Exact and heuristic algorithms for finding envy-free allocations in food rescue pickup and delivery logistics

David Rey[a,*], Khaled Almi'ani[a,b,c], Divya J. Nair[a]

[a] *School of Civil and Environmental Engineering, UNSW Sydney, Australia*
[b] *Princess Sumaya University for Technology, Amman, Jordan*
[c] *Al-Hussein Bin Talal University, Ma'an, Jordan*

Abstract

Food rescue organizations collect and re-distribute surplus perishable food for hunger relief. We propose novel approaches to address this humanitarian logistics challenge and find envy-free allocations of the rescued food together with least travel cost routes. We show that this food rescue and delivery problem is NP-hard and we present a cutting-plane algorithm based on Benders' decomposition for its exact solution. We introduce a novel heuristic algorithm that combines greedy and local search. We test our approaches using real data from food rescue organizations. Our results show that the proposed algorithms are able to efficiently provide envy-free and cost-effective solutions. This paper has been published in *Transportation Research Part E: Logistics and Tranportation Review*, see https://doi.org/10.1016/j.tre.2018.02.001

Keywords: Humanitarian logistics, food relief, fairness, pickup and delivery, Benders decomposition, algorithms

1. Introduction

Hunger is one of the main humanitarian challenges faced by emerging and developed countries. One of the externalities of our consumer society is that food is often wasted. For instance, to keep up with competitors, grocery stores continually store fresh food and have a tight product turnover which typically leads to extensive waste. The same holds for catering events which often prefer to overestimate than underestimate attendance. In an attempt to reduce this societal imbalance, charities and not-for-profit organisations have promoted food rescue initiatives and engineered logistical solutions to re-distribute rescued food to welfare agencies (Foodbank US, Australia; Second Harvest Canada; Food Rescue Australia; OzHarvest Australia; Secondbite Australia; Foodbank Singapore; City Harvest US; Food Shuttle US; Feeding India, India; Leket Israel, etc.) (OzHarvest, 2017; SecondBite, 2017; FoodBank, 2017; SecondHarvest, 2017; Leket Israel, 2017; Feeding India, 2017). The

*Corresponding author
URL: d.rey@unsw.edu.au (David Rey)

past several years have seen a burgeoning in food rescue operations with recovering tons of excess food and distributing it to the people in need. Foodbank Australia delivers 65 million meals per year whereas OzHarvest Australia re-distributes 5 million kg of food per year. In comparison, Food bank Central and Eastern North Carolina delivers 30 million kg of food per year, Feeding America claims to feed 46 million people per year and Second Harvest Canada delivers 11 million meals per year.

It is frequent that the total request of welfare agencies exceeds the available food supply. Hence, it is necessary to fairly allocate the rescued food among welfare agencies. In addition, food rescue organizations typically focus on collecting perishable food which must be re-distributed on the same day, without any storage possible. Due to the perishability of food products collected, these products cannot always be stored in warehouses, and same-day deliveries may be required. This is particularly relevant for the delivery of ready-to-serve meals.

Food rescue programs have become increasingly popular and so has the scale of their operations, thus motivating the need for customized quantitative decision-making frameworks. From a logistical standpoint, the food rescue and delivery problem involves two types of decisions: food allocation and distribution. The former is concerned with the problem of finding fair or equitable allocations of the rescued food among welfare agencies. In turn, the latter is focused on identifying cost-effective vehicle routes to pick-up and deliver the rescued food. While allocation and routing problems have received a significant attention in the literature, the vast majority of the proposed approaches focus on cost-driven solutions which are oblivious to fairness considerations. In this paper, we propose novel solution methods for the fair allocation and re-distribution of rescued food.

Our approach is designed to promote envy-free allocations which are fair with regards to welfare agencies' requests. An allocation is said to be envy-free if all the welfare agencies receive the same amount of food or if the welfare agencies which receive less than the maximum amount delivered are satisfied. Although envy-freeness is a desirable property of fair division solution methods, such equitable allocations may not exist, especially in the context of resource-constrained logistical operations. We highlight this challenge on a toy problem and show that this food rescue and delivery problem is NP-hard. We present a mixed-integer linear program (MILP) representation of the problem which combines fair allocation and vehicle routing decisions. We propose a cutting-plane algorithm based on Benders' decomposition for its exact solution. We then introduce a heuristic algorithm for finding (near) envy-free and cost-effective solutions for the food rescue and delivery problem. Our approach is evaluated through numerical experiments designed to be representative of realistic food rescue programs.

