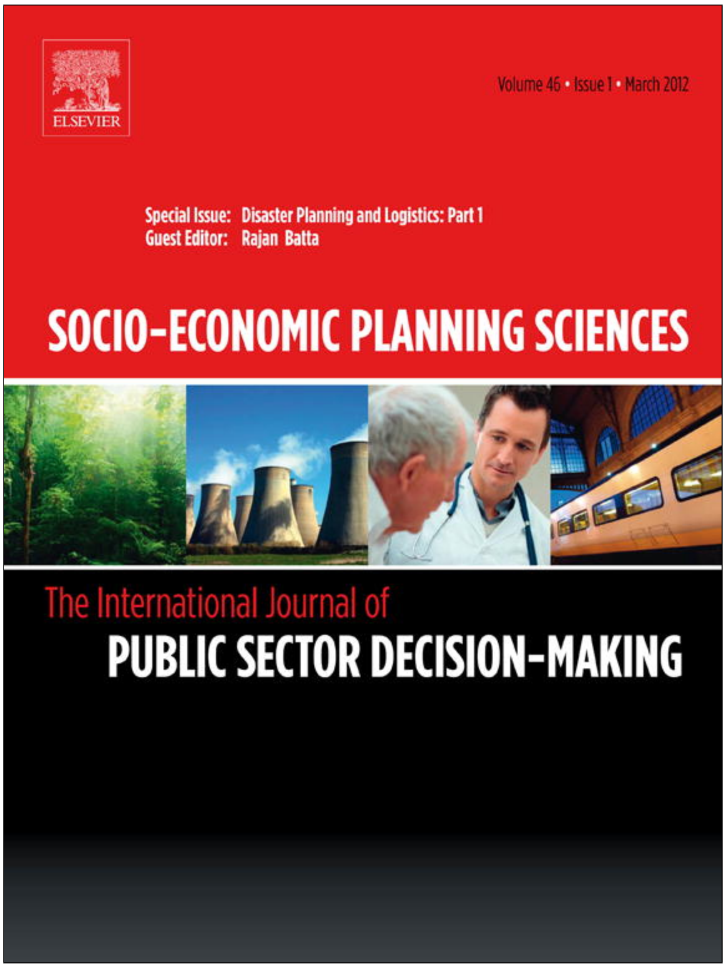
ELSEVIER
Volume 46 • Issue 1 • March 2012
Special Issue: Disaster Planning and Logistics: Part 1
Guest Editor: Rajan Batta
SOCIO-ECONOMIC PLANNING SCIENCES
The International Journal of
PUBLIC SECTOR DECISION-MAKING

Socio-Economic Planning Sciences 46 (2012) 23–32

Contents lists available at ScienceDirect

Socio-Economic Planning Sciences

journal homepage: www.elsevier.com/locate/seps

Transportation in disaster response operations

Djamel Berkoune [a], Jacques Renaud [a,b,*], Monia Rekik [a,b], Angel Ruiz [a,b]

[a] Interuniversity Research Center on Enterprise Networks, Logistics and Transportation (CIRRELT), Canada
[b] Faculté des Sciences de l'administration, Laval University, Quebec G1K 7P4, Canada

ARTICLE INFO

Article history:
Available online 13 June 2011

Keywords:
Transportation
Emergency logistics
Heuristics
Genetic algorithms

ABSTRACT

Disasters are extraordinary situations that require significant logistical deployment to transport equipment and humanitarian goods in order to help and provide relief to victims. An efficient response helps to reduce the social, economic and environmental impacts. In this paper, we define and formulate a practical transportation problem often encountered by crisis managers in emergency situations. Since optimal solutions to such a formulation may be achieved only for very small-size instances, we developed an efficient genetic algorithm to deal with realistic situations. This algorithm produces near optimal solutions in relatively short computation times and is fast enough to be used interactively in a decision-support system, providing high-quality transportation plans to emergency managers.

1. Introduction

Disasters, be they anthropogenic or natural, have always affected humans. Today, with the progress in information technology, disasters are followed live throughout the planet. The accumulation of risks associated with such factors as increasing urbanization, dependence on critical infrastructure, terrorism, climate change and variability, and animal and human diseases, as well as the greater mobility of people and goods around the world, have increased the potential for various types of disasters. These disasters (e.g., floods, earthquakes, chemical spills, plant explosions) require significant logistical deployment in order to provide relief to victims and to transport equipment and humanitarian goods.

According to Altay and Green [1], who reviewed OR/MS research in disaster operations management, there is increasing recognition of the need for research in this area. Though the field is not yet well established, *emergency logistics* can be defined as "a process of planning, managing and controlling the efficient flows of relief, information, and services from the points of origin to the points of destination to meet the urgent needs of the affected people under emergency conditions" [28,29]. Emergency management is divided into four main phases: mitigation, preparedness, response and recovery [1,12]. *Mitigation* is defined as sustained action to reduce or eliminate the risk to people and property from hazards and their effects. *Preparedness* is the set of measures taken to avoid the negative consequences associated with a threat, which includes actions taken to prepare efficient response during a crisis or emergency. *Response* is using resources and emergency procedures as dictated by emergency plans to preserve life, property, the environment, and the community's social, economic, and political structure. *Recovery* involves the long-term actions taken to stabilize the community and to restore normalcy after the disaster's immediate impact has passed. Again, according to Altay and Green [1], nearly half of the research concerned mitigation.

In this paper, we focus on one of the most important aspects of the response phase: the transportation of humanitarian aid (e.g., water, food, medical goods and survival equipment) to people at fixed distribution points. To this end, we propose a formal definition and a mathematical model for the *Transportation Problem in Disaster Response Operations* (TP-DRO). Three solution approaches are proposed to solve this problem. The first approach uses the classic branch-and-bound procedure of the commercial solver CPLEX applied to our mathematical model with a heuristic stopping criterion. The second approach consists of a fast construction heuristic to generate a set of feasible solutions. The third approach is based on a genetic algorithm that uses some of the solutions output by the second approach.

The motivation for this research dates back to January 1998, when the province of Québec in eastern Canada faced its first major disaster ever. From January 5th to 9th, three successive storms left from 85 mm to 100 mm of rain, freezing rain, hail and snow in the region south of Montreal. The accumulation of ice caused a technological disaster, as over 300 electrical towers fell down, resulting in an interruption of the electricity supply over a wide area. More than 700 municipalities – nearly half of Quebec's population of six million – were affected by power outages during the cold month of January. In addition, nearly 1.4 million subscribers in Quebec were

* Corresponding author. Faculté des Sciences de l'administration, Laval University, Quebec G1K 7P4, Canada. Tel.: +1 418 656 7029; fax: +1 418 656 2624.
E-mail address: jacques.renaud@fsa.ulaval.ca (J. Renaud).

0038-0121/$ – see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.seps.2011.05.002

completely without power, which was the most important consequence of the ice storm.

Establishing a technological climate disaster, this incident also caused a major malfunction of critical infrastructure: the telecommunications, banking and financial systems, as well as the health, water, computer, heating and lighting networks were all affected. A “black triangle” – extending from St. Hyacinthe in the north, Granby to the east and Saint-Jean-sur-Richelieu to the west – disrupted the daily lives of 41,000 inhabitants. Inside this black triangle, the complete failure of the electrical power grid lasted for more than five weeks. The economic impact of the ice storm has been estimated at over one billion Canadian dollars. Since many areas had to suspend their activities for several days, Canada’s economic output in January 1998 decreased by 0.7%. For Quebec, the gross domestic product decreased by 1.9% [22].

This disaster demonstrated that the political system was not ready to handle such a situation. In response, the Civil Protection Act was adopted by the Quebec government and went into effect on December 20, 2001. Now, each municipality must develop and update its own emergency preparedness plan, which includes all topics related to emergency logistics. As these requirements are relatively new, there are almost no tools or software to help and train municipality emergency managers. This led us to develop a complete decision-support system (DSS) for emergency logistics [27], which includes a database, a location and transportation optimization module and a resource allocation module. In this article, we describe the transportation module.

