Received May 17, 2020, accepted July 14, 2020, date of publication July 20, 2020, date of current version July 31, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3010176

An Optimization Model and Solution Algorithms for the Vehicle Routing Problem With a "Factory-in-a-Box"

JUNAYED PASHA[1], MAXIM A. DULEBENETS[1], (Member, IEEE), MASOUD KAVOOSI[2], OLUMIDE F. ABIOYE[3], HUI WANG[4], AND WEIHONG GUO[5]

[1]Department of Civil and Environmental Engineering, Florida A&M University-Florida State University, Tallahassee, FL 32310, USA
[2]HNTB Corporation, Tallahassee, FL 32312, USA
[3]Airbus Group Inc., Ashburn, VA 20147, USA
[4]Department of Industrial and Manufacturing Engineering, Florida A&M University-Florida State University, Tallahassee, FL 32310, USA
[5]Department of Industrial and Systems Engineering, Rutgers, The State University of New Jersey, Piscataway, NJ 08854, USA

Corresponding author: Maxim A. Dulebenets (mdulebenets@eng.famu.fsu.edu)

This work was supported in part by the National Science Foundation under Grant CMMI-1901109. The opinions, findings, and conclusions, expressed in this publication, are those of the authors and do not necessarily reflect the views of the National Science Foundation.

ABSTRACT The "factory-in-a-box" concept involves assembling production modules (i.e., factories) in containers and transporting the containers to different customer locations. Such a concept could be highly effective during emergencies, when there is an urgent demand for products (e.g., the COVID-19 pandemic). The "factory-in-a-box" planning problem can be divided into two sub-problems. The first sub-problem deals with the assignment of raw materials to suppliers, sub-assembly decomposition, assignment of sub-assembly modules to manufacturers, and assignment of tasks to manufacturers. The second sub-problem focuses on the transport of sub-assembly modules between suppliers and manufacturers by assigning vehicles to locations, deciding the order of visits for suppliers, manufacturers, and customers, and selecting the appropriate routes within the transportation network. This study addresses the second sub-problem, which resembles the vehicle routing problem, by developing an optimization model and solution algorithms in order to optimize the "factory-in-a-box" supply chain. A mixed-integer linear programming model, which aims to minimize the total cost of the "factory-in-a-box" supply chain, is presented in this study. CPLEX is used to solve the model to the global optimality, while four metaheuristic algorithms, including the Evolutionary Algorithm, Variable Neighborhood Search, Tabu Search, and Simulated Annealing, are employed to solve the model for large-scale problem instances. A set of numerical experiments, conducted for a case study of "factory-in-a-box", demonstrate that the Evolutionary Algorithm outperforms the other metaheuristic algorithms developed for the model. Some managerial insights are outlined in the numerical experiments as well.

INDEX TERMS Factory-in-a-box, metaheuristics, supply chains, urgent demand, vehicle routing problem.

I. INTRODUCTION

Different supply chains have witnessed significant changes over the past decades, which can be attributed to several factors, including a high degree of flexibility, low-volume, low-cost production, and short delivery times [1]–[5]. Moreover, a diverse and dynamic nature of market demand has driven some of the innovative methods used by manufacturers to satisfy customer demands on time. Some manufacturers employ sub-contractors that offer specific services at various locations in order to improve the efficiency of the supply chain network by reducing delivery times [6]. To ensure timely delivery of products, some manufacturers adhere to the distributed manufacturing process, which seeks to move their production sites closer to the location of demand [7]. An emerging concept, which is adopted by certain production companies to address the change and uncertainty associated with customer demand and other challenges related to supply chain management, is a "factory-in-a-box."

The "factory-in-a-box" concept involves installing production modules or factories in containers and transporting the containers using vehicles. The vehicles, carrying the

The associate editor coordinating the review of this manuscript and approving it for publication was Dimitrios Katsaros.

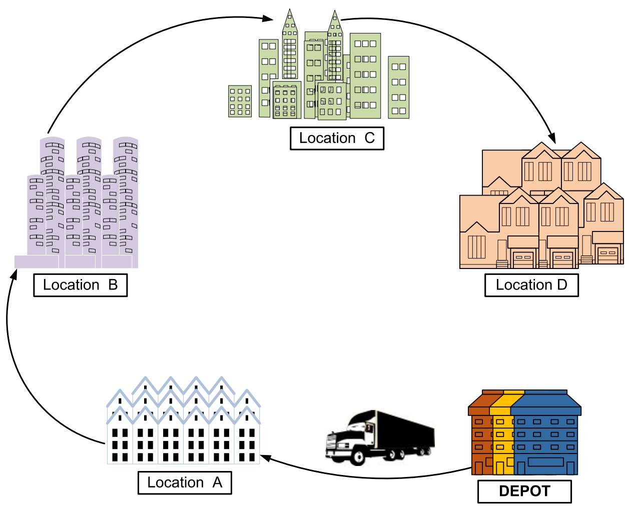


FIGURE 1. An example of a "factory-in-a-box" route.

production modules, are then deployed to the area, where there is a high demand for certain products for a period of time, during which the desired products are manufactured. Also, the vehicles may travel to a new location afterwards in order to satisfy the demand for commodities. The "factory-in-a-box" concept can be implemented under different settings, including the following:

- When there is a high demand over a given time period in an area, the factory will be temporarily stationed and deployed in a production location near customers. When the factory is transported to the next production location after completion, the associated supply chain network has to be updated for the new location; and
- When the demand is in a small batch but of a high variety over a given time period in an area, vehicles can travel between the supper and manufacturer locations to pick-up necessary raw materials and production modules before traveling to the customer locations for final production.

Fig. 1 illustrates an example of a "factory-in-a-box", where a vehicle transports production modules in a container from a depot to four different locations ("A", "B", "C", and "D"), at which the demanded products will be manufactured. As discussed earlier, in some cases, the vehicle may have to stop at supplier and/or manufacturer locations to pick-up necessary raw materials and production modules before traveling to certain customer locations. Some companies have been using the "factory-in-a-box" concept to improve the efficiency of the supply chain management. For instance, in the year of 2018, GE Healthcare inaugurated a "factory-in-a-box" system, named "KUBio," which involves the assembly and transport of pre-fabricated factory modules in containers to a production site that is close to the customers [8], [9]. The factory is further set-up directly at the customer location, and viral-vector-based therapeutics could be mass-produced to satisfy customer demand. Nokia has also adopted the "factory-in-a-box" concept. The production modules, required to make an electronic device, are packaged in a container and shipped to any location with a high demand for the products [10].

Several "factories-in-a-box" could be integrated into a supply chain to jointly deliver products to customers [8], [9]. The manufacturer may require raw materials from various suppliers at specific locations. Moreover, other semi-finished components that are needed to complete production can be fabricated at a fixed location or factory and transported to other sites for production (i.e., transportable manufacturer). The main objective of the "factory-in-a-box" is to improve the mobility of production while ensuring that companies maintain their capacity to satisfy customer demand. The flexibility, introduced by the "factory-in-a-box" concept, assists companies with satisfying customer demands in a short time period and has a tendency to reduce the manufacturing cost. The "factory-in-a-box" manufacturing is of a great importance, especially where there is an urgent demand for certain products, such as immunization vaccines, medical products, medical devices, and medical support during outbreaks of diseases or natural disasters, as well as consumable parts for weapons during a war. For instance, a substantial number of patients, especially the elderly, lost their lives during the Coronavirus Disease 2019 (COVID-19) pandemic. In the U.S. alone, the COVID-19 fatality rate was more than one individual per minute during the month of April 2020 [11]. At such times, delays along the supply chain could literally mean more deaths. If medical supplies are provided in a timely manner during such an emergency, many lives could be saved. Thus, the "factory-in-a-box" concept is beneficial during global emergencies, such as the COVID-19 pandemic, as it can reduce delays along the supply chain by addressing logistic challenges.

The traditional supply chain is generally designed with the idea that a factory is static at a particular location for an extended time period. However, the "factory-in-a-box" concept introduces new flexibility to the supply chain network design, since a factory may be at a new production site for a relatively short time period. The change in the production site will lead to a change in the supply chain network and routing of the vehicles with "factories-in-a-box". Some of the supply chain players may provide raw materials and semi-finished products to multiple production sites, while other supply chain players may provide raw materials and semi-finished products just to one production site. It may be more advantageous to stop at the locations where suppliers and manufacturers provide raw materials and semi-finished products for multiple production sites and pick up these raw materials and semi-finished products during the same visit before traveling to the corresponding production sites. However, due to the vehicle capacity limits, some return visits to certain suppliers and manufacturers may be unavoidable. Therefore, the decisions regarding the transport of sub-assembly modules between suppliers and manufacturers, the order of visits for suppliers, manufacturers, and customers, and selection of the appropriate routes within the transportation network become more challenging as compared to the canonical vehicle routing problem that does not involve any ''factories-in-a-box.''

The design and optimization of the supply chain network have been well-studied in the literature [12]–[14]. Researchers have developed different mathematical models and solution algorithms to optimize various supply chain operations. However, none of the previous studies attempted to optimize operations for the supply chains with a ''factory-in-a-box''. The ''factory-in-a-box'' planning problem can be divided into two sub-problems. The first sub-problem deals with the assignment of raw materials to suppliers, sub-assembly decomposition, assignment of sub-assembly modules to manufacturers, and assignment of tasks to manufacturers. On the other hand, the second sub-problem focuses on the transport of sub-assembly modules between suppliers and manufacturers by assigning vehicles to locations, deciding the order of visits for suppliers, manufacturers, and customers, and selecting the appropriate routes within the transportation network. This paper will address the second sub-problem, which has some similarities with the vehicle routing problem.

The remainder of this manuscript is divided into the following sections. The second section presents a review of the scientific studies that focused on the ''factory-in-a-box'' concept and the most recent efforts dealing with the vehicle routing problem. The third section provides a detailed description of the vehicle routing problem with a ''factory-in-a-box'', while the fourth section presents the mathematical model that was formulated for the studied problem. The fifth section discusses the primary solution approach adopted in this study. The sixth section presents the numerical experiments that were conducted to assess the performance of the proposed solution methodology and highlights the key managerial insights. The final section provides concluding remarks along with a set of future research extensions.

II. LITERATURE REVIEW

This part of the manuscript presents a review of the scientific studies that were collected from the operations research literature with a focus on the following: (1) the ''factory-in-a-box'' concept; and (2) the vehicle routing problem.

A. THE ''FACTORY-IN-A-BOX'' CONCEPT

The review of the literature revealed that only a few studies have focused on the ''factory-in-a-box'' concept. Bengtsson *et al.* [15] discussed how maintenance and monitoring the condition of a ''factory-in-a-box'' might assist companies with achieving the desired flexibility, mobility, and speed. The study suggested that maintenance operations could be used to improve availability in the ''factory-in-a-box'' concept. Jackson and Zaman [16] suggested that there is a need to provide a flexible, reconfigurable, and responsive manufacturing system to address the problem of uncertainty in market demand. The authors highlighted the role of the ''factory-in-a-box'' concept in providing production companies with the mobility and capacity required to deal with uncertain customer demand. Moreover, the study stated that the mobility, flexibility, and speed, introduced by the ''factory-in-a-box'' concept, might allow companies to reduce prohibitive production costs. Some of the modules may be combined and reconfigured into a new factory that could manufacture a new product or change the volume of manufactured products.

Jackson *et al.* [17] investigated the potential of achieving a product service system, based on the results obtained from several ''factory-in-a-box'' projects that were considered by the study. The authors stated that the ''factory-in-a-box'' concept might assist companies with achieving ''result services'', ''product-life extension services'', ''shared utilization services'', and ''demand side management''. Moreover, the study underlined that the ''factory-in-a-box'' might be an efficient and feasible method for reducing the amount of carbon dioxide emissions released around the world. Jiang *et al.* [18] argued that the ''factory-in-a-box'' induces additional challenging decisions in the supply chain network design, such as supply chain reconfiguration and sub-assembly planning, when vehicles travel to different sites. A binary nonlinear model was formulated to minimize the sum of production costs and reconfiguration costs, primarily focusing on supply chain reconfiguration and sub-assembly planning. The numerical experiments were also conducted, which provided a guideline for the supply chain network design as well as reconfiguration for the ''factory-in-a-box.''

B. THE VEHICLE ROUTING PROBLEM

The vehicle routing problem has been well-researched over the past decades. A detailed review of the state-of-the-art on vehicle routing can be found in Eksioglu *et al.* [19] and Braekers *et al.* [20]. The literature review presented herein primarily focuses on the recent vehicle routing studies. Several studies have proposed a number of mathematical models and solution algorithms for different variants of the vehicle routing problem. Some studies addressed the open vehicle routing problem, where vehicles do not return to the depot after they provide service to customers [21]–[25]. Yu *et al.* [21] formulated a mixed-integer linear mathematical model for the open vehicle routing problem with cross-docking. The study aimed to minimize the total cost of providing service for a group of customers. A Simulated Annealing (SA) algorithm was developed to solve the mathematical model for large-scale problem instances, and CPLEX was used to solve the problem to the global optimality. A set of numerical experiments were conducted to evaluate the proposed solution methodology. The results showed that both CPLEX and SA were able to solve small- and medium-scale problem instances to the global optimality. In the meantime, SA provided good-quality solutions for large-scale problem instances as well. Atefi *et al.* [22] studied the open vehicle routing problem with decoupling points, where several carriers transported some products in a large network. The problem was formulated as a mixed-integer programming model. The objective of the study was to minimize the total cost, which included the following components: (i) the load-based transportation cost; (ii) the detour cost; and (iii) the drop cost. An Iterated Local Search Algorithm was developed to solve the model. The results from the computational experiments highlighted the benefits of using decoupling points.

Brandão [23] focused on the open vehicle routing problem, where service time windows were enforced on customers. The study assumed that a customer could be visited only once, and a vehicle was not allowed to return to the depot after serving the last customer. The problem was formulated as a mixed-integer programming model, and the objective was to minimize the total travel distance. An Iterated Local Search Algorithm was proposed to solve the model. The numerical experiments showed that when 100 to 2,000 customers were served, the Iterated Local Search Algorithm provided good-quality solutions within a reasonable computational time. Ruiz *et al.* [25] formulated the open vehicle routing problem, which aimed to minimize the total distance traveled by the vehicles. Homogenous and capacitated vehicles were used to serve the customers. A Biased Random-Key Genetic Algorithm was designed to solve the model. A set of numerical experiments were conducted to assess the performance of the solution algorithm for three sets of benchmark problem instances. The results demonstrated the promising performance of the designed algorithm. Specifically, the algorithm was able to improve the results for 16 out of 30 problem instances.

The vehicle routing problem with stochastic demand is another variant of the generic vehicle routing problem, where the customer demand at nodes in not fixed. Helal *et al.* [26] addressed the capacitated vehicle routing problem with stochastic customer demand. The theory of evidence was used to represent the stochasticity in customer demand. A total of two models were developed for the problem. The first model was formulated as a variant of the chance-constrained programming and ensured that the sum of the customer demands did not exceed the vehicle capacity. The second model was formulated as an extension of the first model, which took into account the stochastic demand recourse strategy, where some corrective actions could be taken if the upper limit on vehicle capacity was violated. The objectives of the two models were to minimize the cost of serving a given number of customers. An SA was used to solve both models. The computational experiments demonstrated that SA was able to solve realistic-scale problem instances and obtained the results that were close to the theoretical results.

