

ABSTRACT

Title of Document: SCHEDULING DELIVERIES WITH BACKHAULS
 IN THAILAND'S CEMENT INDUSTRY

Chutipong Paraphantakul, Master of Science, 2011

Directed By: Associate Professor Elise Miller-Hooks
 Department of Civil and Environmental Engineering

In this study, the Truckload Delivery with Backhaul Scheduling Problem (TDBSP) is formulated and an Ant Colony Optimization methodology developed for a related problem, the Vehicle Routing Problem with Backhaul and Time Windows (VRPBTW), is adapted for its solution. The TDBSP differs from the VRPBTW in that shipments are in units of truckloads, multiple time windows in multiple days are available for delivery to customers, limited space for servicing customers is available and multiple visits to each customer may be required. The problem is motivated by a real-world application arising at a leading cement producer in Thailand. Experts at the cement production plant assign vehicles to cement customers and lignite mines based on manual computations and experience. This study provides mathematical and computational frameworks for modeling and solving this real-world application.

SCHEDULING DELIVERIES WITH BACKHAULS IN THAILAND'S CEMENT
INDUSTRY

By

Chutipong Paraphantakul

Thesis submitted to the Faculty of the Graduate School of the
University of Maryland, College Park, in partial fulfillment
of the requirements for the degree of
Master of Science
2011

Advisory Committee:

Associate Professor Elise Miller-Hooks, Chair/Advisor
Professor Ali Haghani
Professor Paul Schonfeld

© Copyright by
Chutipong Paraphantakul
2011

Dedication

To my family.

Acknowledgements

First and foremost I would like to thank Dr. Elise Miller-Hooks, my advisor, who dedicated her invaluable time and had immeasurable patience to aid me through this thesis. I also would like to give my thanks to committee members, Dr. Ali Haghani and Dr. Paul Schonfeld, who are very flexible for my changes and gave very constructive comments during my thesis defense.

I would like to thank Dr. Sathaporn Opananon who directed me to this interesting thesis topic. Also, I would like to give many thanks to Mr. Nopporn Thepsithar, Ms. Piyaporn Limkatanyoo and Mr. Thana Ruttanawetwong for the helpful advices and priceless efforts to support me in providing data from SCCC.

I would like to personally thank Krist Wongsuphasawat who gave plenty of insightful computer programming advices during my algorithm development process. Moreover, I want to thank Pratt Hetrakul and Abhisit Phoowarawutthipanich who always being very supportive during my thesis writing period.

Last but not least, I have many more thanks to my CEE Transportation fellows whom I did not mentioned.

Table of Contents

Dedication	ii
Acknowledgements	iii
Table of Contents	iv
List of Tables	vi
List of Figures	vii
Chapter 1: Introduction	1
Chapter 2: Background and Literature Review	7
Chapter 3: Problem Formulation	11
Parameters	15
Formulation TDBSP	17
Chapter 4: Application of formulation on small problem instances	21
Chapter 5: Insertion based ants metaheuristic	25
5.1 The insertion based ants optimization method for the VRPBTW	25
5.2 A modified insertion based ants optimization method for the TDBSP	27
5.2.1 Generation of tours by an ant.....	32
5.2.2 Improvement procedure	35
5.2.3 Pheromone update.....	35
Chapter 6: Numerical experiments	38
6.1 Parameter tuning	38
6.1.1 Visibility function parameters.....	40
6.1.2 Pheromone function parameters	41

6.1.3 Number of random time period candidates.....	42
6.2 Solution quality and convergence.....	43
Chapter 7: Case Study Results.....	46
Chapter 8: Conclusions.....	51
Appendix.....	54
Bibliography.....	64

List of Tables

Table 1 Distance between plant, customers and mines (in kilometers)	5
Table 2 Summary of the computational time of each instance from <i>CPLEX</i>	22
Table 3 Exact and LP Relaxed Solutions from <i>CPLEX</i>	24
Table 4 Parameters in the modified insertion based ants algorithm	40
Table 5 Top ten scores by visibility parameter settings.....	41
Table 6 Top ten scores by pheromone parameter settings	42
Table 7 Average solution value for problem instance I9 by number of iterations completed.....	44
Table 8 The capability of the size of problem that can be solved efficiently by heuristic.....	45
Table 9 Average solution value for the case study by number of iterations completed	46
Table 10 Average solution value for the case study with restrictive time windows by number of iterations completed	49

List of Figures

Figure 1 Locations of customers, mines and plant	4
Figure 2 Illustration of an instance where $D_0 = D_C$	14
Figure 3 Illustration of an instance where $D_0 > D_C$	15
Figure 4 Illustration of an instance where $D_0 < D_C$	15
Figure 5 Comparison of the solving time from each instance size.	23
Figure 6 Computational efforts of small problem instances' relaxed solutions	24
Figure 7 Illustration of a solution by modified insertion based ants algorithm	31
Figure 8 Flowchart of modified insertion based ants optimization algorithm.	37
Figure 9 Comparison of the scores from different μ	43
Figure 10 Optimality gap as a function of number of iterations	44
Figure 11 Comparison of the solution value in different runs by the number of iterations	47
Figure 12 Comparison of the solution value in different runs by the number of iterations	49

Chapter 1: Introduction

Thailand's government, along with private investors, have for several decades been investing in large civil infrastructure projects throughout Thailand to benefit society. Examples of such projects include the development of a rapid rail line between the central business district of Bangkok and the airport, expansion of the Bangkok Sky Train, and a network of highways providing both national and international connectivity. Consequently, construction is booming and this boom has significantly benefitted the cement industry. In fact, the cement industry in Thailand is one of the largest industries in the nation. This industry consists of several companies, but two companies take the largest share of the market: Siam Cement Group (SCG), and Siam City Cement Public Company Limited (SCCC). SCG is the largest Thai cement company and the product of a private-public partnership. Both companies are publically traded. The needed high capital costs prevent others from entering the market.

With many projects simultaneously running, national transportation costs associated with obtaining raw materials for cement production, delivering cement to construction sites, and distributing cement for resale, are on average 19% of total costs*. Thus, it is crucial to the bottom line for the company to carefully plan its logistics operations.

This thesis seeks to develop a methodology to aid the cement industry with reducing its logistics costs through efficient route planning and scheduling. Specifically, the problem of routing a cement company's fleet of trucks from the plant

* Information on the Thailand logistics costs report in 2008 from www.NESDB.go.th

to distribution centers (i.e. customers) and incorporation of materials pickup on the backhaul is mathematically formulated. The problem is referred to herein as the Truckload Delivery with Backhaul Scheduling Problem (TDBSP). A solution methodology based on ant colony optimization, a metaheuristic, is suggested and tested on a case study involving typical operations at SCCC in the northern portion of Thailand.

SCCC has a single cement production plant located in Central Thailand. Its main product is bagged cement. The company operates a fleet of nearly 200 trucks in Northern Thailand alone for use in distributing the cement. Trucks are sent out from the plant to distributors. In addition, the same fleet of trucks is used to pick up lignite from lignite mines. Lignite is the main source of energy used in cement production at the plant. As the trucks would either travel empty to the lignite mines or return empty to the plant after hauling cement to the distributors, these pickup and delivery jobs are often coupled. This coupling reduces inefficiencies associated with empty linehauls and backhauls.

SCCC has customers scattered throughout the provinces of Thailand. As customer names and locations are proprietary in nature, a single customer in each of the provinces is used to represent all customers of the province. Cement demand estimates in terms of units of truckloads were provided for the study. SCCC works with four lignite mines in the region. Their locations, along with the location of SCCC's production plant (the depot), are noted in Figure 1. Distances between customer locations, lignite mines and the depot, obtained from Google Maps, are provided in Table 1. All trucks are assumed to travel at a speed of 50 km/hr as

suggested by SCCC. Trip times between customers range between approximately 2 and 14 hours. Customer time windows were also supplied by SCCC. All delivery time windows are set to an earliest arrival time of 9:00 a.m. and latest arrival time of 9:00 p.m. The lignite mines operate 24 hours per day. Queues along with service times on the order of one to two days at the mines can be expected. Moreover, there is a limit on the number of vehicles that can serve a customer simultaneously. A schedule is needed for a seven day planning horizon. Available lignite of any required blend is available at all lignite mines. Typical loading times per vehicle at the depot are on the order of three hours. It is also typical that the vehicle will need 15 hours at each customer for unloading and resting. Waiting, resting and unloading times of 38 hours at the lignite mines is common, as well.



Figure 1 Locations of customers, mines and plant^{†‡}

[†] Customer mine and plant locations are depicted in circle, triangle and square, respectively. Number and alphabet denote the name as in Table 1

[‡] Map picture obtained from www.LEWMANOMONT.com

Table 1 Distance between plant, customers and mines (in kilometers)

	(0) Production Plant	(A) Chiang Khong	(B) Mae Teeb	(C) Na Glang	(D) Mae Ramad
(0) Production Plant	-	784	574	520	481
(1) Chiang Mai	708	335	255	176	398
(2) Mae Hong Son	764	492	383	253	283
(3) Lamphun	568	372	242	49	262
(4) Lampang	631	271	84	172	329
(5) Chiang Rai	724	110	158	313	475
(6) Phayao	660	138	126	285	442
(7) Phrae	506	266	66	206	321
(8) Nan	606	260	151	306	421
(9) Uttaradit	454	405	205	265	337
(10) Tak	392	473	284	147	101
(11) Pichit	274	556	357	349	249
(12) Phitsanulok	315	462	262	249	273
(13) Kamphaeng Phet	314	583	384	256	213
(14) Sukhothai	418	420	220	140	201
(15) Nakhon Sawan	194	607	407	371	328
(16) Phetchabun	260	604	404	392	415
(17) Uthai Thani	254	714	515	371	328
(18) Chai Nat	179	674	474	402	360

Despite the complexity involved in routing, scheduling and dispatching the large fleet of trucks, or in the determination of when to pick up lignite and selection of the specific mines, the company makes such logistics-related decisions manually.

The contributions of this thesis are: (1) a mixed integer programming formulation of this truckload delivery with backhaul scheduling problem (the TDBSP) that can be used to model the Thai cement delivery with backhaul problem, (2) customization and application of a metaheuristic based on insertion techniques developed for routing problems and ant colony optimization to this real-world case

study, and (3) evaluation of the benefits of heuristic solution compared with manually obtained solution for the case study.

In the next chapter, relevant works from the literature are reviewed. This is followed by problem description and formulation in chapter 3. To assess the applicability of direct solution of the mathematical program for real-world problem instances, a set of small problem instances in which their exact solutions are obtained is illustrated in chapter 4. In chapter 5, description of the insertion based ant colony optimization algorithm, including details associated with customization of a proposed approach from the literature for the problem posed herein, is given. Results of numerical experiments designed to assess the performance of the algorithm are described in chapter 6. In chapter 7, the metaheuristic is applied on the real-world case study in Thailand involving a typical set of demands at SCCC. Solutions are compared with average reported tour durations from SCCC required to serve similar demands.

Chapter 2: Background and Literature Review

An enormous number of works in the literature address the related classical Vehicle Routing Problem (VRP) and the myriad variants that arise in practical applications. In this chapter, only those variants with particular relevance to this thesis are reviewed. See (Laporte 2009; Toth and Vigo, 2002; Golden et al., 2008; and Bodin et al., 1983) for general background.

A number of works address the VRP with backhaul (VRPB). Techniques proposed in these works can be categorized as belonging to one of the following general approaches: pickup then delivery; delivery then pickup; simultaneous pickup and delivery. In addressing the TDBSP herein, trucks are loaded at a depot. The goods are first delivered to linehaul customers and then goods are picked up from backhaul customers. The vehicle then returns to the depot. Thus, a "delivery then pickup" approach is taken. A similar "delivery then pickup" approach is considered by Anily (1996), for example. See (Bodin et al., 1983; Toth and Vigo, 2002) for a comprehensive review of contributions in the VRPB. Less than truckload customer demands are usually considered, but the VRPB can also directly address customers with truckload demands.

A related problem to the VRPB is the General Pickup and Delivery Problem (GPDP) in which the vehicle need not start from or return to a depot. A review of GPDP works is given by Savelsbergh and Sol (1995). Regan et al. (1995, 1996a, 1996b, 1998) and Yang et al. (2002) address real-time extensions of the GPDP, referred to as the Truckload Pickup and Delivery Problem (TPDP) in which vehicles are dispatched to simultaneously pick up and deliver goods. Loads are in units of

truckloads. Customers arise over time and may be rejected. No pickup or delivery time requirements are imposed in the VRPB, GPDP or TPDP.