When designing the transportation algorithms, our primary objective was to produce high-quality solutions quickly (i.e., within a few seconds) so that these algorithms could be integrated in the DSS transportation module to help crisis managers efficiently deploy logistical resources. This DSS is also designed to be used to train municipal managers. The algorithms had thus to be fast enough to allow the decision-makers test and evaluate different deployment scenarios or transportation routes, for example. Moreover, as emergency situations are highly dynamic, new urgent requests may arrive at different times and must be scheduled immediately.

The main contributions of this article are the following. First, we model a practical transportation problem faced by crisis managers in emergency situations. We show that the exact branch-and-bound procedure used by CPLEX can yield optimal solutions in short computation times for very small problems with the proposed formulation. Second, we develop efficient algorithms for large problems, which produce near optimal solutions in relatively short computation times. In fact, the genetic algorithm proposed is fast enough to be used interactively in a decision-support system and provide high-quality transportation plans for emergency managers. Finally, computational results show that solutions produced by the genetic algorithm remain very close to the optimum even if demands and travel times are subject to small changes.

The remainder of the paper is organized as follows. Section 2 presents a detailed description of the *Transportation Problem in Disaster Response Operations* (TP-DRO) and its mathematical formulation. A literature review follows in Section 3. Section 4 introduces the developed algorithms. Section 5 reports our computational results, and Section 6 presents our conclusions.

2. Problem description

A disaster is a highly complex situation in which a flow of various goods must be transported to a stricken population using the vehicles available. Requested goods are generally shipped from a set of distribution centers, already open and staffed, to a set of delivery points that represent the locations where the disaster victims can go to obtain help. Since the transportation requests are numerous and heterogeneous, disaster managers often requisition almost all the available vehicles, even if some of them are not really efficient for delivering some kinds of goods or cannot be easily loaded/unloaded at some locations (e.g., school, sport center, town hall, private or public warehouse).

2.1. Problem definition and assumptions

The *Transportation Problem in Disaster Response Operations* (TP-DRO) can be formally defined as follows. Let u be the number of distribution centers (DC), indexed $l = 1,\ldots,u$, from which the humanitarian products are shipped. The number and the location of these distribution centers are assumed to be already determined (by the location module of the DSS, for example). Let n represent the number of delivery or distribution points, $i = 1,\ldots,n$, where people can go to obtain help (e.g., a refugee camp or a dormitory). Let p denote the total number of products types (or groups of humanitarian goods), $j = 1,\ldots,p$, needed for people relief. The nature of these products is closely related to the type of disaster (e.g., earthquake, chemical spill). The quantity of product j available at DC l is denoted as p_{jl}, and the quantity of product j required at delivery point i is denoted as d_{ij}. Throughout the paper, the total quantity available at DCs for each product type j is assumed to be sufficient to cover the demand of all delivery points for this product type (i.e., $\sum_{i=1}^{n} d_{ij} \le \sum_{l=1}^{u} p_{jl}$).

In addition, at each distribution center l, it is assumed that there are m_l vehicle types, $h = 1,\ldots,m_l$, and u_{hl} vehicles of each type h. Since all distribution centers may not be equally equipped for receiving a particular vehicle type, different docking times, τ_{hl}, are considered, one for each vehicle type h and the corresponding DC, l. Similarly, some vehicles may have certain handling equipment that makes them more efficient at manipulating some products. The time needed for loading and unloading one unit (i.e., a pallet) of product j into a vehicle of type h is defined as α_{jh}, where $\alpha_{jh} = \infty$ if product j cannot be loaded into a type-h vehicle.

There are also some restrictions on the total weight and the total volume associated with certain vehicles. These restrictions depend on the vehicle type used. Formally, a loaded vehicle of type h must not weigh more than Q_h weight units nor have a volume over V_h volume units. To determine the total weight (the total volume) corresponding to a given vehicle’s load, the weight w_j in weight units (the volume s_j in volume units) of each product j is assumed to be known with certainty.

Finally, a maximum daily work time, L_h (in time units) for each vehicle type h is imposed. A given vehicle can perform as many trips as needed during a day as long as the corresponding work time limit is respected. In disaster situations, determining travel times is not an easy task since updated information on the state of the transportation infrastructure has to be collected in order to obtain reliable estimations (see Yuan and Wang [37] for path selection models under emergency conditions). In this study, the travel times between the different points of origin and destination are assumed to be known. The travel time between delivery point i and distribution center l is denoted as t_{il}.

As requested quantities are generally large in terms of vehicle capacity (in weight and/or volume), each vehicle trip is assumed to visit only one delivery point at a time. In other words, only back and forth trips are considered. Obviously, a delivery point may be visited many times. However, because of the maximum daily work time, the number of trips performed to delivery point i by a specific vehicle will be limited to a maximum value r. In practice, deciding on the appropriate value for r may have strong consequences on both the quality of the solutions and the solution time. For example, if the number of trips r is strongly limited, the problem is manageable and solved quickly, but the solutions are likely to be of bad quality, sometimes even leading to the problem becoming infeasible. On the other hand, if the value of r is too large, the problem is more difficult to solve but the solutions produced are potentially of higher quality. In our experimental study, we first set $r=3$ and solved each instance to optimality. Then we set $r=4$ and $r=5$ and solved again each instance to see if some improvement can be achieved. We found that for all instances, $r=3$ is the smallest value leading to the optimal solution.

2.2. Mathematical model

The TP-DRO can be stated as follows.

Given a set of distribution centers where a certain number of vehicles of different types are located, determine vehicle trips that minimize the total transportation duration, such that (1) each delivery point receives the required quantity of each product type, (2) all vehicle constraints are satisfied, and (3) the distribution centers' product availability is respected.

We propose a mixed integer programming model (P) for TP-DRO, which uses the following decision variables.

- $x_{ilhk\nu}$, equal to 1 if delivery point i is visited from DC l with the kth vehicle of type h on its νth trip to i; and
- $Q_{ijlhk\nu}$, the quantity of product j delivered to point i from DC l with the kth vehicle of type h on its νth trip to i.

As already mentioned, TP-DRO aims to minimize the total transportation duration (i.e., the sum of all vehicles trip durations). The duration of the νth trip of the kth vehicle of type h from DC l to delivery point i, denoted $D_{ilhk\nu}$, is given by:

$$D_{jlhk\nu} = \left(2t_{il}x_{ilhk\nu} + \tau_{hl}x_{ilhk\nu} + \sum\nolimits_{j=1}^{p} \alpha_{jh}Q_{ijlhk\nu}\right)$$

where the first part ($2t_{il}$) represents the back and forth travel times, the second part (τ_{hl}) is the docking time, and the last part ($\sum_{j=1}^{p}\alpha_{jh}Q_{ijlhk\nu}$) is the loading and unloading time of all the products delivered from DC l to point i. If t'_{ilh} is defined as $t'_{ilh} = 2t_{il} + \tau_{lh}$, then the trip duration reduces to

$$D_{ilhk\nu} = \left(t'_{ilh}x_{ilhk\nu} + \sum\nolimits_{j=1}^{p} \alpha_{jh}Q_{ijlhk\nu}\right)$$

