INFORMATICA, 2019, Vol. 30, No. 3, 481–502

DOI: http://dx.doi.org/10.15388/Informatica.2019.215

Endosymbiotic Evolutionary Algorithm for an Integrated Model of the Vehicle Routing and Truck Scheduling Problem with a Cross-Docking System

Kun-Young LEE, Ji-Soo LIM, Sung-Seok KO*
Department of Industrial Engineering, Konkuk University,
120 Neungdong-ro, Gwangjin-gu, Seoul, South Korea
e-mail: ssko@konkuk.ac.kr

Received: March 2018; accepted: March 2019

Abstract. This paper presents a model which integrates inbound and outbound logistics with a cross-docking system. This model integrates the problem of routing inbound vehicles between suppliers and cross-docks and outbound vehicles between cross-docks and retailers, considering logistics costs and the products properties. This model aims to minimize the total cost by optimizing assignment of products to suppliers and retailers and operations of inbound and outbound vehicles. We developed an endosymbiotic evolutionary algorithm, which yields good performance in concurrent searches for the solutions of multiple subproblems and validate the performance using several numerical examples.

Key words: logistics, cross-docking, vehicle routing, truck scheduling, endosymbiotic evolutionary algorithm.

1. Introduction

Distribution operations reportedly account for 30% of product prices (Apte and Viswanthan, 2000). Hence, it is very important to optimize distribution networks so as to reduce logistics costs and realize a profitable supply chain management policy. Among various distribution activities, the storage of goods is expensive because of space requirements, inventory holding costs, and labour-intensive order picking tasks. Compared to traditional warehousing with its high cost functions, the cross-docking strategy can increase the product flow while reducing the storage space requirements, inventory holding costs, and delivery lead time (Boysen and Fliedner, 2010; Wen *et al.*, 2009). In addition, transportation costs can be decreased by using cross-docking owing to the economies of scale in transportation caused by consolidating different shipments (Apte and Viswanthan, 2000). For example, the reduction in order picking and storage costs resulting from the use of cross-docking decreased warehousing costs by up to 70% (Vahdani and Zandieh, 2010). Therefore, the cross-docking strategy has been used by many companies in different industries and is well known to be one of the most efficient distribution systems.

*Correspoding author.

Cross-docking systems are better suited to stable-demand products such as groceries or agricultural products; perishable bulk materials, including various chemical materials and food compounds requiring prompt shipment; and pharmaceutical materials that need to be transported in a cold-chain environment (Dondo and Cerdá, 2014). In addition, hazardous materials are usually transported via cross-docks to remedy sites for treatment and disposal. As a result, chemical companies such as Eastman Kodak reported that they successfully implemented cross-docking strategies to gain competitive advantages (Van Belle *et al.*, 2012).

Cross-docking is defined as continuous processing to the final destination through a cross-dock, without storing products and materials in a distribution centre (Apte and Viswanthan, 2000). A cross-dock is usually an I-shaped facility with strip and stack dock doors located on opposite sides of the facility and minimal storage space in between (Dondo and Cerdá, 2014). The cross-docking process includes three operations: receiving products from inbound vehicles at strip docks, consolidating the products into groups according to their destinations, and shipping them on outbound vehicles from stack docks. Both the pickup and delivery operations need to be considered to effectively apply cross-docking. Moreover, in addition to the pickup and delivery operations, vehicle routing and scheduling also need to be considered. Therefore, the physical flow in cross-dock operations can be improved by synthetic optimization of all operations, including pickup, cross-docking, and delivery. Moreover, to synthetically optimize the physical flow, the routes and schedules of vehicles need to be considered. Therefore, in this paper, the vehicle routing and truck scheduling problems are addressed along with cross-docking to improve material flow in the supply chain.

Boysen and Fliedner (2010) and Van Belle *et al.* (2012) reviewed cross-docking systems thoroughly, and Buijs *et al.* (2014) presented a research classification and framework for cross-docking network synchronization. Although cross-docking is rapidly becoming important in academia and industry, most studies on cross-docking have focused on the concept, physical design, cases in point, determination of optimal locations, and vehicle allocation. Sung and Song (2003) addressed a service network design problem for finding the optimal location of cross-docks and optimal allocation of vehicles. Jayaraman and Ross (2003) and Gumus and Bookbinder (2004) dealt with a distribution design problem that incorporates cross-docking into the supply chain by deciding whether to operate each cross-dock and the locations of open cross-docks.

The truck scheduling problem handles operational issues at the cross-dock, which include assigning vehicles to dock doors, determining the processing sequence of trucks at strip and stack doors, and transferring goods from inbound to outbound trucks. Tsui and Chang (1992) introduced a bilinear programming model to deal with the truck scheduling problem. Chen *et al.* (2006) studied the truck scheduling problem for a network of cross-docks considering delivery and pickup time windows and warehouse capacities. Lee *et al.* (2006) considered both cross-docking operations and truck scheduling; they assumed that all vehicles departing from suppliers arrive at the cross-dock simultaneously. Yu and Egbelu (2008) presented two approaches to scheduling trucks at the strip and stack docks. Kreng and Chen (2008) studied production–distribution planning for a traditional ware-

housing strategy versus a cross-docking strategy. Li *et al.* (2009) considered a situation where the number of vehicles is higher than the number of cross-dock doors.

For the deterministic truck scheduling problem, where the trucks in the pickup and delivery processes are the same, Wen *et al.* (2009) proposed a tabu search algorithm, Liao *et al.* (2010) developed a new tabu search algorithm, Santos *et al.* (2011, 2013) studied branch-and-price algorithms, and Morais *et al.* (2014) developed some heuristics algorithms. In addition, Alpan *et al.* (2011) studied schedule operations in a cross-dock where preemption and temporary storage are allowed to increase the operational flexibility. Dondo *et al.* (2011) formulated the truck scheduling problem using a mixed integer programming model where products are transported from manufacturers to customers using warehousing and/or cross-docking strategies. Miao *et al.* (2012) considered a multiple cross-dock transshipment problem with time windows. Dondo and Cerdá (2013) investigated a truck scheduling problem assuming an unlimited number of doors. For distribution networks under uncertainty, Mousavi *et al.* (2014) considered the location and the truck scheduling problem, proposing a hybrid fuzzy possibilistic-stochastic model.

The problem considered in this study extends these pickup and delivery problems for a single cross-dock, integrating cross-docking with vehicle route scheduling. Owing to the NP-hardness of this problem, even small-scale instances cannot be optimally solved within a reasonable amount of time. Hence, an endosymbiotic evolutionary algorithm (EEA) has been developed to solve the problem; EEA is known to perform well in concurrent searches for the solutions of multiple subproblems, especially when the original problem consists of multiple interconnected subproblems that need to be solved as a whole instead of solving each subproblem separately.

The rest of the paper is organized as follows. The description and mathematical formulation of the problem are presented in the next section. The proposed EEA for the problem is described in Section 3, and in Section 4, the algorithm performance is validated using several numerical examples in comparison with the mixed integer programming model as well as two genetic algorithms. Finally, the conclusions and future research directions are presented in the last section.

2. Problem Formulation

2.1. *Problem Description*

The integrated model of the cross-dock vehicle routing and truck scheduling problem (VRTSP) is defined as the problem of transporting a set of products from suppliers to customers (or retailers) by passing them through an intermediate cross-docking facility at the minimum transportation cost. Figure 1 illustrates how a cross-docking facility operates in VRTSP. A fleet of inbound vehicles picks up products from suppliers and is assigned to strip (receiving) doors in the cross-dock. Picked-up products are consolidated, transported to outbound vehicles assigned to stack (shipping) doors, and immediately delivered to customers without storage.