Other works consider the VRP with time-windows (VRPTW). Time windows indicate earliest and latest permitted pickup or delivery times from or to the customers. Time windows may be modeled as hard constraints, where no solution in which pickup or delivery occurs outside the time window is considered. Alternatively, soft time windows may be used in which solutions may be penalized for violating time window constraints. The canonical works on the VRPTW include (Savelsbergh,1984; Solomon, 1986; Solomon et al., 1988). Complications due to including both pickup and delivery within the routes simultaneously are not modeled. Additionally, each customer is assumed to have associated with it only one time window.

Thangiah et al. (1996) and Reimann et al. (2002) addressed the Vehicle Routing Problem with Backhauls and Time-Windows (VRPBTW), a problem that combines elements of the VRPB and the VRPTW. Thangiah et al. introduce a specialized construction heuristic for this problem class. Reimann et al. propose an insertion-based ant colony optimization technique. More recently, Cho and Wang (2005) provided an assignment-based formulation of the VRPBTW and propose a threshold accepting variant of Simulated Annealing for its solution. Their formulation extends the VRPTW formulation in (Goetschalckx and Jacobs-Blecha, 1989). Aghdaghi and Jolai (2008) suggest a goal programming formulation of the VRPBTW and introduce the Two-Phase Clustering and Sequencing heuristic, designed specifically for this problem class. Their formulation builds on the VRPTW

formulation of (Calvete et al., 2007) and VRPB formulation of (Toth and Vigo, 1997).

The Generalized Traveling Salesperson Problem (GTSP) also has relevance here. In this variant of the Traveling Salesperson Problem (TSP), a single vehicle version of the VRP, there are clusters of customers. It is sufficient to visit only one customer within each cluster. Thus, the problem becomes one of not only developing a tour for the TSP, but choosing which of the customers in each cluster to visit. This problem was introduced by Noon (1988). A stochastic version of the GTSP, in which customers probabilistically arise, was later studied by Tang and Miller-Hooks (2005).

The TDBSP considered herein can be viewed as a VRPBTW with multiple time windows and multiple visits. Routes are determined over multiple days, each day with a time window for delivery. Two additional works in the literature have applicability to this problem. Favaretto et al. (2007) formulated the Vehicle Routing Problem with Multiple Time Windows (VRPMTW). The VRPMTW also involves multiple visits and periodicity. They propose an ant colony optimization metaheuristic for its solution. The VRPMTW does not consider both pickup and delivery, i.e. there are no backhaul customers.

Thus, the TDBSP addressed herein has elements of the VRPBTW, GTSP and VRPMTW. A formulation that combines elements of the three problem classes is provided in the next chapter. This formulation extends the mixed integer program given by Solomon et al. (1988) for the VRPTW. The ant colony optimization technique proposed by Reimann et al. (2002) is modified (Chapter 5) to solve the

TDBSP. Modifications to this technique for the specific application addressed in this thesis are discussed.

Chapter 3: Problem Formulation

The TDBSP as applied to the cement delivery with backhaul problem arising at SCCC is formulated as a mixed integer mathematical program. The objective of the TDBSP is to assign trucks over the planning horizon (e.g. one week) to linehaul customers (i.e. cement customer) within the requested time window (i.e. between 9:00 a.m. and 9:00 p.m.) on a chosen day, schedule their arrival, and choose backhaul customers (i.e. lignite mines) so as to minimize total travel time required for delivery of cement to all linehaul customers and meet requirements for lignite at the depot (i.e. production plant). Thus, a solution consists of a set of vehicle tours, where each tour includes one linehaul customer and at most one backhaul customer or, alternatively, only a backhaul customer. Customer demand must be served within the planning horizon. The earliest possible starting time is preferred for every vehicle. Demand at any customer may be served by multiple vehicles, since more than one truckload of deliveries (pickups) may be required at a linehaul (backhaul) customer. Demand for pickups at the depot must be met. This demand can be met through a visit to any backhaul customer. While in reality a fixed fleet of vehicles is available to serve this demand, each vehicle becoming available for redeployment upon its return to the depot, an infinite fleet size is assumed as SCCC can obtain vehicles to serve the demand whenever needed. In reality, it is very rare that their demand needs exceed their fleet capacity.

Solomon et al. (1988)'s VRPTW formulation is employed as the backbone of the proposed TDBSP formulation. Terms and notation presented in Solomon et al.'s work are employed herein whenever possible.

Before proceeding to the formulation, assumptions and additional details of the model not yet discussed are mentioned.

1. Each 24 hour duration is discretized into θ blocks of time. Each block is referred to herein as a time period. If θ is set to 24, then each day will consist of 24 one hour blocks. The larger the block, the faster the solution is obtained, but the less precise the schedules will be in terms of arrival times at customers.
2. The vehicle fleet is homogeneous.
3. Capacity limitations at linehaul and backhaul customer facilities exist due to space limitations. It is assumed that a fixed number of vehicles, $\lambda(t)$, can serve or be served by a customer in any time period t . This number may vary not only by time, but by customer. For simplicity, this is taken to be constant over all locations.
4. Service times are used to model loading times at the depot, waiting and unloading times at the linehaul customers and waiting and loading times at the backhaul customers. For simplicity, service times are assumed to be constant and based only on the customer type (i.e. depot, linehaul and backhaul customer).

Let $G = (\mathcal{N}, \mathcal{A})$ be a digraph, where \mathcal{N} is the set of nodes and \mathcal{A} is the set of arcs. Specifically, $\mathcal{N} = \{0\} \cup \mathcal{P}^- \cup \mathcal{P}^+ \cup \{n+1\}$, where $\{0\}$ and $\{n+1\}$ represent a single depot, 0 representing the depot when used for a vehicle's departure and $n+1$ when used for vehicle's arrival. The set of delivery nodes \mathcal{P}^- and set of pickup nodes \mathcal{P}^+ represent linehaul customers and backhaul customers, respectively. Let delivery node, i_k , denote the k^{th} truckload required at linehaul customer i in \mathcal{P}^- and pickup node, j_l , denote the l^{th} truckload of available pickup at backhaul customer j in \mathcal{P}^+ . For each truckload required (available) at a linehaul (backhaul) customer, a copy node is included. Thus, if three truckloads of cement are required at linehaul customer i , there will be three copies of node i , one for each required truckload. $\mathcal{A} = \{(0, i_k) \cup (0, j_l) \cup (i_k, j_l) \cup (i_k, n+1) \cup (j_l, n+1) \mid i_k \in \mathcal{P}^-, j_l \in \mathcal{P}^+\}$.

Let $D_C = |\mathcal{P}^-|$, the total delivery demand of all linehaul customers, and D_0 be the total amount of pickup (lignite) demand at the depot (production plant). There are three possible scenarios related to demand imbalance that may arise: (i) pickup demand is equal to delivery demand ($D_0 = D_C$), (ii) pickup demand is greater than delivery demand ($D_0 > D_C$), and (iii) pickup demand is less than delivery demand ($D_0 < D_C$). To account for all three scenarios, additional notation is required: $\mathcal{P}_1^-, \mathcal{P}_1^+, \mathcal{P}_2^-, \mathcal{P}_2^+$ in which $\mathcal{P}^- = \mathcal{P}_1^- \cup \mathcal{P}_2^-$ and $\mathcal{P}^+ \supseteq \mathcal{P}_1^+ \cup \mathcal{P}_2^+$.

$\mathcal{P}_1^- =$ subset of \mathcal{P}^- including only those delivery nodes, i_k , that are paired in a tour with a node $j_l \in \mathcal{P}_1^+$.

$\mathcal{P}_1^+ =$ subset of \mathcal{P}^+ including only those pickup nodes, j_l , that are paired in a tour with a node $i_k \in \mathcal{P}_1^-$.

\mathcal{P}_2^- = subset of \mathcal{P}^- including only unpaired delivery nodes, i_k , i.e. those nodes that are contained in a tour that includes only a visit to the delivery node.

\mathcal{P}_2^+ = subset of \mathcal{P}^+ including only unpaired pickup nodes, j_l , i.e. those nodes that are contained in a tour that includes only a visit to the pickup node.

Figures 2 through 4 illustrates the effects of imbalances in pickup and delivery demand on route topology on the network for two linehaul customers and two backhaul customers. The total pickup demand at the depot is given for each of the three possible scenarios associated with demand imbalance.

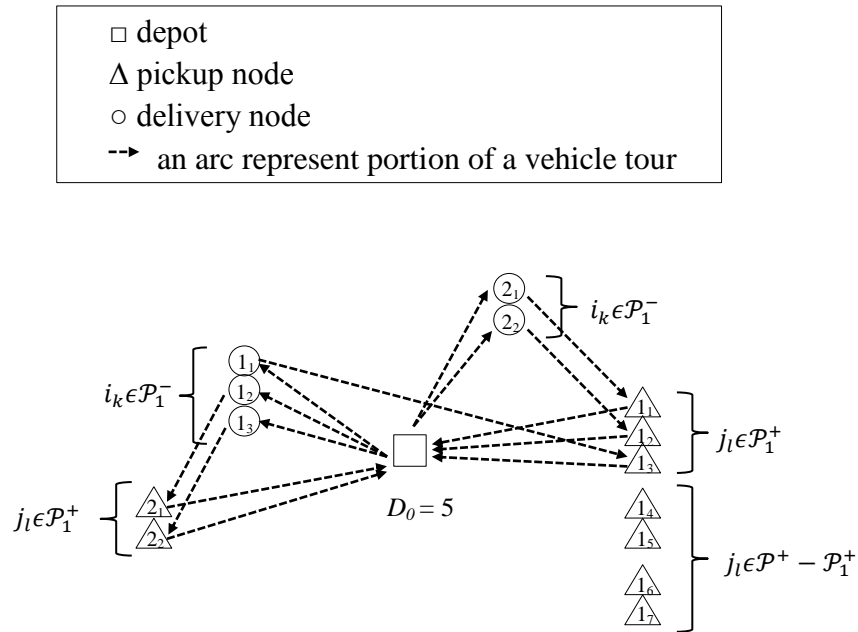


Figure 2 Illustration of an instance where $D_0 = D_C$

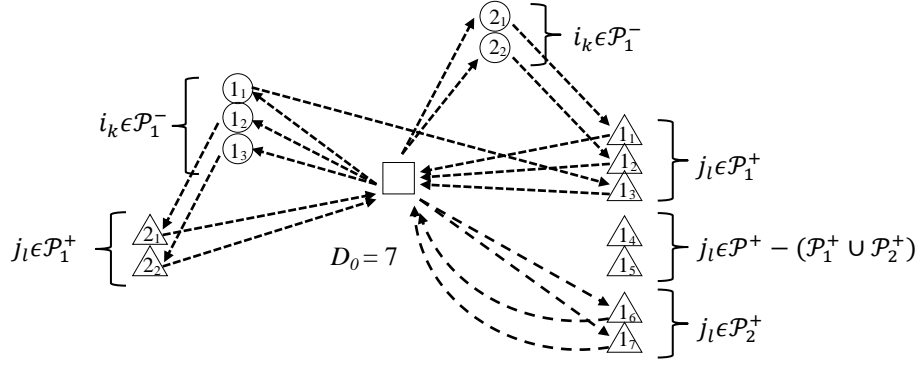


Figure 3 Illustration of an instance where $D_0 > D_C$

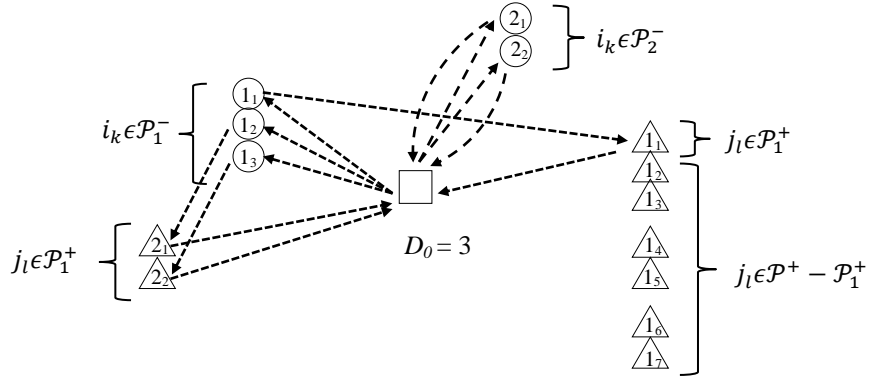


Figure 4 Illustration of an instance where $D_0 < D_C$

Additional notation employed within the formulation are given next.