Hence, our model (P) is given by

$$\text{Min } Z = \sum_{i=1}^{n}\sum_{l=1}^{u}\sum_{h=1}^{m_l}\sum_{k=1}^{u_{hl}}\sum_{\nu=1}^{r}\left(t'_{ilh}x_{ilhk\nu} + \sum_{j=1}^{p}\alpha_{jh}Q_{ijlhk\nu}\right) \quad (1)$$

subject to

$$\sum_{l=1}^{u}\sum_{h=1}^{m_l}\sum_{k=1}^{u_{hl}}\sum_{\nu=1}^{r} Q_{ijlhk\nu} \geq d_{ij} \qquad \begin{array}{l} i = 1,\ldots,n \\ j = 1,\ldots,p \end{array} \quad (2)$$

$$\sum_{i=1}^{n}\sum_{h=1}^{m_l}\sum_{k=1}^{u_{hl}}\sum_{\nu=1}^{r} Q_{ijlhk\nu} \leq p_{jl} \qquad \begin{array}{l} j = 1,\ldots,p \\ l = 1,\ldots,m \end{array} \quad (3)$$

$$\sum_{i=1}^{n}\sum_{\nu=1}^{r}\left(t'_{ilh}x_{ilhk\nu} + \sum_{j=1}^{p}\alpha_{jh}Q_{ijlhk\nu}\right) \leq L_h \qquad \begin{array}{l} l = 1,\ldots,m \\ h = 1,\ldots,m_l \\ k = 1,\ldots,u_{hl} \end{array} \quad (4)$$

$$\sum_{j=1}^{p} w_j Q_{ijlhk\nu} \leq Q_h x_{ilhk\nu} \qquad \begin{array}{l} i = 1,\ldots,n \\ l = 1,\ldots,m \\ h = 1,\ldots,m_l \\ k = 1,\ldots,u_{hl} \\ \nu = 1,\ldots,r \end{array} \quad (5)$$

$$\sum_{j=1}^{p} s_j Q_{ijlhk\nu} \leq V_h x_{ilhk\nu} \qquad \begin{array}{l} i = 1,\ldots,n \\ l = 1,\ldots,m \\ h = 1,\ldots,m_l \\ k = 1,\ldots,u_{hl} \\ \nu = 1,\ldots,r \end{array} \quad (6)$$

$$Q_{ijlhk\nu} \in \mathbb{R}^{+} \qquad \begin{array}{l} i = 1,\ldots,n \\ j = 1,\ldots,p \\ l = 1,\ldots,m \\ h = 1,\ldots,m_l \\ k = 1,\ldots,u_{hl} \\ \nu = 1,\ldots,r \end{array} \quad (7)$$

$$x_{ilhk\nu} \in \{0,1\} \quad (8)$$

The objective function (1) minimizes the total duration of all trips. The constraints (2) ensure that each delivery point i receives the requested quantity of each product j. The constraints (3) guarantee that the total quantity of a given product j delivered from a DC l does not exceed this DC's capacity. The constraints (4) are the maximum daily work time restrictions associated to each vehicle k of type h located at DC l. Constraints (5) and (6) impose the vehicle capacity constraints for each trip, in terms of weight (Q_h) and volume (V_h). Finally, the constraints (7) and (8) are, respectively, non-negativity and binary constraints on the quantity (7) and routing (8) variables.

3. Literature review

There is a vast and very rich literature about routing problems. The *Traveling Salesman Problem* (TSP), which is at the center of almost all routing problems, has received considerable attention, with literally thousands of papers devoted to it, as well as many books [2,10,21] and literature reviews [15,18]. Its generalization, the classic *Vehicle Routing Problem* (VRP), has also been studied quite a bit, with many books [11,31] and literature reviews [8,16,17,19] dedicated to it. The VRP has been used as test bed for the development of metaheuristics, and we consider that algorithms currently used are not only quite sophisticated but also produce results that are very close to the optimal [20,26].

In recent years, research has moved to more sophisticated, realistic routing problems, such as dynamic problems [33], delivery problems with time windows [9] and fleet composition [25]. Among these new realistic problems, problems considering routing in emergency situations are beginning to emerge and lead to many new challenges, as pointed out by Sheu [28,29]. Altay and Green [1] presented a review of OR/MS research in disaster operations management. Breaking down the disaster operations management activities into four phases (mitigation, preparedness, response and recovery), they showed that most of the research was done in the pre-disaster phases of mitigation and preparedness. Among the response activities (e.g., emergency plan activation, evacuation, emergency rescue and medical care, fire fighting), they judged that routing operations for delivering massive amounts of aid have retained little attention.

Haghani and Oh [13] studied a particular version of disaster relief operations as a multi-commodity, multi-modal network flow model with time windows. They considered that a shipment can change from one mode to another at some given nodes, that earliest delivery times are given for some commodities and that arc capacity may be time-dependent. The time aspect was considered using a time-space network. Their goal was to minimize the total transportation cost. Heuristic solution procedures were proposed for solving problems with three modes, three points of origin and two destinations.

Barbarosolu et al. [5] studied helicopter routing and incorporated many specific constraints. They minimized the cost of assigning a helicopter to a given airbase and the cost of assigning a pilot to a helicopter/airbase combination. Özdamar et al. [23] considered a multi-period transportation problem in which vehicles do not have to return to the depot since the model regenerates plans incorporating new requests for aid. Thus, they repetitively solved their model at given time interval during ongoing aid delivery with the objective of minimizing the number of unsatisfied demands over time.

Sheu [28–30] modeled an emergency logistics network composed of relief suppliers, relief-distribution centers and affected areas. Their research can be viewed as a decision-support system that performs the following procedures: time-varying relief forecast, grouping of the affected areas, distribution priority determination, group-based relief distribution and dynamic relief supply. Yi and Özdamar [36] described an integrated location-distribution model for coordinating logistics support and evacuation operations in which they consider the evacuation and transfer of wounded people to emergency units. An important part of their model is the allocation of medical personnel between emergency units to maintain the people affected. Yi and Kumar [35] used the model of Yi and Özdamar [36] without considering the facility location problem.

Tzeng et al. [32] studied a multi-objective relief-distribution model in order to minimize total cost and travel time and maximize the minimal satisfaction when using their planning model. Their network considered five collection points, eight demand points and four transfer depots. Chern et al. [7] studied a similar network with four supply nodes, four distribution nodes, eight demand nodes, and also considered fuel stations. They consider two types of demands – incoming (e.g., food, water, medical supplies) and outgoing (e.g., dead, injured, sound) – as well as the due date for each type of demand. Campbell [6] considered that cost function may be irrelevant in emergency situations, and she proposed two alternative objective functions, one that minimizes the maximum arrival time and another that minimizes the average arrival time.

Balcik et al. [3] studied delivery of relief supplies from local distribution centers to beneficiaries affected by disasters, which they called the *last mile distribution*. They minimized the sum of transportation costs and penalty costs for unsatisfied and late-satisfied demands for two types of relief supplies. Their method works in two phases: (1) a route generation phase that creates a list of candidate routes and (2) a route selection phase based on a mathematical model. In comparison, the model of Özdamar et al. [23] can be seen as a mass supply directed to the local distribution centers, while Balcik et al. [3] performs the *last mile distribution*. Jotshi et al. [14] has developed a methodology for dispatching and routing emergency vehicles in post-disaster environment, with the objective of creating better routes from casualty pickup locations to hospitals. Balcik et al. [4] has discussed the research challenges and opportunities included in humanitarian relief chains.

Conceptually, the Balcik et al. [3] paper is most similar to our proposition since they considered a heterogeneous limited fleet, multiple vehicle routes, and two product types. They solved a single depot problem having four demand nodes using two identical vehicles.

4. Solution approaches

In emergency situations, good operational decisions must be made in a relatively short time since people's lives may be in danger. This is even more important when this decision process is a part of a whole decision-support system and could influence other subsequent decisions of the crisis manager. As discussed in Section 5, when applied to our model (P), the exact branch-and-bound procedure of CPLEX 12.1 produces optimal solutions in a reasonable time only for very small instances. This section presents two other approaches for solving the TP-DRO problem. The first approach is a simple, extremely fast list heuristic. The second approach is a genetic algorithm that uses the solutions obtained by the first heuristic as the initial population, producing high-quality solutions in relatively short times. Hence, with the exact approach and the two other approaches proposed in this section, the crisis manager gets to choose the right solution approach in terms of the problem size and the available computation time.

4.1. Set enumeration heuristic

The *set enumeration heuristic* (SEH) is a greedy heuristic that iteratively assigns the available vehicles at the distribution centers to delivery points until all the demands for all the products for all the delivery points are satisfied. More precisely, SEH considers an ordered list S composed of the triplet (l, h, k) (i.e., distribution center, vehicle type, vehicle number), in which all vehicles k $(k = 1\ldots,u_{hl})$ of all types h available at this DC $(h = 1,\ldots,m_l)$ are enumerated for each DC l $(l = 1,\ldots,u)$. For example, if we consider two DC $(u = 2)$, DC l_1 having three vehicles of type h_1 and DC l_2 having one vehicle of type h_1 and two vehicles of type h_2, then the ordered list is $S = ((l_1, h_1, 1); (l_1, h_1, 2); (l_1, h_1, 3); (l_2, h_1, 1); (l_2, h_2, 1); (l_2, h_2, 2))$.