This work builds on prior research by Nair et al. (2016, 2017a,b), especially the latter. In Nair et al. (2016, 2017a) the effort is focused on minimizing routing cost for periodic pick-up and delivery vehicle

routing problems *i.e.*, there is no food allocation component within the optimization problems solved therein. In Nair et al. (2017b), the authors solve a food rescue and delivery problem similar to that of the present paper with different fairness criteria, *i.e.*, egalitarian and maximum deviation *versus* envy-freeness; and propose a meta-heuristic based solution approach. In contrast, in the present paper we combine relaxations and valid inequalities in a novel cutting-plane algorithm for the exact solution of food rescue and delivery problems and introduce a new heuristic algorithm based on local search.

The paper is organized as follows. Section 2 reviews the state-of-the-art on humanitarian logistics as well as allocation and routing solution approaches. Section 3 formally introduces the food rescue and delivery problem, its mathematical programming representation and a cutting-plane algorithm for its exact solution. Section 4 presents a novel heuristic approach for the food rescue and delivery problem. Section 5 demonstrates the performance of the proposed heuristic algorithm through numerical experiments and Section 6 summarizes the findings of this paper and outlines possible extensions.

2. Literature Review

We first review the literature on vehicle routing with pickup and delivery operations which is closely related to the routing component of the food rescue and delivery problem. We then present the main relevant efforts in humanitarian logistics before focusing on fairness-driven allocation and routing approaches.

2.1. Vehicle routing for pickup and delivery operations

The routing component of the food rescue and delivery problem can be represented as a single-commodity pick-up and delivery vehicle routing problem. We hereby briefly review the literature on pick-up and delivery routing problems and position our paper in this field.

In the context of the traveling salesman problem (TSP) Hernández-Pérez & Salazar-González (2004a) introduced the single-commodity, unpaired pickup and delivery problem (1-PDTSP) which has many real-world applications such as the transportation of milk, sand, gas, eggs or vaccines. They proposed integer-linear programming formulations and a branch-and-cut algorithm able to solve instances with up to 40 customers. They also proposed a heuristic approach based on a greedy algorithm improved with a k-optimality criterion for the same problem (Hernández-Pérez & Salazar-González, 2004b) and latter extended this work by introducing new inequalities (Hernández-Pérez & Salazar-González, 2007). Other approaches for the unpaired PDTSP are based on different heuristic algorithms including a hybrid algorithm combining greedy randomized adaptive search procedure (GRASP) and variable neighborhood descent (VND) (Hernández-Pérez et al., 2009), a genetic algorithm approach (Zhao et al., 2009), an exact permutation algorithm (Martinovic et al., 2009) and a variable neighborhood search (VNS) algorithm (Mladenović et al., 2012).

In the case of multiple vehicles, Chen et al. (2014) introduced the multi-commodity unpaired pickup and delivery vehicle routing problem (PDVRP) with split loads and presented a mathematical formulation and a VNS algorithm to find near optimal solution for instances with up to 10 customers and 6 commodities. Recently, Xu et al. (2017) presented a multi-visit unpaired PDVRP, which is an extension of the model presented by Chen et al. (2014). They proposed a tabu search solution approach and the heuristic evaluation study reports an average and maximum gap between the best solution and its lower bound of 33.04% and 83.1% for instances with 5 products and a number of customers ranging from 5 to 10. Azadian et al. (2017) presented an unpaired PDVRP with time-dependent assignment cost and implements an efficient solution method based on a decomposition approach for an air cargo transportation problem. The solution method is evaluated by comparing the best solution with the optimal solution (for instances of the problem cases with 7 customers and 1 airport) and lower bound values (instances of the problem cases with 15 customers and 2 airports). For large instances, the average gap reported is approximately 11% with a worst-case scenario of 22.7% gap against the lower bound. However, these models are significantly different from the model presented in the current study where we jointly model allocation and routing decisions under limited supply conditions. Recently, Nair et al. (2016) and Nair et al. (2017a) presented a periodic unpaired PDVRP and proposed a tabu search based heuristic solution approach. These studies focus mainly on the scheduling aspect of the logistics problem.

The proposed methods to solve the food rescue and delivery problem shares some similarities with the methods proposed for PDVRP. In particular, the existing mathematical programming formulations can be used to model the routing component of the food rescue and delivery problem. In particular, illegal tours can be eliminated using a cutting-plane algorithm which iteratively identifies infeasible tours. Our main contribution is to incorporate fair allocation decision in a single-commodity pick-up and delivery vehicle routing formulation and propose customized solution methods for its resolution.

We next discuss the application context of the food rescue and delivery problem.