Parameters

τ_{ij} = travel time from linehaul customer i to backhaul customer j

θ = number of time periods per day

U = number of days in the planning horizon

\mathcal{U} = set of days in the planning horizon

a_i = earliest acceptable time to reach the linehaul customer i , where

$$a_i \in \{0, 1, \dots, \theta - 1\}$$

- b_i = latest acceptable time to reach the linehaul customer i ,
 where $b_i \in \{0, 1, \dots, \theta - 1\}$
- V = number of vehicles to be used, equivalent to $\max \{D_C, D_0\}$
- \mathcal{V} = set of vehicles
- T = total number of time periods in a planning horizon, $T = \theta \cdot U$
- \mathcal{T} = set of time periods
- s_i = service time at site i (i.e. i can represent depot, linehaul customer or backhaul customer)
- M = large number

In addition, each linehaul customer has multiple time-windows as a function of θ and U . For any $u \in \mathcal{U}$, the time window for linehaul customer i can be expressed as:

$$[(u - 1) \cdot \theta + a_i, (u - 1) \cdot \theta + b_i] \quad (\text{eq.1})$$

Thus, if $\theta = 24$, for day 2 ($u=2$), given earliest and latest arrival times of 9:00 a.m. (time period 9) and 9:00 p.m. (time period 21), respectively, the time window will be $[24+9, 24+21]$ or $[33, 45]$.

Decision variables

$X_{i_k j_l}^v$ = 1 if delivery node $i_k \in \mathcal{P}_1^-$ and pickup node $j_l \in \mathcal{P}_1^+$ are assigned to vehicle v ; 0, otherwise

$Y_{i_k}^v$ = 1 delivery node $i_k \in \mathcal{P}_2^-$ is assigned to vehicle v ; 0, otherwise

$Y_{j_l}^v$ = 1 pickup node $j_l \in \mathcal{P}_2^+$ is assigned to vehicle v ; 0, otherwise

T_0^v = time period within the planning horizon that vehicle v is ready to start out from the depot or begin loading for delivery customer

T_{n+1}^v = time period within the planning horizon that vehicle v returns to the depot

$T_{i_k}^v$ = time period in which delivery node, i_k , is visited by vehicle v

$T_{j_l}^v$ = time period in which pickup node, j_l , is visited by vehicle v

u_{i_k} = day of week in which node i_k will be visited

S_t^v = 1 if vehicle v starts from depot 0 or returns to depot $n+1$ in time period t ; 0, otherwise

\ddot{S}_t^v = 1 if vehicle v arrives at a delivery or pickup node in time period t ; 0, otherwise

Formulation TDBSP

$$\text{Minimize } c \cdot \sum_{v \in \mathcal{V}} (T_{n+1}^v - T_0^v) + (1 - c) \cdot \sum_{v \in \mathcal{V}} T_0^v \quad (1)$$

Subject to

$$\sum_{v \in \mathcal{V}} \sum_{j_l \in \mathcal{P}_1^+} X_{i_k j_l}^v = 1 \quad \forall i_k \in \mathcal{P}_1^- \quad (2)$$

$$\sum_{v \in \mathcal{V}} \sum_{i_k \in \mathcal{P}_1^-} X_{i_k j_l}^v = 1 \quad \forall j_l \in \mathcal{P}_1^+ \quad (3)$$

$$\sum_{i_k \in \mathcal{P}_1^-} \sum_{j_l \in \mathcal{P}_1^+} X_{i_k j_l}^v = 1 \quad \forall v \in \mathcal{V} \quad (4)$$

$$\sum_{v \in \mathcal{V}} Y_{i_k}^v = 1 \quad \forall i_k \in \mathcal{P}_2^- \quad (5)$$

$$\sum_{v \in \mathcal{V}} Y_{j_l}^v = 1 \quad \forall j_l \in \mathcal{P}_2^+ \quad (6)$$

$$\sum_{i_k \in \mathcal{P}_2^-} Y_{i_k}^v = 1 \quad \forall v \in \mathcal{V} \quad (7)$$

$$\sum_{j_l \in \mathcal{P}_2^+} Y_{j_l}^v = 1 \quad \forall v \in \mathcal{V} \quad (8)$$

$$T_0^v + s_0 + \tau_{0,i} - T_{i_k}^v \leq (1 - X_{i_k j_l}^v) \cdot M \quad \forall i_k \in \mathcal{P}_1^-, j_l \in \mathcal{P}_1^+, v \in \mathcal{V} \quad (9)$$

$$T_{i_k}^v + s_i + \tau_{ij} - T_{j_l}^v \leq (1 - X_{i_k j_l}^v) \cdot M \quad \forall i_k \in \mathcal{P}_1^-, j_l \in \mathcal{P}_1^+, v \in \mathcal{V} \quad (10)$$

$$T_{j_l}^v + s_j + \tau_{j,n+1} - T_{n+1}^v \leq (1 - X_{i_k j_l}^v) \cdot M \quad \forall i_k \in \mathcal{P}_1^-, j_l \in \mathcal{P}_1^+, v \in \mathcal{V} \quad (11)$$

$$T_0^v + s_0 + \tau_{0,i} - T_{i_k}^v \leq (1 - Y_{i_k}^v) \cdot M \quad \forall i_k \in \mathcal{P}_2^-, v \in \mathcal{V} \quad (12)$$

$$T_{i_k}^v + s_i + \tau_{i,n+1} - T_{n+1}^v \leq (1 - Y_{i_k}^v) \cdot M \quad \forall i_k \in \mathcal{P}_2^-, v \in \mathcal{V} \quad (13)$$

$$T_0^v + \tau_{0,j} - T_{j_l}^v \leq (1 - Y_{j_l}^v) \cdot M \quad \forall j_l \in \mathcal{P}_2^+, v \in \mathcal{V} \quad (14)$$

$$T_{j_l}^v + s_j + \tau_{j,n+1} - T_{n+1}^v \leq (1 - Y_{j_l}^v) \cdot M \quad \forall j_l \in \mathcal{P}_2^+, v \in \mathcal{V} \quad (15)$$

$$a_i + \theta \cdot (u_{i_k} - 1) \leq T_{i_k}^v \leq b_i + \theta \cdot (u_{i_k} - 1) \quad \forall i_k \in \mathcal{P}^-, u_{i_k} \in \mathcal{U}, v \in \mathcal{V} \quad (16)$$

$$M \cdot |T_0^v - t| \geq 1 - S_t^v \quad \forall v \in \mathcal{V}, t \in \mathcal{T} \quad (17)$$

$$M \cdot |T_{n+1}^v - t| \geq 1 - S_t^v \quad \forall v \in \mathcal{V}, t \in \mathcal{T} \quad (18)$$

$$M \cdot |T_{i_k}^v - t| \geq 1 - \check{S}_t^v \quad \forall i_k \in \mathcal{P}^-, v \in \mathcal{V}, t \in \mathcal{T} \quad (19)$$

$$M \cdot |T_{j_l}^v - t| \geq 1 - \check{S}_t^v \quad \forall j_l \in \mathcal{P}^+, v \in \mathcal{V}, t \in \mathcal{T} \quad (20)$$

$$\sum_{t \in \mathcal{T}} S_t^v = 2 \quad \forall v \in \mathcal{V} \quad (21)$$

$$\sum_{v \in \mathcal{V}} S_t^v \leq \lambda(t) \quad \forall t \in \mathcal{T} \quad (22)$$

$$\sum_{v \in \mathcal{V}} \check{S}_t^v \leq \lambda(t) \quad \forall t \in \mathcal{T} \quad (23)$$

$$X_{i_k j_l}^v, S_t^v, \check{S}_t^v = \{0,1\} \quad \forall i_k \in \mathcal{P}_1^-, j_l \in \mathcal{P}_1^+, v \in \mathcal{V}, t \in \mathcal{T} \quad (24)$$

$$Y_{i_k}^v, Y_{j_l}^v = \{0,1\} \quad \forall i_k \in \mathcal{P}_2^+, j_l \in \mathcal{P}_2^+, v \in \mathcal{V} \quad (25)$$

$$T_0^v, T_{i_k}^v, T_{j_l}^v, T_{n+1}^v \geq 0 \text{ \& Integer} \quad \forall i_k \in \mathcal{P}^-, j_l \in \mathcal{P}^+, v \in \mathcal{V} \quad (26)$$

The objective of this problem is to minimize the total time required for the vehicles to serve linehaul customers and obtain required supplies from the backhaul customers; that is, minimize all vehicle time spent outside the depot, while simultaneously minimizing total elapsed time between time period zero and the start of all tours.

While one may treat this problem as one involving multiple objectives, the problem is reduced to one with a single additive objective, where $0 \leq c \leq 1$.

Constraints (2) through (8) are assignment constraints, where constraints (2) and (3) together ensure a single pairing of linehaul customers in \mathcal{P}_1^- to backhaul customers in \mathcal{P}_1^+ . If there is a demand imbalance, constraints (5) ensure that only one delivery node will be included in a tour and constraints (6) ensure that only one pickup node will be included in a tour. That only one vehicle will be assigned to each tour (i.e. each pickup and delivery pair or each pickup or delivery alone) is guaranteed through constraints (4), (7) and (8).

Constraints (9) through (15) together force the order in which vehicles arrive at the nodes to follow the sequence: depot, linehaul customer, backhaul customer, depot. To permit tours that skip a linehaul customer and visit only a backhaul customer, constraints (14) and (15) are applied. Similarly, constraints (12) and (13) allow tour configurations that exclude a backhaul customer. Constraints (9) through (15), thus, ensure route formation compatibility. That is, for example, a backhaul customer can be visited after departing the depot or a linehaul customer only if the linehaul customer is included in the tour. Constraints (16) enforce the time window limitations (described by eq. 1), ensuring that no arrival at a linehaul customer occurs outside the specified time windows.

Constraints (22) and (23) ensure that the total number of vehicles visiting any customer at time period t is no greater than $\lambda(t)$. The total number of times each vehicle visits the depot must equal two as enforced through constraint (21). This is

accomplished through constraints (17) through (20) that convert each vehicle's arrival time at the depot, T_0^v, T_{n+1}^v , to binary variable S_t^v and each vehicle's arrival at a delivery or pickup node, $T_{i_k}^v$ and $T_{j_l}^v$, respectively, to \check{S}_t^v . Integrality and non-negativity are enforced through constraints (24) - (26).

The objective function and constraints (2) through (16) are identical to, or modeled after, the objective and constraints used in the VRPTW formulation given in (Solomon et al., 1988). Modifications, including the use of additional notation and additional decision variables, required to address demand imbalances that arise from simultaneously addressing pickup and delivery in a truckload trucking operation are applied in constraints (5) through (8) and (9) through (15). The time window constraints (16) are modified to allow for multiple time windows, allowing delivery to a linehaul customer from during one or more given time windows for each day in the planning horizon. Remaining functional constraints are specific to the TDBSP.

Chapter 4: Application of formulation on small problem instances

The formulation is applied in solving a number of small problem instances designed to assess the applicability of direct solution of the mathematical program for real-world problem instances. *IBM's ILOG CPLEX OPL 12.2* was used for this purpose. Runs were conducted on a personal computer with Intel Core i5 CPU and 2.0 GB of RAM. Nine problem instances are considered as described in Table 2. The instances vary in size from two linehaul customers and two backhaul customers to six linehaul customers and six backhaul customers. Total delivery demand at each linehaul customer and amount of available pickup at backhaul customers is assumed to be one truckload. Thus, there is only one delivery node and one pickup node at each linehaul and backhaul customer, respectively. θ is set to 4 periods. A limit of one vehicle is imposed at each site per time period, $\lambda(t) = 1, \forall t \in \mathcal{T}$. To give priority to solutions with shorter tour duration, the coefficient c in the objective function is set to be 0.99.

Computation times required to obtain the optimal solution are reported in Table 2 for each setting of the parameters tested. Note that TW indicates whether or not time windows were enforced, i.e. $TW = 1$ if time windows are applied and 0, otherwise. Additional details, including exact input values and final objective function values, for these test instances are provided in the appendix.

Table 2 Summary of the computational time of each instance from *CPLEX*

Instance	Input (θ : U : D_C : D_0 : TW)	Computational time (seconds)
I1	4:4:5:3:1	45.44
I2	4:4:5:3:0	69.56
I3	4:5:2:2:0	0.63
I4	4:5:3:3:0	5.19
I5	4:5:4:4:1	35.46
I6	4:5:4:4:0	37.13
I7	4:5:5:5:1	651.86
I8	4:5:5:5:0	1198.26
I9	4:5:6:6:0	65279.36

Solution of problem instances I1, I2, I5, I6, I7, and I8 indicate that the inclusion of time-windows affects required computational effort. For example, a 35% reduction in computation time was required to solve problem instance I2 in which time windows were enforced as compared with problem instance I1 in which no time windows were enforced. A reduction in solution time was also obtained through the inclusion of time windows as evidenced through a comparison of solution times for problem instances I5 and I6 (by 4%), as well as I7 and I8 (by 46%). This is consistent with results from previous works related to the VRPTW. One can expect a reduction in computational effort with the inclusion of time windows as such inclusion reduces the solution space.