Gutierrez *et al.* [27] studied the vehicle routing problem with stochastic demand, and the classical recourse approach was used to model the problem. The objective of the developed model was to minimize the expected route cost. A hybrid algorithm, which was a combination of the Memetic Algorithm and the Greedy Randomized Adaptive Search Procedure, was used to solve the model. The results revealed that the mathematical model was solved within an acceptable computational time for large-scale problem instances with up to 385 customers. It was found that the proposed solution algorithm outperformed other state-of-the-art hybrid algorithms. Salavati-Khoshghalb *et al.* [28] examined the vehicle routing problem with stochastic demand, where the demand of the customers could only be known, when a vehicle arrived at the customer location. The vehicles that could not satisfy the customer demand at a node were required to either split the service or return to the depot and provide enough capacity for the service. An integer programming model was formulated for the problem, and the objective was to minimize the total cost of transportation. An Integer L-Shaped Algorithm in a Branch-and-Cut framework was developed to solve the model. The results showed that the proposed algorithm was able to solve the problem instances with 60 customers and 4 vehicles.

The vehicle routing problem with soft time windows (where a penalty is imposed on vehicles for early or late arrivals at customer nodes) [29]–[34] and strict time windows (when vehicles cannot satisfy customer demand outside a defined arrival time window) [35]–[38] have also been addressed in the state-of-the-art. Bae and Moon [36] focused on the vehicle routing problem with strict time windows, where several depots were considered. A mixed-integer programming model was formulated for the problem. The model aimed to minimize the total cost, which included the following components: (1) travel cost; (2) fixed cost; and (3) labor cost. A Genetic Algorithm and a heuristic were developed to solve the model. The results showed that the Genetic Algorithm outperformed the heuristic for small-scale and medium-scale problem instances. However, the heuristic provided good-quality solutions for large-scale problem instances.

Keskin and Çatay [37] tackled the electric vehicle routing problem. The study allowed partial recharge of vehicles. Moreover, strict time window restrictions were enforced. A mixed-integer programming model was developed to minimize the total travel distance, and an Adaptive Large Neighborhood Search was used to solve the model. The results demonstrated that partial recharge could lead to improved vehicle routes. Huber and Geiger [29] examined the neighborhood operators of a Variable Neighborhood Search (VNS) for the swap-body vehicle routing problem with soft time windows. The study revealed that the sole neighborhood operators could provide better solutions. Hojabri *et al.* [30] proposed a vehicle routing problem with soft time windows. The model was formulated in such a way that the arrival of two different vehicles at separate customer locations must be connected. The model aimed to minimize the total distance traveled. A constraint programming-based Adaptive Large Neighborhood Search algorithm was developed to solve the model. A set of numerical experiments were conducted to evaluate the proposed methodology for realistic-scale problem instances. The results showed that the algorithm was able to obtain good-quality solutions.

Qiu *et al.* [32] formulated the vehicle routing problem with soft time windows, which facilitated discrete split deliveries and pickups, as a mixed-integer linear mathematical model. The study aimed to minimize the total distance traveled by the vehicles. A Tabu Search (TS) algorithm with a unique batch combination and item creation operation was developed to solve the model. The results showed that the item creation operation improved the algorithmic performance and enhanced the algorithmic exploration ability. Furthermore, when compared with some other solution algorithms in the state-of-the-art, the TS algorithm performed better in terms of the solution quality. Subramanyam *et al.* [38] considered operational uncertainties in the vehicle routing problem with strict time windows. A two-stage stochastic optimization model was formulated. Time window assignments were included in the first-stage decisions, while vehicle routes maintaining the assigned time windows were included in the second-stage decisions. The computational experiments demonstrated that the developed Scenario Decomposition Algorithm outperformed the existing solution methods.

The vehicle routing problem and its variants are generally classified as NP-hard problems, based on their combinatorial characteristics. Hence, the majority of studies apply heuristic optimization algorithms [39]–[42], metaheuristic optimization algorithms [43], [44], and hybrid optimization algorithms [45]–[49] to solve the problem. The results from the numerical experiments, conducted in the reviewed studies, suggest that metaheuristic optimization algorithms can obtain near-optimal solutions for large-scale problem instances when compared with the results from exact optimization algorithms.

C. LITERATURE SUMMARY AND CONTRIBUTIONS

A detailed review of the collected literature indicates that only a few studies have discussed the "factory-in-a-box" concept without providing any supporting models for vehicle routing with a "factory-in-a-box" that can be used by the relevant supply chain stakeholders. On the other hand, the literature dealing with the vehicle routing problem is fairly broad. Several variants of the vehicle routing problem have been studied, including the vehicle routing problem with soft time windows, the vehicle routing problem with strict time windows, the vehicle routing problem with stochastic demand, the open vehicle routing problem, and others. Considering the existing trends in supply chains and potential benefits that can be offered by the "factory-in-a-box" concept to different supply chain stakeholders, this study aims to provide the following contributions to the state-of-the-art:

- A novel mathematical formulation is proposed for the vehicle routing problem with a "factory-in-a-box", aiming to minimize the total supply chain cost.
- Based on the "factory-in-a-box" operational features, the problem is formulated as an open capacitated vehicle routing problem with soft time windows and heterogeneous vehicle fleet.
- The model allows capturing complex supplier-to-customer and manufacturer-to-customer relationships.
- Due to the problem complexity, a customized nature-inspired Evolutionary Algorithm is developed to solve the problem.
- A set of numerical experiments are performed to assess the computational performance of the developed solution algorithm against the exact optimization method and alternative metaheuristics.
- Some managerial insights are drawn using the proposed mathematical model and the developed solution algorithm, which would be of interest to different supply chain stakeholders.

III. PROBLEM DESCRIPTION

This section of the manuscript provides a comprehensive description of the problem investigated herein. Overall, there are five different types of nodes, which are the depot node, the supplier nodes, the manufacturer nodes, the customer nodes, and the dummy depot node. The vehicles will start their trips from the depot node, which is denoted by "0". Then, the vehicles will travel from the depot node to the supplier nodes, where they will pick up necessary raw materials, and the manufacturer nodes, where semi-finished products will be picked up. After visiting the required supplier and manufacturer nodes, the vehicles will travel to the customer nodes, where factories for manufacturing the final products will be assembled. In this study, it is assumed that the factory will be assembled at each customer location in order to meet the existing demand. Upon completing the service of the last customer, each vehicle returns to the dummy depot. The travel time and all the costs that are associated with travel from each customer location to the dummy depot are assumed to be zero, which allows modeling the open vehicle routing problem. Fig. 2 illustrates a schematic overview of typical routes, where three vehicles depart from a depot for "factory-in-a-box" manufacturing, pick up raw materials from suppliers and semi-finished products from manufacturers, and set up factories at the customer locations to produce finished products in order to satisfy the customer demands. Based on the provided examples of vehicle routing, two vehicles are expected to serve one customer each, while one vehicle is expected to serve two customers after visiting the required supplier and manufacturer nodes (see Fig. 2).

A set of supplier nodes will be denoted as $N^s = \{1, 2, 3, \ldots, m^1\}$, a set of manufacturer nodes will be denoted as $N^m = \{1, 2, 3, \ldots, m^2\}$, while a set of customer nodes will be denoted as $N^c = \{1, 2, 3, \ldots, m^3\}$. The set of supplier, manufacturer, and customer nodes is represented by $N' = N^s \cup N^m \cup N^c$. A heterogeneous fleet of vehicles $K = \{1, 2, 3, \ldots, m^4\}$ that carry "factories-in-a-box" on

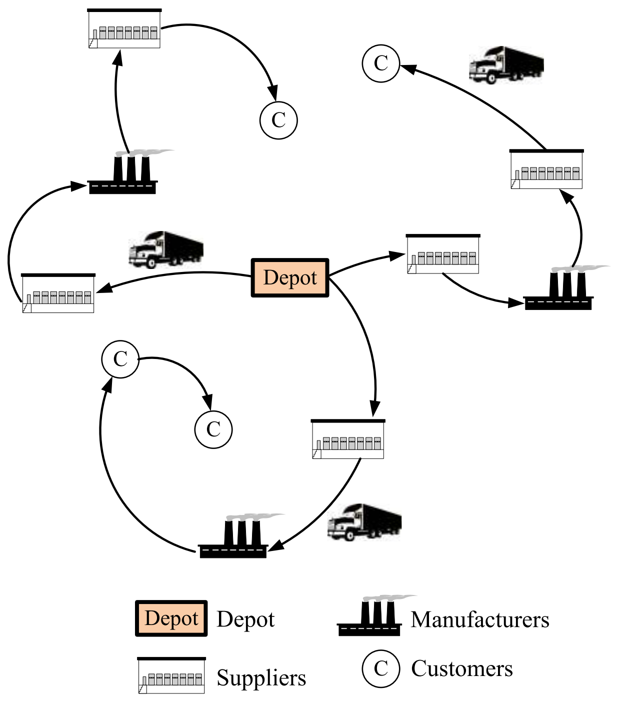


FIGURE 2. **Examples of vehicle routing with a "factory-in-a-box".**

board will be used to serve the nodes, each starting from the depot node "0". Every vehicle has its associated travel cost that is imposed per unit of time – $c_k^v, k \in K$ (USD/hour). The load carrying capacity of any vehicle ($Q_k, k \in K$ – lbs) must not be exceeded. Furthermore, each supplier, manufacturer, or customer node must be visited by only one vehicle exactly once. The problem investigated herein can be represented using a directed graph $G = (N, E), N = N' \cup \{0\} \cup \{m\}$, where N denotes the set of all the nodes (including the depot node 0 and the dummy depot node $\{m\}$, $m = m^1 + m^2 + m^3 + 1$), and $E = \{(i, j), i \in N, j \in N\}$ stands for the edge set. Every edge $(i, j) \in E$ has an associated travel time $t_{ij}, i \in N, j \in N$ (hours), while each node has a specific demand $q_i, i \in N$ (lbs). Positive values of q_i denote the pick-up demand, whereas negative values indicate the delivery demand (i.e., either linehaul or backhaul can be performed for a given node).

In this study, a soft time window $[a_i, b_i], i \in N$ (hours) is imposed at each node, where $a_i, i \in N$ (hours) denotes the start of the time window at node i, while $b_i, i \in N$ (hours) denotes the end of the time window at node i. Therefore, a vehicle arrival before the time window start and after the time window end is permitted at each node. However, due to practical reasons, the vehicles will incur extra costs in case of early or late arrivals. More specifically, if a vehicle arrives at a node before the start of the associated time window, an early arrival cost ($c_i^e, i \in N$ – USD/hour) must be paid. In a similar fashion, a late arrival cost ($c_i^l, i \in N$ – USD/hour) will be incurred if a vehicle arrives at a node after the end of its time window. Moreover, a processing time ($tp_i, i \in N$ – hours) is associated with each node. For the supplier nodes, loading times will be interpreted as processing times, while manufacturing times will be considered as processing times in case of the manufacturer nodes. In addition, processing times at the customer nodes will constitute offloading and manufacturing times.

A precedence level $pl_i, i \in N$ is used in this study in order to enforce a particular order of visiting the nodes. The vehicles will not follow decreasing precedence levels. For instance, if there are four nodes with precedence levels "1", "2", "3", and "3"; then, a vehicle cannot visit the node with precedence level "1" after visiting the node with precedence level "2". Similarly, a vehicle cannot visit the node with precedence level "1" or the node with precedence level "2" after visiting any of the nodes with precedence level "3". In order to imitate real-world scenarios, this study assumes that a customer's demand can only be met after collecting the raw materials and semi-finished products from the pre-specified suppliers and manufacturers. Therefore, before visiting a given customer, its suppliers and manufacturers must be visited by the same vehicle that serves the customer. Such supplier-to-customer and manufacturer-to-customer relationships are critical for the vehicle routing problem with a "factory-in-a-box" and will be modeled using a set of binary parameters. In particular, binary parameter $b_{ij}^{sc}, i \in N^s, j \in N^c$ is used to denote if supplier i must be visited before visiting customer j (i.e., $= 1$ if supplier i must be visited before visiting customer j; else $=0$), while binary parameter $b_{ij}^{mc}, i \in N^m, j \in N^c$ is used to denote if manufacturer i must be visited before visiting customer j (i.e., $=1$ if manufacturer i must be visited before visiting customer j; else $= 0$).

IV. MATHEMATICAL MODEL

This section of the manuscript presents the mathematical model, which addresses the vehicle routing problem with a "factory-in-a-box" (**VRPFIB**).

A. NOMENCLATURE

1) SETS

$N^s = \{1, 2, 3, \ldots, m^1\}$	set of supplier nodes (nodes)
$N^m = \{1, 2, 3, \ldots, m^2\}$	set of manufacturer nodes (nodes)
$N^c = \{1, 2, 3, \ldots, m^3\}$	set of customer nodes (nodes)
$N' = N^s \cup N^m \cup N^c$	set of supplier, manufacturer, and customer nodes (nodes)
$N = N' \cup \{0\} \cup \{m\}$	set of all the nodes (nodes)
$E = \{(i, j), i \in N, j \in N\}$	set of edges (edges)
$K = \{1, 2, 3, \ldots, m^4\}$	set of vehicles (vehicles)

2) DECISION VARIABLES

$\boldsymbol{x}_{ijk} \in \mathbb{B}\ \forall i \in N, j \in N, k \in K$	$=1$ if vehicle k traverses edge (i, j) ($=0$ otherwise)

3) AUXILIARY VARIABLES

$z_{ik} \in \mathbb{B}\ \forall i \in N', k \in K$	=1 if vehicle k visits node i (=0 otherwise)
$\boldsymbol{y}_{ik} \in \mathbb{R}^+\ \forall i \in N, k \in K$	used capacity of vehicle k upon arrival at node i (lbs)
$\boldsymbol{s}_{ik} \in \mathbb{R}^+\ \forall i \in N, k \in K$	service start time at node i by vehicle k (hours)
$\boldsymbol{e}_{ik} \in \mathbb{R}^+\ \forall i \in N, k \in K$	early arrival time at node i by vehicle k (hours)
$\boldsymbol{l}_{ik} \in \mathbb{R}^+\ \forall i \in N, k \in K$	late arrival time at node i by vehicle k (hours)
$\boldsymbol{TTC} \in \mathbb{R}^+$	total travel cost (USD)
$\boldsymbol{TEAC} \in \mathbb{R}^+$	total early arrival cost (USD)
$\boldsymbol{TLAC} \in \mathbb{R}^+$	total late arrival cost (USD)

4) PARAMETERS

$m \in \mathbb{N}$	total number of nodes (nodes)
$m^1 \in \mathbb{N}$	number of supplier nodes (nodes)
$m^2 \in \mathbb{N}$	number of manufacturer nodes (nodes)
$m^3 \in \mathbb{N}$	number of customer nodes (nodes)
$m^4 \in \mathbb{N}$	available number of vehicles (vehicles)
$Q_k \in \mathbb{R}^+\ \forall k \in K$	load carrying capacity of vehicle k (lbs)
$q_i \in \mathbb{R}\ \forall i \in N$	demand at node i (lbs)
$a_i \in \mathbb{R}^+\ \forall i \in N$	start of the time window at node i (hours)
$b_i \in \mathbb{R}^+\ \forall i \in N$	end of the time window at node i (hours)
$t_{ij} \in \mathbb{R}^+\ \forall i \in N, j \in N$	time to travel from node i to node j (hours)
$tp_i \in \mathbb{R}^+\ \forall i \in N$	processing time at node i (hours)
$pl_i \in \mathbb{N}\ \forall i \in N$	precedence level of node i (precedence level)
$b^{sc}_{ij} \in \mathbb{B}\ \forall i \in N^s, j \in N^c$	binary parameter denoting if supplier i must be visited before visiting customer j
$b^{mc}_{ij} \in \mathbb{B}\ \forall i \in N^m, j \in N^c$	binary parameter denoting if manufacturer i must be visited before visiting customer j
$c^v_k \in \mathbb{R}^+\ \forall k \in K$	unit travel cost of vehicle k (USD/hour)
$c^e_i \in \mathbb{R}^+\ \forall i \in N$	unit early arrival cost at node i (USD/hour)
$c^l_i \in \mathbb{R}^+\ \forall i \in N$	unit late arrival cost at node i (USD/hour)
$M \in \mathbb{R}^+$	large positive number

