Integrate Vehicle Routing and Truck Sequencing in Cross-docking Operations

Ching-Jung Ting  
Yuan-Ze University, ietingcj@saturn.yzu.edu.tw

Ting-Don Chen  
Yuan Ze University, s1005436@mail.yzu.edu.tw

Follow this and additional works at: https://digitalcommons.georgiasouthern.edu/pmhr_2014

Part of the Industrial Engineering Commons, Operational Research Commons, and the Operations and Supply Chain Management Commons

Recommended Citation

https://digitalcommons.georgiasouthern.edu/pmhr_2014/32
XXX. INTEGRATE VEHICLE ROUTING AND TRUCK SEQUENCING IN CROSS-DOCKING OPERATIONS

Ching-Jung Ting
Department of Industrial Engineering and Management, Yuan Ze University, Taiwan
ietingcj@saturn.yzu.edu.tw

Ting-Dong Cheng
Department of Industrial Engineering and Management, Yuan Ze University, Taiwan
s1005436@mail.yzu.edu.tw

Abstract

Cross-docking is a an important logistics strategy in which freight is unloaded from inbound vehicles and directly loaded into outbound vehicles, with little or no storage in between. This study considers a cross-docking system which combines the vehicle routing problem with cross-docking (VRPCD) for both inbound and outbound operations and truck sequencing problem at docks. The objective is to minimize the logistics center operation costs and transportation costs. We first formulated the integrated problem with a mixed integer programming model. Since VRPCD and sequencing problems are NP-hard, the integrated problem is also an NP-hard problem. We propose an ant colony optimization (ACO) algorithm to solve the VRPCD and sequencing problem by two independent ant colonies sequentially. The proposed ACO is tested with 15 randomly generated instances. The results show that ACO can obtain the optimal solutions in small size instances. We believe the proposed ACO algorithms can be used for practical use for the cross-docking system.

1 Introduction

Cross-docking has become an increasingly popular distribution strategy implemented by
organizations to improve supply chain efficiency and minimize distribution cost (Napolitano, 2000). In a cross-docking system a product is received at a terminal, consolidated with other products going to the same destination, and then shipped at the earliest opportunity, without going into long-term storage (Bartholdi and Gue, 2004). Thus, the cross docking could eliminate two of the four major activities in a traditional warehouse, the storing and picking activities. The advantage of cross docking is cost savings in storage cost, inventory cost, transportation cost, and labor cost (Saddle Creek Corp., 2008).

Cross docking is defined as a process of moving merchandise from the receiving dock to shipping for shipping without placing it first into storage locations (Material Handling Institute). Agustina et al. (2010) provided a general picture of the mathematical models used in cross-docking planning. Stephan and Boysen (2011) discussed the cross-docking concept, classified relevant cross-dock settings and defined important decision problems. Van Belle et al. (2012) presented an extensive overview of the cross-docking concept and described several characteristics. At a cross-docking center (CDC), products flow through every day, involving suppliers, cross-docking operators, and customers. When implementing a cross-docking operation, two key points are simultaneous arrival and consolidation (Wen et al., 2009). Obviously, the truck scheduling is closely related to inbound and outbound vehicle routing. Buijs et al. (2014) reviewed the cross-docking system operations which include local and network related issues. The authors also mentioned that synchronization between local and network scheduling is important but rare research discussed.

In order to efficiently process the transshipment at the CDC, both inbound and outbound schedules should be synchronized, several procedures have been introduced in recent years (Stephan and Boysen, 2011). The truck scheduling/sequencing problem assumed that the products on each inbound vehicle are known and arrival and departure times are also known. The problem could be reduced to sequencing all trucks when the number of docks is one. On the inbound side, the vehicle routing determines the arrival times of trucks. On the outbound side, succeeding vehicle routings possibly set the earliest and latest departure time based on the interaction between inbound and outbound trucks. Because of the consolidation however, the pickup and delivery routes are not independent. Obviously, both problems could be solved in a simultaneous manner so that considerable improvements of the overall planning task might occur.

So far vehicle routing problem and truck sequencing problem in the cross-docking system have been treated separately. The truck (or some models using trailer) scheduling is important decision in the cross docking operation. It determines the sequences of inbound and outbound trucks. For the truck sequencing/scheduling problem, Boysen and Fliedner (2010) structured a classification scheme for the cross-docking scheduling problem. Yu and Egbelu (2008) addressed a truck scheduling problem where the product assignments from inbound trucks to outbound trucks are determined simultaneously with the docking sequences of the inbound and outbound trucks. Boysen et al. (2008) introduced a base model for scheduling trucks at cross-docking terminals, where a "one
inbound dock serves one outbound dock” problem is considered. Chen and Lee (2009) studied the truck scheduling problem as a two-machine flow shop scheduling problem and showed that the problem is strongly NP-hard. Li et al. (2009) considered a multiple dock cross-docking where all docks are the same and could be used either as inbound or outbound, the problem is formulated as a parallel machine scheduling problem, where there is no differentiation between inbound and outbound operations.