2.2. Humanitarian logistics

Despite their wide applicability, logistical problems have received less attention in charity operations. Earlier studies focused on the logistics of meals-on-wheels programs (Bartholdi III et al., 1983). Several studies were conducted to explore the potential use of geographical information systems (GIS) in improving the efficiency of pre-cooked meals deliveries (Gorr et al., 2001; Wong & Meyer, 1993). This work was then extended by generalizing the problem as that of organizing home deliveries, and meta-heuristics were proposed to solve this location-routing problem (Yildiz et al., 2012). Deliveries in food bank networks have received an increasing attention due to the complexity of their operations, however most studies have focused on location and routing models (Davis et al., 2014; Solak et al.,

2014). In these studies, the decisions are generally focused on locating welfare infrastructure rather than allocating resource to this infrastructure.

There is a growing literature that addresses routing and allocation policies in non-profit sectors. Most of the work focuses on humanitarian relief logistics, where the total transportation cost (Haghani, 1997; Tzeng et al., 2007; Campbell et al., 2008), the unsatisfied demand of all aid recipients (Rawls, 2009), the latest arrival time (Campbell et al., 2008; Huang et al., 2012; Nolz et al., 2010) and the total response time (Tzeng et al., 2007; Campbell et al., 2008) are minimized while travel reliability is maximized. Extensive reviews of the field can be found in Ichoua (2010) and in Luis et al. (2012).

In the context of social welfare, the priorities within logistical systems are less cost-driven than in pure profit-oriented organizations. Besides minimizing operations cost, a central approach to social welfare and humanitarian relief logistics consists in accounting for fairness-driven objectives. We next review the literature on fair allocation and routing logistics and position our paper in this field.

2.3. Fair allocation and routing

Several efforts in humanitarian relief have explored different social welfare utility functions as an indicator of equity and fairness (Campbell et al., 2008; Balcik et al., 2014; Lien et al., 2014; Tzeng et al., 2007; Huang et al., 2012). These studies show that the choice of the objective function has a significant impact on the routing structure and the resource allocation. Tzeng et al. (2007) presented a multi-objective model to identify suitable transfer points in distributing supplies from a set of pickup nodes to delivery nodes. The authors considered an egalitarian objective function to maximize the minimum service satisfaction at demand points, in addition to minimizing operational and facility setup costs. In this model, demand nodes are allowed to be visited multiple times from multiple transfer points. Carotenuto et al. (2007) presented a model for generating equitable risk routes for the transportation of hazardous materials by bounding the maximum risk sustained by the links in the network. Campbell et al. (2008) presented two objective functions for an allocation and routing problem: an egalitarian objective function to minimize the maximum arrival time of supplies, and a utilitarian objective function to minimize the sum of arrival times of supplies and discuss the impact of these objective functions on total route costs and response time. The demand nodes are allowed to be visited only once and it is ensured that demand nodes' requests are fully met. Luis et al. (2012) extended this work by considering three additional equity metrics: minimize the maximum pairwise differences in demand-weighted arrival time, minimize the standard deviation in demand-weighted arrival time and minimize the disutility-weighted arrival time. The demand nodes are allowed to be visited multiple times and it is ensured that each delivery node receives precisely the exact amount of product required. In all these studies, the authors focused on providing equitable arrival times among delivery nodes. The amount of goods delivered is fixed and assumed given prior to the optimization;

hence fairness with regards to the allocation of the supply among the delivery nodes is not a priority.

A few studies have addressed the need for fair and equitable allocation policies in food relief programs. Lien et al. (2014) presented a single-vehicle sequential resource allocation model for a food rescue program in Chicago, aimed at an effective and equitable allocation of rescued food. They considered an egalitarian welfare utility function as an indicator of equity. However, the model assumes that the order of the pickup nodes in the route is fixed and that the deliveries are performed after all the donors are visited. These constraints considerably modify the logistical problem at hand and ignore the impact on route costs. Balcik et al. (2014) extend this single route problem to a multi-route setting. They developed a decomposition-based heuristic that decomposes the problem into three stages: clustering to assign donors and agencies into a vehicle, sequencing to determine the order of the visit of the agencies and allocation to determine the food provided at each delivery node. They considered the same egalitarian objective function, *i.e.*, maximizing the minimum fill rate (satisfaction) in all the three stages. A precedence criterion imposing that all donors be visited before agencies and a route length criterion are imposed. Orgut et al. (2016) developed a joint decision-making model for fair distribution of additional storage capacities among the counties of North Carolina, to prevent food wastage due to lack of storage space. They incorporated fairness by minimizing the absolute deviation between the proportion of food delivered and the need of counties, while maximizing the total food delivered. This work was later extended by Orgut et al. (2017) which modeled a two-stage stochastic optimization model for equitable and effective distribution of food donations under stochastic receiving capacities. However, these studies are focused on designing equitable and effective inventory management strategies and does not account for travel cost. Recently, Nair et al. (2017b) proposed fair allocation and routing formulations for food rescue and re-distribution. The authors assumed that all delivery nodes shared the same utility functions and investigated the behavior of the problem for several fair allocation mechanisms. They proposed a heuristic solution algorithm that combines traditional vehicle routing operators with a tabu search approach.