B. MODEL FORMULATION

The **VRPFIB** mathematical model is formulated as follows:

$$min\,(\boldsymbol{TTC} + \boldsymbol{TEAC} + \boldsymbol{TLAC}) \tag{1}$$

Subject to:

$$\boldsymbol{x}_{iik} = 0 \quad \forall i \in N,\ k \in K \tag{2}$$

$$\sum_{i\in N}\sum_{k\in K}\boldsymbol{x}_{ijk} = 1 \quad \forall j \in N' \tag{3}$$

$$\sum_{i\in N}\boldsymbol{x}_{ijk} = \sum_{i\in N}\boldsymbol{x}_{jik} \quad \forall j \in N', k \in K \tag{4}$$

$$\sum_{j\in N}\boldsymbol{x}_{ijk} = 1 \quad \forall i = 0, k \in K \tag{5}$$

$$\sum_{i\in N^s}\boldsymbol{x}_{ijk} = 0 \quad \forall j = m, k \in K \tag{6}$$

$$\sum_{i\in N^m}\boldsymbol{x}_{ijk} = 0 \forall j = m, k \in K \tag{7}$$

$$\boldsymbol{y}_{ik} \le Q_k \quad \forall i \in N,\ k \in K \tag{8}$$

$$\boldsymbol{y}_{ik} + q_i - M\left(1 - \boldsymbol{x}_{ijk}\right) \le \boldsymbol{y}_{jk} \quad \forall i \in N, j \in N, k \in K \tag{9}$$

$$\boldsymbol{s}_{ik} + tp_i + t_{ij} - M\left(1 - \boldsymbol{x}_{ijk}\right) \le \boldsymbol{s}_{jk} \quad \forall i \in N, j \in N, k \in K \tag{10}$$

$$\boldsymbol{e}_{ik} \ge a_i - \boldsymbol{s}_{ik} - M\left(1 - \boldsymbol{z}_{ik}\right) \quad \forall i \in N, k \in K \tag{11}$$

$$\boldsymbol{l}_{ik} \ge \boldsymbol{s}_{ik} - b_i - M\left(1 - \boldsymbol{z}_{ik}\right) \quad \forall i \in N, k \in K \tag{12}$$

$$\boldsymbol{z}_{ik} = \sum_{j\in N}\boldsymbol{x}_{ijk} \quad \forall i \in N', k \in K \tag{13}$$

$$\sum_{i\in N}\boldsymbol{x}_{ijk} = \frac{\sum_{i\in N^s} b^{sc}_{ij}\boldsymbol{z}_{ik}}{\sum_{i\in N^s} b^{sc}_{ij}} \quad \forall j \in N^c, k \in K \tag{14}$$

$$\sum_{i\in N}\boldsymbol{x}_{ijk} = \frac{\sum_{i\in N^m} b^{mc}_{ij}\boldsymbol{z}_{ik}}{\sum_{i\in N^m} b^{mc}_{ij}} \quad \forall j \in N^c, k \in K \tag{15}$$

$$pl_i - M\left(1 - \boldsymbol{x}_{ijk}\right) \le pl_j \quad \forall i \in N, j \in N, k \in K \tag{16}$$

$$\boldsymbol{TTC} = \sum_{i\in N}\sum_{j\in N}\sum_{k\in K} c^v_k t_{ij}\boldsymbol{x}_{ijk} \tag{17}$$

$$\boldsymbol{TEAC} = \sum_{i\in N}\sum_{k\in K} c^e_i\boldsymbol{e}_{ik} \tag{18}$$

$$\boldsymbol{TLAC} = \sum_{i\in N}\sum_{k\in K} c^l_i\boldsymbol{l}_{ik} \tag{19}$$

$$\boldsymbol{x}_{ijk}, \boldsymbol{z}_{ik}, b^{cs}_{ij}, b^{cm}_{ij} \in \mathbb{B} \quad \forall i \in N, j \in N, k \in K \tag{20}$$

$$m, m^1, m^2, m^3, m^4, pl_i \in \mathbb{N} \quad \forall i \in N \tag{21}$$

$$\boldsymbol{y}_{ik}, \boldsymbol{s}_{ik}, \boldsymbol{e}_{ik}, \boldsymbol{l}_{ik}, \boldsymbol{TTC}, \boldsymbol{TEAC}, \boldsymbol{TLAC}, Q_k, a_i, b_i, t_{ij}, tp_i, c^v_k, c^e_i, c^l_i, M \in \mathbb{R}^+ \forall i \in N, j \in N, k \in K \tag{22}$$

$$q_i \in \mathbb{R} \forall i \in N \tag{23}$$

The objective function (1) of the **VRPFIB** mathematical model minimizes the total supply chain cost, which includes the total travel cost, the total early arrival cost, and the total late arrival cost. Constraint set (2) indicates that a vehicle cannot travel from a node to the same node. Constraint set (3) ensures that each supplier, manufacturer, and customer node is visited once and by only one vehicle. Constraint set (4) guarantees that each supplier, manufacturer, and customer node is arrived at and departed from by the same vehicle. Constraint set (5) implies that each trip starts from the depot. Constraint sets (6) and (7) indicate that a vehicle cannot travel to the dummy depot from a supplier node or a manufacturer node, respectively. Constraint set (8) implies that the capacity of a given vehicle is not exceeded at any time during a trip. Constraint set (9) maintains the consistency of used capacity (i.e., if a vehicle travels from node i to node j, then the used capacity of the vehicle upon arrival at node j should not be less than the sum of the used capacity of the vehicle upon arrival at node i and the demand at node i).

Constraint set (10) ensures the consistency of time (i.e., if a vehicle travels from node i to node j, then the service start time at node j should not be less than the sum of the service start time at node i, the processing time at node i, and the travel time from node i to node j). Constraint sets (11) and (12) estimate the early arrival time and the late arrival time at each node, respectively. Constraint set (13) determines if a vehicle departs from a supplier, manufacturer, or customer node. Constraint sets (14) and (15) imply that before visiting a given customer, a vehicle visits the customer's suppliers and manufacturers, respectively. Constraint set (16) indicates that each vehicle visits the nodes following the precedence levels. Constraint sets (17) to (19) estimate the total travel cost, the total early arrival cost, and the total late arrival cost, respectively. Constraint sets (20) to (23) define the nature of the decision variables, auxiliary variables, and parameters used in the **VRPFIB** mathematical model.

V. SOLUTION METHODOLOGY

In this study, an Evolutionary Algorithm (EA) was developed to solve the **VRPFIB** mathematical model. Considering the findings from the conducted literature review, EA is one of the most popular solution methodologies, showing a promising performance in solving different types of vehicle routing problems [19], [20], [25], [35], [36]. Fig. 3 depicts the general layout of the EA developed in this study. Based on Fig. 3, the input data required for different parameters of the mathematical model as well as the algorithmic parameters are inserted in the algorithm. Then, the initial population is generated. All the solutions in the initial population are evaluated in the following step of the algorithm using equation (1), which was developed for the objective function. Afterwards, the stopping criterion is checked, and the algorithm returns the best solution in case of satisfaction. Note that the stopping criterion was defined as reaching out to a specific number of generations. The number of generations will be further determined throughout the parameter tuning analysis (see section VI.B) [50].

In the next step, the parent chromosomes (in this study, the terms such as "chromosome", "individual", and "solution" have the same meaning and, therefore, are used interchangeably) are selected to generate children for the next generation. The children are produced utilizing specific

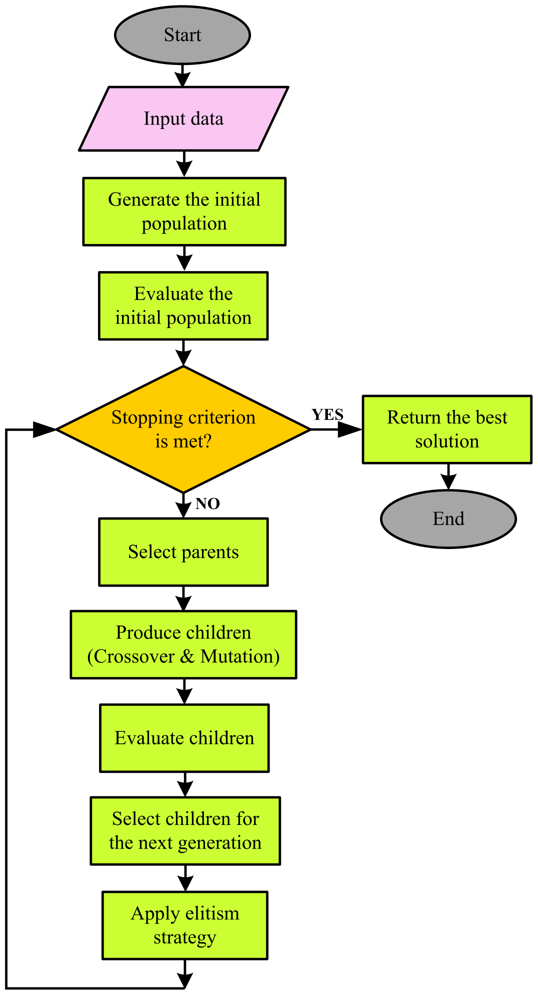
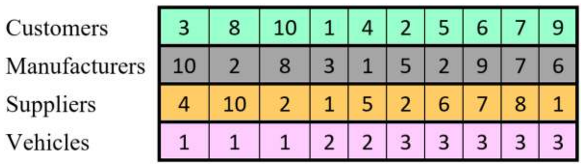

Start
Input data
Generate the initial population
Evaluate the initial population
Stopping criterion is met?
YES
Return the best solution
End
NO
Select parents
Produce children (Crossover & Mutation)
Evaluate children
Select children for the next generation
Apply elitism strategy

FIGURE 3. The main steps of EA.

crossover and mutation operators, developed exclusively for this study. The children that were generated in the previous step are evaluated in the next step for their fitness. A selection strategy is applied to the generated children chromosomes to select a specific number of them for the next generation. An elitism strategy is applied to the newly generated population to guarantee the survival of the fittest solution existing in the current generation. Based on the stochastic nature of the entire EA, including the stochastic way that the children chromosomes are generated, there is no guarantee to have better solutions for the next generation. Hence, the elitism strategy is applied in the algorithm to prevent any solution quality retrogression throughout the algorithmic run [50]. Afterwards, the algorithm checks the stopping criterion and in case of satisfaction, the discovered best solution is returned; otherwise, the main loop of EA is iterated again. In the following sections, all the main algorithmic steps are further described in detail.

A. CHROMOSOME REPRESENTATION

In the considered vehicle routing problem with a "factory-in-a-box", it is assumed that the desired product for each



Customers	3	8	10	1	4	2	5	6	7	9
Manufacturers	10	2	8	3	1	5	2	9	7	6
Suppliers	4	10	2	1	5	2	6	7	8	1
Vehicles	1	1	1	2	2	3	3	3	3	3

FIGURE 4. The chromosome representation.

customer requires a set of different raw materials (can be picked up from suppliers) and a set of different semi-finished products (can be picked up from manufacturers). Moreover, it is assumed that each vehicle should visit the required suppliers and manufacturers first, and after that move towards the customer location. In order to configure the chromosomes and represent the solutions for the aforementioned problem, 4-dimensional integer chromosomes will be used (an example of a chromosome is shown in Fig. 4). Each chromosome component (i.e., vehicle identifiers, supplier identifiers, manufacturer identifiers, and customer identifiers) is called "gene". The location of a gene along the chromosome is determined by "locus" (note that "loci" is the plural form of "locus"), and the value of each gene is called "allele" [50]–[52]. In the considered chromosome example (see Fig. 4), locus "2" contains the genes with allele "1" (corresponds to vehicle "1"), allele "10" (corresponds to supplier "10"), allele "2" (corresponds to manufacturer "2"), and allele "8" (corresponds to customer "8"). A total of 10 unique customers should be served. There are three vehicles "1", "2", and "3", which travel along the transportation network to meet the demand at the customer locations.

In the considered chromosome example, it is assumed that there are multiple suppliers and multiple manufacturers, and each product, ordered by a given customer, would require for a given vehicle to stop at several suppliers to pick up the raw materials and several manufactures to pick up the semi-finished products. Two 2-dimensional binary data structures were used in this study to determine a set of suppliers and a set of manufacturers that have to be visited before arriving at the location of a given customer (i.e., the supplier-to-customer and manufacturer-to-customer assignments that are defined using parameters $b^{sc}_{ij}, i \in N^s, j \in N^c$ and $b^{mc}_{ij}, i \in N^m, j \in N^c$ in the **VRPFIB** mathematical model). Fig. 5 shows an example of the binary data structures for the supplier-to-customer and manufacturer-to-customer assignments, where the products that were ordered by customer "1" require the raw materials from suppliers "2" and "10" as well as the semi-finished products from manufacturers "1" and "7". Note that initials "V", "S", "M", and "C" will be used throughout the manuscript for the vehicles, suppliers, manufacturers, and customers, respectively.

Based on Fig. 4, vehicle "1" should serve customers "3", "8", and "10". Moreover, before visiting customer "3", vehicle "1" should make stops at the locations of suppliers "2", "8", and "10" as well as at the locations of manufacturers "1" and "8" (see Fig. 5). Note that the order of visiting different nodes for each vehicle (i.e., a route of each vehicle) can be determined using the chromosome representation (Fig. 4). Some of the suppliers or manufacturers can be repeated more than once in a given chromosome, when the unique number of suppliers and/or the unique number of manufacturers is less than the unique number of customers (see Fig. 4, where suppliers "1" and "2" are represented with four separate genes). In the latter case, the first repetition (from left) determines the order of visiting a given supplier or manufacturer. For example, supplier "2" is placed in loci "3" and "6"; hence, locus "3" (i.e., the outmost left locus) will be used to determine the order of visiting supplier "2".

The route of vehicle "1" is presented in Fig. 6, where it can be observed that the trip starts from the depot. Then, vehicle "1" visits suppliers "10", "2", and "8" (in that specific order since the genes with suppliers "10", "2", and "8" are placed in loci "2", "3", and "9", respectively – see Fig. 4) as well as manufacturers "8" and "1" (in that specific order since the genes with manufacturers "8" and "1" are placed in loci "3" and "5", respectively – see Fig. 4) before stopping at the location of customer "3". After serving

	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10
C1	0	1	0	0	0	0	0	0	0	1
C2	0	1	0	0	0	0	0	0	0	1
C3	0	1	0	0	0	0	0	1	0	1
C4	0	0	0	0	1	0	0	1	0	1
C5	0	0	0	0	1	0	1	1	0	0
C6	1	0	0	0	0	0	1	0	0	0
C7	1	0	0	1	0	0	0	0	0	0
C8	1	0	0	1	0	0	1	0	0	1
C9	0	0	0	1	0	0	0	1	0	0
C10	1	0	0	0	1	1	0	1	0	0

	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10
C1	1	0	0	0	0	0	1	0	0	0
C2	1	0	0	0	0	0	1	1	0	0
C3	1	0	0	0	0	0	0	1	0	0
C4	0	0	0	0	0	0	0	1	0	0
C5	1	0	0	0	1	0	0	1	0	0
C6	0	1	0	0	0	0	1	0	0	0
C7	0	0	0	0	1	0	1	0	0	1
C8	0	0	0	0	1	0	0	0	0	1
C9	0	1	1	0	1	0	0	0	0	0
C10	1	1	0	0	0	1	0	0	1	0

FIGURE 5. A sample of binary data structures developed for the required suppliers and manufacturers for each customer.