Then, the triplets are treated one by one following the order in the list. For the current triplet (l, h, k), SEH determines a set $I(l,h,k)$of admissible delivery points for which the demand of some products is still unsatisfied. A delivery point i is admissible for a triplet (l, h, k) if there is at least one product j, such that (1) the demand of delivery point i for product j is not entirely satisfied, (2) product j is available at DC l, and (3) a quantity of product j can be delivered by a trip of the kth vehicle of type h without violating the work time duration restrictions. Next, the set of admissible delivery points $I(l,h,k)$is sorted in terms of the increasing distance between the delivery point and DC l. The current triplet (l, h, k) is assigned to the closest admissible delivery point i in $I(l,h,k)$ (i.e., the first element in the sorted list), and its remaining work time is updated.

Then, the sets $I(l,h,k)$ and S are updated, and the process is reiterated until the set $I(l,h,k)$ is empty. The set $I(l,h,k)$ may be empty because:

- *Alternative 1* – The demand of all the delivery points for all products is satisfied. In this case, SEH has already found a feasible solution.
- *Alternative 2* – All the products of DC l are already assigned (i.e., the DC is empty). In this case, SEH considers the following triplet (l,h,k) in list S and repeats the process.
- *Alternative 3* – The kth vehicle *of type* h is no longer available. Since a vehicle can perform many trips from DC l to a delivery point, a vehicle is unavailable only if the remaining work time is not sufficient to visit other delivery point. In this case, SEH considers the following triplet (l, h, k) in list S and repeats the process.

Remember that there are m_l vehicle types available at a DC l, u_{hl} vehicles of each type h. Hence, the number of triplets in the list S corresponding to a DC l is $\sum_h^{m_l} u_{hl}$, and the total list size is $\sum_{l=1}^{u}\sum_{h=1}^{m_l} u_{hl}$. Since the assignment of triplets to delivery points is made in a myopic manner, the order in which triplets are placed in the list S influences the heuristic result and performance. To give the approach more robustness, we considered different orders of triplets in the list. The first list S is obtained by a lexicographic enumeration of the distribution centers, vehicles types and vehicle number as presented before. Thus, $w-1$ other lists can be obtained by a simple rotation of the elements of S; the first triplet of the current list being placed at the end of the next one. For example, a rotation of a list $S=((l_1, h_1, 1); (l_1, h_1, 2); (l_1, h_1, 3); (l_2, h_1, 1); (l_2, h_2, 1); (l_2, h_2, 2))$ yields a new list $S'=((l_1, h_1, 2); (l_1, h_1, 3); (l_2, h_1, 1); (l_2, h_2, 1); (l_2, h_2, 2); (l_1, h_1, 1))$. If more than w lists are needed, a random permutation of the triplets within S is performed. The heuristic is executed for each enumerated set and the best solution produced is kept as the SEH solution. In the following, the number of different lists considered in SEH is denoted *MaxIteration*.

4.2. Genetic algorithm

Genetic algorithms are iterative search algorithms that aim to optimize a pre-specified function, called the *fitness* function. They consider a set of candidate solutions called the *population*. Each candidate solution within a population is referred to as a *chromosome* or an *individual*. In a genetic algorithm (GA), the initial population is generated either randomly or by using simple constructive algorithms. Then, new solutions, or individuals, are generated iteratively, using genetic operators, namely selection, crossover and mutation.

4.2.1. Individual encoding

In our case, we model a solution as a feasible vehicle assignment at the distribution centers to each product to be delivered to each distribution point. Thus, we propose to represent an individual (i.e., a feasible solution) Γ by np genes $(\gamma_{(1,1)}, \ldots \gamma_{(i,j)}, \ldots \gamma_{(n,p)})$, with n being the number of delivery points and p being the number of products. Each gene corresponds to a combination of a delivery point and a product and has an associated combination of DC and vehicle type to serve it. Formally, to each gene $\gamma_{(i,j)}$ is associated a pair (l, h), denoted $A(\gamma_{(i,j)})$ which identifies the distribution center and the vehicle type used to deliver product j to point i in solution Γ. Using this representation, $A(\gamma_{(4,3)}) = (l_1, h_2)$ states that product 3 is delivered to point 4 by a vehicle type h_2 from DC l_1. Unlike the Set Enumeration Heuristic, which uses triplets (l, h, k), the genetic algorithm uses only pairs (l, h), which gives us more flexibility in planning trips for the vehicles (see Section 4.2.4).

4.2.2. Population initialization

Assuming that the size of the initial population is s, we generate the initial population by considering the s best solutions produced by the Set Enumeration heuristic described in Section 4.1. To this end, we set the *MaxIteration* parameter of the Set Enumeration heuristic to the value s.

4.2.3. Individual evaluation: the fitness function

In our context, the lower a solution's total transportation duration, the better the solution. Hence, we define the fitness function f as the function that associates to each individual Γ the value $f(\Gamma) = 1/D(\Gamma)$, where $D(\Gamma)$ denotes the total duration corresponding to solution Γ.

4.2.4. Solution construction

Please note that the encoding approach proposed does not give a complete solution. In fact, specifying only the set of pairs (l, h) associated with $\gamma_{(i,j)}$ ignores the number and the loads of the trips performed by each vehicle. Deducing explicit solutions is not straightforward because of the vehicle constraints. This section describes the procedure used to explicitly construct feasible solutions or to prove that they do not exist. Let $\Gamma = (\gamma_{(1,1)}, \ldots, \gamma_{(i,j)}, \ldots, \gamma_{(n,p)})$ be an individual in the current population.

- *Step 1: Partial solution construction*

For each pair (l, h) used in Γ and each delivery point i served by this pair, construct a trip with a vehicle of type h from DC l to point i by assigning a quantity of a product j that respects the volume and weight capacity of vehicle type h. This trip is denoted $v(i,l,h)$. The order in which the products are considered is irrelevant. Note that a trip may include more than one product. Given the delivered quantities in a trip $v(i,l,h)$, its duration is computed as described in Section 2.2. At the end of step 1, the set of trips $V(l,h)$ associated with each vehicle type h used in each DC l for the solution Γ is known. Each trip satisfies the capacity constraints.

- *Step 2: Complete solution construction*

To check the feasibility of sets $V(l,h)$ with respect to the number and types of available vehicles, the GA solves a series of bin packing problems, one for each pair (l, h) used in Γ, in which the trips in $V(l,h)$ are the "objects" to be packed. The bin packing problem associated with a pair (l, h) considers u_{hl} bins (i.e., vehicles), each with a capacity L_h, and the "size" of an object (i.e., a trip) in $V(l,h)$ is its duration. If all the bin packing problems associated with all the pairs (l, h) used in Γ are feasible, the individual Γ is admissible, and the solution is explicit. Otherwise, Γ is discarded because more vehicles are needed than are available. In our implementation, the bin packing problems are solved by the well-known *first-fit decreasing* algorithm.

