and set $s = s + 1$. Here, λ^s is the step size in Proposition 2. Go to Step 11. Otherwise, go to Step 12.

Step 11. If *LB* is not improved in five consecutive iterations, let $\sigma = \sigma/2$ and go back to Step 2. Otherwise, directly go back to Step 2.

Step 12. If $(UB - LB) / LB \leq \varepsilon_2$, the current upper bound is an approximate optimal solution. Otherwise, go to Phase 2. Fig. 8 is the flowchart of Phase 1.

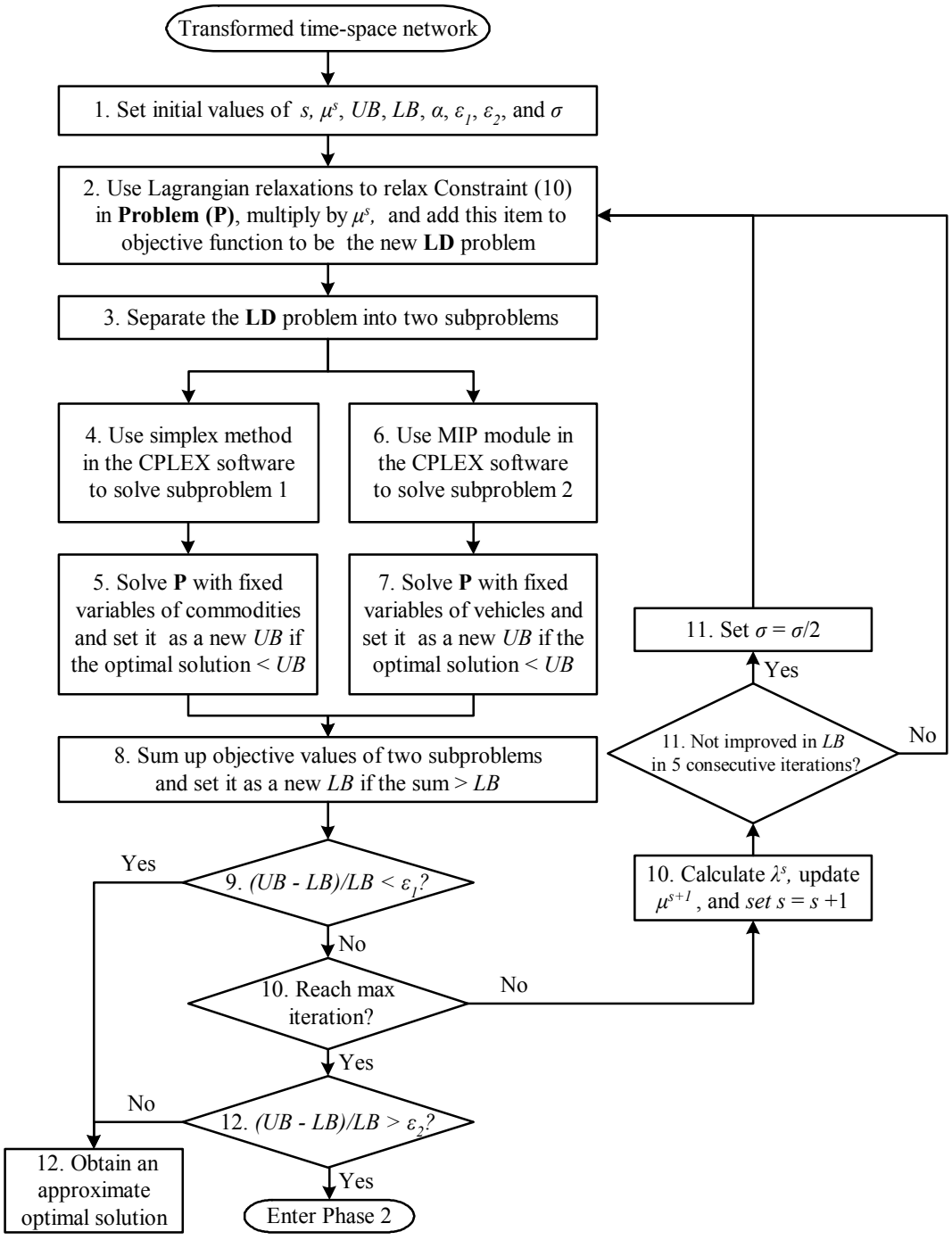


Fig. 8. Flowchart of Phase 1.

Phase 2 proposes two methods to sort commodities and to dispatch each commodity to the cheapest available mode. The basic concept of Phase 2 is to decompose a multi-commodity, multi-mode problem into multiple single-commodity, single-mode subproblems. Vehicles are viewed as resources in the model. According to two methods to sort commodities, it obtains two sequences. For each sequence, the first-order commodity is assigned to the cheapest available mode. Then, each subproblem is solved by the CPLEX software and remaining resources on all arcs are computed. The second-order commodity is assigned to the cheapest available mode and the above procedures are repeated until there is no commodity left.

After all subproblems of each method are solved, values of commodities in **Problem (P)** are substituted by values of commodities obtained from all subproblems of each method. Two problems, **P**, are solved. If the minimum, z^*, of two problems, **P**, is less than *UB*, then a new *UB* is set to z^*. Hence, *UB* is improved and GAP_1 is reduced.

The reason for decomposing a multi-commodity, multi-mode problem into multiple single-commodity, single-mode subproblems is that it is easier to solve each subproblem than the original problem. In addition, if a subproblem is feasible, it is guaranteed to have an optimal solution. Two methods can provide two opportunities to improve GAP_1.

If vehicles are viewed as resources, multiple single-commodity, single-mode problems are similar to multi-job, multi-machine scheduling problems. In order to search for a feasible and near-optimal solution, early due date and total transportation costs are applied according to dispatching rules. They are used to sort commodities and to assign each commodity to the cheapest available mode. Since the objective function of

Problem (P) includes transportation, holding, and overdue-date costs related to commodities, the early due date (EDD) rule focuses on reducing overdue-date costs. If due dates of commodities are tied, the sequence order is sorted by high overdue-date costs first. The total transportation costs (TTC) rule consists of transportation costs of commodities and transportation costs of vehicles divided by the number of commodities transported by vehicles. If total transportation costs are tied, the sequence order is sorted arbitrarily to break a tie. Fig. 9 is the flowchart of Phase 2.

Phase 2

Step 1.1 Apply the early due date rule to sort commodities into $\{1, \ldots, k\}$. If due dates of commodities are tied, choose the commodity with the highest overdue-date cost to break it.

Step 1.2 Set $j = 1$.

Step 1.3 Assign commodity j to the cheapest available mode and solve this subproblem j to obtain optimal values of commodities, x_j^*.

Step 1.4 Calculate remaining arc resources of all arcs of vehicles on all modes.

Step 2 If $j = k$, then go to Step 3. Otherwise, set $j = j + 1$. Go back to Step 1.3.

Step 3 Substitute values of commodities from all subproblems in Step 1.3 into **Problem (P)**.

Step 4 Solve **Problem (P)** in Step 3 and obtain an optimal solution, z_1^*.

Step 5.1 Apply the total transportation costs rule to sort commodities into $\{1, \ldots, k\}$. If tie, break it arbitrarily.