Although the aforementioned efforts provide insight into the various resource allocation strategies and the challenges of food rescue and delivery logistics, there exists a lack of approaches that jointly model fair allocation and cost-effective routing. In this paper, we address this gap by proposing a new formulation for the food rescue and delivery problem without any assumption on welfare agencies' utility function and use envy-freeness as the fairness criterion for the allocation. We contribute to the field by presenting an exact cutting-plane algorithm and a new heuristic algorithm which relies on local search to efficiently find balanced solutions.

3. Formulation and Exact Solution Algorithm

In this section, we present the mathematical formulation of the food rescue and delivery problem. We show that this problem may not admit any envy-free solution and introduce a new MILP formulation for its representation. We then present a new cutting-plane algorithm based on Benders' decomposition for its exact solution.

3.1. Problem Statement

Let $G = (N, A)$ be a directed graph where N is the set of nodes and A is the set of arcs. The set of nodes is partitioned into three subsets: the depot node denoted 0, the set of pickup nodes N_p and the set of delivery nodes N_d; *i.e.*, $N = \{0\} \cup N_p \cup N_d$. Without any loss of generality, we assume that the graph is complete, *i.e.*, there exists an arc between every pair of nodes. In reality, if that is not the case, one can always find the shortest path between any two nodes and use its length to weight a virtual arc between these nodes.

Food supplies and deliveries are assumed to be divisible and represented as a single commodity. For each delivery node $i \in N_d$, we denote R_i its request (in units of food); and for each pickup node $i \in N_p$, we denote S_i its supply. In the food rescue and delivery problem presently addressed, we assume that the total food supply is lower than the total food request *i.e.*, $\sum_{i \in N_p} S_i \leqslant \sum_{i \in N_d} R_i$. If this assumption does not hold, then there is enough food supply to fully satisfy all welfare agencies and the food rescue and delivery problem reduces to an unpaired, PDVRP which has been extensively addressed in the literature. In contrast, the logistical problem at hand seeks to determine a fair allocation of the rescued food among welfare agencies and, in addition, find least-cost vehicles routes to achieve these operations. Further, we assume any uncollected food is wasted and seek a solution with zero waste.

We select envy-freeness to guide the food allocation to welfare agencies since it is a prominent fairness criterion widely used within resource allocation schemes for social welfare (Foley, 1967). Let d_i represents the food delivered at node $i \in N_d$. Formally, we say that node $i \in N_d$ envies node $j \in N_d$ if $E_{ij} > 0$; where the envy of i to j, E_{ij}, is defined as

$$E_{ij} \equiv \min\{R_i, d_j\} - d_i \tag{1}$$

Hence, node i envies node j if and only if i is not fully satisfied ($R_i > d_i$) and i is allocated less food than j ($d_j > d_i$) (Varian, 1974). Further, i can only envy j as much as $R_i - d_i$, which means that one cannot envy another for more than its unsatisfied request. An allocation is envy-free if and only if $E_{ij} = 0, \forall i, j \in N_d : i \neq j$.

Let $t_{ij} \geqslant 0$ be the travel cost on arc $(i,j) \in A$ and let L be the maximum tour length, *i.e.*, the tour of each vehicle must not exceed L travel cost units. This constraint is used to represent the available working hours during daily operations. In addition, let C be the vehicle capacity, we assume $C \geqslant \max\{\max_{i \in N_p} S_i, \max_{i \in N_d} R_i\}$. Further, we enforce that pickup and delivery nodes must be visited exactly once and that vehicles must leave and return with an empty load.

The food rescue and delivery problem examined in this paper can be stated as follows: *find the allocation of rescued food among welfare agencies which deviates the least from an envy-free allocation, and minimize travel costs, while ensuring zero waste and respecting vehicle capacity and tour length constraints*. This logistical problem is hereby referred to as the Food Rescue and Delivery Problem (FRDP).