4.2.5. Selection, crossover and mutation

- *Selection*: individuals of a population $P = (\Gamma_1, \ldots, \Gamma_s)$ are selected using a roulette-wheel-selection procedure in which individuals with a higher fitness value have higher probability of being selected. The probability of selecting an individual Γ_g is computed as $\text{Prob}(\Gamma_g) = f(\Gamma_g)/\sum f(\Gamma_g)$.
- *Crossover*: assume that two individuals Γ_1 and Γ_2, encoded, respectively, as $(\gamma^1_{(1,1)}, \ldots, \gamma^1_{(i,j)}, \ldots, \gamma^1_{(n,p)})$ and $(\gamma^2_{(1,1)}, \ldots, \gamma^2_{(i,j)}, \ldots, \gamma^2_{(n,p)})$ have been selected in the current population P. A delivery point i is randomly selected. Two new individuals, Γ_1' and Γ_2', are generated as follows. Γ_1' is generated by copying all the genes of Γ_1 except those corresponding to delivery point i (i.e., the genes $\gamma^1_{(i,1)}, \ldots, \gamma^1_{(i,j)}, \ldots, \gamma^1_{(i,p)}$), which are replaced by the genes of individual Γ_2 corresponding to i (i.e., the genes $\gamma^2_{(i,1)}, \ldots, \gamma^2_{(i,j)}, \ldots, \gamma^2_{(i,p)}$). Similarly, Γ_2' is generated by copying all the genes of Γ_2 and replacing those corresponding to delivery point i by the genes of Γ_1 corresponding to i. Other types of crossover have been tested, but none of them led to better results.
- *Mutation*: each new individual Γ' obtained with the crossover procedure is mutated with a given probability. More precisely, each gene $\gamma'_{(i,j)}$ of Γ' is selected for mutation with a probability of 1%. When selected, the gene is mutated by randomly changing the vehicle types assigned to the corresponding pair (delivery point, product).

4.2.6. Local improvement

As the problem considers multiple vehicle types and involves many constraints, we developed three fast improvement procedures that are applied in sequence to a feasible solution. The first procedure, called *Single Move*, iteratively considers each delivery point i and evaluates the cost of delivering it from other DCs while respecting all the problem constraints. The second procedure, called *Double Move*, iteratively considers each pair of delivery points (i, i') delivered from two different DCs and exchanges their respective DC while respecting all the problem constraints. Both procedures implement the first improving move found and are repeated until no more improvements can be found. Finally, the last procedure, called *Vehicle Check*, makes sure that each delivery point is delivered by the vehicle type resulting in the smallest duration.

4.2.7. Diversification procedure

For each generation, the GA computes the average solution value over the s individuals of the population, μ_p. If the value of μ_p does not decrease over the last Max_{div} generations, this means that the algorithm may be trapped in a local optimum. In order to diversify the search, the GA removes the $\Delta\%$ worst individuals of the population and replaces them by new random individuals by using the *Set Enumeration Heuristic*. Our preliminary experiments showed that $\text{Max}_{\text{div}} = 10$ and $\Delta\% = 10\%$ give good results.

4.2.8. Complete genetic algorithm

To fully describe the genetic algorithm, the following parameters are considered: the maximum number of generations (Max_{gen}), the iteration counter (Iter_{gen}), the population size (s), the mutation probability (p_{mut}), the maximum number of generations without improvement in the population's average solution value (Max_{div}) and the diversification replacement percentage ($\Delta\%$). The complete algorithm can be described as follows.

- *Step 1 – Population initialization* – Use the *Set Enumeration Heuristic* to generate an initial population P^0 of s individuals. Set $\text{Iter}_{\text{gen}} = 0$.
- *Step 2 – Crossover and mutation* – Apply the selection, crossover and mutation operators to generate s new individuals, resulting in a new population P'.
- *Step 3 – Solution construction* – Construct explicit solutions from the individuals in P', using construction procedure based on the bin packing problem, and discard infeasible solutions.
- *Step 4 – Local improvement* – Each individual in P' is improved, using *Single Move*, *Double Move* and *Vehicle Check* procedures.
- *Step 5 – New population* – Generate a new population P'' by keeping the s best individuals over P and P'. Set $P = P''$ and $\text{Iter}_{\text{gen}} = \text{Iter}_{\text{gen}} + 1$. If necessary, update the best solution found and the average solution value of the entire population μ_p.
- *Step 6 – Diversification* – If μ_p does not improve over the last Max_{div} generations, replace the $\Delta\%$ worst solution in P by new ones by using the *Set Enumeration heuristic*.
- *Step 7 – Stopping criteria* – If $\text{Iter}_{\text{gen}} < \text{Max}_{\text{gen}}$, return to step 2; otherwise, the algorithm terminates.

5. Computational results

In this section, we first detail the problem generation procedure. Next, we evaluate the size of the instances that may be solved to optimality using the mathematical model presented in Section 2. Then, the results of our set enumeration heuristic and our genetic algorithm are reported.

5.1. Problem generation

A problem instance is defined by the number of distribution centers u, the number of delivery points n, and the number of products p. First, the u distribution centers (DC) and the n delivery points are randomly generated in a $[100 \times 100]$ interval. Then, for each delivery point, the product demand (in number of pallets) is randomly generated between a minimum and maximum value. Each product unit (pallet) has a certain weight and volume given in pounds and ft^3, respectively. Also, each product has a loading time (minutes by pallet) that depends on its compatibility with the vehicle used. We generated instances with four products (P1–P4) and three vehicle types (T1–T3). Table 1 provides the product generation parameters.

For each vehicle type, we defined the weight and volume capacity, the maximum traveling time per day and the docking time at each of the depots. In our problem, we used up to three distribution centers (DC1–DC3). These data are provided in Table 2. For the one-DC instances, there are 20, 30, 40, 50 and 60 delivery points and three, four, five and seven vehicles of each type available at the distribution center. For the two-DC instances, two vehicles of each type are available at each distribution center for the instances including 20 and 30 delivery points and three vehicles of each type for the instances including 40, 50 and 60 delivery points. For the three-DC instances, two vehicle of each type are available at each distribution center.

Once the product demand has been generated for each delivery point, the total capacity of all depots for each product is randomly generated between 120% and 150% of the product demand. For the two-DC instances, each distribution center receives between 40% and 60% of the total capacity for each product. For the three-DC instances, the capacity of each distribution center is feasibly sited between 35% and 50% of the capacity for each product. We generated three sets of instances with one, two and three distribution centers, respectively. For each set, we generated 10 instances with 20, 30, 40, 50 and 60 delivery points. Thus, 50 instances for each number of distribution center were solved, for a total of 150 instances. Tests were performed on an IBM x3550 with an Intel Xeon E5420 running at 2.5 Ghz with 4 Gig RAM.

Clearly, the generated instances are too small to represent realistic situations For example, since the Pan American Health Organization (PAHO) [24] and the US Government use a standard operational classification [34] for donated relief supplies composed of 10 broad classes (i.e., medicines, health supplies/equipment, water and environmental health, food/beverages, shelter/housing/electrical/construction, logistics/administration, human resources, personal needs, agriculture, unsorted), using only four products is unrealistic. However, despite the fact that CPLEX produced the optimal results needed as a basis for comparison for the evaluation of the set enumeration heuristic and GA proposed in the paper, CPLEX could not handle anything larger.

5.2. Optimal results

This section analyzes the solving capability of our model, using CPLEX 12.1. Since the proposed algorithms are designed to be used within a real-time decision-support system, it should be possible to obtain solutions quickly (few seconds). We solved the model by allowing CPLEX run for 30 s, which is the maximum tolerable computation time for a DSS. However, to assess CPLEX's capabilities, we also recorded the solution values after 60 and 600 s. Afterward, we let CPLEX run as long as needed to find the optimal solution. Please note that, when solving some instances, out-of-memory errors occurred.

The data reported in Table 3 are the average results for 10 instances. Columns DC and n refer to the number of distribution

Table 1
Product generation parameters.

Product	Demand (pallets)		Volume (ft^3)	Weight (pounds)	Loading time per vehicle type (min/pallet)		
	Minimum	Maximum			T1	T2	T3
P1	10	30	40	300	0.1	0.1	0.1
P2	20	40	35	250	0.2	0.2	0.1
P3	30	50	30	200	0.3	0.1	0.1
P4	30	50	30	150	0.3	0.3	0.4

Table 2
Vehicle type parameters.