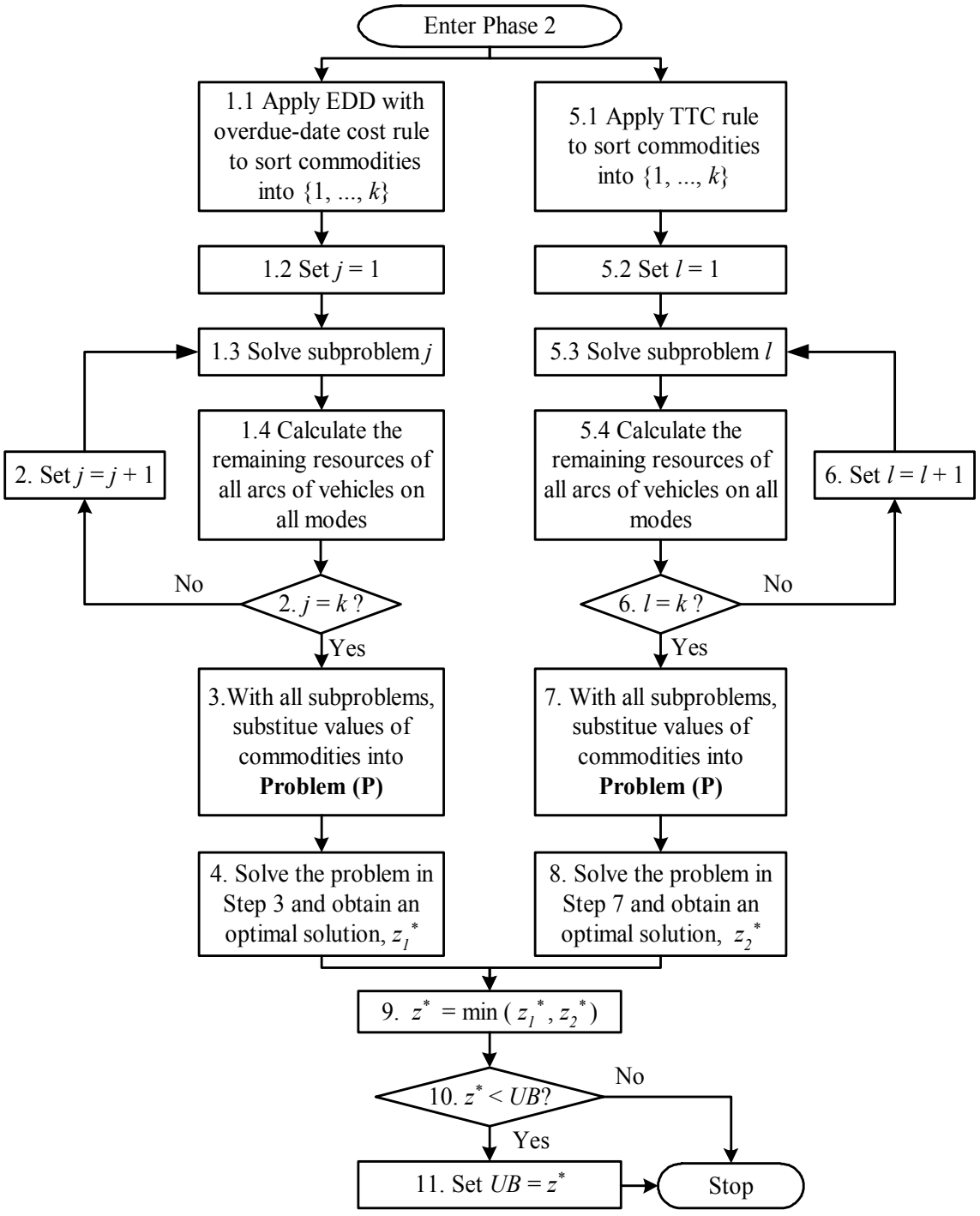


Fig. 9. Flowchart of Phase 2.

Step 5.2 Set $l = 1$.

Step 5.3 Assign commodity l to the cheapest available mode and solve this subproblem l to obtain optimal values of commodities, x_l^*.

Step 5.4 Calculate remaining resources of all arcs of vehicles on all modes.

Step 6 If $l = k$, then go to Step 7. Otherwise, set $l = l + 1$. Go to Step 5.3.

Step 7 Substitute values of commodities from all subproblems in Step 5.3 into **Problem (P)**.

Step 8 Solve **Problem (P)** in Step 7 and obtain an optimal solution, z_2^*.

Step 9 Set $z^* = \min (z_1^*, z_2^*)$.

Step 10 If $z^* < UB$, then go to Step 11. Otherwise, there is no improved solution found for the current upper bound and the procedure is terminated.

Step 11 Set a new $UB = z^*$ and the procedure is terminated.

4.3 Example of HA

Fig. 6 is used as the original network to execute one iteration of **HA**. Before applying **HA**, this network needs to be changed into two transformed networks, Fig. 10 and 11. Fig. 10 is the transformed network for commodities and Fig. 11 is the transformed network for vehicles. This example includes 5 commodities and 3 modes. After the transformation, origin nodes are *A1*, *A2*, and *A3* and destination nodes are *H1*, *H2*, and *K*. The planning time periods are 20 periods. Appendix A1 and A2 are the transformed networks of Fig 7. for commodities and vehicles.

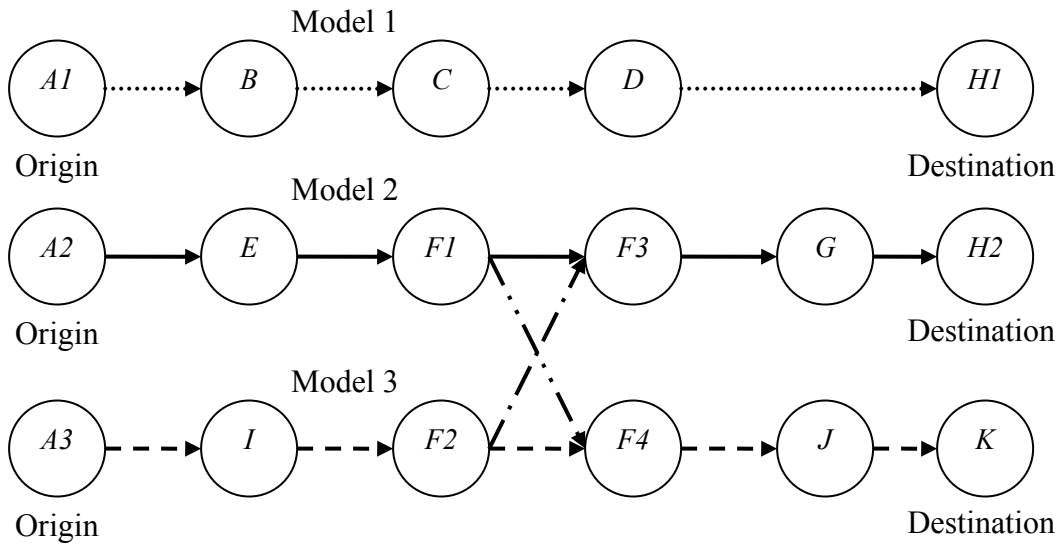


Fig. 10. Transformed network of Fig. 6 for commodities.

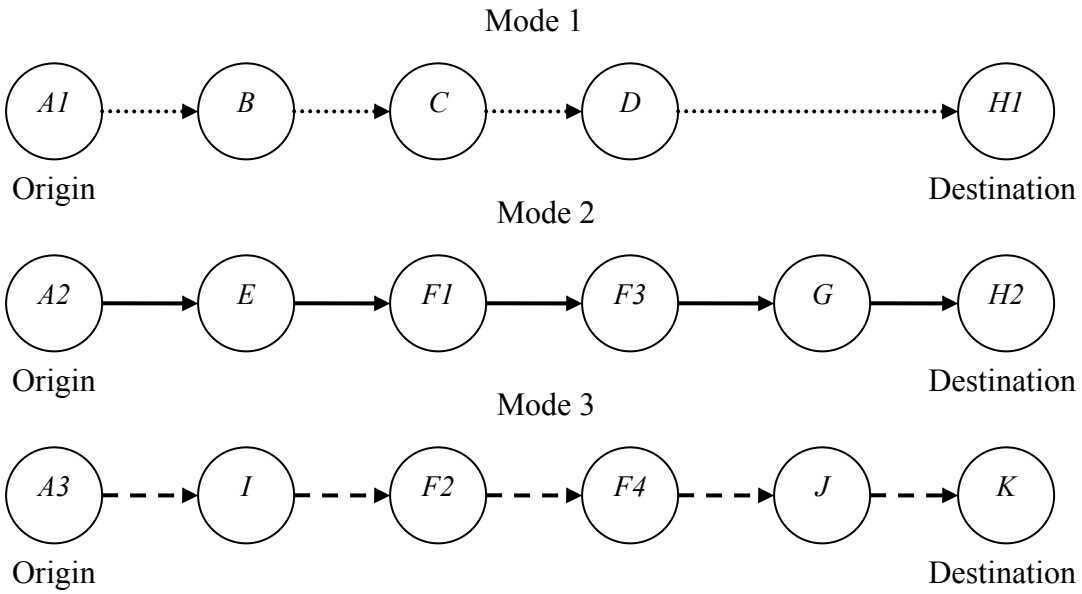


Fig. 11. Transformed network of Fig. 6 for vehicles.

4.3.1 Example of Phase 1

Initially, *LB* is set as -1.0e+018, *UB* as 1.0e+012, μ^0 as 0, α as 200, *s* as 1, ε_1 as 0.0001, ε_2 as 0.1, and σ as 1 in Step 1 of Phase 1. After Step 2 and Step 3, the **LD** problem is separated into two subproblems. Step 4 adds a new constraint, (17). Constraint (17) represents the minimum between capacities and upper bounds of commodities on each mode, arc, and time period. It ascribes from Constraints (10), (12), and (13). The purpose of adding Constraint (17) is to reduce the feasible region and to shorten solving times. Then, the new problem is solved by the CPLEX software. The optimal solution is round to 1,527,031. Since, Step 5 obtains an infeasible solution, *UB* is not updated.

$$\sum_k \left(x^{km}_{ijtt'} \,/\, V^{km} \right) \le \min\left(CA^{m}_{ijtt'}, U^{m}_{ijtt'}\right) \qquad \text{for all } m, i, j, t, t' \qquad (17)$$

Two new constraints, (18) and (19), are added in Step 6. Constraint (18) is desired for vehicles for demands and Constraint (19) is the restriction of vehicles for delivery dates. Both constraints help reduce the feasible region. An optimal integer solution is 415,080. Step 7 obtains an infeasible solution. The sum in Step 8 is 1,942,111, which is greater than the lower bound. Thus, a new lower bound is set as 1,942,111. $(UB - LB) / LB$ = (1e+12 − 1,942,111) / 1,942,111 = 514,903 > 0.0001 = ε_1 in Step 9. Thus, go to Step 10. Since $s = 1 < 200$ in Step 10, values of λ^1 are needed to be calculated. The step size, λ^1, is (1e+12 − 1,942,111) / 38,846 = 25,742,626, where $\left\| \hat{g}^k \right\|^2$ is 38,846. Then, $\mu^1_{ijtt'}$ is updated as max (0, $\mu^0_{ijtt'}$ + 25,742,626) for all *i, j, t, t'*. Then, *s* is set as 2. It

has found a new *LB* in this iteration and the procedure goes back to Step 2. An iteration of Phase 1 is finished. Here, DE^k is the demand of *k*.