We now show that the FRDP admits non-trivial solutions. It is well-known that assuming divisible goods, an envy-free allocation can be obtained in polynomial time using the max-min fair share algorithm summarized in Algorithm 1 (Steinhaus, 1948). This greedy algorithm returns a fair share allocation vector $\boldsymbol{F}$[1]. To see that the fair share allocation $\boldsymbol{F}$ returned by Algorithm 1 is envy-free, that is, if $d_i = F_i$ for each $i \in N_d$, then $E_{ij} = 0$ for each pair of nodes $i, j \in N_d$; observe that delivery nodes are first sorted by increasing request values R_i. Hence, as long as Algorithm 1 enters the **else** condition within the main **for** loop, the fair share value of each delivery node is equal to its request. Hence all the delivery nodes explored so far are fully satisfied and do not envy any other node. Let i' be the first delivery node such that Algorithm 1 enters the **if** condition: the fair share of i' is equal to the average total remaining supply $S/|N_d^o|$. Since i' received more than any other node previously explored, i' does not envy any of these nodes. The remaining supply S is then reduced by the average total remaining supply and i' is removed from the set N_d^o, thus the average total remaining supply remains the same. All delivery nodes explored after i' thus receive the same amount and none envies another. Further, observe that by construction the fair share allocation is a complete re-distribution

[1] We use bold face symbols to denote vectors throughout the paper.

of the available supply, *i.e.*, $\sum_{i \in N_p} S_i = \sum_{i \in N_d} F_i$.

Algorithm 1: EnvyFree

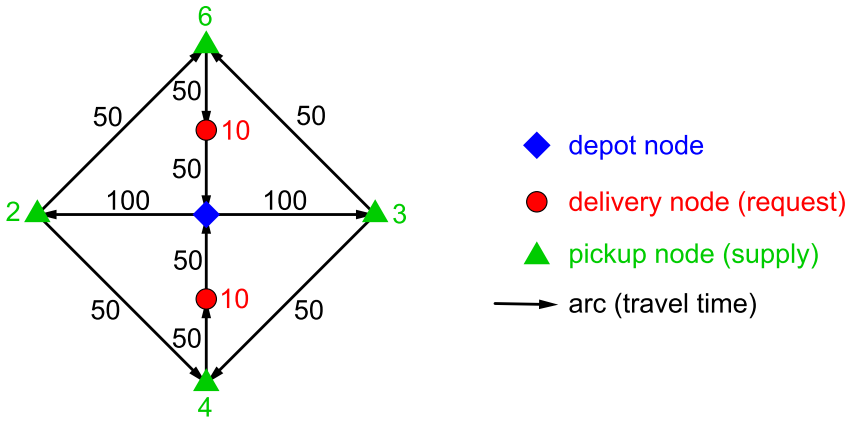
Input: N_d, N_p, $\boldsymbol{R}$, $\boldsymbol{S}$
Output: $\boldsymbol{F}$
$N_d^o \leftarrow$ Sort delivery nodes N_d by increasing request R_i
$S \leftarrow \sum_{i \in N_p} S_i$
for $i \in N_d^o$ **do**
  if $S/|N_d^o| \leqslant R_i$ **then**
    $F_i \leftarrow S/|N_d^o|$
  else
    $F_i \leftarrow R_i$
  $S \leftarrow S - F_i$
  $N_d^o \leftarrow N_d^o \setminus \{i\}$

The envy-free allocation obtained using Algorithm EnvyFree represents an ideal solution for the allocation component of the FRDP but may fail to satisfy the operational constraints imposed therein. Consider the example depicted in Figure 1a: in this toy instance the tour length constraint is set to 250 time units and vehicle capacity is assumed to be large enough to handle any combination of node visits. The envy-free allocation obtained using Algorithm EnvyFree allocates 7.5 food units to both delivery nodes. However, this allocation is infeasible due to the tour length constraint which prevents a single vehicle to visit all pickup and delivery nodes. Two feasible solutions are illustrated in Figure 1b: the left-hand side solution allocates 8 and 7 food units to delivery nodes whereas the right-hand side solution allocates 9 and 6 food units. The deviation to the envy-free allocation is 0.5 in the left solution and 1.5 in the right one, hence the left solution is preferred although none of them are envy-free.