Vehicle type	Capacity		Maximum length (min)	Docking time at depot (min)		
	Weight (pounds)	Volume (ft^3)		D1	D2	D3
T1	32,000	10,000	720	10	10	10
T2	38,000	12,000	720	5	5	5
T3	48,000	15,000	720	10	10	10

centers and the number of delivery points. For the 30, 60 and 600 s, the average gap is calculated in terms of the best-known solution if we let CPLEX run as long as needed to prove optimality. In this case, the best-known solution is either the optimal solution, the best solution found before CPLEX ran out of memory or, if this is the case, the best solution found by the genetic algorithm. Columns NbOptimal show the number of optimal solutions obtained after the corresponding computation time. Finally, the column Nb OOM shows the number of times CPLEX ran out of memory when we let CPLEX take as much computation time as it needed.

These data show that for very small instances (DC $=1$ and $n \le 40$, DC $=2$ and $n \le 40$, CD $=3$ and $n \le 20$), results are almost optimal within 30 s. The optimal was reached within 600 s for 95 instances out of 150. The ** indicate that two additional optimum were found after 152,893 and 25,626 s. For the 60 instances with $n=50$ and $n=60$, only 8 optimums were found, with an average computation time of 5.78 h; CPLEX ran out of memory for the other 52 instances. The instances with two distribution centers show that CPLEX began to have difficulty within the 30-s time limit since the average gap for 60 points goes from 0.34% with one DC to 0.93% with two DC. Clearly, with 3 DC, CPLEX encountered serious problems with an average gap of 2.04% for the 60-point instances. For the 3 DC with $n=60$, CPLEX ran out of memory eight times after an average computation time of 63,719 s (17.7 h). These results show that CPLEX cannot be used within a DSS to generate high-quality solutions for real-life instances.

5.3. *Results for the set enumeration heuristic and genetic algorithm*

This section presents results for the set enumeration heuristic and the genetic algorithm that we developed. Results are reported in terms of the best-known solution for each instance (95 optimal solutions out of 150).

Table 4 displays the results for the Set Enumeration Heuristic (Section 4.1) after 20, 30, 50 and Max iterations where Max: $=50(u(w+n))$. For the one-DC instances, the Set Enumeration heuristic produced solutions within 2% of the optimal values in a fraction of a second (50 iterations). For the two-DC instances, solutions within 3.8% of optimality are obtained within 10 seconds. Unfortunately, the performance of SEH decreases with the three-DC instances.

Table 5 provides the results of the genetic algorithm (Section 4.2) for eight combinations of numbers of generations and individuals. For each combination, we give the average gap in terms of either the optimal solution or the best known solution and the computation time in seconds. The fastest combination (50g100i for 50 generations with 100 individuals) produces very interesting results for one-DC instances, with average gaps lower than 0.24% in less than 10 s. For two- and three-DC instances, the maximum average gaps are 1.38% and 2.11%, respectively, in less than 10 s of computation time. For high-quality solution in a relatively short computation time, the algorithm with 50 generations and 200 individuals (50g200i) produced gaps lower than 0.12%, 1.00% and 1.26% for one-, two- and three-DC instances, respectively, in less than 14 s. Consequently, this combination would be very attractive for use within a decision-support system.

Table 5 also shows that the genetic algorithm can produce even higher quality solutions if more computation time is allowed. For the one-DC instances, the maximum gap is 0.08% (for $n=50$) in less than 38 s (compared to 600 s for CPLEX). For the two-DC instances, the maximum gap is 0.36% for 50 delivery points in 29 s. For the largest instances solved, including 3 distribution centers and 60 delivery points, the average gap is 0.72% in 63 s of computation time.

Finally, Table 6 shows the number of optimal solutions obtained by the two best versions of the genetic algorithm. The 300g300i version produces 61 optimal solutions within an average computation time of 28 s. In comparison, CPLEX found 85 optimal solutions within 30 s and 93 optimal solutions within 600 s. Note that, for DC $=1$ and $n=60$, CPLEX was unable to complete the search, going out of memory for all the 10 instances. For two of these instances, the solution output by the genetic algorithm was better

Table 3
CPLEX results.

DC	n	CPLEX time limit						
		30 s		60 s		600 s		
		Average gap with best (%)	Nb optimal	Average gap with best (%)	Nb optimal	Average gap with best (%)	Nb optimal	Nb OOM
1	20	0.00	10	0.00	10	0.00	10	0
	30	0.01	8	0.01	9	0.01	9	0
	40	0.00	10	0.00	10	0.00	10	0
	50	0.11	0	0.09	0	0.08	0	10
	60	0.40	0	0.25	0	0.08	0	10
2	20	0.00	10	0.00	10	0.00	10	0
	30	0.00	10	0.00	10	0.00	10	0
	40	0.02	9	0.00	9	0.00	9	1
	50	0.24	3	0.23	3	0.17	3	7
	60	0.97	0	0.58	0	0.07	0	10
3	20	0.00	10	0.00	10	0.00	10	0
	30	0.07	8	0.01	9	0.01	9	0
	40	0.26	5	0.09	8	0.00	10	0
	50	1.80	2	1.17	3	0.17	3	7
	60	2.04	0	0.84	0	0.17	0**	8

Table 4
Results for the set enumeration heuristic.

DC	n	Number of iterations			
		20	30	50	Max
		Gap (%)	Gap (%)	Gap (%)	Gap (%)
1	20	1.0	0.9	0.9	0.9
	30	1.6	1.6	1.5	1.0
	40	2.0	1.9	1.9	1.3
	50	1.8	1.8	1.8	1.2
	60	1.9	1.9	1.8	1.2
Seconds		0.1	0.1	0.1	5.3
2	20	10.1	8.5	6.9	3.8
	30	8.9	8.0	6.8	3.6
	40	13.4	9.0	6.8	2.6
	50	10.9	8.2	6.7	3.1
	60	8.6	6.5	5.5	2.4
Seconds		0.1	0.1	0.1	9.4
3	20	30.9	21.1	18.1	11.7
	30	34.9	24.1	20.3	10.3
	40	37.0	25.2	20.1	9.8
	50	17.3	12.3	10.3	5.3
	60	16.6	12.5	11.4	5.0
Seconds		0.1	0.1	0.1	17.6

Table 5
Results for the genetic algorithm.

DC	n	Number of generations															
		50g100i		50g200i		100g100i		100g200i		200g100i		200g200i		300g100i		300g300	
		Gap (%)	Sec	Gap (%)	Sec	Gap (%)	Sec	Gap (%)	Sec	Gap (%)	Sec	Gap (%)	Sec	Gap (%)	Sec	Gap (%)	Sec
1	20	0.00	3	0.00	4	0.00	3	0.00	5	0.00	4	0.00	9	0.00	8	0.00	13
	30	0.07	5	0.05	6	0.05	5	0.05	7	0.05	6	0.04	13	0.05	11	0.04	19
	40	0.11	6	0.04	8	0.09	7	0.02	9	0.02	7	0.02	18	0.05	15	0.02	23
	50	0.24	8	0.12	10	0.19	9	0.11	11	0.11	9	0.09	23	0.13	18	0.08	29
	60	0.09	9	0.07	11	0.08	10	0.06	18	0.06	11	0.06	20	0.07	27	0.05	38
2	20	0.61	4	0.21	5	0.41	5	0.16	6	0.39	5	0.08	10	0.24	6	0.07	13
	30	0.74	5	0.16	7	0.38	7	0.14	9	0.37	7	0.05	14	0.12	8	0.02	19
	40	1.03	7	0.30	10	0.75	9	0.25	12	0.68	9	0.07	15	0.18	12	0.04	22
	50	1.38	9	1.00	12	1.21	10	0.95	15	1.17	11	0.46	22	0.69	15	0.36	29
	60	0.91	10	0.41	14	0.62	14	0.39	22	0.47	18	0.35	26	0.36	31	0.25	56
3	20	0.62	4	0.26	5	0.58	5	0.22	8	0.44	5	0.17	12	0.40	8	0.14	15
	30	0.89	5	0.40	7	0.78	7	0.31	10	0.64	7	0.27	15	0.55	12	0.20	24
	40	1.80	7	1.18	9	1.63	8	0.98	13	1.51	9	0.89	18	1.45	16	0.63	29
	50	1.68	8	1.17	12	1.46	10	0.95	15	1.28	11	0.94	23	1.24	18	0.65	33
	60	2.11	10	1.26	14	1.71	14	1.15	22	1.39	16	0.88	24	1.08	27	0.72	63

than the best feasible solution yielded by CPLEX before the search was aborted.