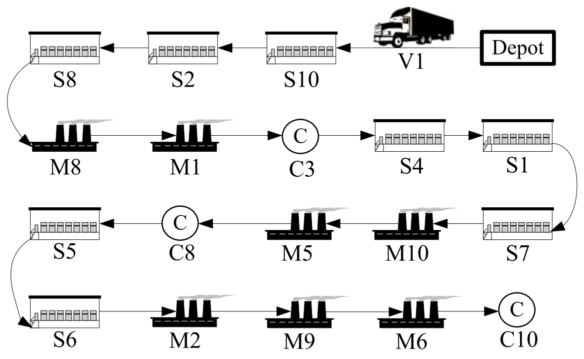


FIGURE 6. The route of vehicle "1".

customer "3", vehicle "1" travels to suppliers "4", "1", and "7" as well as manufacturers "10" and "5" before stopping at the location of customer "8". Upon service completion of customer "8", vehicle "1" travels to suppliers "5" and "6" as well as manufacturers "2", "9", and "6" before stopping at the location of customer "10". Note that if two or more customers, served by the same vehicle, require raw materials or semi-finished products from the same supplier or manufacturer, the corresponding vehicle can visit that supplier or manufacturer just one time and pick up a sufficient amount of raw materials or semi-finished products. Such a strategy is expected to facilitate timely service of customers as long as the vehicle capacity is not violated.

B. INITIAL POPULATION GENERATION

In order to start the process of algorithmic search, an initial population should be generated. The initial population determines the areas of the search space where the algorithmic search should be started from. In this study, a specific strategy was devised to generate the initial population. In order to increase the population diversity, the rows of customers and vehicles are generated randomly, while the rows of suppliers and manufacturers are generated based on the requirements of customers. In particular, all the required suppliers and manufacturers for the corresponding customers are extracted from the binary data structures, shown in Fig. 5. Afterwards, the list of suppliers and the list of manufacturers are perturbed to consider a random nature of the initial population generation and are inserted in the chromosome. Note that the population size (i.e., the total number of chromosomes available in the population) will be determined based on the parameter tuning analysis (see section VI.B) [50].

C. PARENT SELECTION

In EAs, more promising chromosomes are selected based on a specific strategy to play the role of parents and produce children for the next generation [50]. Different selection strategies have been introduced in the EA literature. A roulette wheel selection strategy was deployed to select the parents in the developed EA. As illustrated in Fig. 7, the roulette wheel selection strategy has a rolling wheel and a fixed pointer. Each solution is allocated a portion of the wheel proportional to its objective value. Thus, the solution with the best objective function is allocated the largest portion of the rolling wheel. In order to select a chromosome, the rolling wheel is turned and once it stops, the chromosome pointed by the fixed pointer is selected as a parent. The rolling wheel is turned several times until the required number of parents (equal to the population size) is selected [50].

D. CROSSOVER OPERATOR

After selecting the parent chromosomes, a crossover operator should be applied to them to produce the children chromosomes. The crossover operator provides the algorithm with an opportunity to explore new areas in the search space [50]. Based on the problem description, each customer should be

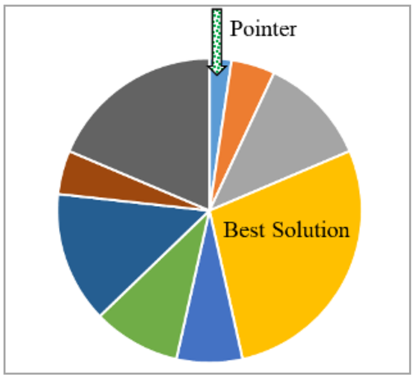
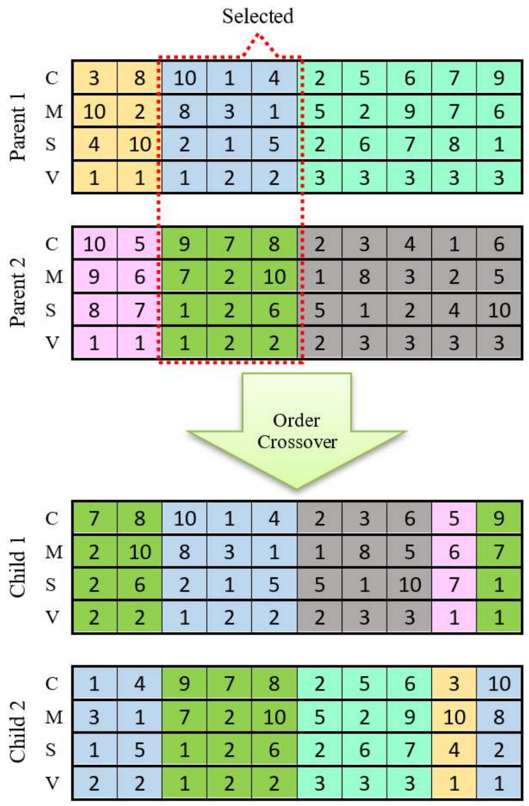


FIGURE 7. Roulette wheel selection.



Parent 1

C	3	8	10	1	4	2	5	6	7	9
M	10	2	8	3	1	5	2	9	7	6
S	4	10	2	1	5	2	6	7	8	1
V	1	1	1	2	2	3	3	3	3	3

Parent 2

C	10	5	9	7	8	2	3	4	1	6
M	9	6	7	2	10	1	8	3	2	5
S	8	7	1	2	6	5	1	2	4	10
V	1	1	1	2	2	2	3	3	3	3

Child 1

C	7	8	10	1	4	2	3	6	5	9
M	2	10	8	3	1	1	8	5	6	7
S	2	6	2	1	5	5	1	10	7	1
V	2	2	1	2	2	2	3	3	1	1

Child 2

C	1	4	9	7	8	2	5	6	3	10
M	3	1	7	2	10	5	2	9	10	8
S	1	5	1	2	6	2	6	7	4	2
V	2	2	1	2	2	3	3	3	1	1

FIGURE 8. Implementation of the order crossover.

served only once, which means that no repetitive allele should be present in the row of customers. Hence, the order crossover was considered for this study, as the order crossover prevents generation of repetitive alleles and has been widely used for the chromosomes that have integer-coded representations [50]. The process of the order crossover is depicted in Fig. 8. As it can be observed in Fig. 8, a portion of the parents is selected randomly. The portion, selected from parent "1" (including customers "10", "1", and "4"), is copied into the same loci of child "1". Then, those genes, which are not present in the copied portion (i.e., customers "2", "3", "6", "5", "9", "7", and "8"), are selected from parent "2" in the order, which starts after the selected portion towards the end followed from the beginning of the chromosome in the

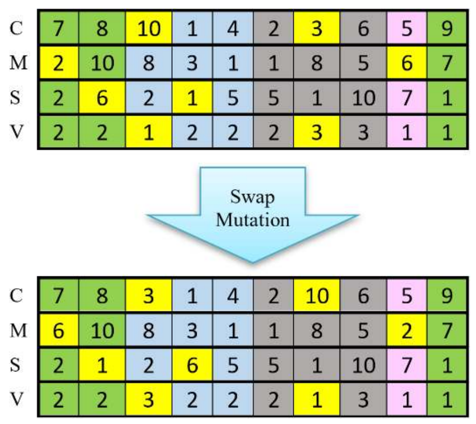


FIGURE 9. Implementation of the swap mutation.

direction of left to right. The latter process should be repeated to produce child "2" [50].

E. MUTATION OPERATOR

The children, produced using the crossover operator, undergo the mutation operation. Particularly, the mutation operator helps the algorithm to exploit the areas of the search space, discovered by the crossover operator [50]. In this study, the swap mutation operator was deployed, considering the fact that it does not make significant genetic changes in the chromosomes, which could be helpful with the exploitation of the search space. As illustrated in Fig. 9, the swap mutation is applied to each one of the chromosome rows independently, considering the probability of 25% (i.e., only one of the chromosome rows will be mutated at a time to prevent significant genetic changes). In particular, the swap mutation selects two genes and exchanges their alleles. The genes, which are colored in yellow, are selected randomly and their alleles are exchanged [50]. For example, the genes with customers "10" and "3" exchange their alleles (see Fig. 9).

F. FITNESS FUNCTION ESTIMATION

Random generation of the initial population and also implementation of the algorithmic operators (i.e., the crossover and mutation operators) might result in generation of some infeasible solutions. In the chromosomes devised for this study, violation of the vehicle capacity requirements makes the solutions infeasible. In order to manage the infeasible solutions, the penalization method was adopted in this study. Particularly, the fitness function value of chromosome c, belonging to the set of chromosomes $Chrm = \{1, \ldots, PopSize\}$, in generation g, belonging to the set of generations $Gen = \{1, \ldots, gens\}$, is estimated using the following equation:

$$FitVal_{cg} = \mathbf{Z}_{cg} + \alpha \varpi_{cg} \quad \forall c \in Chrm,\ g \in Gen \qquad (24)$$

Here, $FitVal_{cg}$ and $\mathbf{Z}_{cg}$ are the fitness value and the objective function value of chromosome c in generation g, respectively; α is the penalty coefficient; and ϖ_{cg} is the cumulative violation of the vehicle capacities in chromosome c in generation g. Note that ϖ_{cg} can be estimated as follows:

$$\varpi_{cg} = \sum_{i \in N} \sum_{j \in N} \sum_{k \in K} max\left\{0; (L_k - Q_k)\,\mathbf{x}_{ijk}\right\} \quad \forall c \in Chrm, g \in Gen \qquad (25)$$

Here, Q_k and L_k are the capacity and the load of vehicle k, respectively. Considering the fact that the developed mathematical model aims to minimize the total cost, increasing the cumulative violation of vehicle capacity would decrease the chance of a given chromosome to participate in the algorithmic operations. The latter would also consequently allow avoiding propagation of the genes of the low-quality chromosomes throughout the algorithmic run.

G. CHILDREN SELECTION

In this study, a binary tournament selection strategy was adopted to implement the children selection. Considering the complicated structure of the chromosomes and the high computational efforts required to complete the process of implementation of the algorithmic operators and chromosome evaluations, the binary tournament selection strategy was adopted. The binary tournament selection does not require a high computational effort. In the binary tournament selection strategy, two children are selected randomly, and the one with higher fitness will be further selected to be one of the chromosomes for the next generation. The binary tournament is repeated several times until the required number of chromosomes, which is equal to the population size, is selected for the next generation [50].

VI. NUMERICAL EXPERIMENTS

This section of the manuscript demonstrates some numerical experiments to showcase the performance of the proposed solution methodology and draw some managerial insights using the developed mathematical model for vehicle routing with a "factory-in-a-box". Three other widely used solution algorithms, namely VNS, TS, and SA, were used in this study to assess the computational performance of the developed EA algorithm. Note that a comprehensive description of VNS, TS, and SA can be found in Hansen *et al.* [53], Cordeau *et al.* [54], and Emde and Boysen [55], respectively. Small-scale problem instances were generated in order to conduct the optimality gap analysis for the considered solution algorithms, while large-scale problem instances were developed to conduct a comprehensive assessment of the algorithms in terms of different metrics (i.e., objective function, CPU time, convergence patterns, and stochastic variations). During the optimality gap analysis, CPLEX was used as the exact solution approach, and it was executed with General Algebraic Modeling System (GAMS) for small-scale problem instances. The metaheuristic algorithms (i.e., EA, VNS, TS, and SA), on the other hand, were encoded with MATLAB (2016a). All the numerical experiments were performed on an Intel(R) Core$^{\text{TM}}$i7-7700K processor with a RAM of 32 GB and an operating system of Windows 10. The following sections of the manuscript provide more

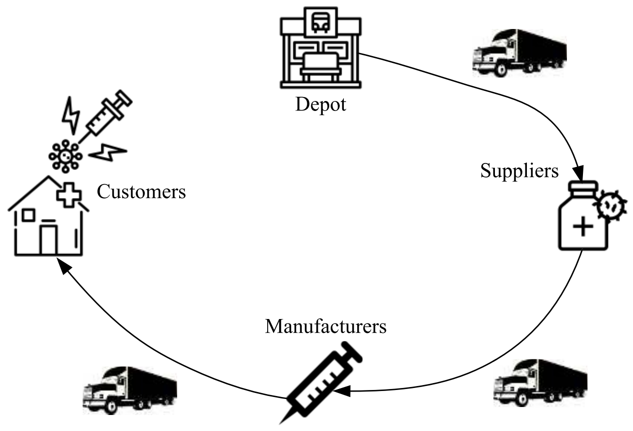


FIGURE 10. **A vaccination project with a "factory-in-a-box".**

details regarding the case study that was used throughout the experiments, parameter tuning for the considered solution algorithms, detailed comparative analysis of the algorithms, and managerial insights.

A. CASE STUDY

In order to evaluate the performance of the proposed solution methodology and draw some managerial insights using the developed mathematical model, a vaccination project involving "factories-in-a-box" was considered as a case study. Based on the vaccination project, raw products (i.e., medical liquids) were picked up from suppliers, while semi-finished products (i.e., syringes) were picked up from manufacturers. The final products (i.e., vaccines) were assembled at the customer locations, as inspired by the "factory-in-a-box" concept (see Fig. 10). Table 1 showcases different parameter values adopted for the case study. During the optimality gap analysis, where small-scale problem instances were used, a maximum of 22 suppliers, 22 manufacturers, and 22 customers were considered for vehicle routing (i.e., $m^1 = m^2 = m^3 = 22$). However, for large-scale problem instances, which were generated to compare the performance of the metaheuristic algorithms and to obtain managerial insights from the developed methodology, 55 to 75 customers as well as 10 suppliers and 10 manufacturers were considered. The fleet size was set to 20 vehicles (i.e., $m^4 = 20$) and 15-30 vehicles (i.e., $m^4 = [15; 30]$) for small-scale and large-scale problem instances, respectively. Note that the mathematical model allows to have unused vehicles. Therefore, not all the vehicles might be used, especially when the total number of nodes is fairly low.