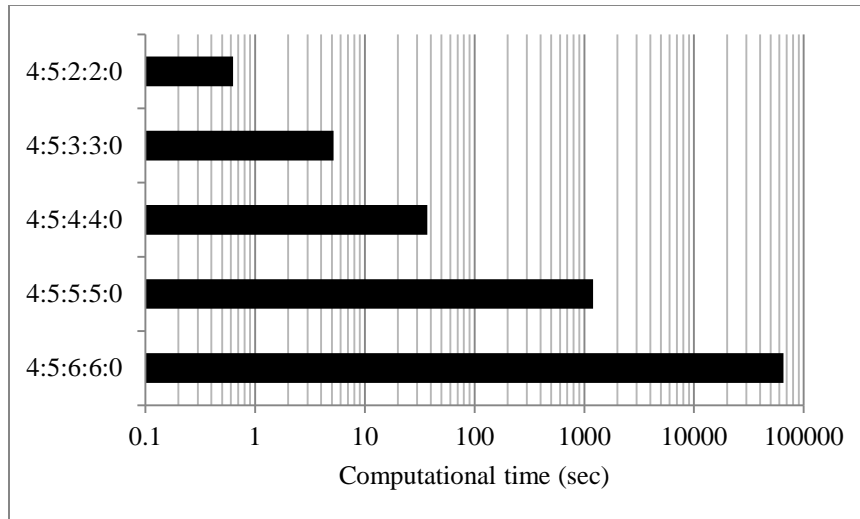


Figure 5 Comparison of the solving time for the exact solution of each instance size.

A comparison of computation times required for problem instances I3, I4, I6, I8, and I9, in which no time windows are enforced, is given in Figure 5. One can see from the figure that the required computation time grows exponentially with increasing problem size, even for the small problem instances considered here. While less than one second was required to solve the smallest problem instance, with only two delivery and two pickup nodes, over eighteen hours was required to solve the largest problem instance with six delivery and six pickup nodes. Exact solution of problem instances arising in the real-world is likely beyond reach. Moreover, the study investigates further if there is any kind of lower bound that could be solved exactly without the exhaustive effort. To investigate the possibility of employing the LP relaxation as a method for obtaining tight bounds for large problem instances that cannot be solved exactly, the bound obtained from the LP relaxation of problem instances I3, I4, I6, I8 and I9 is computed. Table 3 Shows that the optimality gap of the LP relaxed solutions are below 1%, but increases slightly with the size of the problem. Figure 6 shows that the computational effort of the tested set of instances

grows exponentially with problem size. Thus, obtaining even a lower bound on a large problem instance may be problematic. A metaheuristic is employed in the next chapter to address such large-scale problem instances.

Table 3 Exact and LP Relaxed Solutions from *CPLEX*

Instance	Exact solution	LP Relaxed solution	Optimality gap (%)	Computational time (seconds)
4:5:2:2:0	7.92	7.93	0.12	0.41
4:5:3:3:0	10.89	10.95	0.54	1.47
4:5:4:4:0	15.84	15.9	0.38	15.76
4:5:5:5:0	18.81	18.94	0.69	225.36
4:5:6:6:0	23.76	23.97	0.88	5053.54

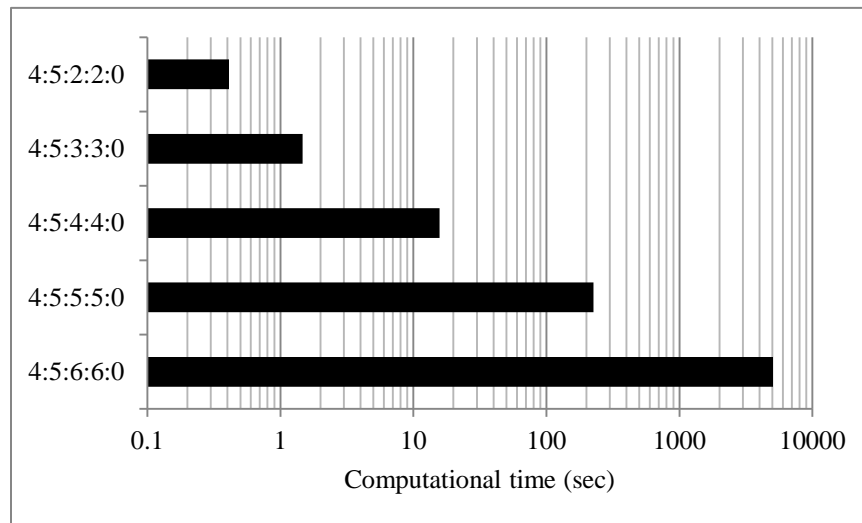


Figure 6 Computational efforts of small problem instances' relaxed solutions

Chapter 5: Insertion based ants metaheuristic

In this chapter, the ant colony optimization methodology developed by Reimann et al. (2002) for the VRPBTW is adapted for solution of the TDBSP. These adaptations are required to address multiple time windows, multiple visits to the same customers and truckload demands. For completeness, details of the entire methodology are provided. Adaptations are noted. Heuristics have been developed for nearly all, if not all, variants of the VRP. Solomon (1987), Cordeau et al. (2001), and Favaretto et al. (2007) provide excellent reviews of heuristics designed for the VRPTW. Likewise, comprehensive reviews of heuristics designed for the VRPB are given in (Potvin et al., 1996; Jacobs-Bleacha and Goetschalckx, 1998; and Mingozzi et al., 1999). Few works address the VRPBTW. Reimann et al. (2002) proposed an insertion based ants optimization methodology based on concepts of ant colony optimization that was shown to outperform an existing heuristic of Thangiah et al. (1996) for the same problem. Cho and Wang (2005) introduced a threshold accepting variant of simulated annealing for this problem, but did not compare their solutions to any benchmark. The Two-Phase Clustering and Sequencing heuristic proposed by Aghdaghi and Jolai (2008) for the VRPBTW with soft time windows was also introduced, but no comparison to exact or known solutions for benchmark problems was provided.

5.1 The insertion based ants optimization method for the VRPBTW

The insertion based ants metaheuristic of Reimann et al. (2002) builds on concepts of two methodologies: the ant colony optimization (ACO) metaheuristic and traditional insertion techniques (e.g. the cheapest insertion or nearest insertion) designed for

routing problems. The ACO concept was first introduced by Dorigo et al. (1991, 1992, and 1996). ACO mimics the food searching behavior of ants. The technique has been further developed since its inception and has been applied in solving problems in a wide array of applications. Dorigo et al. (2006) provide an overview of the approach and comprehensive review of its application in literature since its introduction.

The insertion based ants metaheuristic is described next in the context of the VRP, based on contributions from several works (Dorigo et al., 1991, 1992, and 1996; and Bullnheimer et al., 1997). The steps exploit a network representation of the problem in which nodes represent customers and edges provide existing connections between nodes. A complete graph is presumed.

A colony of ants is sent through the network. In a given iteration, these ants are thought to work in parallel networks and, thus, their actions do not affect one another in a given iteration. The number of ants working to develop tours is a parameter that is set at the start of the procedure.

A single ant generates a tour by meandering through the network. When at a decision point (a node), the ant chooses the next edge probabilistically in direct proportion to the level of pheromones already laid on the edges by ants that have traversed the edge previously. The level of pheromones present depends not only on what was laid during previous visits, but by the length of time (measured in terms of iterations) that has elapsed since the visits. That is, pheromone evaporates over time. When traversing an edge, the ant lays down one unit of pheromone along the edge, dispersed at a constant rate. Thus, the amount of pheromone laid at any location on

the edge is equivalent to the reciprocal of the edge's length. The greater the pheromone intensity, the more likely the ant is to follow the edge. The ant continues to traverse the network, visiting customers until it reaches a customer whose inclusion in the tour would violate feasibility. The ant then returns to the depot from the previous customer and begins another tour. Once all customers have been included in a tour of all the ants, the pheromone levels along the edges are updated to account for the passage of time and the pheromones laid during that iteration.

Elitism is incorporated in the heuristic by giving more weight to the pheromone laid by an ant that produces the best solution in any iteration. The weight of that ant's pheromone depends on a parameter that is set by the user. In a rank-based implementation, the pheromone from only those ants providing the top ranked solutions will be laid for the next iteration. The higher the ranking, the more weight the associated ant's pheromone will have. The ant with the best found solution over all previous iterations is referred to as the elite ant.

Reimann et al. (2002) proposed the use of an insertion technique developed in (Solomon, 1987) within the ACO as a strategy for addressing time-windows within the context of the VRPBTW. This insertion technique is employed herein.

5.2 A modified insertion based ants optimization method for the TDBSP

The insertion based ants optimization technique was developed to solve the VRPBTW. The VRPTW differs from the TDBSP in several respects, as mentioned previously. Thus, changes to the insertion based ants optimization method are required for its application to the TDBSP. Specifically, a linehaul customer cannot be visited after a backhaul customer and at most one delivery and one pickup can be

included in a tour. These specific constraints are handled through minor changes to the logic of the insertion procedure. The number of tours required can be predetermined. Thus, the number of tours that an ant makes is prefixed. It is possible that the ant will build the tours such that not all demand is met. In this case, the solution obtained by the ant is infeasible. When this occurs, the results of the entire iteration, or results for the single ant, can be discarded and either the iteration can be rerun for the single ant or for all ants.

The most significant changes to the heuristic are required to address multiple time windows, a characteristic not shared by the VRPBTW. In addition, more than one truckload may be required at (from) a linehaul (backhaul) customer. Restrictions on the number of vehicles that can visit any customer in each time period exist, since there is limited space at each customer for delivery or pickup. To address this, a vector, δ_q , of size $T = \theta \cdot U$ is employed at each site q for each ant. Each element of the vector maintains the number of visits to the site included in tours for the associated time period. That is, the fifth element of the vector maintains the number of vehicles that have thus far (in already constructed tours) been accommodated at the site in the fifth time period. The vector is employed over all tours made by a single ant.

Additional notation required in the description of the modified insertion based ants algorithm employed herein for the TDBSP, adapted from Reimann's insertion based ants algorithm for the VRPBTW, are given next.

- δ_q = $\{\delta_q^t\}_{t \in \mathcal{T}}$, for $q \in \mathcal{N}$, each element δ_q^t of which represents the number of visits to site q at time period t
- K_q = amount of delivery (pickup) demand in truckloads at linehaul (backhaul) customer $q \in \mathcal{N} - \{0, n + 1\}$
- Λ_q = arrival time in terms of time period at site $q \in \mathcal{N}$
- Ω = number of ants deployed in a run of the algorithm
- P_j = probability that backhaul customer j is chosen
- η_j = attractiveness of backhaul customer j
- σ_{ij} = visibility value of backhaul customer j from linehaul customer i
- π_{ij} = pheromone intensity on arc $(i,j) \in \mathcal{A}$
- $\Delta\pi_{ij}$ = added pheromone from ant along arc $(i,j) \in \mathcal{A}$
- $\Delta\pi_{ij}^r$ = added pheromone from ant ranked r along arc $(i,j) \in \mathcal{A}$
- α = visibility coefficient factor, $0 \leq \alpha \leq 2$
- β = visibility coefficient factor, $0 \leq \beta \leq 1$
- ρ = evaporation rate
- φ = number of ranks that will be added to the pheromone calculation
- ε = intensity of pheromones laid by the elite ant
- μ = parameter of algorithm
- Υ = set of random time period candidates with $|\Upsilon| = \mu \cdot T$

With this notation, the following equations can be given (Reimann et al., 2002).

$$P_j = \frac{\eta_j}{\sum_{j \in \mathcal{P}^+} \eta_j} \quad (\text{eq.2})$$

$$\eta_j = \sigma_{ij} \cdot \pi_{ij} \quad (\text{eq.3})$$

$$\sigma_{ij} = \max \{0, \alpha \cdot \tau_{o,j} - \beta \cdot (\tau_{ij} + \tau_{j,n+1} - \tau_{i,n+1}) - (1 - \beta) \cdot (\Lambda_{n+1}^* - \Lambda_{n+1})\} \quad (\text{eq.4})$$

$$\pi_{ij} = (1 - \rho) \cdot \pi_{ij} + \sum_{r=1}^{\varphi} \Delta\pi_{ij}^r + \varepsilon \cdot \Delta\pi_{ij} \quad (\text{eq.5})$$

$$\Delta\pi_{ij}^r = (\varphi - r + 1) \cdot (1/\tau_{ij}) \quad r = 1, \dots, \varphi \quad (\text{eq.6})$$

$$\Delta\pi_{ij} = (1/\tau_{ij}) \quad (\text{eq.7})$$

When a backhaul customer is selected, this selection is based on the perceived distance (or duration) from the linehaul customer i to the backhaul customer j . This perceived distance is referred to as visibility, σ_{ij} . The visibility function is calculated from equation (eq.4). How attractive each backhaul customer j , i.e. η_j , is depends on both the visibility of the link between the current linehaul customer i and backhaul customer j , and pheromone intensity on link (i,j) , π_{ij} . The intensity is computed from the amount of pre-existing pheromone along the arc given evaporation rate, ρ , the pheromone added to the link by ants that are included in a set of ranked ants, φ , and additional pheromone laid by the elite ant. The strength of pheromone laid by each ranked ant is a function of the ant's ranking. The top ranked ants lay pheromone with greater strength than the lower ones. Choice of a backhaul customer is made probabilistically in direct proportion to its attractiveness. The probability of selecting any particular available backhaul customer, P_j , is computed in equation (eq.2).