$$\sum_{t,t'} y^{m}_{i\,j\,t\,t'} * V^{\,k\,m} \geq DE^{\,k} \qquad \text{for } j \text{ in destination set} \quad (18)$$

$$y^{m}_{i\,j\,t\,T} = 0 \qquad \text{for all } m, i, j, t < ED^k \quad (19)$$

4.3.2 Example of Phase 2

After 200 iterations of Phase 1, *LB* is 1,942,111 and *UB* is 2,237,394. $GAP_1 = (UB - LB) / LB = 0.1520 > 0.1 = \varepsilon_2$. Therefore, the procedure is needed to enter Phase 2. The EDD rule in Step 1.1 is applied to sort 5 commodities. The sequence is 2 → 4 → 1 → 3 → 5. Commodity 2 is assigned to mode 2 in Step 1.3. Commodity 4 is assigned to mode 3. Commodity 1 is assigned to mode 1. Commodity 3 is assigned to mode 3. Commodity 5 is assigned to mode 3. Therefore, $z_1^* = 2{,}154{,}408$ in Step 4.

Similarly, from Step 5.1 to Step 5.4, the sequence by the TTC rule is 2 → 3 → 1 → 5 → 4. Commodity 2 is assigned to mode 2. Commodity 3 is assigned to mode 3. Commodity 1 is assigned to mode 1. Commodity 5 is assigned to mode 3. Commodity 4 is assigned to mode 3. Hence, $z_2^* = 2{,}154{,}408$ in Step 8. $z^* = \min(z_1^*, z_2^*) = 2{,}154{,}408$ in Step 9. Since $z^* = 2{,}154{,}408 < 2{,}237{,}394$ in Step 10, the procedure goes to Step 11. A new *UB* is set as 2,154,408 in Step 11 and the procedure is terminated. Hence, Phase 2 indeed obtains an improved upper bound.

4.4 Computational Complexity of HA

Since the multi-commodity, multi-mode networks with a time windows problem is a mixed integer problem, it is NP-complete. The computational complexity of **HA** can be separated by two phases. Two subproblems and two problems, **P**, with fixed values need to be solved in Phase 1. The first subproblem is related to commodities and the second subproblem is related to vehicles. The former is a multi-commodity generalized flow problem and can be solved in polynomial times. The latter is a multi-mode integer flow and NP-complete problem. **Problem (P)** with fixed commodity values is a multi-mode integer problem, which is a NP-complete problem. **Problem (P)** with fixed vehicle values is a multi-commodity generalized problem, which can be solved in polynomial times. Operations for calculating values of λ and updating values of μ can be done in polynomial times.

Multiple single-commodity, single-mode subproblems in Phase 2 need to be solved. Each subproblem can be solved in polynomial times. Although the complexity of **HA** is still NP-complete, total running times of **HA** are very stable under the same scenario and are quicker than those of solving **Problem (P)** directly by the CPLEX in the worst case. Detailed total running times of different scenarios are summarized in Chapter V.

4.5 Summary

In general, the traditional Lagrangian relaxations with subgradient methods only provide lower bounds for **Problem (P)**. There is no way to know how tight lower bounds are unless an optimal solution of **Problem (P)** is obtained. Nevertheless, Phase 1 of **HA**

provides both lower and upper bounds for **Problem (P)**. If GAP_1 is too large, there are two methods in Phase 2 applied to improve the upper bound. If an improved upper bound is found, GAP_1 will be reduced. In other words, it implies that the quality of the solution provided by **HA** is enhanced.

CHAPTER V

COMPUTATIONS AND ANALYSES IN HA

5.1 Introduction

The computer used in this research is a computer with Pentium 4 CPU 3.2 GHz 512 MB of RAM. **HA** is written by the AMPL 9.0 scripts and its solver is the CPLEX 9.0 software. Two network applications, Fig. 6 and 7, are used for generating different scenarios. The number of commodities and the length of planning time periods are the given parameters. Results include values of lower and upper bounds, total solving times, and total running times. Here, total solving times consist of solving time by the "solve" command in the AMPL codes. Total running times include input/output times, operation times for calculating values of λ and updating values of μ, and solving times by the CPLEX solver.

5.2 Results of Phase 1

For the network application of Fig. 6, it runs the 5-, 10-, and 15-commodity for 10- and 20-planning-time-period scenarios. Max iterations are set as 200 iterations for all runs. After Phase 1, results of solutions in 5-commodity, 10- and 20-planning-time-period scenarios are summarized in Table 3 and 4, respectively. Values of lower and upper bounds are rounded to integers. The other scenarios are summarized in Appendix B1 and B2.

Table 3
5-commodity, 10-planning-time-period scenario in Phase 1

Seed	*LB*	*UB*	GAP_1	Total solving times (seconds)	Total running times (seconds)
1	1,285,612	1,296,482	0.0085	71.14	274.50
2	1,009,004	1,016,305	0.0072	70.05	272.86
3	1,164,821	1,173,927	0.0078	72.08	277.28
4	1,047,729	1,064,333	0.0158	73.09	273.17
5	852,385	856,779	0.0052	73.42	277.09
6	1,213,930	1,217,122	0.0026	71.69	274.03
7	**1,087,437**	**1,235,441**	**0.1361**	**70.34**	**284.31**
8	1,137,633	1,145,321	0.0068	71.70	274.80
9	1,002,444	1,008,527	0.0061	72.13	279.53
10	1,033,500	1,040,782	0.0070	74.30	268.75
		Average	0.0203	71.99	275.63
		Variances	0.0017	1.7775	17.8950

Table 4
5-commodity, 20-planning-time-period scenario in Phase 1

Seed	*LB*	*UB*	GAP_1	Total solving times (seconds)	Total running times (seconds)
1	1,990,812	2,039,472	0.0244	90.89	433.91
2	1,457,712	1,470,399	0.0087	91.34	467.70
3	1,752,458	1,766,059	0.0078	89.31	490.30
4	1,582,588	1,591,935	0.0059	93.17	436.63
5	1,437,083	1,449,538	0.0087	89.06	452.17
6	**1,942,111**	**2,237,394**	**0.1520**	**89.20**	**452.14**
7	**1,222,970**	**1,549,062**	**0.2666**	**96.06**	**530.31**
8	**1,726,965**	**1,915,282**	**0.1090**	**100.83**	**506.38**
9	1,745,483	1,762,788	0.0099	136.09	444.08
10	**1,679,690**	**1,990,176**	**0.1848**	**90.14**	**497.61**
		Average	0.0778	96.61	471.12
		Variances	0.0090	206.2508	1,094.7845

GAP_1 of seed 7 in Table 3 and seeds 6, 7, 8, and 10 in Table 4 are over the threshold of $\varepsilon_2 = 0.1$. These runs are needed to enter Phase 2. GAP_1 of other runs are within the threshold of $\varepsilon_2 = 0.1$ and procedures are terminated. Total running times of 5-, 10-, and 15-commodity, 10-planning-time-period scenarios are shown in Fig. 12. It shows two

observations where the more the number of commodities, the more total running times it takes and variances of total running times of 15-commodity scenarios are larger than those of total running times of 5- and 10-commodity scenarios. Fig. 13 displays that total running times of 5-, 10-, and 15-commodity, 20-planning-time-period scenarios. It has similar observations to total running times of 5-, 10-, and 15-commodity, 10-planning-time-period scenarios.

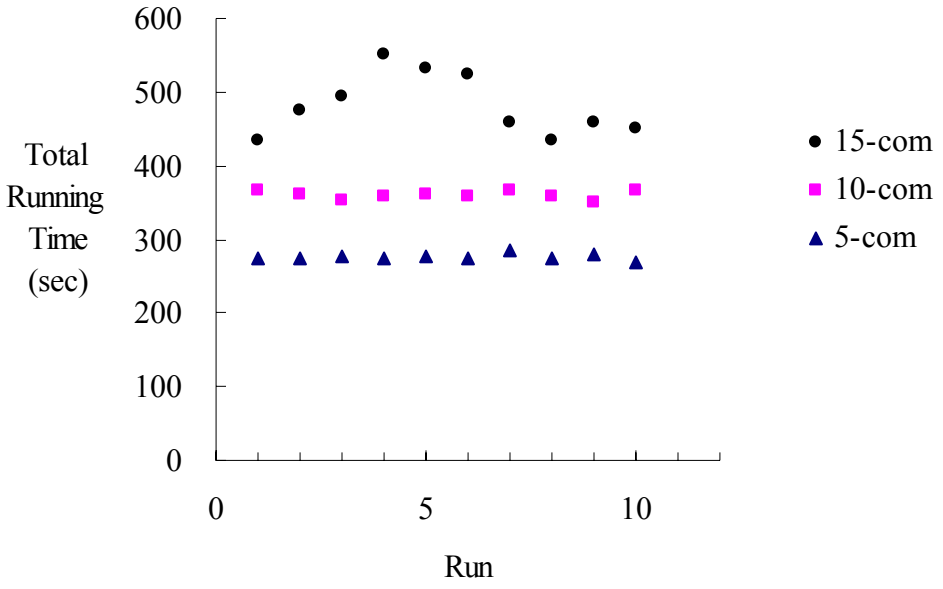


Fig. 12. Total running times of 5-, 10-, and 15-commodity, 10-planning-time-period scenarios.