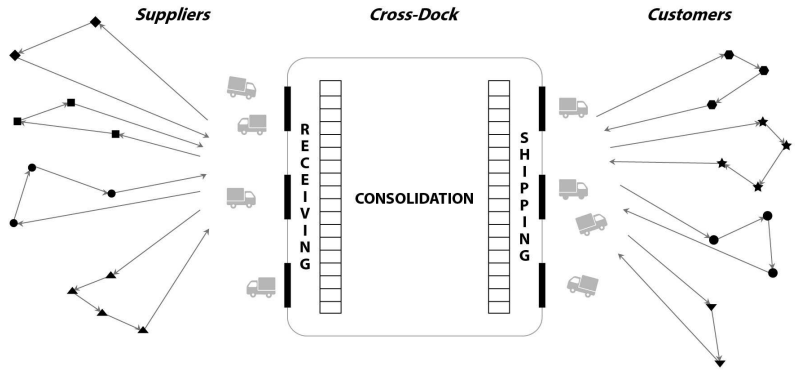

Fig. 1. VRTSP model.

Inbound and outbound vehicles operate exclusively; i.e. suppliers are served only by inbound vehicles, whereas customers are served only by outbound vehicles. Both inbound and outbound vehicles need to be operated efficiently to serve all suppliers and customers with a limited number of vehicles. This is a so-called vehicle routing problem. If each node must be served within a certain time period, the time windows can be considered in terms of an earliness and/or tardiness penalty cost.

When an inbound (outbound) truck arrives at (departs from) the cross-dock, it needs to be assigned to a strip door with the aim of increasing the cross-dock productivity and reducing the handling cost. A truck scheduling problem seeks to find the optimal assignment of inbound/outbound trucks to dock doors. In this study, we assume that the cross-dock has a limited number of strip doors and stack (shipping) doors; consequently, truck scheduling is an important problem because dock doors are scarce resources that need to be scheduled over time, and lines of trucks waiting for service can arise at every dock door.

As simultaneous treatment of both the vehicle routing problem and the truck scheduling problem is usually quite demanding, most studies have solved these integrated problems sequentially. However, in this paper we consider both the vehicle routing problem and the truck scheduling problem at the same time.

2.2. *Assumptions*

The basic characteristics of and assumptions applied to the proposed model are as follows:

- The sum of all demands is equal to the sum of all supplies for all product types; i.e. unloaded product types are the same as loaded product types, and holding inventory at the cross-dock is not allowed.
- The cross-dock yard is unlimited.
- All nodes excluding the cross-dock must be served by one vehicle.
- Service time (loading or unloading time) for each product type is fixed at each node.
- All vehicles have homogeneous capacity and fixed cost.
- An inbound vehicle cannot leave a strip door until its task is completed. Similarly, an outbound vehicle cannot leave a stack door until its task is completed.

- Inbound vehicles and outbound vehicles are operated separately. Therefore, inbound vehicles cannot visit customers, whereas outbound vehicles cannot visit suppliers.
- The vehicle changeover time is fixed.
- Tardiness or earliness is allowed with a separate penalty cost.

2.3. *Notation and the Proposed Model*

The following notations are used to formulate VRTSP:

Sets:

S – set of supplier (inbound) nodes,
C – set of customer (outbound) nodes,
$N = S \cup C \cup \{0\}$ – entire set of nodes, where 0 indicates the cross-dock,
$A_1 = \{(i, j)|i, j \in S \cup \{0\}, i \neq j\}$ – set of feasible arcs in the picking phase,
$A_2 = \{(i, j)|i, j \in C \cup \{0\}, i \neq j\}$ – set of feasible arcs in the delivery phase,
$A = A_1 \cup A_2$ – set of all feasible arcs in the network,
K_1 – set of inbound vehicles,
K_2 – set of outbound vehicles,
$K = K_1 \cup K_2$ – set of all vehicles,
D_1 – set of strip doors,
D_2 – set of stack doors,
$D = D_1 \cup D_2$ – set of dock doors,
G – set of product types.

Parameters:

$[a_i, b_i]$ – time window at node i,
c_{ij} – transportation cost from node i to node j,
CT – changeover time of vehicles at dock doors,
o – operation cost of vehicles,
Q – maximum load capacity of vehicles,
q_i^g – supply/demand of product type g at node i,
st_i^g – service time of product type g at node i,
tt_{ij} – travel time from node i to node j,
α_i^g – earliness penalty cost of product type g at node i,
β_i^g – tardiness penalty cost of product type g at node i,
K_i – maximum number of available inbound vehicles,
K_o – maximum number of available outbound vehicles.

Decision variables:

x_{ij}^k – 1 if vehicle k moves from node i to node j, otherwise 0,
X_d^k – 1 if vehicle k is assigned to strip door d, otherwise 0,
Y_d^k – 1 if vehicle k is assigned to stack door d, otherwise 0,
P_{ij} – 1 if inbound vehicles i and j are assigned to the same strip door and vehicle i is the predecessor of vehicle j, otherwise 0,

Q_{ij} – 1 if outbound vehicles i and j are assigned to the same stack door and vehicle i is the predecessor of vehicle j, otherwise 0,
v_{ij} – 1 if product units have to be transported from inbound vehicle i to outbound vehicle j, otherwise 0,
tf_{ij}^{g} – amount of product type g transported from inbound vehicle i to outbound vehicle j,
y_{ij} – total quantity of products from node i to node j,
AT_i – arrival time at node i,
e_i – earliness time at node i,
t_i – tardiness time at node i,
us_k – unloading start time of inbound vehicle k,
ue_k – unloading end time of inbound vehicle k,
at_k – available unloading start time of inbound vehicle k,
ls_k – loading start time of outbound vehicle k,
le_k – loading end time of outbound vehicle k,
dk_k – available departure time of outbound vehicle k.

The proposed VRTSP model is formulated as follows:

- Objective Function

$$\min \sum_{(i,j)\in A}\sum_{k\in K} c_{ij}x_{ij}^{k} + o\sum_{(0,j)\in A}\sum_{k\in K} x_{0j}^{k} + \sum_{i\in S\cup C} e_i \sum_{g\in G}\alpha_i^g q_i^g + \sum_{\in S\cup C} t_i \sum_{g\in G}\beta_i^g q_i^g; \quad (1)$$

- Constraints
  - Vehicle routing constraints:

$$\sum_{k\in K_1}\left(\sum_{\{p:(p,i)\in A_1\}} x_{pi}^{k} + \sum_{\{j:(i,j)\in A_1\}} x_{ij}^{k}\right) = 2, \quad \forall i \in S, \quad (2)$$

$$\sum_{k\in K_2}\left(\sum_{\{p:(p,i)\in A_2\}} x_{pi}^{k} + \sum_{\{j:(i,j)\in A_2\}} x_{ij}^{k}\right) = 2, \quad \forall i \in C, \quad (3)$$

$$\sum_{\{i:(i,p)\in A\}}\sum_{k\in K} x_{ip}^{k} - \sum_{\{j:(p,j)\in A\}}\sum_{k\in K} x_{pj}^{k} = 0, \quad \forall i \in S\cup C,\ \forall k \in K, \quad (4)$$