Alpan et al. (2010) solved the multiple inbound and outbound dock configuration problem by a bounded dynamic programming. Vahdani and Zandieh (2010) presented an exhausted analysis of the performance of the variable neighborhood search algorithm for the truck scheduling problem. In their study different neighborhood structures and stopping criterion are tested. According to their results VNS is recommended to solve truck scheduling in cross-docking system problems. Boloori Arabani et al. (2011) also presented five metaheuristics to tackle same problem. Alpan et al. (2011) considered a multiple inbound and outbound dock configuration where the objective is to find the best schedule of transshipment operations to minimize the sum of inventory holding and truck replacement cost. Liao et al. (2012) presented two hybrid evolutionary algorithms (EA) to solve the one inbound and one outbound dock truck sequencing problem. They tested the EAs with the instances generated by Yu and Egbelu (2008). Liao et al. (2013) extended Liao et al. (2012) study to a multi-door cross docking operation under a fixed outbound truck departure schedule. The problem was solved by six different metaheuristic algorithms and ant colony optimization (ACO) is the best among all algorithms tested.

Product destined for a cross dock in many cases needs to be picked up at various suppliers, and has to be delivered to multiple customers after consolidation at the cross-dock. Both the pickup and the delivery process in such an environment can be seen as a vehicle routing problem (VRP). Lee et al. (2006) is probably the first that takes both VRP and cross-docking into consideration. They proposed a tabu search (TS) to determine the number of vehicles and the optimal vehicle routing schedule at a cross-dock to minimize the sum of transportation cost and fixed cost of vehicles. Liao et al. (2010) proposed another tabu search algorithm to solve the same problem. Wen et al. (2009) studied the vehicle routing problem with cross-docking (VRPCD). In this problem, products from suppliers are picked up by a homogeneous fleet of vehicles and then consolidated at a cross-dock and immediately delivered to customers by the same set of vehicles. Santos et al. (2011a) proposed a branch-and-bound algorithm to solve the VRPCD, while Santos et al. (2011b) presented a novel column generation formulation and solved it with a branch-and-price algorithm.

The VRP and the operations at the cross-dock are tightly coupled to impact the supply chain process. The vehicle routes could specify loading lists for both inbound and outbound trucks and the departure times of trucks leaving the cross-dock. Therefore, the coordination of cross-docking operations requires a holistic approach. To our knowledge, there is no research considers both VRPCD and truck sequencing for the cross-docking system. This research integrates VRPCD and truck sequencing and presents a new mixed integer programming model for the integrated problem. Since both VRPCD and sequencing are NP-hard, the integrated problem is also NP-hard. Due to the problem
complexity, we propose an ant colony optimization algorithm to solve the problem. To the best of our knowledge, there are limited research apply ACO to the cross-docking related problem. Musa et al. (2010) proposed a model that assigns capacity to the available routes and allocates loads to those routes. Liao et al. (2013) addressed the scheduling of inbound trucks to outbound trucks in a multiple docks setting.

The rest of the paper is organized as follows. In section 2, problem description and mixed integer programming model is presented. The proposed ant colony optimization algorithm is explained in section 3. Section 4 presents the computational experiments for the proposed ACO algorithm. Finally, section 5 concludes the study and suggests the future research.

2 Problem Description

2.1 Assumptions

This study considers a distribution network with a cross-dock terminal. The vehicle routing problem with cross-docking (VRPCD) for both inbound and outbound operations and truck sequencing problem at docks are integrated. Products at various locations are collected in the cross-dock prior to transport to their destinations. After consolidation based to product destination in the cross-dock, products are moved from the cross-dock to their respective destinations. The objective is to minimize the CDC operation times and transportation times for both inbound and outbound trucks. During the process inbound trucks can be scheduled at any time since no restrictions on arrival time are considered. Service times at CDC vary from truck to truck, which depends on the flow each truck carries. The facility layout considered allows an intermediate storage with unlimited capacity in front of outbound dock. As soon as an outbound truck loads its predefined set of products based on its routing, it leaves the terminal.

The assumptions in this research are as follows.
1. One side of the dock is designated to inbound trucks and the other side to outbound trucks.
2. The numbers of both inbound and outbound vehicles are known. Vehicles will start and end at the cross-docking terminal.
3. The pickup and delivery locations are known. The number and types of products to be picked up or delivered at each location are also known in advance.
4. The unloading and loading time at the terminal is constant, while the transfer time between docks is also constant.
5. Transshipment time between docks is fixed.
6. There are multiple products to be pickup and delivered in the cross-docking system. Product units of a specific type can satisfy any demand for this product.
7. Each outbound dock is capable to storage the outbound truck’s freight.

Based on these assumptions, a mixed integer programming is formulated to minimize the total processing time which includes inbound and outbound transportation
time and the consolidation time at the terminal by integrating the vehicle routing problem and truck sequencing problem. Since each of both problems is NP-hard, the integrated problem is also NP-hard. An exact solution approach is difficult to find solution within reasonable time for real world large problem. We propose an ant colony optimization algorithm to solve this problem.

### 2.2 Mathematical Formulation

In developing the mathematical model, the following notations are used.