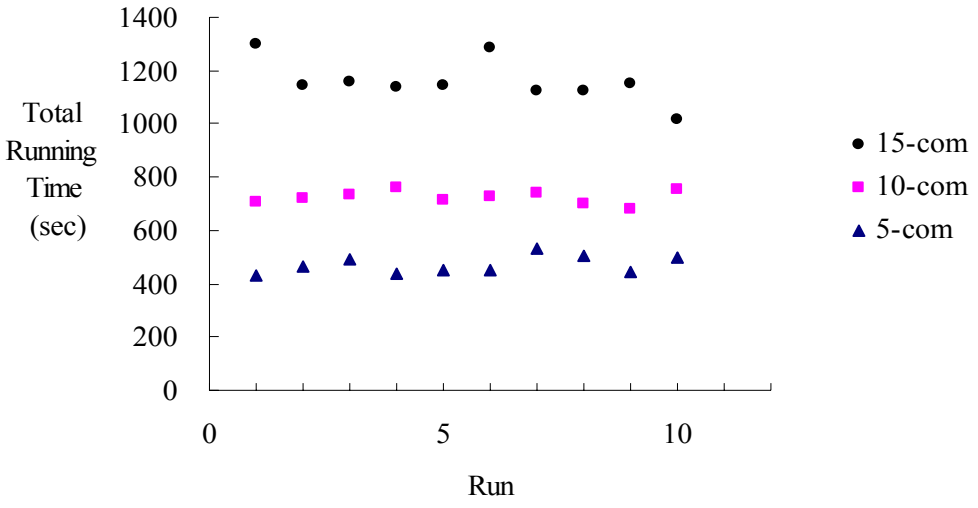


Fig. 13. Total running times of 5-, 10-, and 15-commodity, 20-planning-time-period scenarios.

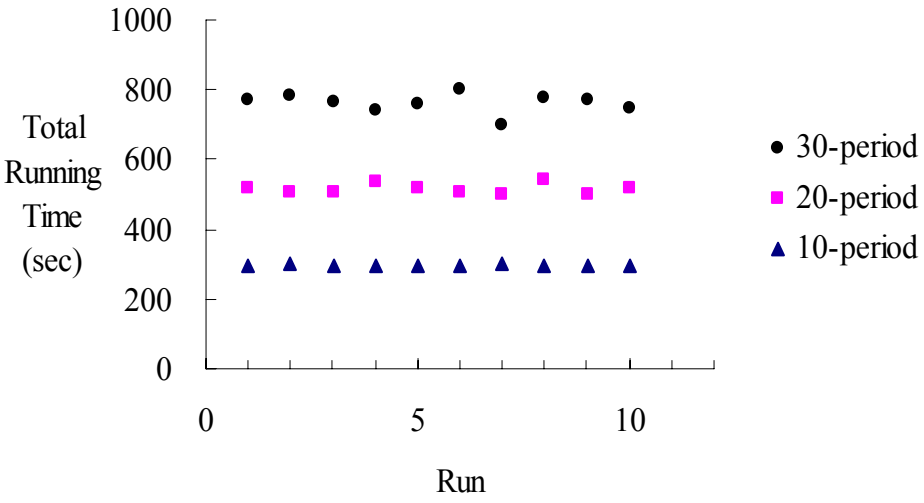


Fig. 14. Total running times of 6-commodity, 10-, 20-, and 30-planning-time-period scenarios.

Fig. 14 and 15 are network applications of Fig. 7. Fig. 14 compares total running times of 6-commodity, 10-, 20-, and 30-planning-time-period scenarios. Fig. 15

compares total running times of 12-commodity, 10-, 20-, and 30-planning-time-period scenarios. Fig. 14 and 15 display that when the length of planning time periods increases, total running times also increase.

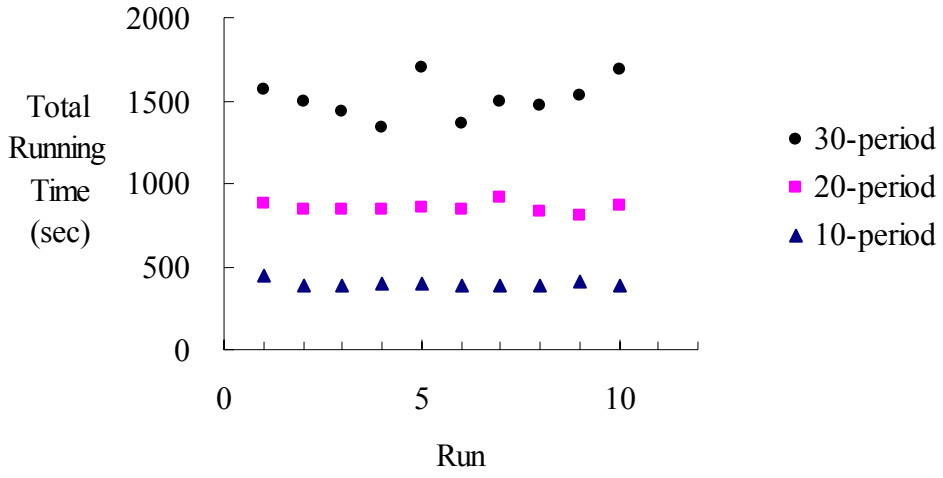


Fig. 15. Total running times of 12-commodity, 10-, 20-, and 30-planning-time-period scenarios.

5.3 Results of Phase 2

The network application of Fig. 6 has 9 runs are needed to enter Phase 2 of **HA**. Similarly, the network application of Fig. 7 has 12 runs are needed to enter Phase 2. Results of Phase 2 are summarized in Table 5 and 6. After Phase 2, only one upper bound of these runs is not improved. Furthermore, GAP_1 of all runs of 5-, 10-, and 15-commodity scenarios are between [0.0019, 0.1393] and the average GAP_1 is 0.0190. Similarly, GAP_1 of all runs of 6- and 12-commodity scenarios are between [0.0007, 0.3330] and the average GAP_1 is 0.0288.

Table 5
GAP_1 in Phase 2 for the network applications of Fig. 6

Commodity_ Period_Seed	EDD with overdue-date cost	Total transportation cost	New *UB*	New GAP_1
5_10_7	1,264,878	1,264,878	**Not found**	–
5_20_6	2,154,408	2,154,408	2,154,408	0.1093
5_20_7	1,686,934	1,314,821	1,314,821	0.0751
5_20_8	2,084,387	1,813,131	1,813,131	0.0499
5_20_10	1,704,900	1,704,900	1,704,900	0.0150
15_10_4	2,905,414	2,916,061	2,905,414	0.0338
15_10_6	3,582,038	3,588,967	3,582,038	0.0201
15_20_3	4,704,616	4,419,479	4,419,479	0.0685
15_20_6	5,387,581	4,990,029	4,990,029	0.1393

Table 6
GAP_1 in Phase 2 for the network applications of Fig. 7

Commodity_ Period_Seed	EDD with overdue-date cost	Total transportation cost	New *UB*	New GAP_1
6_10_7	1,170,022	1,170,022	1,170,022	0.0151
6_20_8	1,771,750	1,771,750	1,771,750	0.0172
6_30_9	4,828,038	4,683,088	4,683,088	0.1213
12_10_1	2,358,806	2,358,806	2,358,806	0.2759
12_10_4	3,771,023	3,820,354	3,771,023	0.3330
12_10_9	2,223,956	2,240,105	2,223,956	0.2018
12_20_1	3,143,917	3,143,917	3,143,917	0.0181
12_20_7	3,101,188	3,100,805	3,100,805	0.0248
12_20_10	3,149,989	3,149,989	3,149,989	0.0048
12_30_2	7,773,376	7,721,244	7,721,244	0.0051
12_30_4	9,173,844	9,445,223	9,173,844	0.0953
12_30_10	7,628,901	7,630,530	7,628,901	0.0389

GAP_1 of 5-, 10-, and 15-commodity scenarios for network applications of Fig. 6 are displayed in Fig. 16(a), (b), and (c), respectively. Fig. 16 (a) GAP_1 of all runs are within 0.04 except 4 runs. Fig. 16 (b) has 1 run with larger GAP_1 than other runs. Fig. 16 (c) has 2 runs with larger GAP_1 than other runs. Similarly, GAP_1 of 6- and 12-commodity scenarios for network applications of Fig. 7 are displayed in Fig. 17 (a) and (b). Fig. 17 (a) has 2 runs with larger GAP_1 than other runs. Fig. 17 (b) has 4 runs with larger GAP_1

than other runs. Therefore, there are 12 runs needed to advance analyses in the later section.