$$\sum_{j\in S}\sum_{k\in K_1} x_{0j}^{k} \leqslant K_i, \quad (5)$$

$$\sum_{j\in C}\sum_{k\in K_2} x_{0j}^{k} \leqslant K_o, \quad (6)$$

$$\sum_{\{k:(j,k)\in A\}} y_{jk} - \sum_{\{i:(i,j)\in A\}} y_{ij} = \begin{cases} \sum_{g\in G} q_i^g & \text{if } i \in S, \\ -\sum_{g\in G} q_i^g & \text{if } i \in C, \\ 0 & \text{otherwise} \end{cases} \quad \forall j \in S\cup C, \quad (7)$$

$$y_{ij} \leqslant Q, \quad \forall i, j \in S\cup C, \quad (8)$$

$$AT_j \geqslant AT_i + \sum_{g\in G} st_i^g q_i^g + tt_{ij} - M\left(1 - \sum_{k\in K} x_{ij}^k\right), \quad \forall i, j \in S \cup C, \tag{9}$$

$$AT_j \geqslant dt_k + tt_{0j} - M\left(1 - x_{0j}^k\right), \quad \forall j \in S \cup C,\ \forall k \in K, \tag{10}$$

$$at_k \geqslant AT_i + \sum_{g\in G} st_i^g q_i^g + tt_{i0} - M\left(1 - x_{i0}^k\right), \quad \forall i \in P \cup D,\ \forall k \in K, \tag{11}$$

$$e_i \geqslant a_i - AT_i, \quad \forall i \in S \cup C, \tag{12}$$

$$t_i \geqslant AT_i - b_i, \quad \forall i \in S \cup C; \tag{13}$$

– Truck scheduling constraints:

$$\sum_{d\in D_1} X_d^k = \sum_{j\in S} x_{0j}^k, \quad \forall k \in K_1, \tag{14}$$

$$\sum_{d\in D_2} X_d^k = \sum_{j\in C} x_{0j}^k, \quad \forall k \in K_2, \tag{15}$$

$$X_d^i + X_d^j - 1 \leqslant P_{ij} + P_{ji}, \quad \forall i, j \in K_1,\ \forall d \in D_1, \tag{16}$$

$$P_{ij} + P_{ji} \leqslant 1, \quad \forall i, j \in K_1, \tag{17}$$

$$Y_d^i + Y_d^j - 1 \leqslant Q_{ij} + Q_{ji}, \quad \forall i, j \in K_2,\ \forall d \in D_2, \tag{18}$$

$$Q_{ij} + Q_{ji} \leqslant 1, \quad \forall i, j \in K_2, \tag{19}$$

$$us_i \geqslant at_i, \quad \forall i \in K_1, \tag{20}$$

$$us_j \geqslant ue_i + CT - M(1 - P_{ij}), \quad \forall i, j \in K_1, \tag{21}$$

$$ue_i \geqslant us_i + \sum_{g\in G} st_0^g \sum_{j\in K_2} tf_{ij}^g), \quad \forall i \in K_1, \tag{22}$$

$$ls_j \geqslant le_i + CT - M(1 - Q_{ij}), \quad \forall i, j \in K_2, \tag{23}$$

$$ls_j \geqslant ue_i - M(1 - v_{ij}), \quad \forall i \in K_1,\ \forall j \in K_2, \tag{24}$$

$$le_j \geqslant ls_j + \sum_{g\in G} st_0^g \sum_{i\in K_1} tf_{ij}^g, \quad \forall j \in K_2, \tag{25}$$

$$dt_k \geqslant le_k, \quad \forall k \in K_2, \tag{26}$$

$$\sum_{g\in G} tf_{ij}^g \geqslant 1 - M(1 - v_{ij}), \quad \forall i \in K_1,\ \forall j \in K_2, \tag{27}$$

$$\sum_{g\in G} tf_{ij}^g \leqslant M v_{ij}, \quad \forall i \in K_1,\ \forall j \in K_2; \tag{28}$$

– Product transshipment constraints:

$$\sum_{k_2\in K_2} tf_{kk_2}^g = \sum_{i\in S} q_i^g \sum_{\{j:(i,j)\in A_1\}} x_{ij}^k, \quad \forall k \in K_1,\ \forall g \in G, \tag{29}$$

$$\sum_{k_1 \in K_1} tf^g_{k_1 k} = \sum_{i \in C} q^g_i \sum_{\{j:(i,j) \in A_2\}} x^k_{ij}, \quad \forall k \in K_2, \ \forall g \in G. \tag{30}$$

Objective (1) minimizes the total supply chain cost, which has four components: total transportation cost, total vehicle operation cost, total earliness penalty cost, and total tardiness penalty cost. We note that material handling cost in the cross-dock yard is ignored by assumption.

Constraints can be divided into three categories. The first type of constraints are related to the vehicle routing problem (constraints (2)–(13)). Constraints (2) and (3) ensure that nodes excluding the cross-dock are serviced once by one vehicle. Route continuity is guaranteed by constraint (4). Constraint (5) determines the number of inbound vehicles to use for pickup routes; similarly, constraint (6) requires that all delivery routes are made with available outbound vehicles. The total quantity transported among nodes is expressed by constraint (7). Constraint (8) restricts the load quantity on all routes, so a vehicle cannot ship more than its maximum capacity. Constraint (9) computes the arrival times at each node as the maximum of the sum of the arrival time, total service time, and travel time from other nodes. Constraint (10) states that the arrival time of vehicles is greater than the arrival time at the first node from the cross-dock. Constraint (11) ensures that the return time of vehicles to the cross-dock is greater than the maximum of the sum of the arrival time, total service time, and travel time. Earliness and tardiness are computed by constraints (12) and (13), respectively.

The second type is related to the vehicle scheduling problem (constraints (14)–(25)). Constraint (14) ensures that each inbound vehicle is assigned to a strip door if it is used for pickup routes. Similarly, constraint (15) ensures that each outbound vehicle is assigned to a stack door if it is used for delivery routes. Constraints (16) and (17) represent the precedence constraints in a strip door when two vehicles are assigned to the same strip door. These constraints are used to compute the unloading time of each vehicle. Constraints (20) and (21) ensure that the unloading start time of each inbound vehicle at the cross-dock is the maximum of the arrival time of the vehicle and the end time of its predecessor plus the vehicle changeover time. Constraint (22) describes the unloading end time of each inbound vehicle. Based on the assumption that a vehicle cannot leave the door before its task is completed, the unloading end time is equal to its start time plus the time required to unload all products transported to outbound trucks. Similarly, constraints (18) and (19) represent the precedent constraints in a stack door. Constraints (23) and (24) determine the loading start time of an outbound vehicle according to its predecessor and the inbound vehicles that transport products to the outbound vehicle. This can be expressed as the larger time among the maximum of the loading end time of the predecessor and the maximum of the unloading end times of the inbound vehicles that transport products to the outbound vehicle. The loading end time of an outbound vehicle can be computed as the sum of its start time and the time required to load all products transported by inbound vehicles, which is enforced by constraint (25). An outbound vehicle can depart from the cross-dock after its loading task is completed; thus, the departure time of an outbound vehicle is equal to its loading end time, which is expressed by constraint (26). Constraints (27) and (28) represent the relationship between v_{ij} and the amount of product transported from inbound vehicle i to outbound vehicle j.