**Set:**
- \{0\}: cross-docking terminal
- \(A\): set of pickup nodes
- \(B\): set of delivery nodes
- \(G\): set of product types
- \(K\): set of pickup trucks
- \(L\): set of delivery trucks

**Parameters:**
- \(g_{ia}\): quantity of product \(g\) that is picked up from node \(i\)
- \(g_{ib}\): demand quantity of product \(g\) for node \(i\)
- \(H\): constant moving time from the inbound dock to the outbound dock
- \(M\): a big number
- \(Q\): truck capacity
- \(t_{ij}\): travel time from node \(i\) to \(j\)
- \(U\): truck changeover time at docks
- \(\alpha_1\): unit unloading time
- \(\alpha_2\): unit loading time

**Decision variables:**
- \(D_{lD}^g\): quantity of product \(g\) on truck \(l\)
- \(e_{ki}^g\): quantity of product \(g\) when truck \(k\) departing from node \(i\)
- \(F_k\): the starting time when truck \(k\) uses the dock
- \(P_k^g\): total quantity of product \(g\) on truck \(k\)
- \(R_l\): the starting time when truck \(l\) use the dock
- \(T\): total operating time for the cross-docking terminal

\[
v_{kl} = \begin{cases} 
1, & \text{if inbound truck } k \text{ transfers products to outbound truck } l \\
0, & \text{Otherwise}
\end{cases}
\]

\[
w_{ki}^g = \begin{cases} 
1, & \text{if truck } k \text{ visit } j \text{ immediately after } i \\
0, & \text{Otherwise}
\end{cases}
\]
\[ y_{ks} = \begin{cases} 1, & \text{if truck } k \text{ departs before truck } s \text{ at receiving dock} \\ 0, & \text{Otherwise} \end{cases} \]

\[ z_{lq} = \begin{cases} 1, & \text{if truck } l \text{ departs before truck } q \text{ at shipping dock} \\ 0, & \text{Otherwise} \end{cases} \]

The mathematical model for this integrated problem is as follows.

Min \[ Z = \sum_{i \in A \cup \{0\}} \sum_{j \in B \cup \{0\}} \sum_{k \in K} x^k_{ij} t_{ij} + \sum_{i \in A \cup \{0\}} \sum_{j \in B \cup \{0\}} \sum_{l \in L} x^l_{ij} t_{ij} + T \] (1)

S.T. \[ \sum_{j \in B \cup \{0\}} \sum_{k \in K} x^k_{ij} = 1 \quad \forall j \in A \] (2)

\[ \sum_{j \in A \cup \{0\}} \sum_{k \in K} x^k_{ij} = 1 \quad \forall i \in A \] (3)

\[ \sum_{j \in A \cup \{0\}} x_{ij}^l = 1 \quad \forall j \in B \] (4)

\[ \sum_{j \in B \cup \{0\}} x_{ij}^l = 1 \quad \forall i \in B \] (5)

\[ \sum_{j \in A \cup \{0\}} x^k_{ih} = \sum_{j \in B \cup \{0\}} x^k_{bj} \quad \forall h \in A, k \in K \] (6)

\[ \sum_{j \in B \cup \{0\}} x^l_{ih} = \sum_{j \in A \cup \{0\}} x^l_{bj} \quad \forall h \in B, l \in L \] (7)

\[ \sum_{j \in A \cup \{0\}} x^k_{0j} = 1 \quad \forall k \in K \] (8)

\[ \sum_{j \in B \cup \{0\}} x^l_{0j} = 1 \quad \forall l \in L \] (9)

\[ e^g_{k0} = 0 \quad \forall g \in G, k \in K \] (10)

\[ e^g_{li} - M (1 - x^l_{ij}) \leq 0 \quad \forall i \in B, \ g \in G, \ l \in L \] (11)

\[ e^g_{ki} - a^g_j - M (1 - x^k_{ij}) \leq e^g_{ij} \quad \forall i \in A \cup \{0\}, \ j \in A, \ g \in G, k \in K \] (12)

\[ e^g_{ij} \leq e^g_{li} - b^g_j + M (1 - x^l_{ij}) \quad \forall i \in B \cup \{0\}, \ j \in B, \ g \in G, l \in L \] (13)
\[ P^g_k = \sum_{i \in A(0)} \sum_{j \in A} x^k_{i j} \cdot a^g_{j} \quad \forall g \in G, k \in K \] (14)

\[ \sum_{g \in G} P^g_k \leq Q \quad \forall k \in K \] (15)

\[ D^g_l = \sum_{i \in B(0)} \sum_{j \in B} x^l_{i j} \cdot b^g_{j} \quad \forall g \in G, l \in L \] (16)

\[ \sum_{g \in G} D^g_l \leq Q \quad \forall l \in L \] (17)

\[ \sum_{l \in L} w^g_{kl} = P^g_k \quad \forall g \in G, k \in K \] (18)

\[ \sum_{k \in K} w^g_{kl} = D^g_l \quad \forall g \in G, l \in L \] (19)

\[ w^g_{kl} \leq M \cdot v_{kl} \quad \forall k \in K, l \in L \] (20)