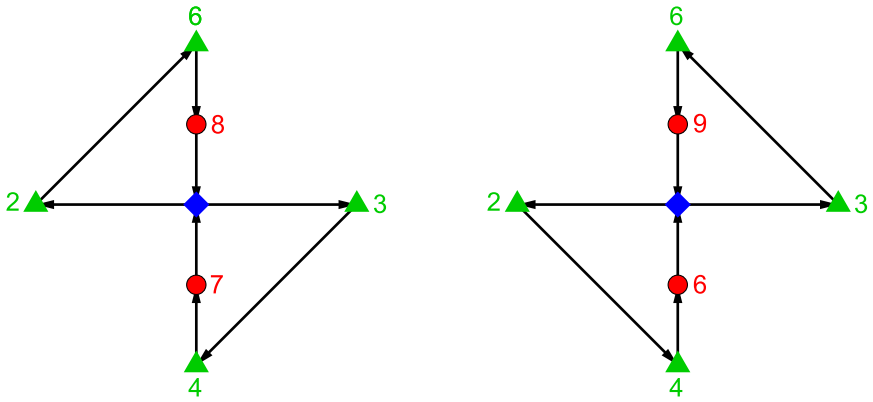
We next show that the FRDP is an NP-hard optimization problem.

Proposition 1. *The FRDP is NP-hard.*

Proof. We prove that the FRDP is NP-hard by reduction from the Hamiltonian cycle problem. Our argument is based on the reduction of the Hamiltonian cycle to the Traveling Salesman Problem. Let $G = (N, A)$ be an instance of the Hamiltonian cycle problem. To construct an instance I of the FRDP, we construct the complete graph $G' = (N', A')$ as follows: we randomly select a node in N and set it as the depot node (0). For each vertex in $n_i \in N \setminus \{0\}$, we add n_i^p and n_i^d to N'. In addition, we add n_i^p to N_p and set its supply to one unit; and we add n_i^d to N_d and set its request to one unit. For each pair $n_i^p, n_i^d \in N'$, we add a directed arc (n_i^p, v_i^d) to A' and set its travel cost to zero. Further, for each arc $(n_i, n_j) \in A$ we add an arc (n_i^d, n_i^p) to A' with a travel cost of zero. We complete the graph G' by adding an arc with a travel cost equal to one between any two unconnected nodes in N'. We set the capacity of the vehicles to one, and the tour length constraint to zero.



(a) Toy instance for the FRDP: $T = 250$ and C is assumed to be a large number. The total supply $S = 15$ and the envy-free allocation is 7.5 units of food at each delivery node.



(b) Two feasible solutions are available. The left solution allocates 8 and 7 food units, thus the deviation to the envy-free allocation is of 0.5. The right solution allocates 9 and 6 food units, thus the deviation to the envy-free allocation is 1.5.

Figure 1: Illustration of an instance of the FRDP which does not admit an envy-free solution.

We show that G has a Hamiltonian cycle if and only if I has an envy-free solution with tour of travel cost equal to 0. If G admits a Hamiltonian cycle, then one can construct a solution for I by randomly selecting a depot node and by traversing the graph using only zero travel cost arcs. This solution is envy-free since all delivery nodes receives a unit of food, has zero waste and minimizes travel cost. On the contrary, suppose that I admits an envy-free solution with total travel cost of zero, and zero waste. Since $|N_d| = |N_p|$ and node supply and request are equal to 1, the route consists of alternated pickup and delivery nodes. In addition, since the travel cost is zero, each pickup node n_i^p must be connected to n_i^d, hence the sequence of node visits can be decomposed into pairs of the form (n_i^p, n_i^d). Since waste is nil and since each node pair (n_i^p, n_i^d) in G' corresponds to a node n_i in G, it is clear that all nodes in N are visited exactly once and thus there exists a Hamiltonian cycle in G. □

We next identify necessary optimality conditions for finding envy-free solutions for the FRDP.

Proposition 2. *The FRDP admits an envy-free solution only if the total supply of the pickup nodes visited by each vehicle is equal to the total fair share of the delivery nodes visited by the same vehicle.*

Proof. Let $S(k)$ be the total supply of pickup nodes visited by vehicle k and let $F(k)$ be the total fair share of the delivery nodes visited by k. If $S(k) < F(k)$, then there must be at least one of the delivery node i visited by k which is delivered strictly less than its fair share, *i.e.*, $d_i < F_i$. Hence, by definition, i must envy at least one other node j visited by k. In turn, if $S(k) > F(k)$, since all the supply is re-distributed, *i.e.*, $\sum_{i \in N_p} S_i = \sum_{i \in N_d} F_i$, there must exist another vehicle k' such that $S(k') < F(k')$. Hence, unless for each vehicle k, $S(k) = F(k)$ the solution of the FRDP cannot be envy-free. □

Proposition 2 provides critical insight on the structure of the FRDP which will be used to design solution improving strategies within the proposed heuristic algorithm. In particular, Proposition 2 underlines the difficulty in finding envy-free solutions for multi-vehicle problems. If a single vehicle is sufficient to visit all nodes, then an envy-free solution can be constructed by ensuring that $d_i = F_i$ for each delivery node i as long as vehicle capacity and tour length constraints do not preclude this allocation (as illustrated in Figure 1). In contrast, if more than one vehicle is required to visit all nodes in the network, then the necessary optimality conditions summarized in Proposition 2 impose specific constraints on the construction of envy-free solutions.