The algorithm is iterative. In each iteration, multiples solutions are produced. The best solution of all completed iterations is saved. An overview of the main steps of the algorithm is given next

```

REPEAT
    FOR  $\omega = 1, \dots, \Omega$ 
        FOR  $v = 1, \dots, V$ 
            Generate tour  $v$  for ant  $\omega$ ;
        END
        Apply swap procedure on tours generated by ant  $\omega$ ;
    END
    Update pheromones based on solution obtained by all ants;
UNTIL (number of iterations reaches threshold)

```

Figure 7 shows a typical solution obtained by a single ant assuming zero service time at customers, unit-time distance between every site, unit available pickup at each backhaul customer, unit delivery demand at each linehaul customer, two pickup demand units at the depot and space for only one vehicle in a given time period at any site.

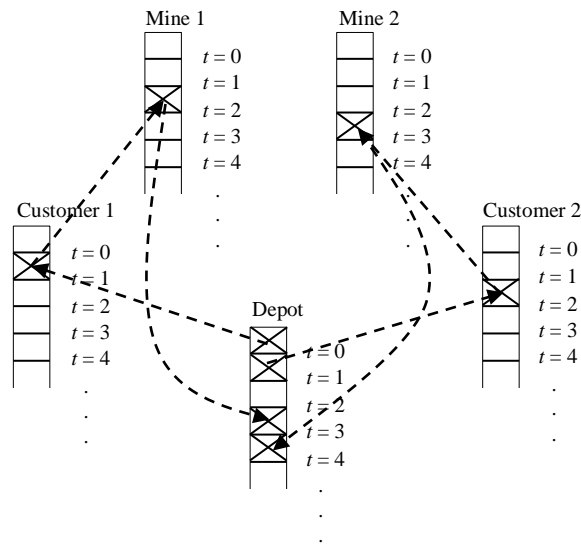


Figure 7 Illustration of a solution by modified insertion based ants algorithm

Specific steps of the algorithm are described in the following subsections.

5.2.1 Generation of tours by an ant

Creating each tour for each ant within each iteration of the proposed insertion based ants optimization procedure consists of three main steps: (1) selection of the starting time from the depot, (2) selection of a linehaul customer for inclusion, and (3) selection of a backhaul customer for inclusion. These steps take advantage of the problem's structure. That is, each tour can be created by selecting a maximum of two customers, one linehaul and one backhaul customer. In each step, feasibility conditions are checked. These feasibility conditions are given as:

- 1) $\lambda(\Lambda_q) - \delta_q^{\Lambda_q} > 0$ $q \in \mathcal{N}$
- 2) $\Lambda_q \geq \Lambda_p + s_p + \tau_{pq}$ $p, q \in \mathcal{N} - \{0, n + 1\}$
- 3) $\sum_{t \in \mathcal{T}} \delta_q^t < K_q$ $q \in \mathcal{N} - \{0, n + 1\}$
- 4) $\Lambda_{n+1} \geq \Lambda_q + s_q + \tau_{q,n+1}$ $q \in \mathcal{N} - \{0, n + 1\}$
- 5) $\lambda(\Lambda_{n+1}) - \delta_{n+1}^{\Lambda_{n+1}} > 0$
- 6) $(u_q - 1) \cdot \theta + a_q \leq \Lambda_q \leq (u_q - 1) \cdot \theta + b_q$
 $q \in \mathcal{N} - \{0, n + 1\} \cup \mathcal{P}^+, u_q \in \mathcal{U}$

Condition 1 forces the number of vehicles visiting site q in time period Λ_q to be less than the available number of slots for vehicles at that time period, $\lambda(\Lambda_q)$. To ensure that the arrival time of a vehicle at site q , Λ_q , occurs after that vehicle's arrival time at the previous site p , Λ_p , plus the service time at site p , s_p , plus the travel time from p to q , τ_{pq} , condition 2 must be met. The total number of visits over the planning horizon at site q , $\sum_{t \in \mathcal{T}} \delta_q^t$, must not exceed the delivery demand (available pickups) at

each linehaul (backhaul) customer q as required by enforcing condition 3. Condition 4 forces the arrival time at the depot, Λ_{n+1} , to occur after visiting site q immediately preceding return to the depot. Specifically, Λ_{n+1} must be no less than the arrival time at site q plus service time at q , s_q , plus travel time between q and the depot, $\tau_{q,n+1}$. Arrival time at depot $n+1$ is permitted only if space for at least one vehicle in time period Λ_{n+1} remains, as is given by condition 5. Finally, condition 6 restricts the arrival time to a linehaul customer q to be within the prespecified time window for the given day, u_q .

Specific steps for generating tours by a single ant are given next.

Step 0: *Initialization.*

Reset $\delta_q^t = 0, \Lambda_q = 0 \quad \forall q \in \mathcal{N}, t \in \mathcal{T}$.

Mark all unavailable linehaul and backhaul customers as available. Let $\nu = 0$.

Step 1: *Selection of the starting time from the depot.*

Update $\nu = \nu + 1$.

Insert in Y the earliest $\mu \cdot T$ possible starting times from depot 0, Λ_0 , such that

Λ_0 satisfy condition 1.

If $Y = \emptyset$, return to step 0; otherwise, randomly select a time period for the

starting time at depot 0, Λ_0 , from Y and update $\delta_0^{\Lambda_0} = \delta_0^{\Lambda_0} + 1$.

Set $\Lambda_{n+1} = \Lambda_0$.

If $\nu > \min \{D_c, D_o\}$ and $D_c < D_o$ go to Step 3; otherwise, go to step 2.

Step 2: *Selection of a linehaul customer for inclusion.*

If no linehaul customer available, return to step 0; otherwise, select an available linehaul customer, i , probabilistically based on direct proportion of the distance from the depot.

Compute the earliest feasible arrival time at the linehaul customer i , Λ_i , and update earliest feasible arrival time at depot $n+1$, Λ_{n+1} , such that conditions 1 through 6 are met.

If $\Lambda_i \leq T$ and $\Lambda_{n+1} \leq T$, then update $\delta_i^{\Lambda_i} = \delta_i^{\Lambda_i+1}$ and $\delta_{n+1}^{\Lambda_{n+1}} = \delta_{n+1}^{\Lambda_{n+1}+1}$; otherwise, mark linehaul customer i as unavailable and restart step 2.

If $v > \min \{D_c, D_o\}$ and $D_c > D_o$ and $v = V$, terminate.

If $v > \min \{D_c, D_o\}$ and $D_c > D_o$ and $v < V$, go to step 1.

Step 3: *Selection of a backhaul customer for inclusion.*

If no backhaul customer is available, return to step 0; otherwise, randomly select an available backhaul customer, j , weighting the selection of each j by probability P_j as in equation (eq.2).

Compute the earliest feasible arrival time at backhaul customer j , Λ_j , and earliest feasible arrival time at the depot $n+1$, Λ_{n+1}^* , such that conditions 1 through 5 are met.

If $\Lambda_j \leq T$ and $\Lambda_{n+1}^* \leq T$, then update $\delta_j^{\Lambda_j} = \delta_j^{\Lambda_j} + 1$, $\delta_{n+1}^{\Lambda_{n+1}^*} = \delta_{n+1}^{\Lambda_{n+1}^*} - 1$,

$\Lambda_{n+1} = \Lambda_{n+1}^*$ and $\delta_{n+1}^{\Lambda_{n+1}} = \delta_{n+1}^{\Lambda_{n+1}} + 1$; otherwise, mark j as unavailable and restart step 3.

If $v < V$, go to Step 1; otherwise, terminate.

This procedure terminates with a feasible solution, if one exists, for the TDBSP. It can be implemented to stop after a fixed number of iterations to prevent an endless loop.

In step 1, the starting time period from the depot is randomly selected from $\mu \cdot T$ earliest feasible time periods at the depot. Such random selection is introduced for solving the TDBSP so as to create an opportunity for finding solutions that might otherwise not be explored.

5.2.2 Improvement procedure

After an ant has constructed a feasible solution, a swap procedure that exchanges backhaul customers between pairs of tours is implemented. The implementation considers the swap in order of tour number. A backhaul customer in tour u will be swapped with a backhaul customer in tour w , $w > u$. During the swap process, the earliest arrival time of the swapped backhaul customers, Λ_u and Λ_w , are computed. If the swap leads to an improved solution, the swap is made and $\delta_u^{\Lambda_u}$, $\delta_w^{\Lambda_w}$ and $\delta_{n+1}^{\Lambda_{n+1}}$ are updated accordingly. The swapping procedure restarts by comparing backhaul customers from tour $u = 1$ and $w = 2$ again. The algorithm continues, comparing backhaul customers from tour pairs until swaps between all pairs of tours have been considered and no improvement is made.

5.2.3 Pheromone update

After every ant has finished its tour construction and the solution has been improved by the swapping procedure, the pheromone level on each arc is updated according to equation (eq. 5). After the pheromones are updated, the solution from each ant is

compared with the previous best solution. The best solution is stored. The algorithm starts the tour construction process over from the first ant until a pre-specified number of iterations is reached. The elite solution is chosen as the final solution.

This concludes the insertion based ants metaheuristic algorithm and its customization for the TDBSP. Results from computational experiments designed to assess the quality of the solutions obtain by the adapted procedure are provided in the next chapter.

5.2.4 Procedure overview

An overview of the procedure is given in Figure 8.

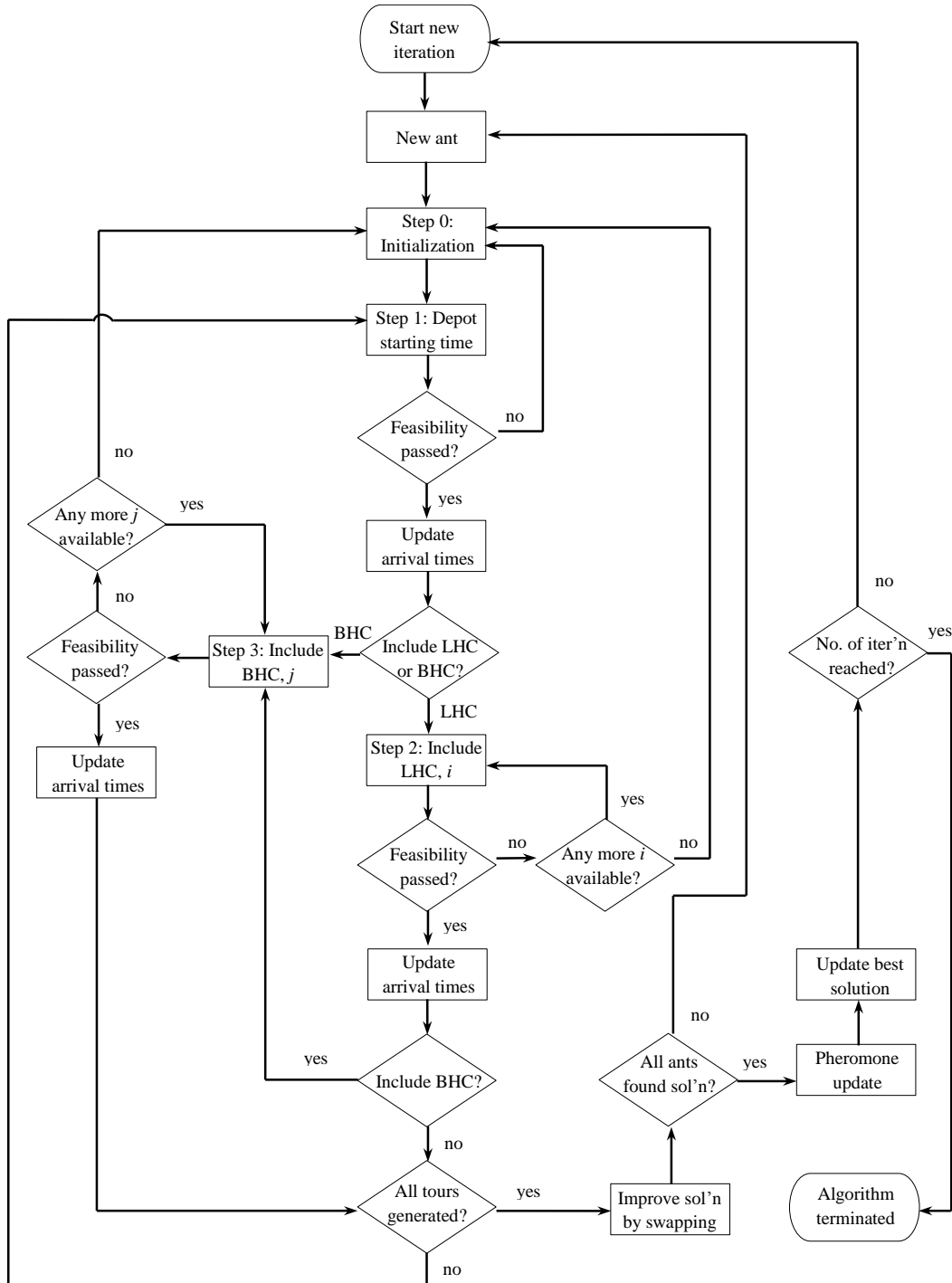


Figure 8 Flowchart[§] of modified insertion based ants optimization algorithm.