\[ F_s \geq \sum_{i \in A(0)} \sum_{j \in A(0)} x^k_{i j} \cdot t_{i j} \quad \forall k, s \in K, k \neq s \] (21)

\[ F_s \geq F_k + \alpha_1 \cdot \sum_{g \in G} P^g_s + U - M(1 - y_{k s}) \quad \forall k, s \in K, k \neq s \] (22)

\[ F_k \geq F_s + \alpha_1 \cdot \sum_{g \in G} P^g_k + U - M \cdot y_{k s} \quad \forall k, s \in K, k \neq s \] (23)

\[ R_1 + \alpha_2 \cdot \sum_{g \in G} D^g_s \geq F_s + \alpha_2 \cdot \sum_{g \in G} w^g_{s l} + H - M(1 - v_{k s}) \quad \forall k \in K, l \in L \] (24)

\[ R_q \geq R_1 + \alpha_2 \cdot \sum_{g \in G} D^g_q + U - M(1 - z_{l q}) \quad \forall l, q \in L, l \neq q \] (25)

\[ R_1 \geq R_q + \alpha_2 \cdot \sum_{g \in G} D^g_q + U - M \cdot z_{l q} \quad \forall l, q \in L, l \neq q \] (26)

\[ T = \max[R_1 + \alpha_2 \cdot \sum_{g \in G} D^g_s] - \min[F_s] \quad \forall k \in K, l \in L \] (27)

\[ x^k_{i j} \in \{0,1\} \quad \forall k \in K, i,j \in A \cup \{0\}, i \neq j \] (28)

\[ x^l_{i j} \in \{0,1\} \quad \forall l \in L, i,j \in B \cup \{0\}, i \neq j \] (29)
\[ y_{ks} \in \{0,1\} \quad \forall k,s \in K, k \neq s \] (30)

\[ z_{lq} \in \{0,1\} \quad \forall l,q \in L, l \neq q \] (31)

\[ e_{ki}^g \geq 0 \quad \forall k \in K, i \in A \cup \{0\}, g \in G \] (32)

\[ e_{li}^g \geq 0 \quad \forall l \in L, i \in B \cup \{0\}, g \in G \] (33)

\[ w_{ki}^g \geq 0 \quad \forall k \in K, l \in L, g \in G \] (34)

\[ F_k \geq 0 \quad \forall k \in K \] (35)

\[ R_l \geq 0 \quad \forall l \in L \] (36)

\[ D_l^g \geq 0 \quad \forall l \in L, g \in G \] (37)

\[ P_k^g \geq 0 \quad \forall k \in K, g \in G \] (38)

The objective function (1) sums up the inbound and outbound transportation cost and the operation cost at the cross-docking terminal. Constraints (2) and (3) ensure that every pickup node can only be served by only one truck. Constraints (4) and (5) state that each pickup node can only be served by only one truck. Constraints (6) and (7) ensure that a vehicle has to arrive and leave a node for inbound and outbound trucks, respectively. Constraints (8) and (9) ensure that every inbound and outbound truck must be used. Constraint (10) states that the inbound truck will leave the cross-docking terminal with no commodity, while constraint (11) ensures that no commodity on outbound truck when it return to the cross-docking terminal. Constraints (12) and (13) express the commodity on the consecutive movement of vehicles for inbound and outbound trucks, respectively. Constraint (14) represents the total number of units of each product type when a truck return to the terminal and (15) states that the quantity of loaded products cannot exceed the truck capacity. Similarly, constraints (16) and (17) state the quantity of loaded products for outbound truck cannot exceed the truck capacity.

Constraint (18) ensures that the total number of units of product \( g \) that transfer from inbound truck \( k \) to all outbound trucks is exactly the initially loaded amount in that truck. Similarly, constraint (19) ensures that the total number of units of product \( g \) that transfer to outbound truck \( l \) from all inbound trucks is exactly the loaded amount in that truck. Constraint (20) enforces the correct relationship between transfer quantity variable and transfer variable. Constraint (21) states that the first truck to be docked must after all inbound trucks arrive at the cross-docking terminal. Constraint (22)-(23) make a valid sequence for starting times for the inbound trucks based on their order. Constraint (24) connects the departure time for an outbound truck to the arrival time of an inbound truck.
if a flow must be transferred from that inbound truck. Similar to constraints (22)-(23),
constraints (25)-(26) function in a similar manner for the outbound trucks. Constraint (27)
sets the total operating time at the cross-docking terminal. Constraints (28)-(31) are the
integrality constraints. Constraints (32)-(38) are the non-negativity constraints.

3 Ant Colony Optimization Algorithm

The ant colony optimization (ACO) algorithm was first proposed by Dorigo et al. (1996).
Subsequently, many variants of ACO have been developed and applied extensively in the
combinatorial optimization problems. Descriptions of available ACO algorithms and
related literature review can be obtained in Dorigo and Stützle (2004).