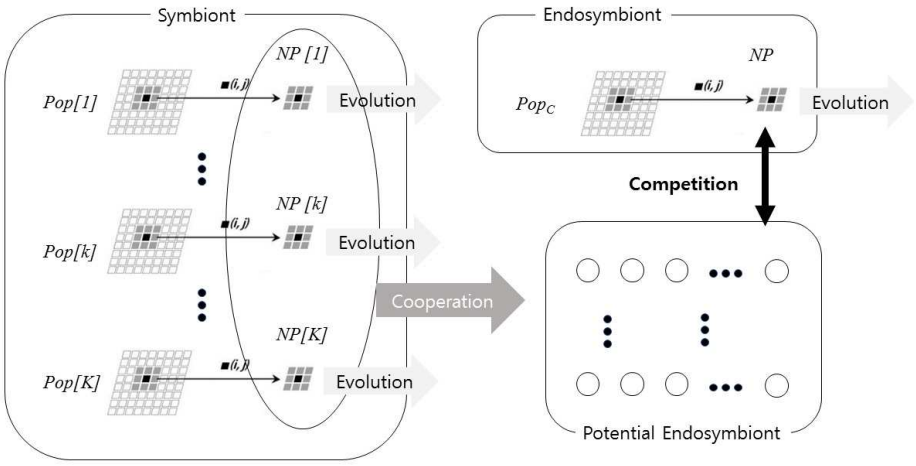

Fig. 2. Endosymbiotic evolutionary algorithm.

Product transshipment between inbound vehicles and outbound vehicles is expressed by constraints (29) and (30). Constraint (29) ensures that the total quantity of product types transported from an inbound vehicle to outbound vehicles is equal to its pickup quantity. Constraint (30) ensures that the total quantity of product types transported from inbound vehicles to an outbound vehicle is equal to its delivery quantity.

3. Proposed Endosymbiotic Evolutionary Algorithm

3.1. *Endosymbiotic Evolutionary Algorithm*

The VRTSP under study is a well-known NP-hard problem. Therefore, we propose an EEA-based method to obtain a good approximation solution within a reasonable time. EEA proposed by Kim *et al.* (2001) is a type of symbiotic evolutionary algorithm that imitates the natural process of endosymbiotic evolution (Margulis, 1980), in which prokaryotes with relatively simple structure enter a larger host prokaryote, where they live together in symbiosis and evolve into a eukaryote. Since the advent of EEA, this algorithm has been applied to combinatorial optimization problems and has been proven to be very effective at solving them. Owing to the intrinsic properties of EEA, it can be a good candidate to search for the solutions of multiple subproblems concurrently, especially when the original problem consists of multiple interconnected subproblems that are solved as a whole instead of being considered separately.

Figure 2 gives an overview of EEA under consideration. In EEA, the original problem is split into several subproblems, and each subproblem has its own population ($Pop[k]$) consisting of symbionts representing partial solutions to the entire problem. To construct a complete solution to the entire problem, all the partial solutions, one from each of the populations, are combined. Each population evolves while cooperating with corresponding symbionts from other populations so as to find better solutions to the original problem.

However, because individuals in each population of subproblems are only partial solutions, we can evaluate the solution only when the corresponding symbionts are combined appropriately into endosymbionts. In addition, the population (Pop_C) of the combined problem in this algorithm consists of the endosymbionts, which carry the genes of all the symbionts (partial solutions). Hence, individuals in the population of the combined problem represent solutions to the original problem. Individuals in the population of the original problem compete with new offspring generated by combining individuals from each population of subproblems. However, the population of the original problem also evolves to find a better solution to the original problem.

Each population forms a two-dimensional structure consisting of a toroidal grid with the same number of individuals, $n \times n$, and individuals in the population are mapped onto the cells of the grid. Let (i, j), $i, j = 1, \ldots, n$ be the location index of the $n \times n$ toroidal grids. Then $NP[k]_{ij}$ is defined as the neighbourhood of individual (i, j) in $Pop[k]$ and forms a 3×3 grid with (i, j) in the centre of this structure and eight neighbouring individuals. Only the individuals in these neighbourhoods are considered for the interactions among these populations in each generation. The neighbourhoods of selected individuals in each population cooperate to find a good solution of the problem. Here, because each neighbourhood contains nine individuals, 9^K (where K is the number of subproblems) combinations are considered as candidate solutions of the original problem. The best combination among them is compared with the current best solution in the algorithm, which competes with nine individuals in NP from Pop_C as well. On the basis of the interactions among these neighbourhoods, a parallel search for partial solutions of the subproblems from each population and an integrated search for complete solutions from the population of the combined problem are conducted simultaneously in all generations.

The overall EEA procedure is as follows. In the procedure, b_k represents an individual in $Pop[k]$, and $a_1 a_2 \cdots a_k \cdots a_K$ is an individual (endosymbiont) in the population of endosymbionts.

Step 1: Initialization

- For each cell in $Pop[k]$ $(k = 1, 2, \ldots, K)$ and Pop_C, generate n^2 individuals randomly.
- Set $f_{best} = \infty$.

Step 2: Neighbourhood construction

- Select an arbitrary location (i, j) and set up the neighbourhoods, $NP[k]$ $(k = 1, 2, \ldots, K)$, and NP in each population.

Step 3: Competition between symbiont and endosymbiont

- If there exists $b_k \in NP[k]$ such that for each $a_1 a_2 \cdots a_k \cdots a_K \in NP$, $f(a_1 a_2 \cdots b_k \cdots a_K)$ is better than $f(a_1 a_2 \cdots a_k \cdots a_K)$, then substitute $a_1 a_2 \cdots b_k \cdots a_K$ for $a_1 a_2 \cdots a_k \cdots a_K \in NP$ and replace b_k with a_k in $NP[k]$.

Step 4: Generation of potential endosymbionts

- Evaluate the fitness of individuals in $NP[k]$ $(k = 1, 2, \ldots, K)$, where symbiotic partners are selected from the neighbourhood of the corresponding other populations.
- Let E_{new} be a potential endosymbiont that is the best combination of symbionts. If $f(E_{new})$ is better than f_{best}, then update f_{best}.

Step 5: Competition between an endosymbiont and a potential endosymbiont

- Compare $f(E_{new})$ to the fitness of NP_w, which is the worst individuals in NP.
- If $f(E_{new})$ is better than $f(NP_w)$, then replace NP_w with E_{new}. The replaced NP_w is separated and moved into the position of component symbionts in the neighbourhood to which it belongs.

Step 6: Evolution

- Call *Genetic Evolution*$(NP[k], R_c, R_m)$ for $k = 1, 2, \ldots, K$.
- Call *Genetic Evolution*(NP, R_c, R_m).
- Release the evolved neighbourhoods, $NP[k]$ $(k = 1, 2, \ldots, K)$, and NP to the populations $Pop[k]$ $(k = 1, 2, \ldots, K)$ and Pop_C.

Step 7: Termination criteria of evolution

- If one of the following conditions is satisfied: (i) the maximum number of iterations is reached, or (ii) the solution is not improved for a fixed number of consecutive iterations, then stop.
- Otherwise, go to Step 2.

The overall procedure of the general genetic algorithm used in Step 6 is as follows. The binary tournament method is applied as a selection mechanism method in this algorithm.

**Genetic Evolution*(P, R_c, R_m)
Substep 1: Selection and reproduction

- Select individuals from P using the binary tournament method.
- Generate a random number for each selected individual. If the random number is smaller than R_c(crossover rate), the individual becomes a candidate for crossover operation.
- Produce two offspring (individuals) by applying a crossover operator to a pair of candidate individuals.

Substep 2: Replacement

- Select individuals for replacement from P, where a high probability is given to those having high fitness.
- Replace the selected individuals with the newly created offspring (individuals).

Substep 3: Mutation

- Generate a random number for each individual from the newly generated P. If the random number is smaller than or equal to R_m(mutation rate), apply mutation operators to the individuals.