3.2. Mathematical Representation

We now present a mathematical formulation for the FRDP. The allocation component of the FRDP is a typical fair division problem and the routing component of our formulation is based on the mixed

integer programming models for the unpaired PDVRP (Hernández-Pérez & Salazar-González, 2007; Chen et al., 2014).

We propose a 2-index formulation for the routing component of the FRDP. Let $x_{ij} \in \{0,1\}, \forall i,j \in N : i \neq j$ be a binary decision variable taking value 1 if arc (i,j) is used in a vehicle tour and 0 otherwise. The routing constraints (2)-(5) impose that each pickup and delivery node is visited exactly once and that flow is conserved at each node. Further, we impose that all tours start by visiting a pickup node and finish after visiting a delivery node.

$$\sum_{(i,j)\in A} x_{ij} = 1 \qquad \forall i \in N_p \cup N_d \tag{2}$$

$$\sum_{(i,j)\in A} x_{ji} = 1 \qquad \forall i \in N_p \cup N_d \tag{3}$$

$$\sum_{i\in N_p} x_{i0} = 0 \tag{4}$$

$$\sum_{i\in N_d} x_{0i} = 0 \tag{5}$$

Note that this formulation represents a multi-vehicle routing problem wherein the number of vehicles used is determined by the optimization and can be determined by counting the number of outgoing links routed from the depot in the solution. Further, this formulation implicitly assumes that all vehicles have homogeneous capacities. In the more realistic and flexible context of heterogeneous vehicle capacities, a 3-index formulation can be used to model each vehicle's tour appropriately. Such formulations offer a richer modeling framework but often compromise on computational performance. The proposed solution approach for the FRDP can be easily adapted to the case of heterogeneous vehicle capacities since cutting planes can be generated independently for each vehicle. Hence, we hereby assume that all vehicles have identical capacities.

To impose that vehicles' tour do not exceed the maximum tour length L, we adopt the same approach as that of Laporte et al. (1984) in which length-violating tours are treated as illegal tours and eliminated using the traditional subtour elimination constraints proposed by Dantzig et al. (1954). Let $N' \subseteq N \setminus \{0\}$ be a subset of customer nodes such that $|N'| \geqslant 2$ and let $V(N')$ be a lower bound on the number of vehicles required to visit all nodes in N'. Both traditional subtours and length-violating tours can be eliminated using the illegal tour elimination constraint (6).

$$\sum_{i,j\in N'} x_{ij} \leqslant |N'| - V(N') \quad \forall N' \subseteq N \setminus \{0\}, |N'| \geqslant 2 \tag{6}$$

In order to ensure capacity-compatible routes, we adopt the same commodity flow model introduced

by Hernández-Pérez & Salazar-González (2007). In this model, the demand of each node is assumed to be positive for pick-up nodes and negative for delivery nodes. The commodity transported (*i.e.*, food) is then represented as a flow in a network subject to vehicle capacity constraints. The allocation decision variables $d_i, \forall i \in N_d$ indicate the amount of food delivered at node i. The domain of the allocation variables is set to $0 \leqslant d_i \leqslant R_i$ to ensure that no delivery receives more than its request. These variables are linked to link flow variables $l_{ij} \geqslant 0$ representing the load of the vehicle on link $(i,j) \in A$ through constraints (7)-(10) given below.

$$\sum_{(i,j)\in A} l_{ij} - \sum_{(j,i)\in A} l_{ji} = S_i \qquad \forall i \in N_p \tag{7}$$

$$\sum_{(i,j)\in A} l_{ij} - \sum_{(j,i)\in A} l_{ji} = -d_i \qquad \forall i \in N_p \tag{8}$$

$$l_{ij} \leqslant x_{ij} C \qquad \forall (i,j) \in A \tag{9}$$

$$l_{0,j} = 0 \qquad \forall (0,j) \in A \tag{10}$$

Constraint (7) ensures that S_i units of food are picked-up after visiting node $i \in Np$ and constraint (8) ensures that d_i units of food are delivered up after visiting node $i \in Nd$. Constraint (9) impose capacity restrictions on the flow of each arc which is routed ($x_{ij} = 1$) and sets arc flow to zero otherwise ($x_{ij} = 0$). Finally, Constraint (10) imposes that all vehicles leave the depot with an empty load.