This section introduces the proposed algorithm for solving the integrated problem.
Our MACO adopts a hierarchical ACO structure with different transition rules for
different ant colonies. The upper level is for the vehicle routing problem with cross-
docking subproblem, while the lower level is for the truck sequencing subproblem which
is solved based on the results of the VRPCD. These two phases are applied sequentially.
Moreover, two different pheromone matrices and pheromone updating rules are adopted
to record pheromone information for each colony, respectively. The idea is to find the
best routing for both inbound and outbound trucks, then the consolidation process can be
achieved by minimizing the total operating time. The procedures of our ACO are
described as follows and introduced in the following sections.

Step 1: Initialization:
   a. Set parameters.
   b. Initialize value of pheromone matrices.
   c. Let $g = 1$, $h = 1$.

Step 2: Construct solutions for VRPCD:
   a. Solve the VRPCD based on eqs. (39) and (40).
   b. Apply the local pheromone updating rule (eq. (41)).
   c. $h = h + 1$.

Step 3: If $h < b$ (number of ants in ACO), go to Step 2. Otherwise, go to Step 4.

Step 4: Apply local search to improve all constructed solutions.

Step 5: Apply the global pheromone updating rule (eqs. (42) and (43)) based on the
global-best and the iteration-best solutions of VRPCD.

Step 6: Update the global best solution of VRPCD.

Step 7: If the maximum number of iterations ($\text{Iter}$) is met, stop. and output the global
best solution. Otherwise, $g = g + 1$, $h = 1$, and go to Step 2.

Step 8: Let $g = 1$, $h = 1$.

Step 9: Given the results of VRPCD, construct solutions for truck sequencing:
   a. Solve the truck sequencing based on eqs. (39) and (40).
   b. Apply the local pheromone updating rule (eq. (41)).
   c. $h = h + 1$.

Step 10: If $h < b$ (number of ants in ACO), go to Step 9. Otherwise, go to Step 11.
Step 11: Apply local search to improve all constructed solutions.
Step 12: Apply the global pheromone updating rule (eqs. (42) and (43)) based on the
global-best and the iteration-best solutions of truck sequencing.
Step 13: Update the global best solution of truck sequencing.
Step 14: If the maximum number of iterations (Iter) is met, stop, and output the global
best solution. Otherwise, $g = g + 1$, $h = 1$, and go to Step 9.

3.1 Vehicle Routing Phase

In this paper, an ant colony optimization is applied to solve the VRPTW for each depot.
The procedures of our ACO are described as follows

3.1.1 Route Construction Rule

In our ACO, when located at node $i$, ant $h$ moves to a node $j$ chosen by the following
state transition rule.

$$
S = \begin{cases} 
\arg \max_{j \in N_i} \left\{ \tau_{ij} \cdot \eta_{ij}^\beta \right\}, & q \leq q_0 \\
S, & q > q_0
\end{cases}
$$

(39)

$$
S : \quad P_{ij}^k (t) = \frac{\tau_{ij}^\beta \cdot \eta_{ij}^\beta}{\sum_{q \in N_i} \tau_{iq}^\beta \cdot \eta_{iq}^\beta}, \text{ if } j \in N_i
$$

(40)

where $N_i$ is the set of nodes which are not visited by ant $h$ at node $i$, $\tau_{ij}$ is the pheromone
of edge $(i, j)$, $\eta_{ij}$ is defined as the reciprocal of travel time of edge $(i, j)$.
$\beta$ is the parameter
that determines the relative effect of $\tau_{ij}$ versus $\eta_{ij}$ ($\beta > 0$), $q$ is a random variable
uniformly distributed in $[0, 1]$, and $q_0$ is a pre-defined parameter ($0 \leq q_0 \leq 1$). If $q \leq q_0$,
then the best depot $v$ for customer $i$ is determined according to eq. (39). On the contrary,
it is chosen according to $S$ which is a random variable selected according to the
probability distribution given in eq. (40). Hence, the parameter $q_0$ determines the relative
importance of exploitation eq. (39) versus exploration eq. (40).

3.1.2 Local Search

Local search is a time-consuming procedure of ACS. The analysis in Ting and Chen
(2013) showed that it is efficient for ACO to only apply local search to the best solution
among all solutions built at the current iteration. To save the computation time, only the
iteration-best solution is applied local search in this paper. In addition, three local search
methods are involved in our ACO, including 2-opt, swap and insertion. The local search
could be applied within route or between routes. This is because that diverse
neighborhood moves can expand the solution searching space. In 2-opt, two non-
consecutive arcs are removed, either in the same route or in two different routes, and the
two paths created are reconnected to restore feasibility. Two customers are exchanged in
swap. Insertion is to move one customer from its current position to another position, in
the same route or in a different route. In each iteration, we randomly implement only one
local search method. Thus, we assume that every approach has the same probability to be
selected for local search.