3.2. *Genetic Representations for VRTSP*

Eq. (1) is used as a fitness function ($f(\cdot)$) to compute the total supply chain cost. And then EEA for VRTSP under consideration breaks down the original problem into four different subproblems and also considers the original problem itself as the combination of the four subproblems. The first subproblem is the inbound vehicle routing problem ($Pop[1]$), and the second is the outbound vehicle routing problem ($Pop[2]$). The inbound vehicle routing problem represents a pickup route, whereas outbound vehicle routing problem represents a delivery route. The third one is the vehicle sequencing problem ($Pop[3]$) for outbound vehicles determined in cross-docking. The last subproblem is the product transshipment problem ($Pop[4]$) to assign product units from inbound trucks to outbound trucks. Then the combined problem (Pop_C) constitutes the VRTSP. Separate populations are maintained for these four subproblems and the combined problem. The four populations of the subproblems correspond to the symbionts in the endosymbiotic theory, and individuals in those populations represent partial solutions of the original problem.

When the populations evolve, EEA follows the steps of the steady-state genetic algorithm: selection, reproduction, replacement, and mutation. The crossover operators used for reproduction not only transmit good genes from parents to offspring, but should also ensure that new offspring can meet the problem constraints. Mutation is necessary in evolutionary algorithms to escape local optima and consistently search for the global optimum; thus, appropriate mutation operators must be introduced. In this study, we applied two crossover operators, one for the vehicle routing population and one for the other populations, and three mutation operators are considered for all populations.

3.2.1. *Genetic Components for Vehicle Routing*

For the vehicle routing problem, a genetic representation based on permutation is generally used. In this study, the two-dimensional representation proposed by Pereira *et al.* (2002) is employed for the inbound and outbound vehicle routing problems. It is more intuitive than the one-dimensional representation using a splitter. In this representation, the genetic material of the chromosome is broken down into several routes, each of which consists of an ordered subset of suppliers or customers. Figure 3 gives an example of chromosomes for an inbound vehicle route with 10 supplier nodes and 3 inbound vehicles. Because all the vehicles must depart from and arrive at the cross-dock, the vehicle routes in the example are interpreted as {0-8-2-3-0}, {0-1-7-6-9-5-0}, and {0-10-4-0}.

Because each vehicle has limited capacity, each route cannot include suppliers whose summed supply exceeds the corresponding vehicle's capacity. In the outbound vehicle routing population, the genetic representation of the chromosome matches that of the chromosome in the inbound vehicle routing population. The genetic material of individuals in the outbound vehicle routing population is composed of customers, whereas the genetic material of individuals in the inbound vehicle routing population is composed of suppliers.

Individuals in vehicle routing populations should obey the constraints imposed by the number of vehicles and vehicle capacity. For reproduction, we apply the *Best-Cost Route Crossover* (BCRC) operator proposed by Ombuki *et al.* (2006). BCRC minimizes not only

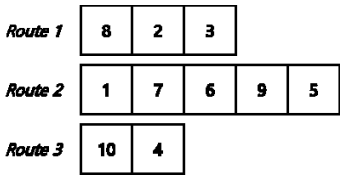

Fig. 3. Examples of pickup route representations.

the number of vehicles but also the cost while checking the feasibility constraints. BCRC is applied as follows.

1. Select a route from two parents, P_1 and P_2.
2. Delete the nodes in the selected route from the opposite parent.
3. For P_1 and P_2, array the deleted nodes in a random order.
4. Find feasible positions at which to insert deleted nodes.
5. Insert the deleted nodes at the position among the feasible positions that minimizes the total cost.
6. Select the offspring from among the newly reproduced individuals by comparing their fitness.

Figure 4 illustrates how BCRC works. Individuals in vehicle routing populations represent subsolutions, not complete solutions. Therefore, the best position in Step 4 of EEA procedure described in Section 3.1 is the position that minimizes the sum of the total transportation cost and vehicle operation cost. BCRC usually aims to generate two offsprings. In EEA, however, the steady-state genetic algorithm requires only one offspring. Therefore, Step 4 is applied to choose the better offspring by comparing their fitness. For the mutation process, we adopt three mutation operators: sweep, Insertion, and Inversion. Mutation is accepted only under feasible conditions.

3.2.2. *Genetic Components for Vehicle Sequencing and Product Transshipment*
The vehicle scheduling problem determines the dock assignment and the sequence of inbound and outbound vehicles. Dock assignment of inbound vehicles is performed using First-Come First-Serve in most experiments in existing literature. Hence, only the scheduling of outbound vehicles needs to be considered. An outbound vehicle scheduling solution can be represented by a sequence of outbound vehicles obtained using a dispatching rule. The dispatching rule assigns outbound vehicles to the doors at which they can start their task as soon as possible. Figure 5 gives an example of the genetic representation of an outbound vehicle sequence. Suppose that only stack doors 1 and 2 are available. Vehicle 8 is assigned to stack door 1. Vehicle 7 can be assigned to stack door 2 or 1 depending on whether vehicle 8 is working on its task or not. Vehicles 6, 10, and 9 are assigned following the same procedure.

The loading start time of outbound vehicles depends on the unloading end time of inbound vehicles. Therefore, product transshipment is employed to link outbound vehicles to inbound vehicles. The relationship between inbound vehicles and outbound vehicles

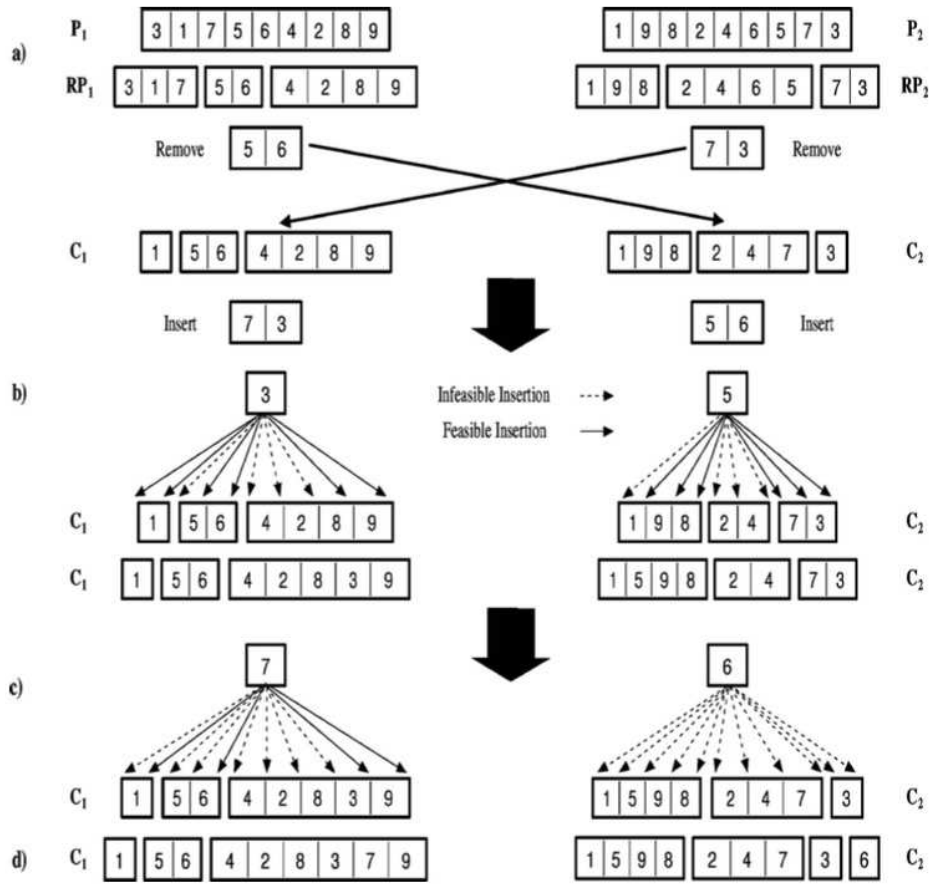

Fig. 4. Example of a BCRC operator.