For the case study, 3 cc syringes, weighing 15 gm, were used [56]. The weight of medical liquid per vaccine was assumed to be 0.007 lbs, and the weight of empty syringes was assumed to be 0.033 lbs. Roughly 50,000 vaccines were finally produced at each customer node. So, the demand at supplier nodes ($q_i, i \in N^s$) was estimated as $0.007 \cdot U[0.8 \cdot 50,000; 1.2 \cdot 50,000]$ lbs or $U[280; 420]$ lbs. Here, U represents uniformly distributed pseudo-random numbers. Similarly, the demand at manufacturer nodes ($q_i, i \in N^m$)

TABLE 1. **The parameter values adopted for the case study.**

Parameter	Value
Total number of nodes (nodes): $m \in \mathbb{N}$	$[11; 97]$
Number of supplier nodes (nodes): $m^1 \in \mathbb{N}$	$[3; 22]$
Number of manufacturer nodes (nodes): $m^2 \in \mathbb{N}$	$[3; 22]$
Number of customer nodes (nodes): $m^3 \in \mathbb{N}$	$[3; 75]$
Available number of vehicles (vehicles): $m^4 \in \mathbb{N}$	$[15; 30]$
Load carrying capacity of vehicle k (lbs): $Q_k \in \mathbb{R}^+ \forall k \in K$	$U[6{,}000; 6{,}300]$
Demand at supplier node i (lbs): $q_i \in \mathbb{R} \forall i \in N^s$	$U[280; 420]$
Demand at manufacturer node i (lbs): $q_i \in \mathbb{R} \forall i \in N^m$	$U[1{,}320; 1{,}980]$
Demand at customer node i (lbs): $q_i \in \mathbb{R} \forall i \in N^c$	$-U[1{,}600; 2{,}400]$
Duration of time window at node i (hours)	10
Time to travel from node i to node j (hours): $t_{ij} \in \mathbb{R}^+ \forall i \in N, j \in N'$	$U[5; 10]$
Processing time at supplier/manufacturer node i (hours): $tp_i \in \mathbb{R}^+ \forall i \in N^s \cup N^m$	$U[1; 3]$
Processing time at customer node i (hours): $tp_i \in \mathbb{R}^+ \forall i \in N^c$	$U[2; 8]$
Precedence level of node i (precedence level): $pl_i \in \mathbb{N} \forall i \in N$	$[0; 4]$
Binary parameter denoting if supplier i must be visited before visiting customer j: $b_{ij}^{sc} \in \mathbb{B} \forall i \in N^s, j \in N^c$	$U[0; 1]$
Binary parameter denoting if manufacturer i must be visited before visiting customer j: $b_{ij}^{mc} \in \mathbb{B} \forall i \in N^m, j \in N^c$	$U[0; 1]$
Unit travel cost of vehicle k (USD/hour): $c_k^v \in \mathbb{R}^+ \forall k \in K$	$U[900; 950]$
Unit early arrival cost at node i (USD/hour): $c_i^e \in \mathbb{R}^+ \forall i \in N'$	$U[0.1; 100]$
Unit late arrival cost at node i (USD/hour): $c_i^l \in \mathbb{R}^+ \forall i \in N'$	$U[0.1; 100]$
Large positive number: $M \in \mathbb{R}^+$	1,000,000

was estimated as $0.033 \cdot U[0.8 \cdot 50,000; 1.2 \cdot 50,000]$ lbs or $U[1,320; 1,980]$ lbs. The demand at customer nodes is the sum of the supplier demand and the manufacturer demand. However, the customer demand is negative, as this study considers pickup demands to be positive and delivery demands to be negative. Therefore, the demand at customer nodes ($q_i, i \in N^c$) was estimated as $-U[1,600; 2,400]$ lbs. Note that the depot (and the dummy depot) had zero demand. In order to transport the goods, semi-trailer trucks with a capacity of $U[12,000; 12,600]$ lbs were used [57]. Note that the materials, which are required to assemble factories, are expected to occupy a significant portion of the trucks' capacity. If half of the load carrying capacity of the trucks is assumed to be covered by the materials that are required to assemble factories; then, the trucks would be able to carry about $U[6,000; 6,300]$ lbs of goods (i.e., $Q_k = U\,[6,000; 6,300]\,\forall k \in K$ lbs).

The start of time windows at each node ($a_i, i \in N$) was estimated as $U[0; 660]$ hours, and duration of a time window at each node was assumed to be 10 hours. The travel time from node i to node j ($t_{ij}, i \in N, j \in N'$) was calculated as $U[5; 10]$ hours. However, the travel time from any node to the dummy depot was considered to be zero in order to reduce the considered decision problem to the open vehicle routing problem. The processing time at the supplier and manufacturer nodes ($tp_i, i \in N^s \cup N^m$) was considered to be $U[1; 3]$ hours, while the processing time at each customer node ($tp_i, i \in N^c$) was assumed to be $U[2; 8]$ hours. Note that the processing time at the depot and the dummy depot was zero. The precedence level of each node ($pl_i, i \in N$) was between 0 and 4. As mentioned before, binary parameter $b_{ij}^{sc}, i \in N^s, j \in N^c$ was applied to denote if supplier i had to be visited before visiting customer j. If the value of b_{ij}^{sc} is 1, then customer j is not allowed to be visited before the respective vehicle serves supplier i. The same principle was applied for binary parameter $b_{ij}^{mc}, i \in N^m, j \in N^c$. If the value of b_{ij}^{mc} is 1, then customer j is not allowed to be visited before the respective vehicle serves manufacturer i.

In the vehicle routing literature, the unit travel cost of a vehicle is assumed to be about 60 USD/hour [36]. However, in this study, the factory is installed on the vehicles, and special precautions are required to haul vehicles with factories. Furthermore, vehicles with "factories-in-a-box" often transport quite expensive production modules (e.g., medical products, medical devices, consumable parts of weapons, electronic devices). So, the unit travel cost of a vehicle ($c_k^v, k \in K$) in this case was assumed to be $U[900; 950]$ USD/hour. However, a sensitivity analysis will be conducted for the unit travel cost to determine how the unit travel cost may impact the routing of vehicles with "factories-in-a-box" (see section VI.D of the manuscript). The unit early arrival cost ($c_i^e, i \in N'$) and the unit late arrival cost ($c_i^l, i \in N'$) were assumed to be $U[0.1; 100]$ USD/hour. For the depot and the dummy depot, the unit early arrival cost and the unit late arrival cost were considered to be zero.

B. PARAMETER TUNING ANALYSIS

Before executing the considered metaheuristic algorithms for small-scale and large-scale problem instances, a parameter tuning analysis was conducted in order to estimate the appropriate parameter values of the algorithms from a pool of candidate values. A total of three candidate values were tested for each parameter, which corresponds to the 3^k factorial design, where k is the number of parameters for a given metaheuristic algorithm [58], [59]. The results from the parameter tuning analysis are underlined in Table 2. The four parameters of EA are the population size (*PopSize*), crossover probability (σ^c), mutation probability (σ^m), and the maximum number of generations (*gens*). The best values of *PopSize*, σ^c, σ^m, and *gens* were found to be 30, 0.50, 0.04, and 500, respectively, considering both solution quality and computational time perspectives. For the rest of the metaheuristic algorithms, a maximum of 2,000 iterations returned the best results (i.e., $iters = 2,000$). The number of neighborhood structures (Ψ^{VNS}) for VNS was tuned as 10. Apart from the maximum number of iterations (*iters*), TS has two other parameters, which are the number of solutions evaluated during the local search (Ψ^{TS}) and Tabu list size (Γ), whose best values were 7 and 15, respectively. Furthermore, three additional parameters are associated with SA, namely the initial temperature (T_0), temperature interval (ΔT), and normalizing

TABLE 2. The parameter tuning results for the considered metaheuristic algorithms.

Algorithm	Parameter Description	Candidate Values	Best Value
EA	Population size (*PopSize*)	[20; 30; 40]	30
	Crossover probability (σ^c)	[0.30; 0.40; 0.50]	0.50
	Mutation probability (σ^m)	[0.02; 0.03; 0.04]	0.04
	Maximum number of generations (*gens*)	[400; 500; 600]	500
VNS	Number of neighborhood structures (Ψ^{VNS})	[10; 15; 20]	10
	Maximum number of iterations (*iters*)	[1,500; 1,800; 2,000]	2,000
TS	Number of solutions evaluated during the local search (Ψ^{TS})	[5; 7; 10]	7
	Tabu list size (Γ)*	[10; 15; 20]	15
	Maximum number of iterations (*iters*)	[1,500; 1,800; 2,000]	2,000
SA	Initial temperature (T_0)	[2,100; 2,300; 2,500]	2,500
	Temperature interval (ΔT)	[0.30; 0.40; 0.50]	0.50
	Normalizing coefficient (ω)**	[0.10; 0.15; 0.20]	0.10
	Maximum number of iterations (*iters*)	[1,500; 1,800; 2,000]	2,000

*The Tabu list size is measured in terms of the number of solutions that can be stored in the Tabu list.
**The normalizing coefficient for Boltzmann selection was set as a percentage of the average solution fitness.

coefficient (ω), whose best values were found to be 2,500, 0.50, and 0.10, respectively.

C. COMPARISON WITH OTHER SOLUTION APPROACHES

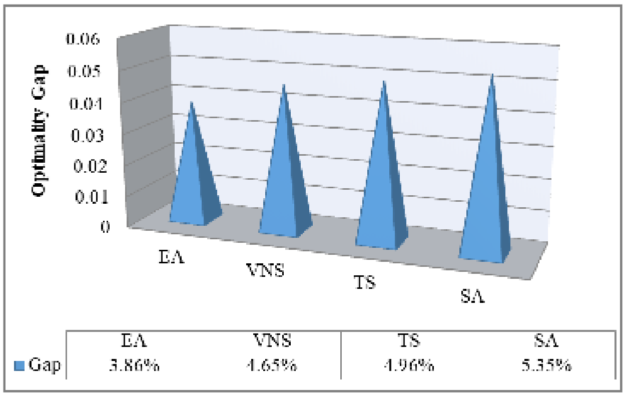
1) SMALL-SCALE PROBLEM INSTANCES

A total of 20 small-scale problem instances were generated to compare the computational performance of the exact solution approach employed in this study (i.e., CPLEX) and the considered metaheuristic algorithms (i.e., EA, VNS, TS, and SA). The number of customer nodes was increased from 3 nodes in small-scale problem instance #1 to 22 nodes in small-scale problem instance #20, with an increment of 1 node. The number of supplier nodes and the number of manufacturer nodes were the same as the number of customer nodes in each small-scale problem instance. Furthermore, the available number of vehicles was set to 20 vehicles for all the small-scale problem instances. The maximum CPU time of CPLEX was limited to 1 hour (3,600 seconds), and its target optimality gap was set to 1.00%. The average over 5 replications objective function values and CPU times of the tested solution approaches for small-scale problem instances are highlighted in Table 3. CPLEX generally returned better objective function values (average = 183,967 USD) among the considered solution approaches. However, its CPU times (average = 1,362.18 seconds) were substantially higher than the other solution approaches. The average objective function values of EA, VNS, TS, and SA comprised 191,069 USD, 192,514 USD, 193,085 USD, and 193,818 USD, respectively. Furthermore, among the metaheuristic algorithms, EA returned the best objective function values in all the small-scale problem instances. Note that the CPU times of

TABLE 3. **The objective function values and CPU times of different solution approaches for small-scale problem instances.**

Small-Scale Problem Instance	Number of Customer Nodes (nodes)	CPLEX		EA		VNS		TS		SA	
		Objective (USD)	CPU Time (sec)	Objective (USD)	CPU Time (sec)	Objective (USD)	CPU Time (sec)	Objective (USD)	CPU Time (sec)	Objective (USD)	CPU Time (sec)
1	3	**44,896**	0.09	45,008	35.34	45,120	21.44	45,562	16.62	45,631	2.55
2	4	**59,572**	0.20	59,961	45.53	60,750	28.70	61,021	21.49	61,086	3.15
3	5	**74,335**	0.42	74,978	57.42	75,960	37.00	76,300	30.46	76,394	3.88
4	6	**88,589**	0.73	89,297	67.48	90,454	43.65	90,640	31.98	90,935	4.64
5	7	**103,445**	1.25	105,857	78.77	106,277	50.92	106,405	37.40	107,533	5.57
6	8	**118,681**	2.59	121,194	88.85	123,173	62.78	123,411	42.32	124,650	6.08
7	9	**132,933**	8.81	136,509	100.14	137,724	69.35	138,034	46.84	138,349	6.87
8	10	**147,287**	26.05	151,986	111.57	153,451	71.71	153,876	52.53	154,197	7.61
9	11	**161,800**	85.75	167,155	121.32	168,671	78.61	168,713	59.06	170,604	8.47
10	12	**176,072**	154.83	181,901	131.88	184,138	90.60	184,292	65.70	185,319	9.11
11	13	**190,486**	548.61	197,512	143.16	198,656	92.15	198,860	67.35	200,635	10.71
12	14	**205,357**	563.59	213,927	152.98	214,054	99.71	215,348	76.70	215,977	10.71
13	15	**220,153**	647.28	229,201	163.31	231,994	107.31	232,161	77.04	232,543	11.25
14	16	**234,825**	3,600.36	245,251	176.13	246,994	112.96	247,813	83.32	247,830	12.03
15	17	**250,066**	3,600.36	261,607	185.15	262,052	120.14	262,725	87.23	263,843	12.67
16	18	**264,629**	3,600.41	273,742	194.89	276,610	127.45	278,696	101.35	279,713	13.40
17	19	**278,930**	3,600.47	291,064	206.40	294,499	137.50	295,085	97.71	296,023	14.28
18	20	**294,347**	3,600.53	309,505	219.22	311,145	155.82	311,737	104.44	312,714	15.06
19	21	**308,498**	3,600.58	324,976	233.19	326,527	160.67	327,452	114.37	328,048	15.75
20	22	**324,446**	3,600.67	340,746	238.48	342,028	155.73	343,567	113.50	344,342	16.30
Mean		**183,967**	1,362.18	191,069	137.56	192,514	91.21	193,085	66.37	193,818	9.50

***Bold font** is used to underline the best objective function values achieved.



	EA	VNS	TS	SA
■ Gap	3.86%	4.65%	4.96%	5.35%

FIGURE 11. **The average optimality gaps of different algorithms for small-scale problem instances.**

all the metaheuristic algorithms were less than 4 minutes (240 seconds). In particular, the average CPU times of EA, VNS, TS, and SA comprised 137.56 seconds, 91.21 seconds, 66.37 seconds, and 9.50 seconds, respectively.

The average optimality gaps for the metaheuristic algorithms are illustrated in Fig. 11, which demonstrates that the solutions provided by the metaheuristic algorithms were near-optimal (as suggested by CPLEX). Here, the optimality gap for an algorithm (e.g., algorithm x) was estimated as follows: $\boldsymbol{OptGap}_x = \frac{Z_x - Z_{CPLEX}}{Z_{CPLEX}}$, where $\boldsymbol{Z}_x$ is the objective function value for algorithm x, and $\boldsymbol{Z}_{CPLEX}$ is the optimal objective function value, which was obtained by CPLEX. For the considered small-scale problem instances, the average optimality gaps for EA, VNS, TS, and SA comprised 3.86%, 4.65%, 4.96%, and 5.35%, respectively. The optimality gaps of the metaheuristic algorithms could be reduced by increasing the maximum number of generations for EA and the maximum number of iterations for VNS, TS, and SA. In the meantime, the computational performance of CPLEX is expected to decline even further with increasing problem size due to the computational complexity of the **VRPFIB** mathematical model.

2) LARGE-SCALE PROBLEM INSTANCES

In order to compare the performance of the considered metaheuristic algorithms for large-scale problems, a total of 20 problem instances were generated, where the number of customer nodes was increased from 55 nodes to 75 nodes. The available number of vehicles was also varied between 15 vehicles and 30 vehicles. However, the number of supplier nodes and the number of manufacturer nodes were unchanged (10 supplier nodes and 10 manufacturer nodes in each problem instance). Note that CPLEX could not solve the problem instances with 50 nodes or more to the global optimality within the specified time limit. Therefore, for the large-scale problem instances, the computational performance of the metaheuristic algorithms only (i.e., EA, VNS, TS, and SA) was analyzed. The average over 5 replications objective function values and CPU times of the considered metaheuristic algorithms for large-scale problem instances are highlighted in Table 4, which reveals that EA returned the best objective function values in 15 out of 20 large-scale problem instances. The average objective function values of EA, VNS, TS, and SA were 1,654,988 USD, 1,688,129 USD, 1,689,691 USD, and 1,716,816 USD, respectively. Throughout the analysis of large-scale problem instances, EA outperformed VNS, TS, and SA by up to 8.86%, 8.99%, and 9.82%, respectively, in terms of the objective function values.

TABLE 4. The objective function values and CPU times of different solution approaches for large-scale problem instances.