5.4. Robustness of the solutions produced by the genetic algorithm

In disaster response operations, travel times and quantities of products requested at delivery points may not be known with certainty. The object of this section is to evaluate the *robustness* of the solutions produced by the genetic algorithm with regard to small perturbations in the estimated demand and travel times. In other words, we seek to evaluate how the quality of a solution produced by the genetic algorithm is affected by small changes in input data. The quality of a solution is measured in this case through the deviation it yields with regard to the optimum.

To assess the robustness of the proposed GA, we proceed as follows. We consider a subset G^O of original instances and modify the demand for each instance g^O in G^O. The resulting set of perturbed instances is referred to as G^P. An instance is perturbed by randomly increasing or decreasing the demand of each product j for each delivery point i by ρ_{ij}%. That is, the new demand of delivery point i for a product j in an instance g^P resulting from the perturbation of instance g^O is $d_{ij}^P = d_{ij}^O \pm 0.01\rho_{ij}d_{ij}^O$. For each original instance $g^O \in$, we also record both the optimal solution output by CPLEX (we consider the instances for which CPLEX does not run out of memory), denoted by $S^*(g^O)$, and the solution yielded by the proposed genetic algorithm: $S^{GA}(g^O)$; as well as the corresponding objective values $Z^*(g^O)$ and $Z^{GA}(g^O)$, respectively. Then, the optimal solution of the perturbed instance g^P is determined by solving the associated model with CPLEX. The resulting optimal solution and objective value are denoted $S^*(g^P)$ and $Z^*(g^P)$, respectively. To assess the robustness of the genetic algorithm, we keep the solution produced by the GA to the original (unperturbed) instance g^O, $S^{GA}(g^O)$, and evaluate it when applied on the perturbed instance g^P. The value of the corresponding objective function, denoted $Z^{GA,O}(g^P)$, is then compared to the optimal value $Z^*(g^P)$. We computed the gap between the GA solution applied to the perturbed instances and the optimal solution for these instances using the ratio: $Z(S) - Z^*/Z^*$, where $Z(S)$ is the objective function value corresponding to solution S and Z^* is the optimal value. Hence, if gaps remain as low as in the previous section, we will be able to confirm the robustness of the solutions produced by the GA algorithm, with respect to small changes in demand. The same procedure was applied in order to perturb travel times between the distribution centers and the delivery points.

We consider three types of experiments. For the first type, only the demand of delivery points is perturbed. For the second type, only travel times between distribution centers and delivery points are perturbed. Finally, for the third type, both demand and travel times are perturbed. For each type of experiment, three levels of perturbation were considered.

Perturbed instances were therefore generated in the following way. We selected three instances from the original set corresponding to DC = 2 and n = 20, 40 and 50, respectively.

Starting from each original instance, five new instances were generated for each type of experiment and level of perturbation by modifying demand quantities, travel times or both by a factor randomly drawn within the pre-specified intervals corresponding to the perturbation levels. Overall, 45 perturbed instances were generated for each of the three original solutions, for a total of 135 perturbed instances. Table 7 reports the results of our experiments. The solution produced by the GA method to the original instances was applied to the corresponding perturbed instances, and the gap with respect to the optimal solution to each perturbed instance (obtained by CPLEX) was computed. We also report in the last line of Table 7 the original gap obtained for the genetic algorithm for the original instances.

As can be seen in Table 7, when the demand is perturbed by ±5% and by ±10%, the original GA solution remains optimal for n = 20. For larger instances (n = 40 and 50), GA solutions are within 0.13%

Table 6
Optimal solutions obtained by the genetic algorithm.

DC	n	300g100i Nb Opt	300g300i Nb Opt
1	20	10	10
	30	7	7
	40	4	4
	50	0	0
	60	0	0
2	20	4	8
	30	5	7
	40	3	9
	50	0	0
	60	0	0
3	20	3	7
	30	3	5
	40	0	2
	50	0	0
	60	0	0

Table 7
GA average gap for original and perturbed instances.

Type of experiment	Perturbation level	(DC, *n*)		
		(2,20)	(2,40)	(2,50)
Demand	[−5%; +5%]	0.00%	0.00%	0.13%
	[−10%; +10%]	0.00%	0.05%	0.13%
	[+5%; +10%]	0.00%5*	0.00%4*	0.17%5*
Travel time	[0%; +5%]	0.07%	0.00%	0.10%
	[0%; +10%]	0.12%	0.03%	0.19%5#
	[+5%; +10%]	0.27%	0.03%	0.13%5#
Demand and travel time	[0%; +5%]	0.17%	0.01%	0.15%5#
	[0%; +10%]	1.40%5*	0.04%1*	0.17%5*
	[+5%; +10%]	0.02%5*	0.02%	0.09%5*
Gap for original instances		0.00%	0.00%	0.10%

of optimality. For the instances perturbed by a factor lying within [+5%, +10%], the solutions produced by the genetic algorithm to the original instances become infeasible when applied to the perturbed ones (the results obtained for these particular instances are marked with an * in Table 7. The number in front of the * corresponds to the number of infeasible instances over the five considered). For all these instances, infeasibility was due to the lack of supplies at the DC's. Recovering solution feasibility was, however, an easy task for all the instances. To build these *modified* solutions, we identified the route which could not be supplied adequately by its DC and made it depart from the other DC, while keeping the rest of the solution unchanged. The results reported in the [+5%, +10%] row give the average gap for the modified solutions. It can be seen that, for instances (2, 20) and (2, 40), the *modified* solutions are optimal for the perturbed instances. For instances in (2, 50), the average gap is of 0.17%.

When analyzing the case where only travel times are perturbed, the results obtained show that the solutions produced by the genetic algorithm remain feasible and close to the optimum for all the perturbed instances corresponding to $(DC, n) = (2, 20)$ and (2, 40). The maximum average gap does not exceed 0.27%. However, for larger problems (i.e., $n = 50$), 10 instances (over the 15 considered) become infeasible (the results obtained for these particular instances are marked with a "#" in Table 7. The number in front of the "#" corresponds to the number of infeasible instances over the five considered). Infeasibility, in this case, is due to the fact that the last trip of a vehicle exceeds the maximum daily work time restriction. Feasibility was recovered by moving the last trip of an unfeasible route to the first vehicle having sufficient time left. The results reported in Table 7 for these instances give the average gap when considering the modified solutions. The average gap obtained for the modified solutions remain relatively small (no more than 0.19%).

Finally, when both demand and travel times are perturbed, the solutions produced by the genetic algorithm for the original instances become infeasible for 26 instances out of the 45 perturbed. For five of these instances (marked with a "#"), infeasibility was due to the maximum daily work time restrictions. For the remaining 21 instances (marked with a "*"), infeasibility was due to the lack of supplies at the DC's. Solutions were modified as explained above to recover feasibility. The values reported in Table 7 for these instances correspond to the average gap obtained for modified solutions. The results of Table 7 show that the original and the modified solutions of the genetic algorithm remain of good quality even for relatively large perturbations affecting both demand and travel times.

In summary, results of Table 7 show that the solutions produced by the GA are very robust since, when applied to perturbed instances, they remain of high quality, the largest average deterioration not exceeding 0.27% for the solutions that remain feasible, and 1.40% for those that need to be slightly modified to become feasible. The robustness of the algorithm can be explained by the use of *back and forth* trips. In fact, imposing that only one delivery point is visited in a vehicle trip results most of the time in some unused capacity (in terms of volumes and/or time) that can be used to absorb demand or travel times fluctuations. For example, a vehicle for which the maximum work time must not exceed 720 min may spend 680 min in performing three trips. The remaining 40 min is clearly not enough to perform another trip but may be very useful to absorb some perturbation in travel times.