3.1.3 Pheromone Updating Rule

The pheromone updating of a typical ACO includes global and local updating rules. The
ants apply a local pheromone update rule immediately after they crossed an edge \((i, j)\)
during the tour construction. The local pheromone updating rule of our ACO is
\[
\tau_{ij}^{\text{new}} = \rho \cdot \tau_{ij}^{\text{old}} + (1 - \rho) \cdot \tau_0, \text{ if } \{\text{edge}(i, j) \in T_h\}
\]
(41)
where \(T_h\) denotes the routes constructed by ant \(h\), \(\rho\) is the pheromone decay parameter in
the range of \([0, 1]\) that regulates the reduction of pheromone on the edges. The \(\tau_0\) is the
initial value of the pheromone matrix for the route construction rule, and is set to be \(1/L_{nn}\),
where \(L_{nn}\) is the length of routes constructed by the Nearest Neighborhood heuristic.

In our ACO, the best elitist tours, including the global-best tour \((T_b)\) and the
iteration-best tour \((T_s)\) of VRPCD, are allowed to lay pheromone on the edges that belong
to them. The idea here is to balance between exploitation (through emphasizing the
global-best tour) as well as exploration (through the emphasis to the iteration-best tour).
The global updating rule of ACO for VRPCD is described as follow.
\[
\tau_{ij}^{\text{new}} = \rho \cdot \tau_{ij}^{\text{old}} + (1 - \rho) \cdot \Delta \tau_{ij}^e
\]
(42)
where
\[
\Delta \tau_{ij}^e = \begin{cases} [(L_w - L_b) + (L_w - L_s)]/L_w & \text{if } \{(i, v) \in T_b \text{ or } T_s\} \\ 0 & \text{otherwise} \end{cases}
\]
(43)
\(L_b\) and \(L_s\) denote the tour length of the global-best solution and the iteration-best solution
of VRPCD, respectively, and \(L_w\) is the tour length of the worst solution of the current
iteration.

3.2 Truck Sequencing Phase

3.2.1 Truck Sequencing Rule

The sequence is represented as a permutation for both inbound and outbound trucks,
respectively. Both inbound and outbound truck sequence use the same mechanism. We
will only describe the inbound truck sequence in details. For each ant, randomly pick the
first inbound truck to visit, say truck \(i\). Select the next inbound truck based on probability
computed as eqs. (39) and (40) to construct the sequence. However, \(N_i\) and \(\eta_{ij}\) are defined
differently from those for VRPCD. \(N_i\) in eq. (39) is defined here as the set of trucks that
are not assigned a sequence, while \(\eta_{ij}\) is defined as the reciprocal of total number of
products on truck $i$.

### 3.2.2 Local Search

Three local search methods are involved in our ACO for truck sequencing, including 2-opt, swap and insertion. In 2-opt, two non-consecutive sequence are removed, and the two subsequence created are reconnected to restore feasibility. The swap involves swapping the values of two randomly chosen unique positions in the sequence. The insertion involves moving the value in the second randomly chosen position to the position before the value of the first randomly chosen position.

#### 3.2.3 Pheromone Updating Rule

The pheromone updating rules for truck sequencing are the same as those for the VRPCD in eqs. (41)-(43). However, the edge $(i, j)$ represents the sequence that truck $j$ is scheduled immediately after truck $i$ in the sequence. The route is changed to the sequence for both inbound and outbound docks.

### 4 Computational Experiments

The proposed reduced variable neighborhood search algorithms were coded in Microsoft Visual Studio C++ 2010 and run on a PC with an Inter Core i5-2433 3.10GHz processor, 8.0GB of RAM and Windows 7 operating system. The instances tested are generated based on the flow pattern in Yu (2002) and Liao et al. (2013) for the truck sequencing problem, while the vehicle routing with cross-docking part are based on Lee et al. (2006). 15 instances are generated randomly. The proposed ACO algorithm was tested and compared with the solution given by Gurobi optimizer 5.1.0 with the running time of two hours. Each instance is run for 20 times.

In preliminary experiments we tried to find a good parameter setting for the proposed ACO algorithm/heuristic information combinations. We consider a set of parameters for the algorithm and then modifying one at a time, while keeping the others fixed. The parameters that were tested include: $\rho \in \{0.8, 0.85, 0.9, 0.95\}$, $q_0 \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$, $\beta \in \{0.2, 0.4, 0.6, 0.8, 1\}$, $b \in \{3, 5, 10, 15\}$, $\text{Iter} \in \{60, 90, 120, 150\}$ (maximum iterations of ACO). We found that for the parameter setting, $\alpha = 0.95$, $q_0 = 0.9$, $\beta = 0.8$, $b = 10$ and $\text{Iter} = 150$, can provide the best average solution.

Table 1 shows the characteristics of the 15 instances that we generated. Column 1 is the instance number. Columns 2–7 present the number of pickup nodes $A$, number of delivery nodes $B$, number of inbound trucks $K$, number of outbound trucks $L$, number of product types $G$, and truck capacity $Q$. The number of pickup nodes ranges from 4 to 50 while the number of delivery nodes ranges from 6 to 50. Column 8 and 9 are the constant material handling time between docks $H$ and truck changeover time $U$. In all instances, the loading and unloading times are assumed to be equal and to be one time unit.
Numbers of product $g$ for both pickup and delivery at node $i$ are generated by a uniform distribution in column 10 based on Yu (2002) and Liao et al. (2013). The travel time of each edge $(i, j)$ is generated by a uniform distribution as discussed in Lee et al. (2006) in column 11. The last two columns show the number of variables and number of constraint for each instance, respectively. It is noted that the instance becomes very complicated when the problem size increases. The integrated problem cannot be solved by the optimization software within reasonable computational time.