Large-Scale Problem Instance	Number of Customer Nodes (nodes)	Available Number of Vehicles (vehicles)	EA		VNS		TS		SA	
			Objective (USD)	CPU Time (sec)	Objective (USD)	CPU Time (sec)	Objective (USD)	CPU Time (sec)	Objective (USD)	CPU Time (sec)
1	55	15	1,289,641	715.67	1,296,027	480.53	**1,287,483**	335.85	1,328,037	43.88
2	55	20	**1,350,563**	715.06	1,423,567	482.10	1,407,273	336.42	1,437,272	44.09
3	55	25	**1,435,379**	712.89	1,476,135	481.63	1,500,877	337.23	1,525,899	44.46
4	55	30	**1,442,434**	718.55	1,570,189	481.51	1,572,172	340.83	1,584,098	44.89
5	60	15	**1,363,144**	782.71	1,409,891	527.95	1,401,673	375.08	1,435,794	54.46
6	60	20	**1,505,319**	791.09	1,540,001	527.15	1,524,835	375.12	1,552,622	54.05
7	60	25	1,624,915	790.11	**1,616,346**	537.74	1,631,169	379.21	1,660,601	55.24
8	60	30	**1,621,518**	787.15	1,701,742	544.29	1,677,674	387.58	1,701,211	54.69
9	65	15	**1,484,364**	845.77	1,517,884	570.51	1,516,967	405.59	1,549,295	58.25
10	65	20	1,636,854	847.70	**1,615,865**	572.74	1,650,780	407.85	1,656,690	57.98
11	65	25	**1,686,508**	859.22	1,770,421	589.46	1,765,553	409.56	1,783,082	58.25
12	65	30	**1,791,250**	859.04	1,851,451	584.05	1,825,624	411.87	1,888,880	58.33
13	70	15	**1,611,918**	916.64	1,619,906	615.12	1,619,404	435.73	1,659,466	64.21
14	70	20	1,758,496	915.84	**1,739,672**	614.80	1,772,906	433.99	1,785,274	62.88
15	70	25	**1,846,379**	920.22	1,890,365	617.60	1,878,471	444.18	1,920,030	63.25
16	70	30	**1,956,778**	927.20	1,971,909	620.72	1,961,551	438.89	1,990,161	64.56
17	75	15	**1,719,287**	961.24	1,731,934	654.22	1,731,276	462.75	1,760,673	67.08
18	75	20	**1,878,129**	967.10	1,903,703	663.44	1,903,181	463.03	1,915,401	66.99
19	75	25	2,019,367	973.34	**2,006,719**	659.90	2,043,393	466.84	2,049,653	68.19
20	75	30	**2,077,512**	975.34	2,108,846	673.10	2,121,553	478.98	2,152,180	69.17
Mean			**1,654,988**	849.09	1,688,129	574.93	1,689,691	406.33	1,716,816	57.75

***Bold font** is used to underline the best objective function values achieved.

The superior performance of EA, when comparing to the other metaheuristic algorithms, can be justified by the fact that EA is a population-based algorithm, while VNS, TS, and SA are the single-solution-based algorithms. As a population-based algorithm, EA has the opportunity to start the search process considering different areas in the search space (as each solution represents a specific area of the search space); hence, EA has a higher chance to find the areas with higher quality solutions in the search space, as compared to the single-solution-based algorithms. Moreover, EA deploys a variety of algorithmic operators to better explore and exploit the search space. For instance, crossover helps EA to discover more areas in the search space, while parent selection and offspring selection facilitate EA to maintain the areas with high quality solutions throughout the search process. On the other hand, mutation helps EA to exploit the promising search space areas for superior solutions. The other metaheuristic algorithms (VNS, TS, and SA) start the search process considering just one solution, which significantly lowers their explorative capabilities. Among the single-solution-based algorithms, SA demonstrated the weakest performance, as it considers just one neighbor of the current solution in each iteration, while VNS and TS discover and evaluate several neighbors of the current solution throughout local search in each iteration. Hence, VNS and TS have higher chances to discover better solutions for the problem, when comparing to SA.

The average CPU times of EA, VNS, TS, and SA comprised 849.09 seconds, 574.93 seconds, 406.33 seconds, and 57.75 seconds, respectively. The higher CPU times recorded for EA, when comparing to VNS, TS, and SA, can be justified based on the following reasons: (1) EA is the population-based metaheuristic, while VNS, TS, and SA are the single-solution-based metaheuristics; (2) EA has more algorithmic operators as compared to VNS, TS, and SA (e.g., crossover, mutation, parent selection, offspring selection), and each one of them imposes certain computational efforts. The number of generations and iterations (the term "generation" is used for EA, while the term "iteration" is used for VNS, TS, and SA) can remarkably affect the computational time for each one of the algorithms. It was demonstrated by the parameter tuning analysis that EA could converge within 500 generations, while VNS, TS, and SA needed up to 2,000 iterations to converge successfully (see section VI.B). Hence, the population-based nature of EA reduced the number of generations to convergence as compared to the other algorithms. Among the single-solution-based algorithms, the lower CPU times of SA against VNS and TS can be explained by the fact that SA considers just one neighbor of the current solution in each iteration.

Fig. 12 demonstrates the convergence patterns of the considered metaheuristic algorithms for the 8 largest problem instances (i.e., large-scale problem instances #13 to #20), where the objective function values are plotted against the mapped number of generations (for EA) or iterations (for VNS, TS, and SA). Instead of showing the actual number of generations and iterations directly on the horizontal axis, they are mapped to the range [0; 1], as the number of generations and iterations differed for the considered algorithms. In particular, 500 generations were adopted for EA, while 2,000 iterations were adopted for VNS, TS, and SA. The following relationship was used to map each generation and iteration to the range [0; 1]: $MV_g^x = g/T^x$, $g \in [1; T^x]$, where MV_g^x is the mapped value of generation or iteration g for algorithm x, and

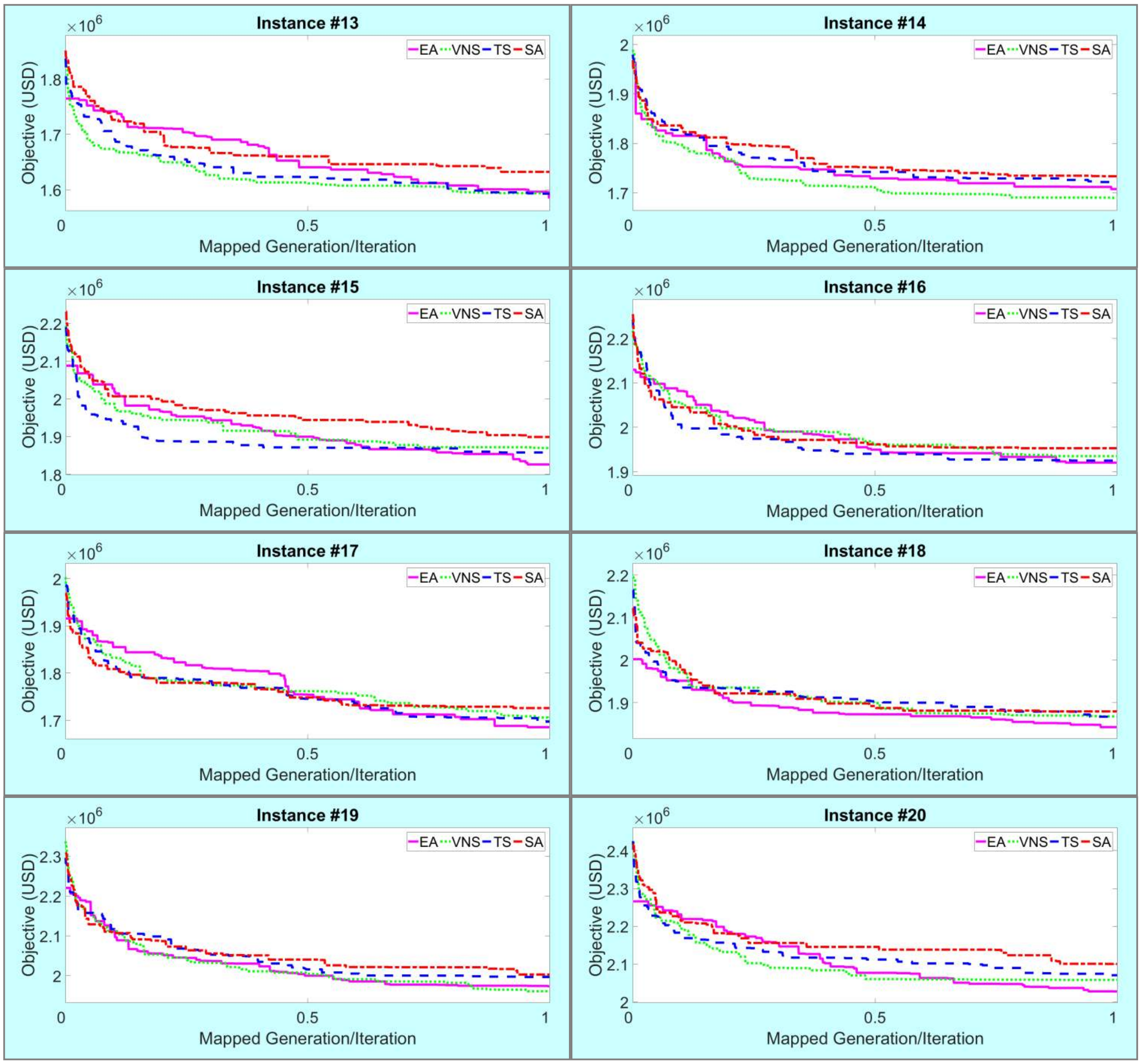


FIGURE 12. The convergence patterns of the considered metaheuristic algorithms for the largest problem instances.

T^x is the total number of generations or iterations, considered as the stopping criterion for algorithm x [60]. Fig. 12 reveals that, in most of the problem instances, EA was moving along the search space in a more effective manner, when comparing to VNS, TS, and SA. Such a finding can be supported by nature of the EA algorithm. As discussed earlier, EA is the population-based algorithm and has the opportunity to start the search process from different areas in the search space and explore the search space more effectively using a variety of algorithmic operators. On the other hand, VNS, TS, and SA are the single-solution-based algorithms and work with the current solution as well as its neighbor(s), which significantly lowers their explorative and exploitative capabilities.

All the considered metaheuristic algorithms are stochastic. It means that the metaheuristic algorithms may return different objective function values after each replication. However, the metaheuristic algorithms with low stochastic variations in the objective function values from one replication to another can be still viewed as reliable decision support tools [60], [61]. The scope of this study included the analysis of stochastic variations in the objective function values returned by the considered metaheuristic algorithms. In order to conduct the analysis, the coefficient of variation was estimated for each metaheuristic algorithm and each large-scale problem instance as follows: $CV_n^x = SD_n^x/M_n^x$, where CV_n^x, SD_n^x, and M_n^x are the coefficient of variation, the standard deviation, and the mean of the objective function values returned by algorithm x over 5 replications for problem instance n. Fig. 13 illustrates the boxplots of the coefficient of variation values for all the large-scale problem instances and

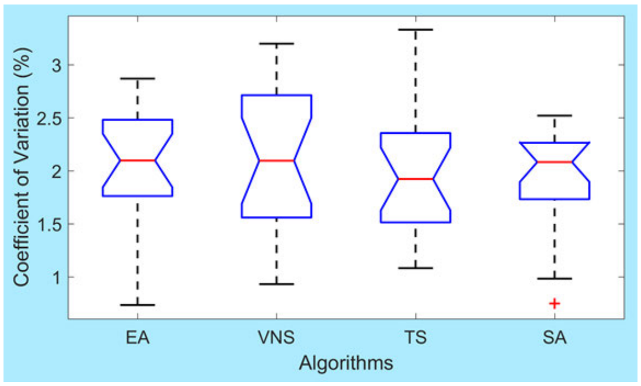


FIGURE 13. The coefficient of variation for the objective function values returned by the considered metaheuristics for large-scale problem instances.

all the performed replications for each one of the metaheuristic algorithms considered in this study. The results from the conducted analysis of the stochastic variations show that all the algorithms generally have a coefficient of variation lower than ≈3% over all the large-scale problem instances, which means that all the algorithms are reliable decision support tools. The average coefficient of variation values comprised 2.05%, 2.09%, 1.96%, and 1.94% for EA, VNS, TS, and SA, respectively.

Based on a set of analyses that were conducted for large-scale problem instances (including evaluation of the algorithmic objective function values, CPU time values, convergence patterns, and stochastic variations), this study recommends EA as the solution approach for the **VRPFIB** mathematical model.

D. MANAGERIAL INSIGHTS

Some significant managerial insights are outlined in this section of the manuscript. In particular, this section demonstrates how changes in the cost components and other parameters can affect the vehicle routing decisions provided by the developed EA algorithm for the **VRPFIB** mathematical model. Throughout the experiments, the largest problem instance (i.e., problem instance #20 with 75 customers, 10 suppliers, 10 manufacturers, and 30 vehicles available) was selected for the analysis of changes in the following parameters: (1) unit travel cost; (2) unit late arrival cost; and (3) fleet size (i.e., available number of vehicles).

In the sensitivity analysis for the unit travel cost, we generated 15 scenarios by increasing the unit travel cost from $U[50; 100]$ USD/hour in scenario #1 to $U[1,450; 1,500]$ USD/hour in scenario #15, with increments of 100 USD/hour. The other parameters, specified in Table 1, were not changed. Changes in the total travel time, the total late arrival time, and the total vehicle utilization (i.e., percentage of vehicles utilized from the vehicle fleet), due to increments in the unit travel cost, are illustrated in Fig. 14. When the unit travel cost was increased, the total travel time was decreased by utilizing fewer vehicles from the vehicle fleet, in order to reduce the total travel cost. The sensitivity analysis for the unit travel cost also revealed that a reduction in the total travel time

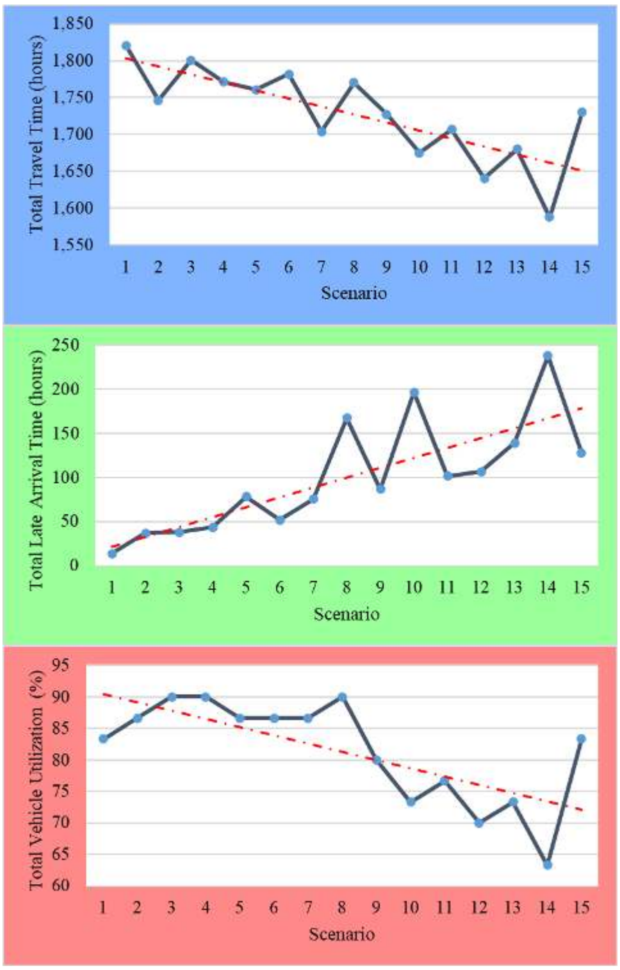


FIGURE 14. Sensitivity of the total travel time, total late arrival time, and total vehicle utilization to the unit travel cost.

further led to an increase in the total late arrival time at supplier, manufacturer, and customer nodes. Moreover, increments in the unit travel cost resulted in an increase in the total cost (i.e., the objective of the **VRPFIB** mathematical model) from 617,207 USD in scenario #1 to 3,036,622 USD in scenario #15. Note that some fluctuations in the patterns of different variables from one scenario to another can be explicated by the complexity of the **VRPFIB** mathematical model and a significant number of variables for the considered problem instance (i.e., changes in the unit travel cost triggered changes in the values of other variables as well).