[§] LHC = linehaul customer, BHC = backhaul customer

Chapter 6: Numerical experiments

Numerical experiments were designed to assess the quality of solutions obtained by the insertion based ants optimization procedure on small problem instances of the TDBSP. Exact solution for these instances was obtained through formulation TDBSP using ILOG's CPLEX. Parameters of the model are first tuned for best performance. This is followed by analysis of results from runs with varying number of iterations to study the algorithm's convergence behavior.

6.1 Parameter tuning

How the heuristic performs is, in part, dependent on the setting of its parameters. A systematic study is conducted on three hypothetical problem instances ranging in size for the purpose of tuning the parameters, i.e. finding the best combination of parameters to use. Problem instance I9 described in Table 2 involving six linehaul and six backhaul customers is the smallest problem instance considered. Two additional instances that are approximately 20 and 64 times larger than problem instance I9 (H1 and H3 in Table 8, respectively), measured in terms of total delivery and pickup demand multiplied by the number of time periods in the planning horizon, are also considered. These additional larger instances were generated by adding pickup and delivery demand at sites included in problem instance I9. Furthermore, the problem instances were enhanced by the inclusion of additional randomly selected sites. These larger problem instances are studied for the purpose of assessing the efficiency of the solution method. The parameters of the algorithm, along with the range of their possible values, are summarized in Table 4.

Solution performance is assessed according to a score, Φ , computed from two criteria. The first criterion, a , is the average value of the solutions found from each ant in each iteration. The average value of the solution is used to determine the overall efficiency of the heuristic. The second criterion, b , is the average value of the best solutions found up to the beginning of each iteration. Solutions obtained at any iteration are compared to the best solution maintained from previous iterations. The better of the two solutions is stored as the best solution up to the current iteration. The average value of the best solution is used to determine how fast the heuristic converges. For each criterion, a score ϕ_a or ϕ_b is computed. This score, for each criterion, is a function of the relative value of the average solution value from the given run to the best obtained average solution value over all runs. Suppose three runs are made and the average solution value measured in terms of criterion a is 10, 9 and 8. ϕ_a for each of these runs is computed from $(1 - (10 - 8)/8)$, $(1 - (9 - 8)/8)$ and $(1 - (8 - 8)/8)$, respectively. ϕ_b is computed similarly with respect to criterion b . The final score $\Phi = \varpi \cdot \phi_a + (1 - \varpi) \cdot \phi_b$, for $0 \leq \varpi \leq 1$. This scoring method provides a standardized score that can be used as a basis of comparison irrespective of problem size.

Table 4 Parameters in the modified insertion based ants algorithm

Parameter	Description
Ω	Number of ants based on the proportion of delivery and pickup demand, can be any positive integer value
α	Coefficient in visibility function with range [0,2]
β	Coefficient in visibility function with range [0,1]
ρ	Evaporation rate of the laid pheromones with range [0,1]
φ	Parameter for setting number of ranked ants with range [0,1]
ε	Parameter for setting pheromone intensity of the elite ant with range [0, 20]
μ	Coefficient used in determining number of candidate starting times from depot with range [0,1]

The following subsections, each addressing parameters that affect different portions of the solution approach, summarize the experimental results. In all experiments, $\Omega = \left\lfloor \frac{D_c + D_0}{2} \right\rfloor$.

6.1.1 Visibility function parameters

The first set of experimental runs was designed to assess the sensitivity of the solution due to changes in α and β . Values of α are set between 0.1 and 2 in increments of 0.1. Similarly, β is set between 0.1 and 1 also in increments of 0.1. These values were suggested by Reimann et al., (2003). Equal weights are given to each score component in determining the final score for each run. Scores for those combinations of α and β that led to the ten highest score values are shown in Table 5. The results indicate that the pair of α and β that led to the best solution is 0.9 and 0.7, respectively. In these runs, the parameters, ρ , φ , ε and μ are fixed at 0.5, 0.5, 5 and 0.1, respectively.

Table 5 Top ten scores by visibility parameter settings

α	β	Score, Φ
0.9	0.7	9955
0.9	0.5	9934
1	0.2	9927
1	0.3	9926
1	0.5	9925
1.1	0.3	9924
1	0.1	9919
0.8	0.6	9909
0.8	0.4	9908
1.2	0.3	9903

6.1.2 Pheromone function parameters

Three parameters, ρ , p and ε , affect pheromone intensity. Runs are based on pairs of settings of these three parameters: $\rho - \varphi$, $\rho - \varepsilon$, and $\varphi - \varepsilon$. Evaporation rate, ρ , is varied in increments of 0.05 from 0.05 to 1. The number of ranked ants, φ , is varied as a percentage of the total number of ants, Ω , ranging in value from 10% to 100% increasing in increments of 10%. Pheromone intensity, ε , is varied between 1 and 20 as an integer coefficient of the total number of ants, Ω . $\varpi = 0.5$. Results of the 10 best performing runs are given in Table 6. The best solution was obtained when ρ , φ , ε are set to 0.5, 0.7, 1, respectively. α , β and μ are fixed at 0.9, 0.7. and 0.1.

Table 6 Top ten scores by pheromone parameter settings

ρ	φ	ε	Score, Φ
0.5	0.7	1	9969
0.25	0.1	17	9967
0.2	1	18	9966
0.1	0.2	15	9959
0.4	0.4	20	9959
0.3	0.4	18	9959
0.05	1	9	9957
0.25	0.1	20	9953
0.1	0.5	15	9953
0.5	0.8	1	9950

6.1.3 Number of random time period candidates

To increase diversification, the starting time for each tour at the depot is chosen randomly from a set of values in Y , where $|Y| = \mu \cdot T$. μ is set to the percentage of the planning horizon in terms of time periods, ranging in value between 1% of the horizon to 50% of the horizon. As in the previous experimental runs, two criteria are used to determine the performance score for each solution. However, the greater the value of μ , the more randomness in the obtained solution values and the less representative ϕ_a is of the solutions obtained by the ants. Therefore, greater weight is given to ϕ_b than to ϕ_a in computing the score, i.e. $\varpi = 0.09$. Figure 9 shows the score from different μ while the other parameter, $\alpha, \beta, \rho, \varphi$ and ε are fixed at 0.9, 0.7, 0.5, 0.7 and 1, respectively. The results show that the best solution was obtained for μ set to 13%.

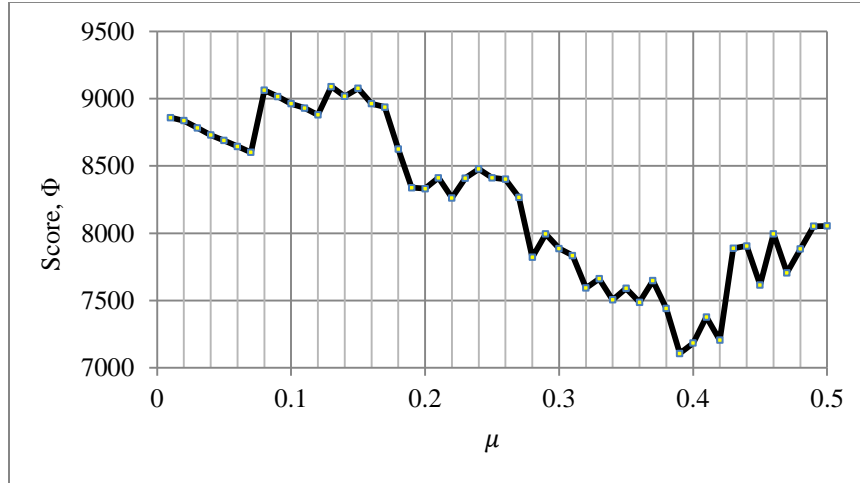


Figure 9 Comparison of the scores from different μ

From all of the studies in this chapter, it is concluded that the best parameters for the insertion based ants heuristic developed are $(\alpha, \beta, \rho, \varphi, \varepsilon, \mu) = 0.9, 0.7, 0.5, 0.7, 1, 0.13$, respectively. Next, the performance of the heuristic using the recommended set of parameters is discussed.

6.2 Solution quality and convergence

The quality of the solution as a function of number of iterations is studied on problem instance I9, the largest of the studied instances. As in the small problem instance runs in chapter 4, c is set to 0.99, thus placing the majority of weight on the first term in objective function (1) of the TDBSP. The optimal solution value for this problem instance with $c = 0.99$ is 23.97. Additional details associated with this solution are given in appendix.

The heuristic is run 100 times for each setting of a varying number of iterations ranging from 100 to 100,000. The average solution value obtained over the 100 runs is reported in Table 7. Convergence toward the optimal solution with

increasing number of iterations is depicted in Figure 10. To obtain solutions within 5% of optimality, more than 7,500 iterations were required.

Table 7 Average solution value for problem instance I9 by number of iterations completed

Number of Iterations	Average value of the best solutions
100	28.12
500	26.85
1,000	26.36
2,500	25.76
5,000	25.34
7,500	25.17
10,000	25.08
25,000	24.75
50,000	24.55
100,000	24.18

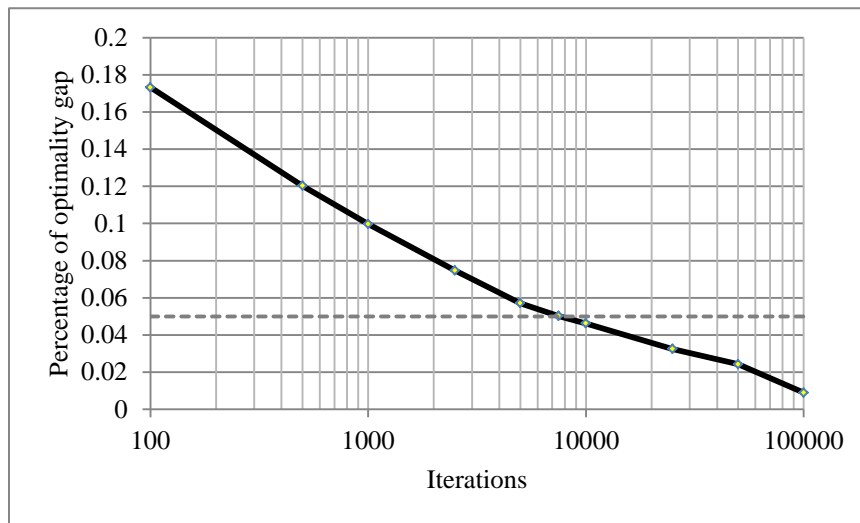


Figure 10 Optimality gap as a function of number of iterations

A second set of experiments was conducted on six randomly generated problem instances, including instances H1 and H3 discussed previously, to assess the increase in computational effort as a function of problem size. The number of iterations in each run was set to 1,000. Results are given in Table 8.

The results indicate that the run time of the modified insertion based ants algorithm grows slightly worse than linearly with problem size. Moreover, problem instances H4 through H6 are similar in size to problems expected in the field, at least in the Thai cement delivery business considered herein and, thus, the solution methodology should be efficient enough for real-world application.

Table 8 The capability of the size of problem that can be solved efficiently by heuristic

Instances	Input ($\theta: U: D_C: D_0: TW$)	Computational time (sec)
H1	24:7:33:35:0	8.48
H2	8:7:100:100:0	14.59
H3	24:7:100:100:0	37.54
H4	8:7:250:250:0	87.39
H5	8:7:350:350:0	228.54
H6	8:7:500:500:0	638.17

Chapter 7: Case Study Results

Exact solution of the Thai cement company could not be obtained through formulation and solution by CPLEX. Thus, the modified insertion based ants algorithm is applied. Since no optimal solution is known for the case study, and actual tours are proprietary to SCCC, the average tour duration over all vehicles supplied by SCCC is used as a basis for comparison. Note that if this value is multiplied by the number of vehicles and number of time periods in the planning horizon, this is equivalent to the first term in objective function (1). Thus, c is set to 1 in seeking the optimal solution for the case study by the heuristic.