Table 1: Problem characteristics of the test instances.

<table>
<thead>
<tr>
<th>Inst.</th>
<th>$A$</th>
<th>$B$</th>
<th>$K$</th>
<th>$L$</th>
<th>$G$</th>
<th>$Q$</th>
<th>$H$</th>
<th>$U$</th>
<th>$a_i^g$, $b_i^g$</th>
<th>$t_g$</th>
<th># of Variables</th>
<th># of Constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>6</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>70</td>
<td>100</td>
<td>75</td>
<td>$U(0,13)$</td>
<td>$U(48,560)$</td>
<td>460</td>
<td>897</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td>8</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>80</td>
<td>100</td>
<td>75</td>
<td>$U(0,13)$</td>
<td>$U(48,560)$</td>
<td>796</td>
<td>1863</td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td>14</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>150</td>
<td>100</td>
<td>75</td>
<td>$U(0,13)$</td>
<td>$U(48,560)$</td>
<td>1176</td>
<td>2691</td>
</tr>
<tr>
<td>4</td>
<td>20</td>
<td>20</td>
<td>4</td>
<td>4</td>
<td>6</td>
<td>200</td>
<td>100</td>
<td>75</td>
<td>$U(0,13)$</td>
<td>$U(48,560)$</td>
<td>4737</td>
<td>20233</td>
</tr>
<tr>
<td>5</td>
<td>25</td>
<td>23</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>170</td>
<td>100</td>
<td>75</td>
<td>$U(0,13)$</td>
<td>$U(48,560)$</td>
<td>6775</td>
<td>21945</td>
</tr>
<tr>
<td>6</td>
<td>25</td>
<td>25</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>270</td>
<td>100</td>
<td>75</td>
<td>$U(0,13)$</td>
<td>$U(48,560)$</td>
<td>5782</td>
<td>22754</td>
</tr>
<tr>
<td>7</td>
<td>30</td>
<td>26</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>210</td>
<td>100</td>
<td>75</td>
<td>$U(0,13)$</td>
<td>$U(48,560)$</td>
<td>7841</td>
<td>26209</td>
</tr>
<tr>
<td>8</td>
<td>30</td>
<td>36</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>150</td>
<td>100</td>
<td>75</td>
<td>$U(0,13)$</td>
<td>$U(48,560)$</td>
<td>10265</td>
<td>27385</td>
</tr>
<tr>
<td>9</td>
<td>30</td>
<td>32</td>
<td>5</td>
<td>6</td>
<td>6</td>
<td>250</td>
<td>100</td>
<td>75</td>
<td>$U(0,13)$</td>
<td>$U(48,560)$</td>
<td>13806</td>
<td>66002</td>
</tr>
<tr>
<td>10</td>
<td>40</td>
<td>40</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>270</td>
<td>100</td>
<td>75</td>
<td>$U(0,13)$</td>
<td>$U(48,560)$</td>
<td>18676</td>
<td>65711</td>
</tr>
<tr>
<td>11</td>
<td>40</td>
<td>40</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>280</td>
<td>100</td>
<td>75</td>
<td>$U(0,13)$</td>
<td>$U(48,560)$</td>
<td>23533</td>
<td>117893</td>
</tr>
<tr>
<td>12</td>
<td>42</td>
<td>48</td>
<td>6</td>
<td>7</td>
<td>3</td>
<td>150</td>
<td>100</td>
<td>75</td>
<td>$U(0,13)$</td>
<td>$U(48,560)$</td>
<td>30010</td>
<td>82383</td>
</tr>
<tr>
<td>13</td>
<td>42</td>
<td>42</td>
<td>7</td>
<td>7</td>
<td>3</td>
<td>150</td>
<td>100</td>
<td>75</td>
<td>$U(0,13)$</td>
<td>$U(48,560)$</td>
<td>28043</td>
<td>76273</td>
</tr>
<tr>
<td>14</td>
<td>40</td>
<td>48</td>
<td>7</td>
<td>8</td>
<td>6</td>
<td>285</td>
<td>100</td>
<td>75</td>
<td>$U(0,13)$</td>
<td>$U(48,560)$</td>
<td>35660</td>
<td>181826</td>
</tr>
<tr>
<td>15</td>
<td>50</td>
<td>50</td>
<td>8</td>
<td>8</td>
<td>4</td>
<td>175</td>
<td>100</td>
<td>75</td>
<td>$U(0,13)$</td>
<td>$U(48,560)$</td>
<td>45409</td>
<td>163401</td>
</tr>
</tbody>
</table>

Table 2 present the minimum, maximum, average, standard deviation, and average CPU time in seconds for each instance over 20 runs. Among these 15 instances, optimization software Gurobi can only solve instance 1 within the 2 hour computation time limit. The best solution of instance 2 in 20 runs are the same, the lower bound of the Gubori is 4845 after 2 hours. We expect that the best solution might be the optimal solution of instance 2. For instances 3-15, we also compute the lower bound of the integrated problem by adopting Vahdani and Zandieh’s (2010) lower bound ($LB_s$) on the truck sequencing as shown in eq. (44) and the minimum VRPCD cost in 20 runs.