6. Conclusion

This article defines and models an important transportation problem encountered in emergency response operations. CPLEX shows that it is a good alternative for very small instances. However, heuristics are needed for realistic problem sizes. The genetic algorithm developed is fast enough to be used in an interactive decision-support system and produces near optimal solutions. For instances involving three distribution centers and 60 delivery points, the GA produces solutions with an average gap lower than 0.72% in only 60 s of computation time. These results show not only that the genetic algorithm is a high performing alternative, but it may also be used to deal with larger instances in a decision-support system. We also prove through an empirical study that the solutions produced by the genetic algorithm are robust with respect to small changes in demand quantities and travel times. Such a result is important in disaster contexts since the demand of affected areas as well as travel times are not known with certainty, especially in the first hours following a disaster.

Acknowledgements

This work was financed by individual grants [OPG 0293307, OPG 0172633 and OPG 0371655], by Collaborative Research and Development Grants [CG 095123] from the *Canadian Natural Sciences and Engineering Research Council* (NSERC), by DMR, a division of Fujitsu Consulting (Canada), and by *Partnerships for Research on Microelectronics*, *Photonics and Telecommunications* (PROMPT). This financial support is gratefully acknowledged. We would also like to thank the emergency managers of our industrial partner, Fujitsu Consulting (Canada), for providing us with relevant data and useful comments. Finally, we thank the guest editor and the referees for their valuables comments.

References

[1] Altay N, Green III WG. OR/MS research in disaster operations management. Eur J Oper Res 2006;175:475–93.
[2] Applegate DL, Bixty RE, Chvatal V, Cook W. The traveling salesman problem. A computational study. In: Princeton series in applied mathematics; 2006.
[3] Balcik B, Beamon BM, Smilowitz K. Last mile distribution in humanitarian relief. J Intell Transp Syst 2008;12:51–63.
[4] Balcik B, Beamon BM, Krejci CC, Muramatsu KM, Ramirez M. Coordination in humanitarian relief chains: practices, challenges and opportunities. Int J Prod Econ 2010;126:22–34.
[5] Barbarosoğlu G, Özdamar L, Çevik A. An interactive approach for hierarchical analysis of helicopter logistics in disaster relief operations. Eur J Oper Res 2002;140:118–33.
[6] Campbell AM. Routing under relief efforts. TranspSci 2008;42:127–45.
[7] Chern CC, Chen YL, Kung LC. A heuristic relief transportation planning algorithm for emergency supply chain management. Int J Comput Mathem; 2009:1–27.
[8] Cordeau J-F, Gendreau M, Laporte G, Potvin J-Y, Semet F. A guide to vehicle routing heuristics. J Oper Res Soc 2002;53:512–22.
[9] Cornillier F, Laporte G, Boctor FF, Renaud J. The petrol station replenishment problem with time windows. Comput Oper Res 2009;36:919–35.

[10] Gutin G, Punnen AP. The traveling salesman problem and its variations. Springer; 2007.
[11] Golden B, Raghavan S, Wasil E. The vehicle routing problem. Latest advances and new challenges. Springer; 2008.
[12] Haddow GD, Bullock JA, Coppola DP. Introduction to emergency management. In: Butterworth-Heinemann homeland security series. Elsevier Inc; 2008.
[13] Haghani A, Oh S-C. Formulation and solution of a multi-commodity, multi-modal network flow model for disaster relief operations. Transp Res A 1996; 30:231–50.
[14] Jotshi A, Gong Q, Batta R. Dispatching and routing of emergency vehicles in disaster mitigation using data fusion. Socio-Econom Plan Sci 2009;43:1–24.
[15] Laporte G. The traveling salesman problem: an overview of exact and approximate algorithms. Eur J Oper Res 1992a;59:231–47.
[16] Laporte G. The vehicle routing problem: an overview of exact and approximate algorithms. Eur J Oper Res 1992b;59:345–58.
[17] Laporte G. Fifty years of vehicle routing. Transp Sci 2009;43:408–16.
[18] Laporte G. A concise guide to the traveling salesman problem. J Oper Res Soc 2010;61:35–40.
[19] Laporte G, Osman Ibrahim H. routing Problems: a bibliography. Ann Oper Res 1995;61:227–62.
[20] Laporte G, Gendreau M, Potvin J-Y, Semet F. Classical and modern heuristics for the vehicle routing problem. Int Trans Oper Res 2000;7:285–300.
[21] Lawler EL, Lenstra JK, Rinnooy Kan AHG, Shmoys DB. The traveling salesman problem. A guided tour of combinatorial optimization. John Wiley & Sons; 1985.
[22] Nicolet R. Pour affronter l'imprévisible: les enseignements du verglas de 98: rapport de la Commission scientifique et technique chargeće d'analyser les ećvećnements relatifs ala tempete de verglas survenue du 5 au 9 janvier 1998. Première partie, Chapitre 2, Les impacts économiques, sociaux et environnementaux, 43–100; 1999.
[23] Özdamar L, Ekinci E, Küçükyazici B. Emergency logistics planning in natural disasters. Ann Oper Res 2004;129:217–45.
[24] Organisation Panaméricaine de la santé. Logistique et gestion des approvisionnement du secteur santé. Washington: Organisation Panaméricaine de la santé; 2003.
[25] Pessoa A, Eduardo Uchoa, de Aragao Marcus Poggi. A robust branch-and-cut algorithm for the heterogeneous fleet vehicle routing problem. Networks; 2009:167–77.
[26] Potvin J-Y. Evolutionary algorithms for vehicle routing. INFORMS J Comput 2009;21:518–48.
[27] Renaud J, Ruiz A, Rekik M, Berkoune D, Paquet S, Paquet R. Emergency logistic decision support. Working paper, Université Laval, DMR Une société Fujitsu; 2010.
[28] Sheu J-B. Editorial: challenges of emergency logistics management. Transp Res E 2007;43:655–9.
[29] Sheu J-B. An emergency logistics distribution approach for quick response to urgent relief demand in disasters. Transp Res E 2007;43:687–709.
[30] Sheu J-B. Dynamic relief-demand management for emergency logistics operations under large-scale disasters. Transp Res E 2010;46:1–17.
[31] Toth P, Vigo D. The vehicle routing problem. In: SIAM monographs on discrete mathematics and applications; 2002.
[32] Tzeng G-H, Cheng H-J, Huang TD. Multi-objective optimal planning for designing relief delivery systems. Transp Res E 2007;43:673–86.
[33] Wen M, Cordeau J-F, Laporte G, Larsen J. The dynamic multi-period vehicle routing problem. Comput Oper Res 2010;37:1615–23.
[34] World Health Organization (WHO). The management nutrition in major emergencies. Geneva: WHO; 2000.
[35] Yi W, Kumar A. Ant colony optimization for disaster relief operations. Transp Res E 2007;43:660–72.
[36] Yi W, Özdamar L. A dynamic logistics coordination model for evacuation and support in disaster response activities. Eur J Oper Res 2007;179:1177–93.
[37] Yuan Y, Wang D. Path selection model and algorithm for emergency logistics management. Comput Ind Eng 2009;56:1081–94.

Djamel Berkoune holds a Ph. D. in Industrial and Human Automation and Computer Science and is a researcher in logistics at Laval University. His main research interests lie in the fields of production, logistics and hospital management.

Jacques Renaud is professor of logistics at Laval University. He is responsible of teaching courses related to logistics, distribution as well as transportation management. His main research interests lie in the fields of distribution management, including vehicle routing and warehousing. He supervised more than sixty graduate students and ten students are currently completing their studies under his supervision.

Monia Rekik is an assistant professor at Laval University. She is responsible of teaching courses related to logistics, mathematical programming and industrial scheduling. Her main research interests lie in the fields of combinatorial optimization, personnel scheduling, transport auctions and emergency logistics.

Angel Ruiz is professor in the department of Operations and Decision Systems at Laval University. He holds a Ph.D. from the Université de Technologie de Compiègne (France). His current research interests are in supply chain management, emergency logistics and medical emergency planning. He is deputy director of the Interuniversity Research Center on Enterprise Networks, Logistics and Transportation (CIRRELT) and head of the CIRRELT's Laboratory on Healthcare Networks.