In the sensitivity analysis for the unit late arrival cost, we generated 15 scenarios by increasing the unit late arrival cost from $U[5; 10]$ USD/hour in scenario #1 to $U[705; 710]$ USD/hour in scenario #15, with increments of 50 USD/hour. The other parameters, specified in Table 1, were not changed. Changes in the total travel time, the total late arrival time, and the total vehicle utilization, due to increments in the unit late arrival cost, are illustrated in Fig. 15. When the unit late arrival cost was increased, the total late arrival time at supplier, manufacturer, and customer nodes was decreased by utilizing more vehicles from the vehicle fleet, in order to

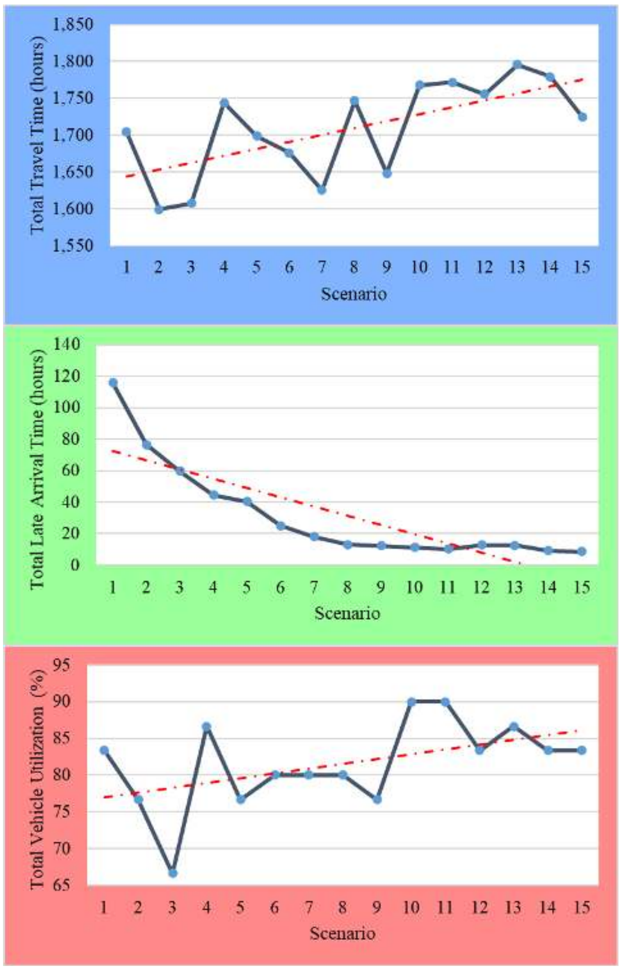


FIGURE 15. Sensitivity of the total travel time, total late arrival time, and total vehicle utilization to the unit late arrival cost.

reduce the total late arrival cost. The sensitivity analysis for the unit late arrival cost also revealed that a reduction in the total late arrival time further led to an increase in the total travel time. Moreover, increments in the unit late arrival cost resulted in an increase in the total cost (i.e., the objective of the **VRPFIB** mathematical model) from 2,032,115 USD in scenario #1 to 2,090,754 USD in scenario #15. Note that some fluctuations in the patterns of different variables from one scenario to another can be explicated by the complexity of the **VRPFIB** mathematical model and a significant number of variables for the considered problem instance (i.e., changes in the unit late arrival cost triggered changes in the values of other variables as well).

In the sensitivity analysis for the fleet size, we generated 15 scenarios by increasing the fleet size from 10 vehicles in scenario #1 to 52 vehicles in scenario #15, with increments of 3 vehicles. The other parameters, specified in Table 1, were not changed. Changes in the total travel time, the total late arrival time, the total vehicle utilization, and the average number of nodes visited by the utilized vehicles, due to increments in the available number of vehicles (i.e., fleet size), are illustrated in Fig. 16. When the fleet size was increased, the percentage of vehicles utilized from the vehicle fleet (i.e., total

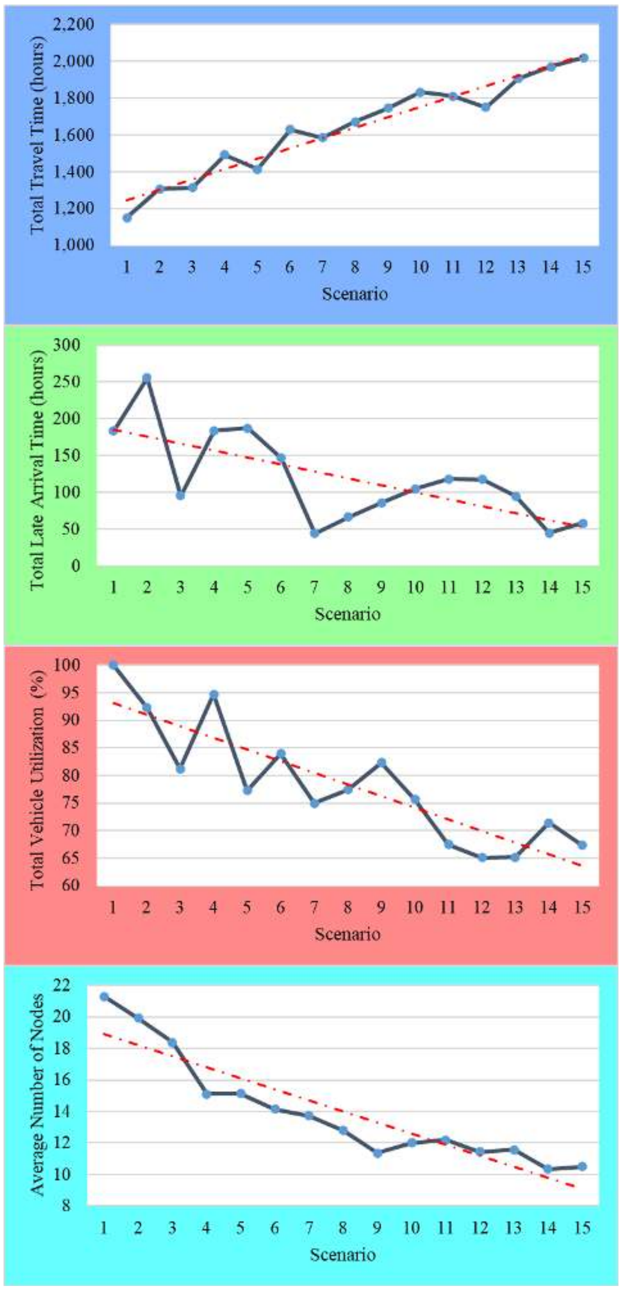


FIGURE 16. Sensitivity of the total travel time, total late arrival time, total vehicle utilization, and average number of nodes visited by the utilized vehicles to the fleet size.

vehicle utilization) decreased. However, the total number of utilized vehicles generally increased from one scenario to another. After increasing the fleet size, the average number of nodes visited by the utilized vehicles decreased, while the total travel time increased. Most importantly, increasing the fleet size led to a reduction in the total late arrival time. Moreover, increments in the fleet size resulted in an increase in the total cost (i.e., the objective of the **VRPFIB** mathematical model) from 1,499,350 USD in scenario #1 to 2,318,095 USD in scenario #15, since the total number of utilized vehicles generally increased from one scenario to another. Note that some fluctuations in the patterns of different variables from one scenario to another can be explicated by the complexity of the **VRPFIB** mathematical model and a significant number of variables for the considered problem instance (i.e., changes in the fleet size triggered changes in the values of other variables as well).

VII. CONCLUSIONS

The "factory-in-a-box" concept could be very useful to meet the urgent demand, as it involves assembling factories in containers and then transporting the containers by the designated vehicles to the locations, where the products are needed the most. This study addressed the "factory-in-a-box" planning problem and focused on the transport of sub-assembly modules between suppliers and manufacturers by assigning vehicles to locations, deciding the order of visits for suppliers, manufacturers, and customers, and selecting the appropriate routes within the transportation network.

The problem was formulated as the open capacitated vehicle routing problem with soft time windows and heterogeneous vehicle fleet, aiming to minimize the total supply chain cost. The proposed mathematical formulation allowed capturing complex supplier-to-customer and manufacturer-to-customer relationships that are common for the vehicle routing with a "factory-in-a-box". Due to the problem complexity, a customized nature-inspired Evolutionary Algorithm (EA) was developed to solve the problem. A set of numerical experiments were performed to assess the computational performance of the developed solution algorithm against the exact optimization method (CPLEX) and alternative metaheuristics (Variable Neighborhood Search – VNS, Tabu Search – TS, and Simulated Annealing – SA).

A case study of a vaccination project involving "factories-in-a-box" was considered throughout the numerical experiments. The analysis of small-scale problem instances revealed that EA had the lowest optimality gaps, when comparing to VNS, TS, and SA. Moreover, the CPU time of CPLEX was exponentially increasing with the problem size. The analysis of large-scale problem instances revealed that EA outperformed VNS, TS, and SA by up to 8.86%, 8.99%, and 9.82%, respectively, in terms of the objective function values. A detailed analysis of the algorithmic convergence patterns and objective function stochastic variations confirmed superiority of EA against the alternative metaheuristic algorithms. A set of important managerial insights were drawn using the developed EA algorithm and the proposed mathematical model. It was found that an increase in the unit travel cost reduced the total travel time and the total vehicle utilization but increased the total late arrival time. On the other hand, an increase in the unit late arrival cost reduced the total late arrival time but increased the total travel time and the total vehicle utilization. Furthermore, the numerical experiments showed that the total late arrival time could be effectively reduced by increasing the vehicle fleet size.

Therefore, the proposed mathematical model and solution algorithm could be instrumental to the supply chain stakeholders who rely on the "factory-in-a-box" concept. In particular, the proposed methodology could assist meeting the urgent demand due to natural disasters (e.g., hurricanes, floods, earthquakes) or epidemics (e.g., COVID-19) as well as serve as an effective planning tool for military applications (e.g., transport of consumable parts for weapons during a war). The avenues of future research may include the following:

- Consideration of the alternative metaheuristic algorithms for the proposed mathematical model (e.g., Whale Optimization Algorithm, Particle Swarm Optimization, Ant Colony Optimization);
- Modeling the assignment of raw materials to suppliers, sub-assembly decomposition, assignment of sub-assembly modules to manufacturers, and assignment of tasks to manufacturers;
- Multiple depots for deployment of vehicles could be incorporated;
- Consideration of conflicting objectives throughout vehicle routing with a "factory-in-a-box" in multi-objective settings (e.g., minimize the total travel time vs. maximize the total customer satisfaction);
- Stochastic stocks and processing times could be considered for suppliers and manufacturers;
- Game theory could be applied to determine the set of suppliers and manufacturers for a given customer;
- Electric vehicles and the associated features (e.g., recharge strategy) could be studied for vehicle routing with a "factory-in-a-box";
- Modeling alternative "factory-in-a-box" operations, where the vehicles with "factories-in-a-box" travel directly to customer sites, while other vehicles (e.g., 3rd party contractors) are responsible for delivery of supplies to customer sites.

REFERENCES

[1] X. Yue and Y. Chen, "Strategy optimization of supply chain enterprises based on fuzzy decision making model in Internet of Things," *IEEE Access*, vol. 6, pp. 70378–70387, 2018.

[2] M. A. Dulebenets, "A comprehensive evaluation of weak and strong mutation mechanisms in evolutionary algorithms for truck scheduling at cross-docking terminals," *IEEE Access*, vol. 6, pp. 65635–65650, 2018.

[3] D.-Y. Lin, C.-J. Juan, and C.-C. Chang, "Managing food safety with pricing, contracts and coordination in supply chains," *IEEE Access*, vol. 7, pp. 150892–150909, 2019.

[4] F. Liu, M. Gui, C. Yi, and Y. Lan, "A fast decomposition and reconstruction framework for the pickup and delivery problem with time windows and LIFO loading," *IEEE Access*, vol. 7, pp. 71813–71826, 2019.

[5] K. Peng, J. Du, F. Lu, Q. Sun, Y. Dong, P. Zhou, and M. Hu, "A hybrid genetic algorithm on routing and scheduling for vehicle-assisted multi-drone parcel delivery," *IEEE Access*, vol. 7, pp. 49191–49200, 2019.

[6] P. Rajagopal, S. Zailani, and M. Sulaiman, "Assessing the effectiveness of supply chain partnering with scalable partnering as a moderator," *Int. J. Phys. Distrib. Logistics Manage.*, vol. 39, no. 8, pp. 649–668, Sep. 2009.

[7] E. Alarcon-Gerbier and U. Buscher, "Integrated scheduling of production and distribution operations with site selection," in *Logistics Management*, C. Bierwirth, T. Kirschstein, and D. Sackmann, Eds. Cham, Switzerland: Springer, 2019, pp. 255–267. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-030-29821-0_17

[8] GE. *KUBio Modular Biomanufacturing Environments*. Accessed: Dec. 6, 2019. [Online]. Available: https://www.gelifesciences.com/en/us/solutions/bioprocessing/products-and-solutions/enterprise-solutions/kubio,

[9] GE. *GE's New Factory-in-a-Box Accelerates the Production of Viral Vector-Based Therapeutics*. Accessed: Dec. 6, 2019. [Online]. Available: https://www.genewsroom.com/press-releases/ge%E2%80%99s-new-factory-boxaccelerates-production-viral-vector-based-therapeutics-284321/

[10] Nokia. *Nokia Showcases Factory in a Box 2.0 With Global IoT Connectivity, Improved Reliability and Security at Hannover Messe 2019*. Accessed: Dec. 6, 2019. [Online]. Available: https://www.nokia.com/about-us/news/releases/2019/04/01/nokia-showcases-factory-in-a-box-20-with-global-iot-connectivity-improved-reliability-and-security-at-hannover-messe-2019/

[11] Johns Hopkins University. *COVID-19 United States Cases by County*. Accessed: May 5, 2020. [Online]. Available: https://coronavirus.jhu.edu/us-map

[12] J. Mula, D. Peidro, M. Díaz-Madroñero, and E. Vicens, "Mathematical programming models for supply chain production and transport planning," *Eur. J. Oper. Res.*, vol. 204, no. 3, pp. 377–390, Aug. 2010.

[13] J. A. Orjuela-Castro, L. A. Sanabria-Coronado, and A. M. Peralta-Lozano, "Coupling facility location models in the supply chain of perishable fruits," *Res. Transp. Bus. Manage.*, vol. 24, pp. 73–80, Sep. 2017.

[14] A. Jabbarzadeh, M. Haughton, and F. Pourmehdi, "A robust optimization model for efficient and green supply chain planning with postponement strategy," *Int. J. Prod. Econ.*, vol. 214, pp. 266–283, Aug. 2019.

[15] M. Bengtsson, S. W. Elfving, and M. Jackson, "The factory-in-a-box concept and its maintenance application," presented at the 19th Int. Congr. Condition Monit. Diagnostic Eng. Manage., Luleå, Sweden, Jun. 2006.

[16] M. Jackson and A. Zaman, "Factory-in-a-box-mobile production capacity on demand," *Int. J. Mod. Eng. Res. Technol.*, vol. 8, no. 1, pp. 12–26, 2007.