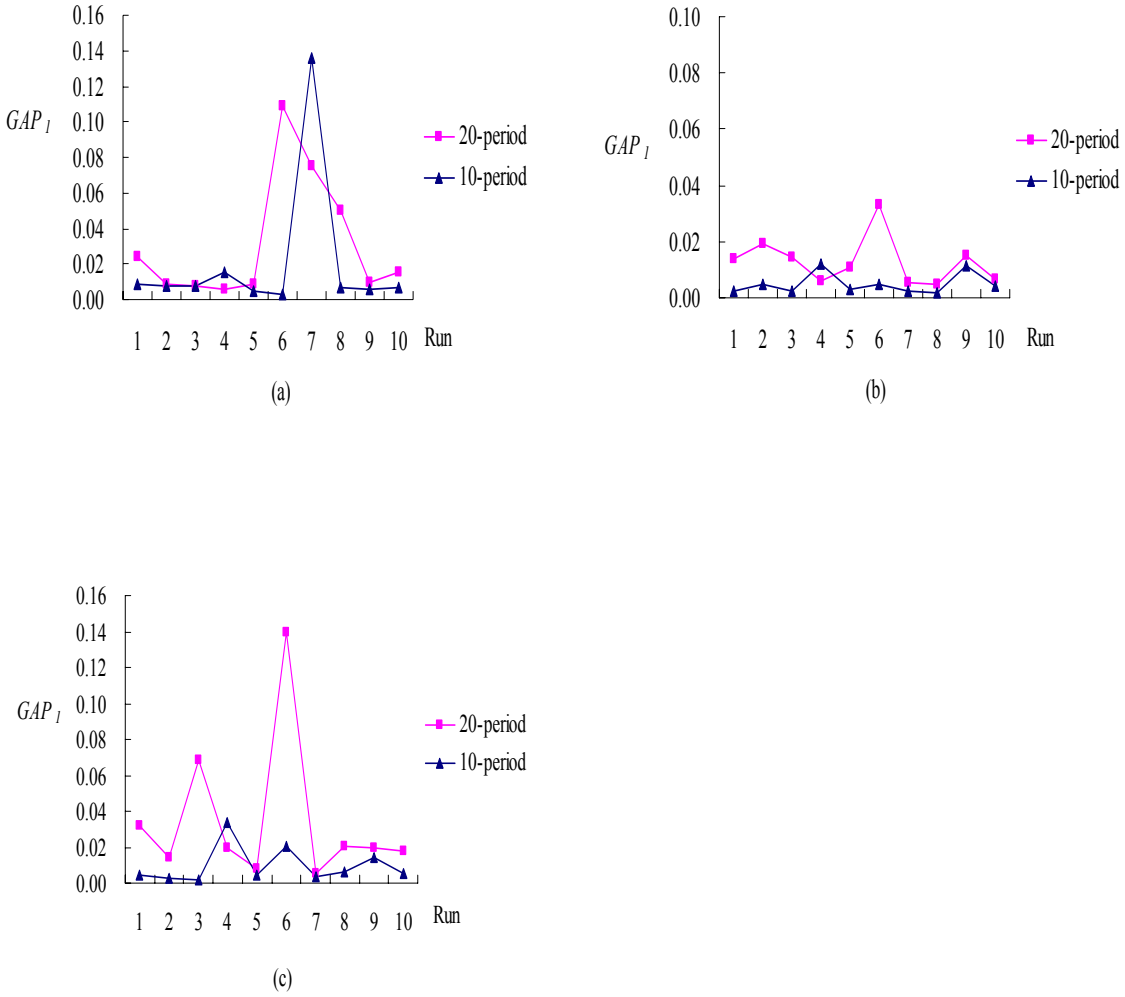


Fig. 16. (a) GAP_1 of 5-commodity, 10- and 20-planning-time-period scenarios. (b) GAP_1 of 10-commodity, 10- and 20-planning-time-period scenarios. (c) GAP_1 of 15-commodity, 10- and 20-planning-time-period scenarios.

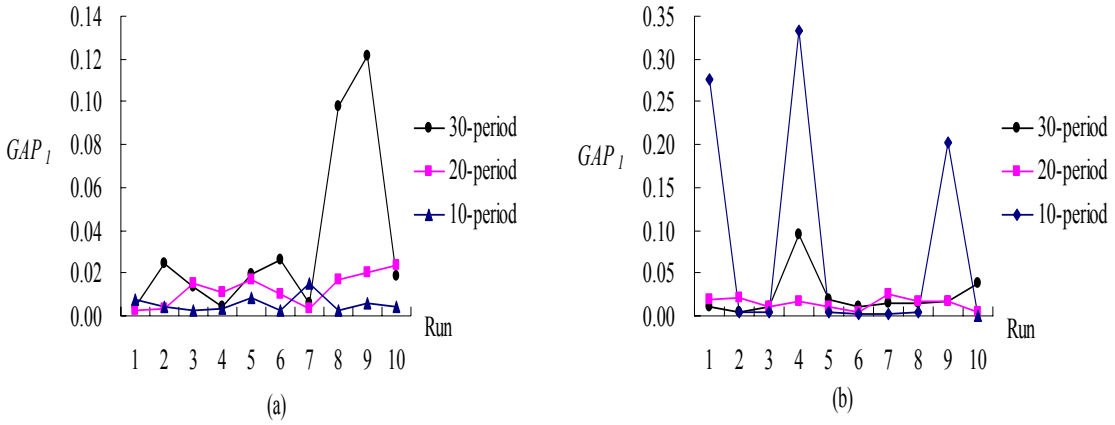


Fig. 17. (a) GAP_1 of 6-commodity, 10-, 20-, and 30-planning-time-period scenarios. (b) GAP_1 of 12-commodity, 10-, 20-, and 30-planning-time-period scenarios.

5.4 Analyses and Conclusions of HA

After Phase 2, 12 runs with larger GAP_1 are needed to calculate their optimal solutions. 12 original problems, **Problem (P),** are solved by the CPLEX software for comparison with **HA**. In the CPLEX software, it defines that optimal solutions are satisfied with either of three situations. The first condition is that value of absmipgap is 0. The second condition is that the value of mipgap is 0.0001. The third condition is that the value of timelimit is 1.0e+75 seconds. Absmipgap means the absolute value between best integer solution and best node solution found. Mipgap represents that |best node solution found − best integer solution| / (1.0 + |best node solution found|). Timelimit means the limit of total running times. If timelimit is too large, it will cost a lot of memory space and may cause the termination of a program because of running out of memory. Thus, the default value of timelimit in this research is 14,400 seconds. After the optimal solution is obtained, it can compare lower and upper bounds of **HA** with an

optimal solution.

There are two definitions given for analyzing relationships among the lower bound, upper bound, and optimal solution. GAP_2 is defined as $|UB - Optimal\ solution| / (Optimal\ solution)$ and the definition of GAP_3 is $(Optimal\ solution - LB) / (Optimal\ solution)$.

Table 7
GAP_2 and GAP_3 of 12 runs with larger GAP_1 after Phase 2

Commodity_ Period_Seed	*LB*	new *UB*	*OP*	GAP_2	GAP_3
5_10_7	1,087,437	1,235,441	1,219,893	0.0127	0.1086
5_20_6	1,942,111	2,154,408	2,019,508	0.0668	0.0383
5_20_7	1,222,970	1,314,821	1,313,273	**0.0012**	0.0688
5_20_8	1,726,965	1,813,131	1,768,013	0.0255	0.0232
15_20_3	4,136,007	4,419,479	4,391,821	0.0063	0.0582
15_20_6	4,379,719	4,990,029	4,685,205	0.0651	0.0652
6_30_8	4,892,077	5,369,557	4,965,393	0.0814	**0.0148**
6_30_9	4,176,342	4,683,088	4,514,575	0.0373	0.0749
12_10_1	1,848,676	2,358,806	2,145,408	**0.0995**	0.1383
12_10_4	2,828,994	3,771,023	3,476,939	0.0846	**0.1864**
12_10_9	1,850,534	2,223,956	2,030,451	0.0953	0.0886
12_30_4	8,375,532	9,173,844	8,504,906	0.0787	0.0152

GAP_2 and GAP_3 of these 12 runs are shown in Table 7. The only run, which can not find an improved solution in Phase 2, is the 7th run of the 5-commodity, 10-planning-time-period scenario. GAP_2 of this run is 0.0127. It means the gap between the upper bound and optimal solution is small. Therefore, it makes sense that there is no improvement on the upper bound found in Phase 2. Values of GAP_2 in Table 7 are between [0.0012, 0.0995]. Therefore, the maximal upper bound in both network

applications by **HA** is within 9.95% ranges of optimal solutions. Since threshold 2 is 0.1 for both applications, it means that GAP_2 of other runs are within 10% ranges of their optimal solutions. Hence, **HA** for both network applications indeed provides good upper bounds, which are close to 10% ranges of optimal solutions. Similarly, Values of GAP_3 are between [0.0148, 0.1864]. Since threshold 2 is 0.1 for both network applications, it means that GAP_3 of other runs must be within 10% ranges of their optimal solutions. Hence, lower bounds provided by **HA** for both network applications are close to 20% ranges of their optimal solutions.