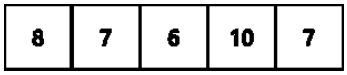

Fig. 5. Example of outbound vehicle sequence representation.

determines the loading start time of outbound vehicles. That is, the loading start time of an outbound vehicle also depends on the maximum of the unloading end time of inbound vehicles that transport products to the outbound vehicle. Therefore, we know that the genetic representation of product transshipment is the same as that of the outbound vehicle sequence.

In this study, we apply Partially Mapped Crossover (PMX), which is the most widely used crossover operator for chromosomes having permutation encoding for reproduction. It was proposed by Goldberg and Lingle (1985) for the traveling salesman problem. In PMX, two chromosomes are aligned, and two crossover points are selected randomly. The two crossover points give a matching selection, called a swab, which is used to perform

cross-through position-by-position exchange operations. The steps in PMX process are as follows:

1. Copy the swap from the first parent (P_1) into the first offspring.
2. Starting from the first crossover point, look for elements in the swap of the second parent (P_2) that have not been copied.
3. For each uncopied element (say i), look in the first offspring to find the elements that have been copied from P_1 (say j).
4. Place i in the position occupied by j in P_2.
5. If the place occupied by j in P_2 is already filled in the offspring by k, then put i in the position occupied by k in P_2.
6. Fill all the positions of the first offspring similarly for all elements in P_2.
7. Repeat this set of steps for the second offspring, reversing the order of parents.

4. Computational Results

In this section, we evaluate the effectiveness of the proposed EEA by comparing with a monolithic method (Dondo and Cerdá, 2014) and other symbiotic evolutionary algorithms.

4.1. *Test Problems and Parameter Settings*

Because no benchmark dataset is available for this study, we generate 20 test problem instances to assess the performance of the algorithm. The crucial parameters of each instance are shown in Table 1, and the locations of all the nodes are generated randomly.

The proposed EEA has been coded in the Java programming language. The binary tournament method is employed as a selection mechanism. The parameters in EEA are determined by preliminary experiments. A 100-cell (10×10) toroidal grid is used for $Pop[k]$, $k = 1, 2, 3, 4$, and Pop_C. The crossover rate and mutation rate are set to 0.7 and 0.01, respectively. To determine the crossover rate (R_c) and mutation rate (R_m), an experimental design has been developed at different levels. We tested crossover rates in the range of [0.5, 0.9] with an increment of 0.1 and mutation rates in the range of [0.01, 0.05] with an increment of 0.01. Finally, we selected the pair, $R_c = 0.7$ and $R_m = 0.01$, that yields the highest performance in case of 'Instance 20'. When the current number of generations reaches 5,000, the proposed algorithm has been terminated. And, we solved 40 times for each problem instance shown in Table 1 and obtained serveral key measures such as mean and standard deviation of objective value, and computation times.

4.2. *Performance Analysis*

First, we compared EEA with the monolithic approach to handle the NP-hardness of VRTSP proposed by Dondo and Cerdá (2014) as well as with the mixed integer programming (MIP) model in Section 2. The main idea of the monolithic approach is to assign

Table 1
Problem instances.

Instance	Suppliers	Customers	Product types	Inbound vehicles	Outbound vehicles	Q	Strip doors	Stack doors
Instance 01	5	5	1	3	3	13	1	1
Instance 02	5	5	2	3	3	30	1	1
Instance 03	4	6	4	3	3	35	1	1
Instance 04	4	6	4	3	3	30	1	1
Instance 05	8	12	1	3	3	30	1	1
Instance 06	8	12	2	3	3	50	1	1
Instance 07	10	10	2	3	3	60	1	1
Instance 08	10	10	3	3	3	60	1	1
Instance 09	10	10	3	3	3	60	1	1
Instance 10	8	12	4	3	3	70	1	1
Instance 11	20	30	1	6	7	50	2	2
Instance 12	20	30	1	6	7	50	2	2
Instance 13	25	25	2	6	6	60	2	2
Instance 14	25	25	2	6	6	60	2	2
Instance 15	25	25	3	6	6	70	2	2
Instance 16	25	25	4	6	6	75	2	2
Instance 17	40	60	1	12	12	70	2	2
Instance 18	50	50	2	12	12	80	2	2
Instance 19	50	50	2	12	12	70	2	2
Instance 20	50	50	4	12	12	90	3	3

Table 2
Computational results from EEA, monolithic approach, and MIP model.

Instance	EEA		Monolithic approach		MIP model	
	Objective (mean)	Run-time (s)	Objective (mean)	Run-time (s)	Objective (mean)	Run-time (s)
Instance 01	2125.92	2.16	2138.40	52.73	2125.92	47.16
Instance 02	3167.91	2.24	3452.45	28.32	3167.91	47.16
Instance 03	2712.35	3.19	3515.15	32.56	2712.35	52.77
Instance 04	3680.49	3.04	3605.62	12.33	3594.65	78.02
Instance 05	4116.70	4.04	4797.14	622.16	4483.69	3600.00
Instance 06	4957.08	4.63	4432.60	3172.53	4982.33	3600.00
Instance 07	3825.29	4.46	5376.53	1098.67	5188.81	3600.00
Instance 08	19023.91	4.31	19678.07	2039.78	22827.99	3600.00
Instance 09	22449.95	4.44	24141.73	3600.00	22885.03	3600.00
Instance 10	30922.97	4.30	31908.43	924.25	37546.22	3600.00

vehicles to nodes and set the driving direction by mimicking the sweep algorithm. This approach has a potential to find a reasonably good solution within a shorter computational time, compared to exact algorithms for the optimal solutionl. The monolithic approach and the MIP model have been implemented on Cplex Optimizer version 12.6.2, and we set the time limit to 1 h (3,600 s). All the experiments in this study have been conducted on a computer with 2.20 GHz CPU and 8.0 GB RAM.

Table 2 summarizes the results of the experiments for all problem instances. But For the monolithic approach and MIP model, Cplex Optimizer could not find any feasible solution

for instance 11–20 with more than 50 nodes (Suppliers + Customers) in the preset time limit (3,600 s). Table 2 shows that the monolithic approach shows reasonable results only in some environments, because of the basic assumption of the monolithic approach that all vehicles must travel counterclockwise and a node must form a route with nodes located at positions counterclockwise from itself. In addition, MIP modelling does not seem to be a good approach to large scale problems.

Unlike other approaches, EEA gives good solutions for all problem instances in the shortest computational time. For small instances, both EEA and Cplex Optimizer found an optimal solution for the MIP model. From the results for these test instances, we can see that while the MIP model approach using Cplex Optimizer does not guarantee an optimal solution in large-scale problems within the preset time limit, EEA gives good solutions within a reasonable computation time. Moreover, the computation time in EEA increases not exponentially but as a quadratic function of the number of nodes (suppliers and customers), which demonstrates the usefulness of our proposed algorithm.

We also compared the proposed algorithm to other genetic approaches: SPA (Separated and Population-based Algorithm) and SNA (Separated and Neighbourhood-based Algorithm). SPA and SNA are symbiotic evolutionary algorithms that divide the problem into subproblems. SPA has been used as the core algorithm in most existing symbiotic evolutionary algorithms (Maher and Poon, 2009). In SPA, a solution is divided into more than one type of partial solution. Individuals representing each type of partial solution are maintained in a population, and they can evolve with the population to which they belong. Although the SNA is similar to SPA, they differ in that SNA uses neighbourhood-based co-evolution to maintain population diversity, whereas SPA uses population-based co-evolution (Kim *et al.*, 2000).