The insertion based ants optimization algorithm is run for 100,000 iterations and the average tour duration for the best obtained solution is reported at each 10,000 iterations. Results of five experimental runs are provided in Table 9.

Table 9 Average solution value for the case study by number of iterations completed

Number of Iterations	Average tour duration of solution (days)					
	1st run	2nd run	3rd run	4th run	5th run	Average
1,000	3.962	3.956	3.975	3.967	3.962	3.964
5,000	3.963	3.956	3.963	3.967	3.962	3.962
10,000	3.954	3.956	3.963	3.965	3.960	3.960
20,000	3.954	3.956	3.963	3.961	3.960	3.959
30,000	3.948	3.956	3.963	3.961	3.951	3.956
40,000	3.948	3.956	3.959	3.961	3.951	3.955
50,000	3.948	3.956	3.959	3.961	3.951	3.955
60,000	3.948	3.956	3.959	3.953	3.951	3.953
70,000	3.948	3.956	3.958	3.953	3.951	3.953
80,000	3.948	3.956	3.953	3.953	3.951	3.952
90,000	3.946	3.956	3.953	3.950	3.951	3.951
100,000	3.946	3.956	3.953	3.950	3.951	3.951

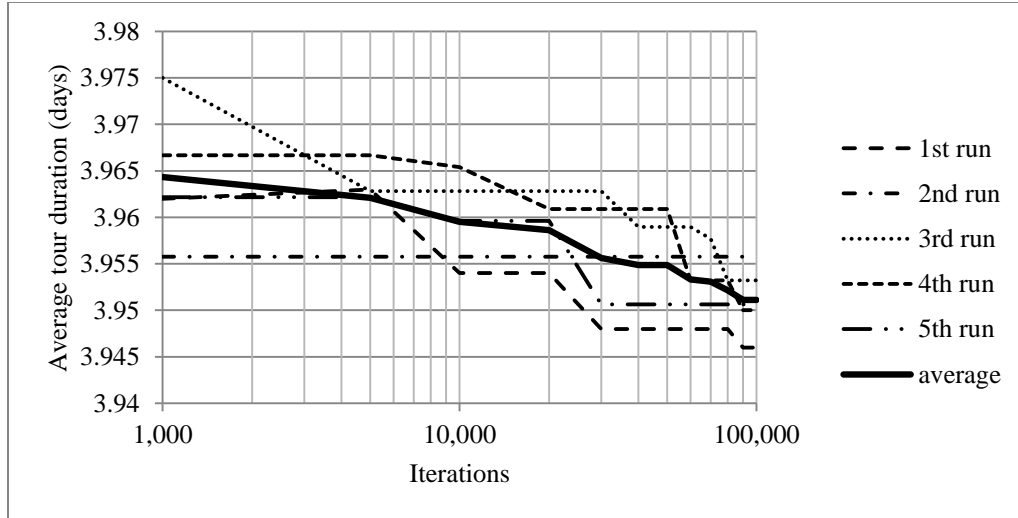


Figure 11 Comparison of the solution value in different runs by the number of iterations

From Figure 11, various improvement behaviors obtained from the algorithm were observed, i.e. rapid improvement at earlier iterations (in the 3rd run), gradual improvement through the iterations (in the 1st run), improvement at later iterations (in the 4th run) or stagnation since earlier iterations (in the 2nd run). However, on average, gradual improvement was noted over the iterations. The average tour duration obtained after 100,000 iterations is 3.95 days. Each run with 100,000 iterations of the algorithm required slightly longer than two hours of run time (including time for input and output).

By comparison, the average tour duration experienced by SCCC for the same customers using the same number of vehicles is 4.5 days. Thus, the obtained solution provides 12.3% improvement as compared with the average solutions obtained manually by experts at SCCC. It is worth noting that the loading time at the depot and backhaul customers and unloading time at the linehaul customers range from 3 to 38 hours. These times are fixed. Thus, a significant portion of the 4.5 days cannot be

reduced and the improvement as a percentage when considering the source of flexibility in the tours is actually significantly larger than 12%.

Another measure of performance of the results found by the algorithm can be obtained by investigating the difference between the average tour length that is obtained and a bound equal to the minimum possible tour length of tours that include both a customer and a mine while using the provided waiting, loading, unloading and resting times by SCCC and simultaneously relaxing the time window constraints. It turns out that the lowest possible tour length requires 3.27 days. This is 20.8% lower than the 3.95 days required for an average tour when time window constraints are enforced. Further, it is 37.6% lower than SCCC's average tour duration. By comparing the result from the algorithm and average tour duration provided by SCCC with this bound, one will note a difference of 0.68 and 1.23 days respectively. That is, the solution obtained by the algorithm leads to improvements in this measure by nearly 45% (i.e. $(1.23-0.68)/1.23$).

The case study involves very long time windows. To further investigate the performance of the algorithm under restrictions that are more difficult to meet, instances of the case study with smaller time windows are considered. In these runs, the time windows are categorized as follows: (1) morning time window [9:00 a.m., 12:00 p.m.]; (2) afternoon time window [12:00 p.m., 3:00 p.m.]; (3) evening time window [3:00 p.m., 6:00 p.m.]; and (4) nighttime window [6:00 p.m., 9:00 p.m.]. A time window falling within one of these categories is randomly assigned to each linehaul customer. The time window is repeated each day through the planning horizon. The results obtained from five runs, each with different seed values, but with

the same inputs are run. Each run involved 100,000 iterations. These results are provided in Table 10.

Table 10 Average solution value for the case study with restrictive time windows by number of iterations completed

Number of Iterations	Average tour duration of solution (days)					
	1st run	2nd run	3rd run	4th run	5th run	Average
1,000	4.358	4.361	4.348	4.345	4.347	4.352
5,000	4.332	4.346	4.342	4.333	4.342	4.339
10,000	4.332	4.346	4.331	4.330	4.328	4.333
20,000	4.331	4.333	4.311	4.329	4.328	4.327
30,000	4.329	4.333	4.311	4.327	4.322	4.324
40,000	4.329	4.325	4.311	4.327	4.322	4.323
50,000	4.329	4.325	4.311	4.327	4.322	4.323
60,000	4.324	4.324	4.311	4.327	4.322	4.322
70,000	4.324	4.324	4.311	4.327	4.319	4.321
80,000	4.324	4.324	4.311	4.327	4.319	4.321
90,000	4.324	4.324	4.311	4.327	4.319	4.321
100,000	4.324	4.324	4.311	4.327	4.319	4.321

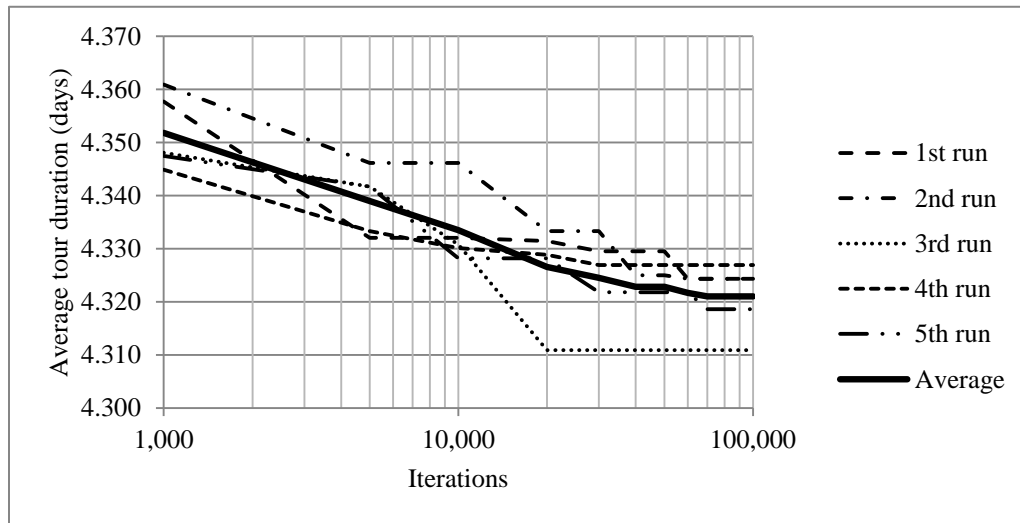


Figure 12 Comparison of the solution value in different runs by the number of iterations

From Figure 12, the average result shows that gradual improvement was observed over the iterations. The average tour duration obtained after 100,000 iterations is 4.32 days. This shows that the algorithm can be used to give solutions

even within more restrictive time windows. Furthermore, as expected, the average tour duration increases with reduced time window range. Even with these significantly reduced time windows, the obtained solution requires shorter average tour length than SCCC obtains.

The TDBSP is formulated with a fixed vehicle fleet; however, in the case study, the fleet size is not imposed. This is because the number of vehicles in SCCC's fleet exceeds their demand. It may be beneficial for the company to consider measures for reducing the number of vehicles employed in serving the customers. The framework proposed herein permits such consideration.

Chapter 8: Conclusions

In this study, the Truckload Delivery with Backhaul Scheduling Problem (TDBSP) is formulated and an ACO methodology developed for a related problem, the VRPBTW, is adapted for its solution. The TDBSP differs from the VRPBTW in that shipments are in units of truckloads, multiple time windows over multiple days are available for delivery to customers, limited space for servicing customers is available and multiple visits to each customer may be required. The problem is motivated by a real-world application arising at the Siam Cement Public Company Limited (SCCC), a leading cement producer in Thailand. SCCC uses lignite as the main energy source for its production plant. Lignite mines are abundant in their area of operations, particularly in Northern Thailand. Thus, to reduce costly empty backhaul movements, the company combines trips to deliver cement to customers with hauling lignite back to the production plant. Experts at the SCCC production plant assign vehicles to cement customers and lignite mines based on manual computations and experience. This study provides mathematical and computational frameworks for addressing this real-world problem.

The primary contributions of this work are (1) the conceptual framework for interpreting the real-world problem arising at SCCC and other companies with similar problems as an optimization problem; (2) development of a mixed linear integer formulation for the conceived TDBSP; (3) modifications to an insertion based ants optimization method designed for a related, but different problem; and (4) analysis of the performance of the proposed techniques on a real-world problem instance.

Additional practicalities of the real-world application might be considered in modeling this problem. For example, it was assumed that the mines have unlimited stocks of lignite. In reality, the amount of stock at each mine may vary from day to day. If the lignite mines are able to predict their stock levels for the planning horizon, time-dependent lignite availability might be considered. If such information is not shared with lignite customers and stock out is possible, the tours may need to be dynamically updated.

The monthly demand for cement at customers and lignite at the company are nearly equivalent over a typical month; however, these demands vary from week to week, creating the imbalances that cause the need for empty backhaul. To reduce such empty backhaul, the company may consider timing sales incentives and promotions or using other demand management strategies accordingly. Additionally, they might consider stocking excess lignite at the production plant.

Finally, the actual demand for lignite at the production plant in reality is a function of the energy potential of the lignite that is purchased from each mine. No information concerning the energy potential of lignite excavated from the various mines is available to the company. However, if this information could be known in advance, the formulation and solution technique could be appropriately modified to provide solutions that incorporate backhaul trips involving a heterogeneous set of lignite mines that meet energy needs rather than requiring a fixed number of truckloads from homogeneous lignite sites.

The performance of the developed insertion based ants optimization methodology might also be compared with that of alternative solution techniques that might be developed on a tabu search or simulated annealing framework.

Appendix

This appendix provides the results of the Table 4.1 including the schedule, total trip duration, objective function value and computational time for each instance. τ_{ij} matrix represent the travel time between linehaul customer and backhaul customer. Vector $\tau_{o,i}$ represents travel time between linehaul customer, i , and depot, as well as vector $\tau_{o,j}$ for backhaul customer, j . a_i and b_i represent earliest and latest time periods of each day that linehaul customer i can be delivered.

The input assumed zero service time, s_i , for all nodes. The symmetric travel time is assumed for every pair of nodes. Total delivery demand at each linehaul customer and amount of available pickup at backhaul customers is assumed to be one truckload. θ is set to 4 periods. A limit of one vehicle is imposed at each site per time period, $\lambda(t) = 1, \forall t \in \mathcal{T}$. To give priority to solutions with shorter tour duration, the coefficient c in the objective function is set to be 0.99.