Table 8
Total solving times (seconds) of Problem (P) in the network application of Fig. 6

Run	5_c_10_p	5_c_20_p	10_c_10_p	10_c_20_p	15_c_10_p	15_c_20_p
Seed 1	2.02	0.25	0.64	6,311.91	51.03	48.52
Seed 2	0.13	0.88	0.23	1.39	0.27	8.80
Seed 3	0.13	0.72	0.17	0.91	1.81	0.86
Seed 4	0.44	0.91	4,205.47	2.05	0.36	3.89
Seed 5	0.14	0.47	0.17	1.78	8,632.58	7,406.11
Seed 6	5.41	0.31	0.2	2.23	0.66	1.91
Seed 7	0.14	0.39	370.99	14,400.02	0.28	1.06
Seed 8	0.11	0.8	0.16	1.44	0.45	47.50
Seed 9	0.13	0.66	0.3	1.73	0.64	3.81
Seed 10	0.14	1.19	0.16	13,544.91	0.27	8.08
Average	0.88	0.66	457.85	3,426.84	868.84	753.05
Std.	1.70	0.30	1,321.92	5,899.47	2,727.95	2,337.72

Results for total solving times for optimal solutions are recorded in Table 8 and 9. The worst case of total solving times is over the limit, 14,400 seconds. This means that even through 14,400 seconds, some cases still can not find the optimal solution to meet

the stop criteria. It is shown that standard deviations of total solving times in Table 8 and 9 are large, except three of 5-commodity, 10-planning-time-period, 5-commodity, 20-planning-time-period, and 6-commodity, 10-planning-time-period scenarios.

Table 9
Total solving times (seconds) of Problem (P) in the network application of Fig. 7

Run	6_c_10_p	6_c_20_p	6_c_30_p	12_c_10_p	12_c_20_p	12_c_30_p
Seed 1	0.13	1.02	67.22	0.25	22.27	11.94
Seed 2	0.11	14,400.20	4,856.20	12.75	14,400.13	5,342.00
Seed 3	2.19	9.39	2.22	0.84	7,627.00	6,739.69
Seed 4	0.11	10.53	1.55	0.14	2.30	2.70
Seed 5	5.92	5,744.67	13.61	5.08	14,400.16	22.88
Seed 6	9.92	4.17	2.70	0.56	22.86	7.78
Seed 7	0.67	4,631.50	0.59	0.23	1,878.08	13,022.52
Seed 8	0.17	2.22	14,400.02	14,400.05	4.17	4.25
Seed 9	0.13	9,945.49	0.36	1.03	43.33	6,249.31
Seed 10	1.20	1,999.06	14,400.01	1.09	5.55	5.16
Average	2.06	3,674.83	3,374.45	1,442.20	3,840.59	3,140.82
Std.	3.30	5,051.77	6,004.07	4,552.93	6,045.46	4,520.98

5.5 Scale Size of the Problem

Fig. 10 is a graph of 17 nodes, including 3 origin nodes, 1 transshipment node, and 3 destinations. Under the 512 MB RAM and 1,455 MB virtual memory, the maximal scale size for the network application of Fig. 6 is either a 15-commodity, 3-mode, and 100-planning-time-period or 50-commodity, 3-mode, and 20-planning-time-period scenario. Appendix A1 is a graph of 21 nodes, including 4 origin nodes, 2 transshipment nodes, and 4 destinations. Under the 512 MB RAM and 1,455 MB virtual memory, the maximal scale size for the network application of Fig. 7 is either a 12-commodity,

4-mode, and 100-planning-time-period or 50-commodity, 4-mode, and 30-planning-time-period scenario.

5.6 Summary

After implementing **HA**, results of lower and upper bounds are recorded and analyzed. It demonstrates that **HA** can provide tight upper bounds for all runs in both network applications. If optimal solutions are known, upper bounds are close to 10% ranges of optimal solutions for all scenarios. If optimal solutions are unknown, the average gap between lower and upper bounds is 0.0239. Minimal and maximal gaps are [0.0007, 0.3330].

Procedures in Phase 2 can improve upper bounds except one run. Total running times of **HA** increase either when the number of commodities increases or when the length of planning time periods increases. Moreover, standard deviations of total solving times by solving **Problem (P)** directly in most scenarios are very large. Total solving times of solving **Problem (P)** directly in some runs are larger than the average total running times of **HA**. Therefore, applying **HA** not only provides good lower and upper bounds, but also prevents unpredicted total solving times of solving **Problem (P)** directly.

CHAPTER VI

SUMMARY, CONCLUSIONS, AND RECOMMENDATIONS

6.1 Summary

This research focuses on multi-commodity, multi-mode generalized networks with time windows problems. These problems are complex mixed integral and NP-complete problems. Moreover, the traditional Lagrangian relaxations only provide the lower bound of the original minimal problem and this bound is usually infeasible, except when it is an optimal solution. Therefore, this research is to develop a heuristic algorithm, **HA**, to solve these problems.

Phase 1 of **HA** is based on applying Lagrangian relaxations to find lower bounds. According to lower bounds, it obtains upper bounds for feasible solutions. Hence, it not only can provide lower and upper bounds for the original problem, but also can give a useful feasible solution for references. After Phase 1, if the gap between lower and upper bounds is too large, procedures are needed to enter Phase 2. Phase 2 provides opportunities to improve upper bounds by two methods. One method is based on early due date with overdue-date costs; the other is based on total transportation costs. Chapter IV describes detailed procedures of two phases of **HA** and draws flowcharts of procedures to help understand overall concepts of **HA**.

Two network applications, Fig. 6 and 7, are used to test **HA** in this research. One is a 1-origin, 2-destination, and 3-mode network; the other is a 2-origin, 3-destination, and 4-mode network. The programs are written by the AMPL 9.0 scripts. It can generate

different random variables for needed parameters based on various random seeds. Each scenario has 10 runs and data are recorded in the AMPL output files. It tests different numbers of commodities and various lengths of planning time periods and summarizes results in Chapter V.

The programs used in this research are executed on a computer with Pentium 4 CPU 3.2 GHz 512 MB of RAM. The software used includes the AMPL 9.0 and CPLEX 9.0. For comparison purposes, **Problem (P)** is also solved directly by the CPLEX software. It provides an optimal solution for some specific runs to calculate the gap between the lower bound and optimal solution and the gap between the upper bound and optimal solution. Moreover, it also provides total solving times of solving **Problem (P)** directly. Thus, total running times of implementing **HA** can be compared with total solving times of solving **Problem (P)** directly. It shows that **HA** can provide a good feasible solution within reasonable ranges of optimal solutions under reasonable running times. Chapter VI gives conclusions, points out contributions, and recommends future research and applications.

6.2 Conclusions and Contributions

Results in Chapter V show that **HA** can provide good lower and upper bounds for original problems. This two-phase algorithm can provide a feasible solution based on the upper bound and within a reasonable range of an optimal solution. For different numbers of commodities, the graphs display that the larger the number of commodities, the longer the total running times it takes. For various lengths of planning time periods, the figures

show that the longer the length of planning time periods, the longer the total running times it takes.

The most significant contributions of this research are as follows: first, it develops a procedure that considers multiple commodities, multiple modes, commodity losses, and time windows. Because only a few papers combine with all four factors simultaneously, this research can provide more insights in this area. Second, the procedure in this research is more computational efficiency than solving original problems directly. This procedure is a heuristic algorithm and provides good heuristic feasible solutions instead of optimal solutions.

Third, it improves procedures for lower bound generations of subgradient methods. Because the step size in this research is adopted in Proposition 2, this step size is larger than the traditional step size. Larger step sizes will reduce convergence times. Finally, it is an effective procedure for the reduction of upper bounds. Two methods, early due date with overdue-date costs and total transportation costs in Phase 2 of **HA**, are applied to seek an improved upper bound. Testing runs generated in Chapter V show that most upper bounds are improved in Phase 2.

6.3 Recommendations for Future Research

Since multi-commodity, multi-mode generalized networks with time windows problems are NP-complete, it still has a lot of spaces to develop different heuristic algorithm to reduce total solving times. The extension researches are as follows: first, for special cases, when the gain/loss factor is equal to 1 in all arcs, it means that

commodities are not damaged during transportation. For example, the term "commodity" represents customers and the term "mode" represents buses or subways. Therefore, variables of commodities are required to be integers. It becomes a multi-commodity, multi-mode pure network with time windows problem. It may be useful for real practical situations and worth doing further research in this area.

Second, the special structure of multi-commodity, multi-mode generalized networks with time windows problems is that one decision variable is a non-integer, the others are integers, and there is a constraint among them. This structure is similar to traditional location problems. The only difference is that the integer variable in traditional location problems is a binary variable. It may be easier to branch and bound for traditional location problems than for multi-commodity, multi-mode networks with time windows problems. In other words, traditional locations problems can be viewed as a special case of multi-commodity, multi-mode generalized networks with time windows problems. Therefore, it may get a lot inspiration from algorithms developed for traditional location problems and apply them to solve multi-commodity, multi-mode generalized networks with time windows problems.

Finally, the network can be considered adding the uncertainties for arcs and modes into the model. In this research, arcs and modes are viewed as known parameters. In the real world, there may be uncertainties for availabilities or reliabilities of arcs and modes. This is also a popular research area, which needs further studies.