[17] M. Jackson, M. Wiktorsson, and M. Bellgran, "Factory-in-a-box—Demonstrating the next generation manufacturing provider," in *Manufacturing Systems and Technologies for the New Frontier*, M. Mitsuishi, K. Ueda, and F. Kimura, Eds. London, U.K.: Springer, 2008, pp. 341–346. [Online]. Available: https://link.springer.com/chapter/10.1007/978-1-84800-267-8_70

[18] Z. Jiang, H. Wang, Q. Tian, and W. Guo, "Co-design of supply chain network and subassembly planning considering the reconfiguration of supply chain structure for factory-in-a-box manufacturing," presented at the 13th Int. Manuf. Sci. Eng. Conf., College Station, TX, USA, Jun. 2018. [Online]. Available: https://asmedigitalcollection.asme.org/MSEC/proceedings/MSEC2018/51371/V003T02A015/274043

[19] B. Eksioglu, A. V. Vural, and A. Reisman, "The vehicle routing problem: A taxonomic review," *Comput. Ind. Eng.*, vol. 57, no. 4, pp. 1472–1483, Nov. 2009.

[20] K. Braekers, K. Ramaekers, and I. Van Nieuwenhuyse, "The vehicle routing problem: State of the art classification and review," *Comput. Ind. Eng.*, vol. 99, pp. 300–313, Sep. 2016.

[21] V. F. Yu, P. Jewpanya, and A. A. N. P. Redi, "Open vehicle routing problem with cross-docking," *Comput. Ind. Eng.*, vol. 94, pp. 6–17, Apr. 2016.

[22] R. Atefi, M. Salari, L. C. Coelho, and J. Renaud, "The open vehicle routing problem with decoupling points," *Eur. J. Oper. Res.*, vol. 265, no. 1, pp. 316–327, Feb. 2018.

[23] J. Brandão, "Iterated local search algorithm with ejection chains for the open vehicle routing problem with time windows," *Comput. Ind. Eng.*, vol. 120, pp. 146–159, Jun. 2018.

[24] Y. Niu, Z. Yang, P. Chen, and J. Xiao, "Optimizing the green open vehicle routing problem with time windows by minimizing comprehensive routing cost," *J. Cleaner Prod.*, vol. 171, pp. 962–971, Jan. 2018.

[25] E. Ruiz, V. Soto-Mendoza, A. E. R. Barbosa, and R. Reyes, "Solving the open vehicle routing problem with capacity and distance constraints with a biased random key genetic algorithm," *Comput. Ind. Eng.*, vol. 133, pp. 207–219, Jul. 2019.

[26] N. Helal, F. Pichon, D. Porumbel, D. Mercier, and É. Lefèvre, "The capacitated vehicle routing problem with evidential demands," *Int. J. Approx. Reasoning*, vol. 95, pp. 124–151, Apr. 2018.

[27] A. Gutierrez, L. Dieulle, N. Labadie, and N. Velasco, "A hybrid metaheuristic algorithm for the vehicle routing problem with stochastic demands," *Comput. Oper. Res.*, vol. 99, pp. 135–147, Nov. 2018.

[28] M. Salavati-Khoshghalb, M. Gendreau, O. Jabali, and W. Rei, "An exact algorithm to solve the vehicle routing problem with stochastic demands under an optimal restocking policy," *Eur. J. Oper. Res.*, vol. 273, no. 1, pp. 175–189, Feb. 2019.

[29] S. Huber and M. J. Geiger, "Order matters—A variable neighborhood search for the swap-body vehicle routing problem," *Eur. J. Oper. Res.*, vol. 263, no. 2, pp. 419–445, Dec. 2017.

[30] H. Hojabri, M. Gendreau, J.-Y. Potvin, and L.-M. Rousseau, "Large neighborhood search with constraint programming for a vehicle routing problem with synchronization constraints," *Comput. Oper. Res.*, vol. 92, pp. 87–97, Apr. 2018.

[31] T. Liu, Z. Luo, H. Qin, and A. Lim, "A branch-and-cut algorithm for the two-echelon capacitated vehicle routing problem with grouping constraints," *Eur. J. Oper. Res.*, vol. 226, pp. 467–487, Apr. 2018.

[32] M. Qiu, Z. Fu, R. Eglese, and Q. Tang, "A tabu search algorithm for the vehicle routing problem with discrete split deliveries and pickups," *Comput. Oper. Res.*, vol. 100, pp. 102–116, Dec. 2018.

[33] Z. Xu, A. Elomri, S. Pokharel, and F. Mutlu, "A model for capacitated green vehicle routing problem with the time-varying vehicle speed and soft time windows," *Comput. Ind. Eng.*, vol. 137, Nov. 2019, Art. no. 106011.

[34] H. Zhang, Q. Zhang, L. Ma, Z. Zhang, and Y. Liu, "A hybrid ant colony optimization algorithm for a multi-objective vehicle routing problem with flexible time windows," *Inf. Sci.*, vol. 490, pp. 166–190, Jul. 2019.

[35] Ç. Koç, T. Bekta , O. Jabali, and G. Laporte, "A hybrid evolutionary algorithm for heterogeneous fleet vehicle routing problems with time windows," *Comput. Oper. Res.*, vol. 64, pp. 11–27, Dec. 2015.

[36] H. Bae and I. Moon, "Multi-depot vehicle routing problem with time windows considering delivery and installation vehicles," *Appl. Math. Model.*, vol. 40, nos. 13–14, pp. 6536–6549, Jul. 2016.

[37] M. Keskin and B. Çatay, "Partial recharge strategies for the electric vehicle routing problem with time windows," *Transp. Res. C, Emerg. Technol.*, vol. 65, pp. 111–127, Apr. 2016.

[38] A. Subramanyam, A. Wang, and C. E. Gounaris, "A scenario decomposition algorithm for strategic time window assignment vehicle routing problems," *Transp. Res. B, Methodol.*, vol. 117, pp. 296–317, Nov. 2018.

[39] C. Expósito-Izquierdo, A. Rossi, and M. Sevaux, "A two-level solution approach to solve the clustered capacitated vehicle routing problem," *Comput. Ind. Eng.*, vol. 91, pp. 274–289, Jan. 2016.

[40] H. Li, L. Zhang, T. Lv, and X. Chang, "The two-echelon time-constrained vehicle routing problem in linehaul-delivery systems," *Transp. Res. B, Methodol.*, vol. 94, pp. 169–188, Dec. 2016.

[41] L. Bertazzi and N. Secomandi, "Faster rollout search for the vehicle routing problem with stochastic demands and restocking," *Eur. J. Oper. Res.*, vol. 270, no. 2, pp. 487–497, Oct. 2018.

[42] Z. Wang, X. Ren, Z. Ji, W. Huang, and T. Wu, "A novel bio-heuristic computing algorithm to solve the capacitated vehicle routing problem based on Adleman–Lipton model," *Biosystems*, vol. 184, Oct. 2019, Art. no. 103997.

[43] V. Leggieri and M. Haouari, "A matheuristic for the asymmetric capacitated vehicle routing problem," *Discrete Appl. Math.*, vol. 234, pp. 139–150, Jan. 2018.

[44] D. Sacramento, D. Pisinger, and S. Ropke, "An adaptive large neighborhood search metaheuristic for the vehicle routing problem with drones," *Transp. Res. C, Emerg. Technol.*, vol. 102, pp. 289–315, May 2019.

[45] M. Avci and S. Topaloglu, "A hybrid metaheuristic algorithm for heterogeneous vehicle routing problem with simultaneous pickup and delivery," *Expert Syst. Appl.*, vol. 53, pp. 160–171, Jul. 2016.

[46] J. Euchi, "The vehicle routing problem with private fleet and multiple common carriers: Solution with hybrid metaheuristic algorithm," *Veh. Commun.*, vol. 9, pp. 97–108, Jul. 2017.

[47] E. Jabir, V. V. Panicker, and R. Sridharan, "Design and development of a hybrid ant colony-variable neighbourhood search algorithm for a multi-depot green vehicle routing problem," *Transp. Res. D, Transp. Environ.*, vol. 57, pp. 442–457, Dec. 2017.

[48] M. Alinaghian and N. Shokouhi, "Multi-depot multi-compartment vehicle routing problem, solved by a hybrid adaptive large neighborhood search," *Omega*, vol. 76, pp. 85–99, Apr. 2018.

[49] A. M. Altabeeb, A. M. Mohsen, and A. Ghallab, "An improved hybrid firefly algorithm for capacitated vehicle routing problem," *Appl. Soft Comput.*, vol. 84, Nov. 2019, Art. no. 105728.

[50] A. E. Eiben and J. E. Smith, *Introduction to Evolutionary Computing*. Berlin, Germany: Springer-Verlag, 2015. [Online]. Available: http://www.springer.com/us/book/ 9783662448731

[51] M. A. Dulebenets, "A delayed start parallel evolutionary algorithm for just-in-time truck scheduling at a cross-docking facility," *Int. J. Prod. Econ.*, vol. 212, pp. 236–258, Jun. 2019.

[52] M. A. Dulebenets, "An adaptive island evolutionary algorithm for the berth scheduling problem," *Memetic Comput.*, vol. 12, no. 1, pp. 51–72, Aug. 2019.

[53] P. Hansen, C. O uz, and N. Mladenović, "Variable neighborhood search for minimum cost berth allocation," *Eur. J. Oper. Res.*, vol. 191, no. 3, pp. 636–649, Dec. 2008.
[54] J.-F. Cordeau, G. Laporte, P. Legato, and L. Moccia, "Models and tabu search heuristics for the berth-allocation problem," *Transp. Sci.*, vol. 39, no. 4, pp. 526–538, Nov. 2005.
[55] S. Emde and N. Boysen, "Berth allocation in container terminals that service feeder ships and deep-sea vessels," *J. Oper. Res. Soc.*, vol. 67, no. 4, pp. 551–563, Apr. 2016.
[56] Universal. *NORM-JECT and HENKE-JECT Plastic Syringe Specifications*. Accessed: Mar. 1, 2020. [Online]. Available: http://universal-ac.com/img/SyringeSpec.pdf
[57] Allen County. *A Guide to Truck Trailers*. Accessed: Mar. 1, 2020. [Online]. Available: https://www.allencounty.us/homeland/images/lepc/docs/TruckTrailerGuide.pdf
[58] M. A. Dulebenets, "Application of evolutionary computation for berth scheduling at marine container terminals: Parameter tuning versus parameter control," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 1, pp. 25–37, Jan. 2018.
[59] M. Dulebenets, "A diploid evolutionary algorithm for sustainable truck scheduling at a cross-docking facility," *Sustainability*, vol. 10, no. 5, p. 1333, Apr. 2018.
[60] M. Kavoosi, M. A. Dulebenets, O. F. Abioye, J. Pasha, O. Theophilus, H. Wang, R. Kampmann, and M. Mikijeljević, "Berth scheduling at marine container terminals: A universal island-based metaheuristic approach," *Marit. Bus. Rev.*, vol. 5, no. 1, pp. 30–66, Nov. 2019.
[61] M. A. Dulebenets, M. Kavoosi, O. Abioye, and J. Pasha, "A self-adaptive evolutionary algorithm for the berth scheduling problem: Towards efficient parameter control," *Algorithms*, vol. 11, no. 7, p. 100, Jul. 2018.

JUNAYED PASHA received the bachelor's degree in civil engineering from the Khulna University of Engineering & Technology, Khulna, Bangladesh, and the Ph.D. degree in civil engineering with concentration in transportation from Florida State University, Tallahassee, FL, USA, in 2020. He is currently a Postdoctoral Research Associate with the Department of Civil and Environmental Engineering, College of Engineering, Florida A&M University-Florida State University (FAMU-FSU). His research interests include, but are not limited to, operations research, optimization, simulation modeling, supply chain management, transportation systems, transportation safety, transportation economics, industrial engineering, and natural hazard preparedness. He is a Licensed Professional Engineer in Bangladesh.

MAXIM A. DULEBENETS (Member, IEEE) received the B.S. and M.Sc. degrees in railway construction from the Moscow State University of Railroad Engineering, Moscow, Russia, and the M.Sc. and Ph.D. degrees in civil engineering with concentration in transportation from the University of Memphis, Memphis, TN, USA. He is currently an Assistant Professor with the Department of Civil and Environmental Engineering, College of Engineering, Florida A&M University-Florida State University (FAMU-FSU). His research interests include, but are not limited to, operations research, simulation modeling, optimization, NP-hard problems, liner shipping scheduling, evolutionary computation, mathematical programming, hybrid algorithms, metaheuristics, transportation engineering, and GPS data processing. He is actively involved in activities of more than ten Standing Committees of the Transportation Research Board (TRB) of the National Academies of Sciences, Engineering, and Medicine. He is an Invited Member of the TRB Standing Committee on International Trade and Transportation (AT020). He is an Affiliated Member of the INFORMS Optimization Society.

MASOUD KAVOOSI received the B.S. degree in civil engineering from Bu-Ali Sina University, Hamedan, Iran, in 2009, the M.S. degree in civil engineering with concentration in hydraulic structures from the Iran University of Science and Technology, Tehran, Iran, in 2014, and the Ph.D. degree in civil engineering with concentration in transportation from Florida State University, Tallahassee, in 2019. He is currently an Operations Research Engineer at HNTB Corporation, Tallahassee, FL, USA. His research interests include, but are not limited to, operations research, optimization, heuristic and metaheuristic algorithms, supply chains and logistics, simulation modeling, transportation systems, freight terminal design, and operations modeling. He has been a registered Professional Engineer in Tehran, since 2014.

OLUMIDE F. ABIOYE received the B.Eng. degree in civil engineering from the Federal University of Technology, Akure, Nigeria, and the M.Sc. and Ph.D. degrees in civil engineering with concentration in transportation from Florida A&M University, Tallahassee, FL, USA. He is currently an Operations Improvement Analyst at Airbus Group Inc., Ashburn, VA, USA. His research interests include, but are not limited to, operations research, freight transportation, liner shipping scheduling, mathematical programming, optimization, simulation modeling, transportation engineering, traffic safety and operations, and hazard preparedness. He is actively involved in activities of more than five Standing Committees of the Transportation Research Board of the National Academies of Sciences, Engineering, and Medicine.

HUI WANG received the B.S. degree in mechanical engineering from Shanghai Jiao Tong University, Shanghai, China, the M.S. degree in mechanical engineering from the University of Michigan, Ann Arbor, MI, USA, and the Ph.D. degree in industrial engineering from the University of South Florida, Tampa, FL, USA. He was a Research Scientist and an Intermittent Lecturer at the University of Michigan. He is currently an Assistant Professor with the Department of Industrial and Manufacturing Engineering, College of Engineering, Florida A&M University-Florida State University (FAMU-FSU). His research interests include, but are not limited to, manufacturing process monitoring/diagnosis/design/control/automation (with applications to automotive manufacturing), manufacturing system design and optimization, process control, and informatics for advanced materials.

WEIHONG (GRACE) GUO received the B.S. degree in industrial engineering from Tsinghua University, Beijing, China, and the Ph.D. degree in industrial and operations engineering from the University of Michigan, Ann Arbor, MI, USA. She is currently an Assistant Professor with the Department of Industrial and Systems Engineering, Rutgers, The State University of New Jersey. Her research interests include, but are not limited to, data mining for manufacturing and healthcare systems modeling, statistical quality control and process monitoring, quality-oriented design and modeling of complex manufacturing systems, and data fusion methods in the interface between applied statistics and system control/optimization. She is a member of IIE, INFORMS, and ASME. She was a recipient of the 2014 ISERC Quality Control and Reliability Engineering Best Student Paper Award Finalist, the 2014 International Conference on Frontiers of Design and Manufacturing Sciences Best Paper Award, the Rackham Predoctoral Fellowship from the University of Michigan, and the Wilson Prize for the Best Student Paper.