The representation of instance inputs, $\theta: U: D_C: D_0: TW$, are defined in Chapter 4. On the schedule element, each column represents time period and each row represents tour assigned for each vehicle. C# in the schedule element means the vehicle arrival time period at linehaul customer #, as well as M# for backhaul customer #. Blackened block means the time period that vehicle starting from or arrival at the depot. Runs were performed using *IBM's ILOG CPLEX OPL 12.2* and were conducted on a computer with Intel Core i5 CPU and 2.0 GB of RAM. The order of instances is the same as in Table 4.1. The following shows the details of obtained results.

Instance II - 4:4:5:3:1

$$\tau_{ij} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 5 & 5 & 5 \\ 5 & 5 & 5 \end{pmatrix}$$

$$\tau_{o,i} = [1 \ 2 \ 2 \ 1 \ 2]$$

$$\tau_{o,j} = [2 \ 3 \ 2]$$

$$a_i = [1 \ 0 \ 1 \ 1 \ 2]$$

$$b_i = [3 \ 1 \ 2 \ 2 \ 3]$$

Result

v\t	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1		■	C4	■												
2						■		C5		■						
3					■	C1	M3		■							
4			■		C2	M1		■								
5	■		C3	M2			■									

Total duration = 21

Objective function value = 20.91

Computational time = 45.44 Sec

Instance I2 – 4:4:5:3:0

$$\tau_{ij} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 5 & 5 & 5 \\ 5 & 5 & 5 \end{pmatrix}$$

$$\tau_{o,i} = [1 \ 2 \ 2 \ 1 \ 2]$$

$$\tau_{o,j} = [2 \ 3 \ 2]$$

Result

v\t	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	█		C3	M2			█									
2						█		C5		█						
3		█	C4	█												
4					█	C1	M1		█							
5			█		C2	M3		█								

Total duration = 21

Objective function value = 20.91

Computational time = 69.56 Sec

Instance I3 – 4:5:2:2:0

$$\tau_{ij} = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

$$\tau_{o,i} = [1 \ 2] \quad \tau_{o,j} = [1 \ 2]$$

Result

v\l	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
1				C2	M2															
2		C1	M1																	

Total duration = 8

Objective function value = 7.93

Computational time = 0.63 Sec

Instance I4 – 4:5:3:3:0

$$\tau_{ij} = \begin{pmatrix} 1 & 2 & 1 \\ 2 & 1 & 2 \\ 1 & 2 & 1 \end{pmatrix}$$

$$\tau_{o,i} = [1 \ 2 \ 1] \quad \tau_{o,j} = [1 \ 2 \ 1]$$

Result

v\t	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
1				C3	M3															
2		C1	M1																	
3				C2	M2															

Total duration = 11

Objective function value = 10.95

Computational time = 5.19 Sec

Instance I5 – 4:5:4:4:1

$$\tau_{ij} = \begin{pmatrix} 1 & 2 & 1 & 1 \\ 2 & 1 & 2 & 1 \\ 1 & 2 & 1 & 2 \\ 1 & 1 & 2 & 1 \end{pmatrix}$$

$$\tau_{o,i} = [1 \ 2 \ 1 \ 2] \quad \tau_{o,j} = [1 \ 2 \ 1 \ 2]$$

$$a_i = [1 \ 0 \ 1 \ 1] \quad b_i = [3 \ 1 \ 2 \ 2]$$

Result

v\t	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
1	■	C3	M3	■																
2		■	C1	M4		■														
3					■			C4	M1	■										
4			■		C2	M2		■												

Total duration = 16

Objective function value = 15.91

Computational time = 35.46 Sec

Instance I6 – 4:5:4:4:0

$$\tau_{ij} = \begin{pmatrix} 1 & 2 & 1 & 1 \\ 2 & 1 & 2 & 1 \\ 1 & 2 & 1 & 2 \\ 1 & 1 & 2 & 1 \end{pmatrix}$$

$$\tau_{o,i} = [1 \ 2 \ 1 \ 2] \quad \tau_{o,j} = [1 \ 2 \ 1 \ 2]$$

Result

v\t	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
1		■	C1	M4		■														
2				■	C3	M3	■													
3			■		C2	M2		■												
4	■		C4	M1	■															

Total duration = 16

Objective function value = 15.90

Computational time = 37.13 Sec

Instance I7 – 4:5:5:5:1

$$\tau_{ij} = \begin{pmatrix} 1 & 2 & 1 & 1 & 1 \\ 2 & 1 & 2 & 1 & 1 \\ 1 & 2 & 1 & 2 & 1 \\ 1 & 1 & 2 & 1 & 2 \\ 1 & 1 & 1 & 2 & 1 \end{pmatrix}$$

$$\tau_{o,i} = [1 \ 2 \ 1 \ 2 \ 1]$$

$$\tau_{o,j} = [1 \ 2 \ 1 \ 2 \ 1]$$

$$a_i = [1 \ 0 \ 1 \ 1 \ 2]$$

$$b_i = [3 \ 1 \ 2 \ 2 \ 3]$$

Result

v\t	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
1					■		C4	M1	■											
2							■	C1	M3	■										
3	■	C3	M5	■																
4		■	C5	M2		■														
5			■		C2	M4		■												

Total duration = 19

Objective function value = 18.94

Computational time = 651.86 sec

Instance I8 – 4:5:5:5:0

$$\tau_{ij} = \begin{pmatrix} 1 & 2 & 1 & 1 & 1 \\ 2 & 1 & 2 & 1 & 1 \\ 1 & 2 & 1 & 2 & 1 \\ 1 & 1 & 2 & 1 & 2 \\ 1 & 1 & 1 & 2 & 1 \end{pmatrix}$$

$$\tau_{o,i} = [1 \ 2 \ 1 \ 2 \ 1]$$

$$\tau_{o,j} = [1 \ 2 \ 1 \ 2 \ 1]$$

Result

v\t	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
1	■	C3	M1	■																
2		■	C5	M2		■														
3			■		C4	M4		■												
4					■		C2	M5	■											
5							■	C1	M3	■										

Total duration = 19

Objective function value = 18.94

Computational time = 1198.26 sec

Instance I9 – 4:5:6:6:0

$$\tau_{ij} = \begin{pmatrix} 1 & 2 & 1 & 1 & 1 & 1 \\ 2 & 1 & 2 & 1 & 1 & 1 \\ 1 & 2 & 1 & 2 & 1 & 1 \\ 1 & 1 & 2 & 1 & 2 & 1 \\ 1 & 1 & 1 & 2 & 1 & 2 \\ 1 & 1 & 1 & 1 & 2 & 1 \end{pmatrix}$$

$$\tau_{o,i} = [1 \ 2 \ 1 \ 2 \ 1 \ 2]$$

$$\tau_{o,j} = [1 \ 2 \ 1 \ 2 \ 1 \ 2]$$

Result

v\l	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
1									■	C3	M5	■								
2	■	C5	M3	■																
3		■		C6	M1	■														
4					■		C2	M2		■										
5			■		C4	M4		■												
6							■	C1	M6		■									

Total duration = 24

Objective function value = 23.97

Computational time = 65279.36 sec

Bibliography

- Anily, S., "The Vehicle-Routing Problem with Delivery and Back-Haul Options," *Naval Research Logistics*, **43**, 415-434 (1996)
- Bodin, L. D., B. L. Golden, A. A. Assad, and M. O. Ball, "Special Issue on the Routing and Scheduling of Vehicles and Crews," *Computers and Operations Research*, **2**, 63-211 (1983)
- Bullnheimer, B., R.F. Hartl, C. Strauss, "A new rank based version of the ant system: A computational study," Working Paper No.1, SFB Adaptive Information Systems and Modelling in Economics and Management Science, Vienna (1997)
- Calvete, H. I., C. Gale, M. J. Oliveros, B. S. Valverde, "A goal programming approach to vehicle routing problems with soft time windows." *European Journal of Operational Research*, **177**, 1720-1733 (2007)
- Cordeau, J.-F., G. Desaulniers, J. Desrosiers, M. M. Solomon, F. Soumis, "The VRP with Time Windows," *Les Cahiers du GERAD*, G-99-13 (2000)
- Cordeau, J.-F., G. Laporte, A. Mercier, "A Unified Tabu Search Heuristic for Vehicle Routing Problems with Time Windows," *Journal of the Operational Research Society*, **52**, 8, 928-936 (2001)
- Dorigo, M., V. Maniezzo, A. Coloni, "Positive feedback as a search strategy," Dipartimento di Elettronica, Politecnico di Milano, Italy, Tech. Rep. 91-016 (1991)
- Dorigo, M., "Optimization, learning and natural algorithms (*in italian*)," Ph.D. dissertation, Dipartimento di Elettronica, Politecnico di Milano, Italy (1992)
- Dorigo, M., V. Maniezzo, A. Coloni, "The Ant System: Optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics—Part B*, **26**, 1, 1-13 (1996)
- Favaretto, D., E. Moretti, P. Pellegrini, "Ant colony system for a VRP with multiple time windows and multiple visits," *Journal of Interdisciplinary Mathematics*, **10**, 2, 263–284 (2007)
- Goetschalckx, M., C. Jacobs-Blecha, "The vehicle routing problem with backhauls," *European Journal of Operational Research*, **42**, 39-51 (1989)

- Golden, B. L., S., Raghavan, E. A., Wasil, “The Vehicle Routing Problem: Latest Advances and New Challenges,” Springer, New York (2008).
- Jacobs-Bleacha, C., M. Goetschalackx, “The Vehicle Routing Problem with Backhauls: Properties and Solution Algorithms,” Georgia Tech Research Corporation, Atlanta, Georgia (1998)
- Mingozzi A., S. Giorgi, R. Baldacci, “An Exact Method for the Vehicle Routing Problem with Backhauls,” *Transportation Science*, **33**, 3 (1999)
- Noon, C., “The generalized traveling problem,” PhD Dissertation, The University of Michigan, Ann Arbor (1988)
- Potvin J., C. Duhamel, F. Guertin, “A Genetic Algorithm for Vehicle Routing with Backhauling,” *Applied Intelligence*, **6**, 345-355 (1996)
- Reimann, M., K. Doerner, R. F. Hartl, “Insertion Based Ants for Vehicle Routing Problems with Backhauls and Time Windows,” *ANTS 2002*, LNCS 2463, 135-148 (2002)
- Reimann, M., K. Doerner, R. F. Hartl, “Analyzing a Unified Ant System for the VRP and Some of Its Variants,” *EvoWorkshops 2003*, LNCS 2611, 300–310 (2003)
- Regan, A.C., H.S. Mahmassani, P. Jaillet, “Improving the Efficiency of Commercial Vehicle Operations Using Real-Time Information: Potential Uses and Assignment Strategies,” *Transportation Research Record*, **1493**, 188-198 (1995)
- Regan, A.C., H.S. Mahmassani, P. Jaillet, “Dynamic Decision Making for Commercial Fleet Operations Using Real-Time Information,” *Transportation Research Record*, **1537**, 91-97 (1996a)
- Regan, A.C., H.S. Mahmassani, P. Jaillet, “Dynamic Dispatching Strategies under Real-Time Information for Carrier Fleet Management,” in Lesort, J.B. (ed), *Transportation and Traffic Theory Pergamon*, 737-756 (1996b)
- Regan, A.C., H.S. Mahmassani, P. Jaillet, “Evaluation of Dynamic Fleet Management Systems: Simulation Framework,” *Transportation Research Record*, **1645**, 176-184 (1998)
- Savelsbergh, M., “Local Search in Routing Problems with Time Windows,” Report OS-R8409, Centre for Mathematics and Computer Science, Amsterdam (1984)

- Savelsbergh, M. W. P., M. Sol, "The General Pickup and Delivery Problem," *Transportation Science*, **29**, 1, 17-29 (1995)
- Solomon, M. M., "On the Worst-Case Performance of Some Heuristics for the Vehicle Routing and Scheduling Problem with Time Window Constraints," *Networks*, **16**, 161-174 (1986)
- Solomon, M. M., "Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints," *Operations Research*, **35**, 2, 254-265 (1987)
- Solomon, M. M., J. Desrosiers, "Time Window Constrained Routing and Scheduling Problems," *Transportation Science*, **22**, 1 (1988)
- Tang, H., E. Miller-Hooks, "Solving a generalized traveling salesperson problem with stochastic customers," *Computers and Operations Research*, **34**, 1963-1987 (2007)
- Toth, P., D. Vigo, "An exact algorithm for the vehicle routing problem with backhauls," *Transportation science*, **31**, 372-385 (1997)
- Toth, P., D., Vigo, "The Vehicle Routing Problem." SIAM Monographs on Discrete Mathematics and Applications, Philadelphia (2002)
- Yang, J., P. Jaillet, H. Mahmassani, "Real-Time Multi-Vehicle Truckload Pick-Up and Delivery Problems," *Transportation Science*, **38**, 2, 135-148 (2004)