REFERENCES

Assad, A.A., 1978. Multicommodity network flows - a survey. Networks 8, 37-91.

Bertsekas, D.P., 1999. Nonlinear Programming. Athena Scientific, Belmont, Massachusetts.

Correa, R., 1993. Convergence of some algorithms for convex minimization. Mathematical Programming 62, 261-275.

Drissi-Kaitouni, O., 1991. Solution approaches for multimode multiproduct assignment problems. Transportation Research Part B 25 (5), 317-327.

Fisher, M.L., 1981. The lagrangian relaxation method for solving integer programming problems. Management Science 27 (1), 1-18.

Fisher, M.L., 1985. An application oriented guide to lagrangian relaxation. Interfaces 15 (2), 10-21.

Goffin, J.L., 1977. On convergence rates of subgradient optimization methods. Mathematical Programming 13, 329-347.

Guélat, J. et al., 1990. A multimode multiproduct network assignment model for strategic planning of freight flows. Transportation Research 24 (1), 25-39.

Haghani, A., Oh, S.C., 1996. Formulation and solution of a multi-commodity, multi-modal network flow model for disaster relief operations. Transportation Research Part A 30 (3), 231-250.

Kennington, J.L., 1978. A survey of linear cost multicommodity network flows. Operations Research 26, 209-236.

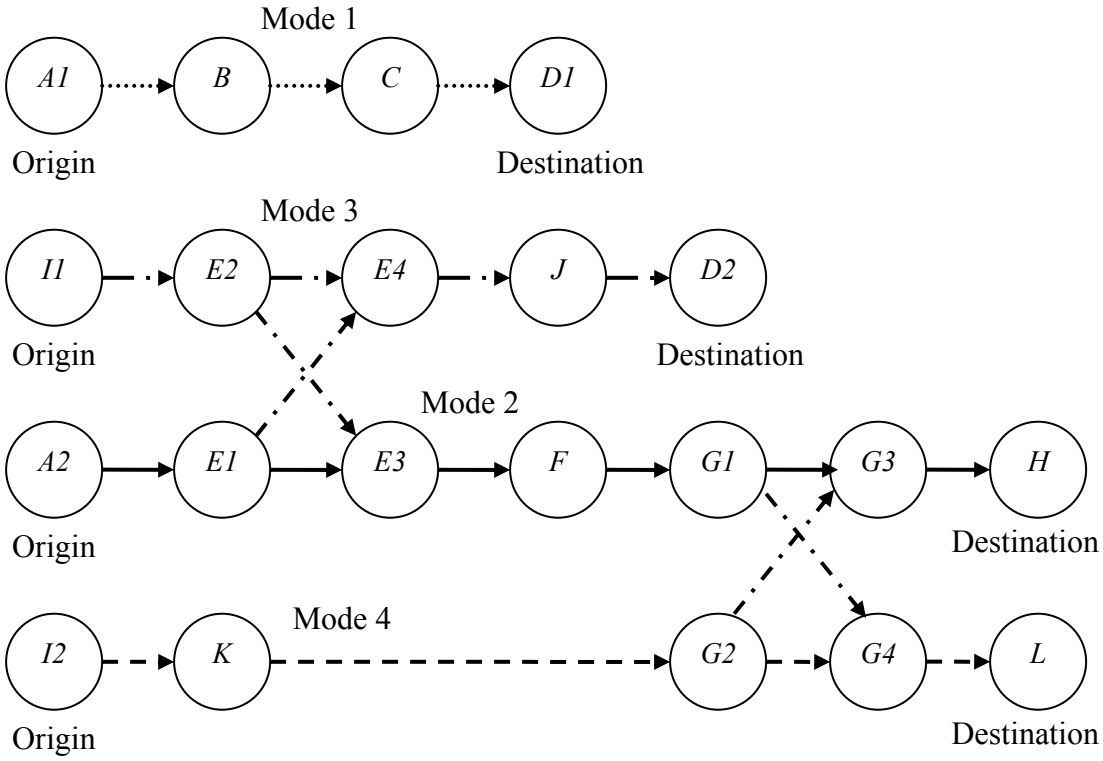
Konnov, I.V., 2003. On convergence properties of a subgradient method. Optimization Methods and Software 18 (1), 53-62.

Rathi, A.K. et al., 1992. Allocating resources to support a multicommodity flow with time windows. Logistics and Transportation Review 28 (2), 167-188.

Wang, S.H., 2003. An improved stepsize of the subgradient algorithm for solving the lagrangian relaxation problem. Computers and Electrical Engineering 29, 245-249.

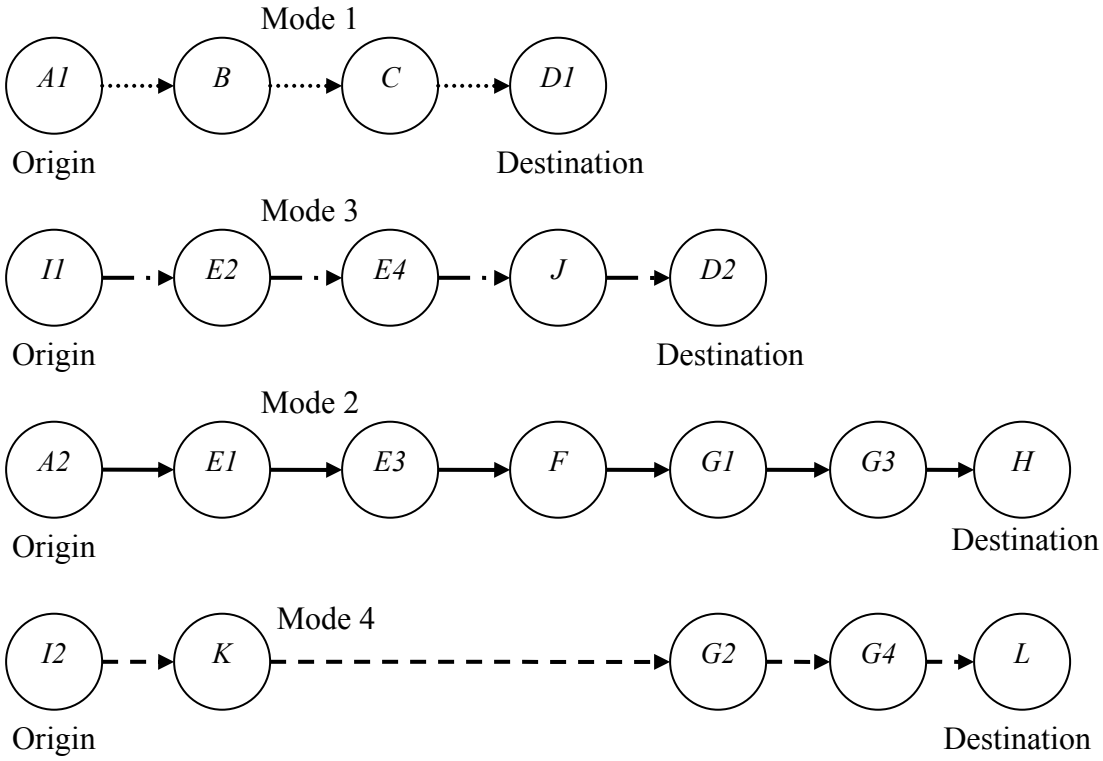
APPENDIX A1

2−ORIGIN, 3−DESTINATION, AND 4−MODE TRANSFORMED NETWORK FOR COMMODITIES



APPENDIX A2

2−ORIGIN, 3−DESTINATION, AND 4−MODE TRANSFORMED NETWORK FOR VEHICLES



APPENDIX B1

DATA OF THE NETWORK APPLICATION OF FIG. 6

10-commodity, 10-planning-time-period scenario in Phase 1

Seed	*LB*	*UB*	GAP_1	Total solving times	Total running times
1	1,378,140	1,381,124	0.0022	91.16	367.84
2	1,798,515	1,807,676	0.0051	81.72	360.25
3	1,607,849	1,612,124	0.0027	81.61	353.28
4	1,837,019	1,859,426	0.0122	84.77	358.20
5	1,646,075	1,651,313	0.0032	80.86	360.42
6	1,766,924	1,775,779	0.0050	84.20	358.89
7	2,114,611	2,119,199	0.0022	88.72	366.42
8	2,174,250	2,178,282	0.0019	81.55	357.33
9	1,539,485	1,557,471	0.0117	89.08	351.58
10	1,761,070	1,768,652	0.0043	88.38	365.20
		Average	0.0050	85.21 (seconds)	359.94 (seconds)
		Variances	0.00001463	14.6022	28.6202

10-commodity, 20-planning-time-period scenario in Phase 1

Seed	*LB*	*UB*	GAP_1	Total solving times	Total running times
1	3,333,195	3,379,438	0.0139	216.20	708.92
2	4,097,613	4,177,729	0.0196	122.00	722.16
3	3,392,406	3,441,578	0.0145	145.69	730.75
4	2,229,500	2,243,384	0.0062	214.92	761.33
5	3,531,723	3,569,902	0.0108	154.55	714.69
6	3,939,470	4,069,389	0.0330	158.73	728.64
7	4,391,479	4,415,827	0.0055	137.06	739.66
8	3,519,998	3,537,750	0.0050	125.09	700.13
9	3,629,948	3,684,010	0.0149	182.08	680.28
10	3,101,907	3,121,706	0.0064	165.97	753.77
		Average	0.0130	162.23 (seconds)	724.03 (seconds)
		Variances	0.00007356	1,119.2577	598.9338