Under same parament setting ($R_c = 0.7$ and $R_m = 0.01$), experimental results are summarized in Table 3 in which the average and the standard deviations of the solutions for each experimental condition are reported. As shown in Table 3, for every problem instance, EEA showed the best results and SNA produced the next-to-best ones. Hence the last column (Gap) indicates the improvement rate, which is computed as

$$\text{Improvement Rate} = \frac{|\text{Mean of EEA} - \text{Mean of SNA}|}{\text{Mean of SNA}} \times 100\%.$$

The main difference between EEA and SNA is the existence of endosymbionts. EEA allows the formation and evolution of endosymbionts that are a combination of good individuals obtained while evolving the populations of the subproblems and combined problem. Hence, we can infer that endosymbionts enhance the performance of the proposed algorithm, although a longer computation time is inevitable. Even though EEA computation time is reasonably short, it is still 3.5 times longer than that of SNA in average computation time for all instances. Therefore SNA may be used for a quick solution while EEA may be used for a better and more exact solution.

For detailed analysis, we compared convergence pattern of objective value of *instance 20* in terms of the number of generation as shown in Fig. 6. While objective values of SNA decrease smoothly without sudden change in trend, those of EEA decrease smoothly

Table 3
Computational results from EEA, SPA, and SNA.

Instance	SPA		SNA		EEA		Gap
	Mean	S.D.	Mean	S.D.	Mean	S.D.	
Instance 01	2125.92	–	2125.92	–	2125.92	–	0
Instance 02	3167.91	–	3167.91	–	3167.91	–	0
Instance 03	2712.35	–	2712.35	–	2712.35	–	0
Instance 04	3680.49	–	3691.30	40.76	3680.49	–	0%
Instance 05	6296.91	444.93	4429.23	341.30	4116.70	224.56	7%
Instance 06	5781.59	302.40	5762.38	521.41	4957.08	683.10	14%
Instance 07	7645.23	644.09	4842.14	680.19	3825.29	408.37	21%
Instance 08	24651.91	1067.31	20031.98	1084.47	19023.91	759.04	5%
Instance 09	27943.59	1162.44	23802.30	1458.23	22449.95	1127.05	6%
Instance 10	38701.39	1578.90	34795.58	2720.43	30922.97	2432.34	11%
Instance 11	19215.70	1426.24	6435.90	742.82	4713.22	375.39	27%
Instance 12	20091.32	1038.84	7308.07	778.87	5259.30	419.87	28%
Instance 13	47341.58	2310.59	19392.46	2559.45	11971.18	1843.43	38%
Instance 14	60928.81	2832.51	27007.88	3037.31	18134.63	3715.49	33%
Instance 15	39016.55	1592.40	28502.67	2189.54	18128.09	937.56	36%
Instance 16	87548.26	3889.85	65141.56	6057.46	44744.94	3899.56	31%
Instance 17	331027.02	10240.01	207381.35	18003.46	95955.32	10741.19	54%
Instance 18	249954.00	12494.57	127543.23	15833.90	37362.67	9148.41	71%
Instance 19	312035.33	9865.61	223908.94	13556.05	170200.87	10126.49	24%
Instance 20	516568.92	18478.62	312789.88	27946.20	154071.06	23627.00	51%

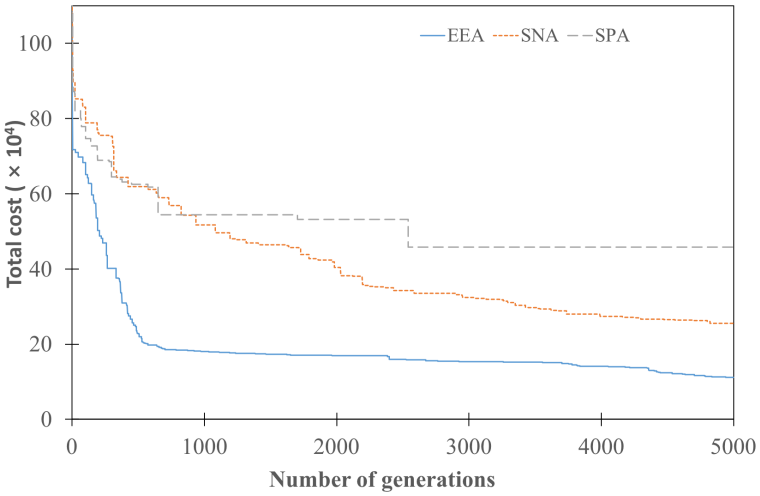

Fig. 6. Sample convergence rate for instance 20.

after high drop improvement in initial generation period. This shows that in a smaller generation, the EEA can produce a better solution than the SNA and we can also find that similar phenomena occur in other instance problems.

We also compared the result of EEA with that of SNA under the same run-time for each instance. These run-times are the average computation time to solve each instance

Table 4
Comparison of SNA and EEA under the same run-time.

Instance	SNA		EEA		Run-time	Gap
	Mean	S.D.	Mean	S.D.		
Instance 01	2125.92	–	2125.92	–		0%
Instance 02	3167.91	–	3167.91	–		0%
Instance 03	2712.35	–	2712.35	–		0%
Instance 04	3691.30	40.76	3680.49	–		0%
Instance 05	4429.23	341.30	4285.00	309.57	3.02	3%
Instance 06	5762.38	521.41	4987.18	672.62	2.83	13%
Instance 07	4842.14	680.19	4201.37	623.19	1.63	13%
Instance 08	20031.98	1084.47	19477.59	1003.17	1.61	3%
Instance 09	23802.30	1458.23	22957.74	1450.34	1.64	4%
Instance 10	34795.58	2720.43	31883.37	2758.46	1.65	8%
Instance 11	6435.90	742.82	4822.21	409.37	3.83	25%
Instance 12	7308.07	778.87	5399.30	374.17	3.80	26%
Instance 13	19392.46	2559.45	13991.27	3090.22	3.65	28%
Instance 14	27007.88	3037.31	20799.50	3664.06	3.67	23%
Instance 15	28502.67	2189.54	18479.78	1048.65	7.00	35%
Instance 16	65141.56	6057.46	45881.88	4999.99	7.06	30%
Instance 17	207381.35	18003.46	109947.28	13523.93	19.15	47%
Instance 18	127543.23	15833.90	51688.50	11049.59	19.10	59%
Instance 19	223908.94	13556.05	188672.99	12146.38	20.74	16%
Instance 20	312789.88	27946.20	185855.84	28857.88	19.84	41%

by SNA with 5,000 generations. Table 4 shows the comparison results. We can see that EEA outperforms SNA for all instances under the same run-time but improvement rate is slightly decreased compared with the values in Table 3. In addition, to check whether EEA is superior to SNA in our experiments, we performed t-test with null hypothesis that the objective values from these two algorithms are equal. The resulting p-values are almost zero for all instances. Therefore, we can conclude that EEA outperforms SNA.

5. Conclusion

This work considered a cross-docking system, which is a promising logistics strategy that distributes products from inbound vehicles directly to outbound vehicles by using the warehouse as temporary storage instead of a place for storage and retrieval, eliminating the need for storage and order-picking. This system is well known to be quite suitable for stable-demand products, perishable bulk materials, and pharmaceutical materials.

Many studies have investigated the vehicle routing and truck scheduling problems with cross-docking separately; however, there are few studies of the integration of these subproblems, although the components of a cross-docking system are interconnected with each other. This study investigated an integrated model with cross-docking for a three-echelon supply chain made up of suppliers, cross-docks, and customers.