$$
LB_s = \begin{cases} 
\sum_{g \in G} \sum_{i \in A} a_i^g + H + (|L| - 1)U & \text{if } |K| \leq |L| \\
\sum_{g \in G} \sum_{i \in A} a_i^g + H + (|L| - 1)U + (|K| - |L|)U & \text{otherwise}
\end{cases}
$$
Comparing to the LB, our ACO can obtain the same results in 13 out of 15 instances. The average of the 15 instances is only 0.06%. It is noted that the lower bound of the VRPCD is based on the best solution of our ACO. Thus, the actual lower bound might be lower than that we obtain. In the future, we might have to apply different approach to obtain better solution in VRPCD.

Table 2: Results of the 15 test instances.

<table>
<thead>
<tr>
<th>Inst.</th>
<th>Min</th>
<th>Max</th>
<th>Avg.</th>
<th>Std. dev.</th>
<th>Time (s)</th>
<th>LB</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3394a</td>
<td>3394</td>
<td>3394.00</td>
<td>0.00</td>
<td>3.87</td>
<td>3394</td>
</tr>
<tr>
<td>2</td>
<td>4932</td>
<td>4932</td>
<td>4932.00</td>
<td>0.00</td>
<td>4.52</td>
<td>4932</td>
</tr>
<tr>
<td>3</td>
<td>4040</td>
<td>4454</td>
<td>4255.40</td>
<td>120.04</td>
<td>6.27</td>
<td>4040</td>
</tr>
<tr>
<td>4</td>
<td>7838</td>
<td>8221</td>
<td>8075.55</td>
<td>104.33</td>
<td>16.14</td>
<td>7751</td>
</tr>
<tr>
<td>5</td>
<td>7713</td>
<td>8337</td>
<td>8062.80</td>
<td>162.89</td>
<td>31.79</td>
<td>7713</td>
</tr>
<tr>
<td>6</td>
<td>7459</td>
<td>7957</td>
<td>7722.55</td>
<td>120.98</td>
<td>27.12</td>
<td>7459</td>
</tr>
<tr>
<td>7</td>
<td>8533</td>
<td>9410</td>
<td>9014.45</td>
<td>267.97</td>
<td>40.66</td>
<td>8533</td>
</tr>
<tr>
<td>8</td>
<td>9630</td>
<td>10262</td>
<td>9868.95</td>
<td>153.45</td>
<td>75.43</td>
<td>9630</td>
</tr>
<tr>
<td>9</td>
<td>11100</td>
<td>11569</td>
<td>11329.80</td>
<td>127.14</td>
<td>73.74</td>
<td>11097</td>
</tr>
<tr>
<td>10</td>
<td>10230</td>
<td>11062</td>
<td>10676.40</td>
<td>244.03</td>
<td>154.87</td>
<td>10230</td>
</tr>
<tr>
<td>11</td>
<td>12587</td>
<td>13173</td>
<td>12865.50</td>
<td>138.78</td>
<td>253.61</td>
<td>12587</td>
</tr>
<tr>
<td>12</td>
<td>12011</td>
<td>13208</td>
<td>12651.75</td>
<td>285.92</td>
<td>279.87</td>
<td>12011</td>
</tr>
<tr>
<td>13</td>
<td>12474</td>
<td>13458</td>
<td>13079.20</td>
<td>276.31</td>
<td>188.15</td>
<td>12474</td>
</tr>
<tr>
<td>14</td>
<td>14155</td>
<td>15004</td>
<td>14609.35</td>
<td>229.17</td>
<td>235.92</td>
<td>14155</td>
</tr>
<tr>
<td>15</td>
<td>16013</td>
<td>17078</td>
<td>16585.45</td>
<td>288.54</td>
<td>1981.56</td>
<td>16013</td>
</tr>
<tr>
<td>Avg.</td>
<td>9473.93</td>
<td>10101.27</td>
<td>9808.21</td>
<td>167.97</td>
<td>224.90</td>
<td>9467.93</td>
</tr>
</tbody>
</table>

*aGurobi obtains the optimal solution

5 Conclusion

In this research a new mixed integer programming model is proposed for the integrated problem, which combines vehicle routing and sequencing at cross-docking terminals. Since this problem encompasses the combination of two NP-Hard problems it cannot be solved in large instances in a polynomial computational time. An ACO algorithm was developed to solve large size instances of the problem. Our ACO solve two subproblems sequentially with two ant colonies. The results of vehicle routing problem were the input for the truck sequencing problem. To test out algorithm, 15 test instances were randomly generated based on the setting from the literature. The model is tested in Gurobi optimizer to find optimal solution for small instances. The effectiveness of the proposed
algorithm was tested for different size problems. The results show that our ACO could provide very good solutions. In the future, we would like to solve the integrated problem with an iterated approach to solve the integrated problem.

References