15-commodity, 10-planning-time-period scenario in Phase 1

Seed	*LB*	*UB*	GAP_1	Total solving times	Total running times
1	2,124,049	2,133,373	0.0044	170.70	434.81
2	2,533,780	2,541,158	0.0029	126.52	475.81
3	2,910,547	2,916,794	0.0021	125.86	495.30
4	**2,810,507**	**3,256,290**	**0.1586**	**113.33**	**551.17**
5	2,438,013	2,449,621	0.0048	145.34	530.88
6	**3,511,417**	**4,060,382**	**0.1563**	**115.30**	**524.48**
7	2,334,082	2,342,584	0.0036	121.03	457.64
8	2,120,239	2,132,633	0.0058	189.28	433.36
9	2,120,175	2,150,454	0.0143	134.17	458.92
10	3,110,860	3,126,120	0.0049	120.89	451.39
		Average	0.0358	136.24 (seconds)	481.38 (seconds)
		Variances	0.0041	635.6707	1,760.5694

15-commodity, 20-planning-time-period scenario in Phase 1

Seed	*LB*	*UB*	GAP_1	Total solving times	Total running times
1	3,874,182	3,997,726	0.0319	373.94	1,298.09
2	6,978,762	7,078,418	0.0143	1,357.38	1,144.00
3	**4,136,007**	**5,406,521**	**0.3072**	**480.14**	**1,155.94**
4	3,631,018	3,702,960	0.0198	190.19	1,134.61
5	4,684,073	4,719,720	0.0076	266.14	1,327.48
6	**4,379,719**	**5,712,791**	**0.3044**	**370.92**	**1,282.75**
7	6,390,000	6,424,466	0.0054	172.69	1,124.83
8	5,537,116	5,649,129	0.0202	246.14	1,122.88
9	5,529,344	5,635,522	0.0192	273.36	1,151.38
10	5,126,667	5,218,051	0.0178	318.91	1,019.58
		Average	0.0748	404.98 (seconds)	1,176.15 (seconds)
		Variances	0.0149	123,101.8307	6415.5682

APPENDIX B2

DATA OF THE NETWORK APPLICATION OF FIG. 7

6-commodity, 10-planning-time-period scenario in Phase 1

Seed	*LB*	*UB*	GAP_1	Total solving times	Total running times
1	908,361	915,072	0.0074	71.88	292.89
2	1,306,366	1,311,951	0.0043	72.06	302.70
3	866,240	868,688	0.0028	75.44	293.63
4	1,006,729	1,010,423	0.0037	71.92	294.56
5	1,056,615	1,065,789	0.0087	72.25	293.20
6	835,213	837,347	0.0026	71.45	293.08
7	**1,152,812**	**1,370,887**	**0.1892**	**72.72**	**301.19**
8	1,131,779	1,134,922	0.0028	73.14	292.25
9	1,208,294	1,215,494	0.0060	71.95	294.14
10	781,809	785,245	0.0044	72.48	294.70
		Average	0.0232	72.53 (seconds)	295.23 (seconds)
		Variances	0.0034	1.2751	13.2111

6-commodity, 20-planning-time-period scenario in Phase 1

Seed	*LB*	*UB*	GAP_1	Total solving times	Total running times
1	1,235,657	1,238,267	0.0021	89.78	516.05
2	1,356,245	1,360,790	0.0034	87.09	506.64
3	1,603,081	1,627,539	0.0153	90.16	507.53
4	1,024,329	1,035,859	0.0113	88.73	538.92
5	1,566,571	1,592,959	0.0168	90.47	516.20
6	1,667,238	1,683,642	0.0098	87.23	508.70
7	1,493,591	1,498,288	0.0031	86.91	50177
8	**1,741,733**	**1,981,240**	**0.1375**	**89.86**	**539.20**
9	1,645,657	1,678,660	0.0201	87.98	502.95
10	1,698,156	1,738,802	0.0239	87.30	516.25
		Average	0.0243	88.55 (seconds)	515.42 (seconds)
		Variances	0.0016	1.9990	182.8617

6-commodity, 30-planning-time-period scenario in Phase 1

Seed	*LB*	*UB*	GAP_1	Total solving times	Total running times
1	3,684,252	3,699,096	0.0040	111.69	770.08
2	4,065,968	4,166,040	0.0246	108.78	781.98
3	3,793,007	3,845,117	0.0137	111.52	763.73
4	3,109,990	3,123,554	0.0044	105.03	738.59
5	3,227,884	3,291,373	0.0197	113.08	757.50
6	3,723,158	3,820,729	0.0262	111.22	799.52
7	4,940,613	4,969,269	0.0058	102.09	696.66
8	4,892,077	5,369,557	0.0976	108.64	777.84
9	**4,176,342**	**5,470,370**	**0.3098**	**109.27**	**772.23**
10	3,453,175	3,517,062	0.0185	106.53	744.58
		Average	0.0524	108.78 (seconds)	760.27 (seconds)
		Variances	0.0089	11.6033	815.8216

12-commodity, 10-planning-time-period scenario in Phase 1

Seed	*LB*	*UB*	GAP_1	Total solving times	Total running times
1	**1,848,676**	**2,811,328**	**0.5207**	**83.23**	**441.81**
2	1,856,943	1,864,051	0.0038	77.77	388.17
3	1,839,938	1,849,030	0.0049	79.64	388.00
4	**2,828,994**	**3,863,170**	**0.3656**	**81.88**	**391.98**
5	2,231,047	2,238,208	0.0032	81.11	394.25
6	1,838,989	1,843,646	0.0025	78.28	388.50
7	2,582,083	2,589,648	0.0025	77.39	387.81
8	1,494,906	1,499,787	0.0033	80.95	388.28
9	**1,850,534**	**2,423,343**	**0.3095**	**82.09**	**414.98**
10	2,540,019	2,541,742	0.0007	78.45	385.45
		Average	0.1217	80.08 (seconds)	396.93 (seconds)
		Variances	0.0392	4.1870	321.0040

12-commodity, 20-planning-time-period scenario in Phase 1

Seed	*LB*	*UB*	GAP_1	Total solving times	Total running times
1	**3,087,930**	**3,593,790**	**0.1638**	**120.41**	**881.30**
2	2,645,635	2,701,095	0.0210	115.16	845.63
3	3,264,770	3,298,322	0.0103	105.61	843.23
4	2,648,994	2,694,851	0.0173	103.05	844.27
5	2,895,344	2,923,176	0.0096	103.00	853.94
6	2,682,773	2,695,517	0.0048	101.83	846.06
7	**3,025,781**	**3,409,229**	**0.1267**	**128.31**	**913.14**
8	2,658,741	2,703,497	0.0168	103.66	830.72
9	2,397,368	2,439,986	0.0178	115.31	804.11
10	**3,134,827**	**3,928,959**	**0.2533**	**128.09**	**866.97**
		Average	0.0641	112.44 (seconds)	852.94 (seconds)
		Variances	0.0075	109.9412	863.8127

12-commodity, 30-planning-time-period scenario in Phase 1

Seed	*LB*	*UB*	GAP_1	Total solving times	Total running times
1	7,286,808	7,362,572	0.0104	132.67	1,568.11
2	**7,682,412**	**8,678,134**	**0.1296**	**160.66**	**1,490.09**
3	5,829,220	5,891,358	0.0107	131.27	1,430.89
4	**8,375,532**	**10,210,964**	**0.2191**	**202.53**	**1,341.53**
5	6,918,224	7,054,205	0.0197	166.27	1,696.25
6	6,726,256	6,794,477	0.0101	229.59	1,360.88
7	7,499,361	7,604,373	0.0140	130.25	1,499.80
8	7,823,209	7,937,334	0.0146	131.14	1,466.27
9	6,343,088	6,450,199	0.0169	132.53	1,526.78
10	**7,343,188**	**8,361,351**	**0.1387**	**180.84**	**1,687.05**
		Average	0.0584	159.78 (seconds)	1,506.76 (seconds)
		Variances	0.0057	1,241.1990	14,319.9547

VITA

Ping-Shun Chen, son of Lien-Fu Chen and Mei-Jung Chen Lin, was born on March 20, 1975 in Kaohsiung City, Taiwan. He obtained his B.S. degree in the Department of Industrial Engineering and Management from National Chiao Tung University, Taiwan, in 1997. Then, he continued his M.S. degree in the same department and graduated in 1999. His master thesis was "An Application of Linear Programming in Releasing for An Electronic Component Factory."

In August 2000, Mr. Chen enrolled in the graduate program of the Department of Industrial Engineering at Texas A&M University for his Ph.D degree. During five years of his study, his research was focused on network programming and application problems. Mr. Chen may be reached at the following address: 4F-2, No 82, Wufu 2nd Rd, Sinsing District, Kaohsiung City, 800, Taiwan. His email address is pingshunchen@yahoo.com.