This study described a MIP formulation for cross-docking, where vehicle routing and truck scheduling were planned cooperatively to achieve the minimum total cost. Further,

we proposed an EEA-based method to obtain a good approximate solution within a reasonable time. The method was applied and proven to be very effective for combinatorial optimization problems; it is a good candidate method for searching for the solutions of multiple subproblems concurrently, especially when the original problem consists of multiple interconnected subproblems that are solved as a whole instead of being considered separately. A computational study has been conducted to quantify the performance of the proposed algorithm, and the computational results show that the proposed algorithm outperforms existing algorithms.

Acknowledgement. This paper was supported by Konkuk University in 2015.

References

Alpan, G., Larbi, R., Penz, B. (2011). A bounded dynamic programming approach to schedule operations in a cross docking platform. *Computers and Industrial Engineering*, 60, 385–396.

Apte, U.M., Viswanthan, S. (2000). Effective cross docking for improving distribution efficiencies. *International Journal of Logistics Research and Applications*, 3, 291–302.

Boysen, N., Fliedner, M. (2010). Cross dock scheduling: classification, literature review and research agenda. *Omega*, 38, 413–422.

Buijs, P., Vis, I.F.A., Carlo, H.J. (2014). Synchronization in cross-docking networks: a research classification and framework. *European Journal of Operational Research*, 239, 593–608.

Chen, P., Guo, Y., Lim, A., Rodrigues, B. (2006). Multiple crossdocks with inventory and time windows. *Computers & Operations Research*, 33, 43–63.

Dondo, R., Cerdá, J., (2013). A sweep-heuristic based formulation for the vehicle routing problem with cross-docking. *Computers & Chemical Engineering*, 48, 293–311.

Dondo, R., Cerdá, J. (2014). A monolithic approach to vehicle routing and operations scheduling of a cross-dock system with multiple dock doors. *Computers & Chemical Engineering*, 63, 184–205.

Dondo, R., Méndez, C.A., Cerdá, J. (2011). The multi-echelon vehicle routing problem with cross docking in supply chain management. *Computers & Chemical Engineering*, 35, 3002–3024.

Goldberg, D.E., Lingle, R., (1985), Alleles, loci, and the traveling salesman problem. In: *Proceedings of an International Conference on Genetic Algorithms and Their Applications*, Vol. 154. Lawrence Erlbaum, Hillsdale, NJ, pp. 154–159.

Gumus, M., Bookbinder, J.H. (2004). Cross-docking and its implications in location-distribution systems. *Journal of Business Logistics*, 25, 199–229.

Jayaraman, V., Ross, A. (2003). A simulated annealing methodology to distribution network design and management. *European Journal of Operational Research*, 144, 629–645.

Kim, Y.K., Kim, J.Y., Kim, Y., (2000). A coevolutionary algorithm for balancing and sequencing in mixed model assembly lines. *Applied Intelligence*, 13.3, 247–258.

Kim, Y.K., Kim, J.Y., Kim, Y., (2001). An endosymbiotic evolutionary algorithm for optimization. *Applied Intelligence*, 15, 117–130.

Kreng, V.B., Chen, F.T. (2008). The benefits of a cross-docking delivery strategy: a supply chain collaboration approach. *Production Planning & Control*, 19, 229–241.

Lee, Y.H., Jung, J.W., Lee, K.M. (2006). Vehicle routing scheduling for cross-docking in the supply chain. *Computers & Industrial Engineering*, 51, 247–256.

Li, Z.P., Low, M.Y.H., Shakeri, M., Lim, Y.G. (2009). Crossdocking Planning and Scheduling: Problems and Algorithms. *SIMTech Technical Reports*, 10 (July–September (3)), 159–167.

Liao, C.J., Lin, Y., Shih, S.C. (2010). Vehicle routing with cross-docking in the supply chain. *Expert Systems with Applications*, 37, 6868–6873.

Maher, M.L., Poon, J. (1996). Modelling design exploration as co-evolution. *Microcomputers in Civil Engineering*, 11, 195–210.

Margulis, L. (1980). *Symbiosis in Cell Evolution*. WH Freeman, San Francisco.

Miao, Z., Yang, F., Fu, K., Xu, D. (2012). Transshipment service through crossdocks with both soft and hard time windows. *Annals of Operations Research*, 192, 21–47.

Morais, V.W.C., Mateus, G.R., Noronha, T.F. (2014). Iterated local search heuristics for the vehicle routing problem with cross-docking. *Expert Systems with Applications*, 41, 7495–7506.

Mousavi, S.M., Vahdani, B., Tavakkoli-Moghaddam, R., Hashemi, H. (2014). Location of cross-docking centers and vehicle routing scheduling under uncertainty: a fuzzy possibilistic-stochastic programming model. *Applied Mathematical Modelling*, 38, 2249–2264.

Ombuki, B., Brian, J.R., Franklin, H. (2006). Multi-objective genetic algorithms for vehicle routing problem with time windows. *Applied Intelligence*, 24, 17–30.

Pereira, F.B., Tavares, J., Machado, P., Costa, E. (2002). GVR: a new genetic representation for the vehicle routing problem. In: *Irish Conference on Artificial Intelligence and Cognitive Science*. Springer, Berlin, Heidelberg, pp. 95–102.

Santos, F.A., Mateus, G.R., da Cunha, A.S. (2011). A branch-and-price algorithm for a vehicle routing problem with cross-docking. *Electronic Notes in Discrete Mathematics*, 37, 249–254.

Santos, F.A., Mateus, G.R., da Cunha, A.S. (2013). The pickup and delivery problem with cross-docking. *Computers & Operations Research*, 40, 1085–1093.

Sung, C.S., Song, S.H. (2003). Integrated service network design for a cross-docking supply chain network. *Journal of the Operational Research Society*, 54, 1283–1295.

Tsui, L.Y., Chang, C.H. (1992). An optimal solution to a dock door assignment problem. *Computers & Industrial Engineering*, 23, 283–286.

Vahdani, B., Zandieh, M. (2010). Scheduling trucks in cross-docking systems: robust meta-heuristics. *Computers & Industrial Engineering*, 58, 12–24.

Van Belle, J., Valckenaers, P., Cattrysse, D. (2012). Cross-docking. State of the art. *Omega*, 40, 827–845.

Wen, M., Larsen, J., Clausen, J., Cordeau, J.F., Laporte, G. (2009). Vehicle routing with cross-docking. *Journal of the Operational Research Society*, 60, 1708–1718.

Yu, W., Egbelu, P.J. (2008). Scheduling of inbound and outbound trucks in cross docking systems with temporary storage. *European Journal of Operational Research*, 184, 377–396.

K.-Y. Lee is an engineer at SK Hynix, Gyeonggi-do, Korea. He received his BS in industrial engineering from Konkuk University, Seoul, in 2016. His research interests are in operations research, production management, supply chain management and algorithm.

J.-S. Lim is an engineer at T-Robotics, Gyeonggi-do, Korea. He received his BS in industrial engineering from Konkuk University, Seoul, in 2016. His research interests are in quality management, production management, supply chain management and algorithm.

S.-S. Ko is a professor in the Department of Industrial Engineering at Konkuk University, Seoul, Korea. He received his PhD in 2003 and MS in 1999 from the School of Industrial and Systems Engineering at the Georgia Institute of Technology and a BS in industrial engineering from Hanyang University, Seoul. His research interests are in operations research, production and inventory control, parallel processing, collaboration systems and financial engineering.