To ensure that no food is wasted, we introduce a zero-waste constraint (11) which imposes that all the rescued food must be delivered (note that this constraint enforces all vehicles to return to the depot with an empty load).

$$\sum_{i\in N_d} d_i = \sum_{i\in N_p} S_i \tag{11}$$

To address the fair-allocation component of the problem, we propose to minimize the maximum deviation from the envy-free allocation $\boldsymbol{F}$ which can be obtained efficiently with Algorithm EnvyFree.

Let $E \geqslant 0$ be a decision variable, we impose the following constraints (12) below and seek to minimize the value of E. Note that the absolute value can be linearized using two linear constraints.

$$E \geqslant |F_i - d_i| \quad \forall i \in N_d \tag{12}$$

To minimize the total travel cost, we minimize the classical TSP objective function $\sum_{(i,j)\in A} x_{ij} t_{ij}$.

The objective of the FRDP is to find an allocation which deviates the least from an envy-free allocation while minimizing the total travel cost. In this respect, the FRDP can be viewed as bi-objective optimization problem. In practice, charity organizations prioritize the fair-allocation of the

rescued food over logistical costs. Hence, a possible approach to solve the FRDP consists in adopting a goal programming formulation wherein an envy-free allocation is first sought by minimizing E subject to constraints (2)-(12); before the total travel cost is minimized subject to the same constraints as well as an upper bound on the maximum deviation from envy-freeness. Extensive computer experiments have indicated that solving the first step of this 2-step goal programming approach to optimality presented considerable challenges. This can be explained by the high level of symmetry involved in the problem if travel costs are ignored (Xu et al., 2017). To overcome this difficulty, we adopt a weighted-sum approach to solve the FRDP by combining both objectives in a single function.

Let $w \geqslant 0$, we propose the following objective function for the FRDP:

$$\min Ew + \sum_{(i,j)\in A} x_{ij}t_{ij} \tag{13}$$

This objective aims to find a compromise between finding a fair-allocation of the rescued food while minimizing the total travel cost. The weight w represents the price of envy-freeness compared to logistical costs and can be viewed as an input from the decision-maker. Since our prime objective is to find envy-free—or least deviating to—solutions, we seek sufficiently high values for w such that high deviation from envy-freeness are strongly penalized in the optimization. The numerical results presented in Section 5 will show that the proposed weighted-sum formulation can find envy-free solutions, thus achieving its prime objective.

The mathematical programming formulation for the FRDP is the MILP consisting of constraints (2)-(12), the domain of variables $\boldsymbol{d}$, $\boldsymbol{l}$, $\boldsymbol{x}$ and the objective function (13)—we denote $\mathcal{P}$ this program

and provide a summary below.

$$\mathcal{P}: \quad \min \quad Ew + \sum_{(i,j)\in A} x_{ij}t_{ij}$$

subject to

$$\sum_{(i,j)\in A} x_{ij} = 1 \qquad \forall i \in N_p \cup N_d$$

$$\sum_{(i,j)\in A} x_{ji} = 1 \qquad \forall i \in N_p \cup N_d$$

$$\sum_{i\in N_p} x_{i0} = \sum_{i\in N_d} x_{0i} = 0$$

$$\sum_{i,j\in N'} x_{ij} \leqslant |N'| - V(N') \qquad \forall N' \subseteq N \setminus \{0\}, |N'| \geqslant 2$$

$$E \geqslant F_i - d_i \qquad \forall i \in N_d$$

$$E \geqslant d_i - F_i \qquad \forall i \in N_d$$

$$\sum_{i\in N_d} d_i = \sum_{i\in N_p} S_i$$

$$\sum_{(i,j)\in A} l_{ij} - \sum_{(j,i)\in A} l_{ji} = S_i \qquad \forall i \in N_p$$

$$\sum_{(i,j)\in A} l_{ij} - \sum_{(j,i)\in A} l_{ji} = -d_i \qquad \forall i \in N_p$$

$$l_{0,j} = 0 \qquad \forall (0,j) \in A$$

$$0 \leqslant l_{ij} \leqslant x_{ij}C \qquad \forall (i,j) \in A$$

$$0 \leqslant d_i \leqslant R_i \qquad \forall i \in N_d$$

$$x_{ij} \in \{0,1\} \qquad \forall (i,j) \in A$$

Since the number of subtour elimination constraints is exponential, we use a cutting-plane algorithm to iteratively generate subtour elimination constraints. Let $\mathcal{P}_R$ be the relaxation of $\mathcal{P}$ in which constraints (6) are omitted. At each iteration of the algorithm, we solve a relaxed version of $\mathcal{P}$ and examine the solution for both types of illegal tours, *i.e.*, length-violating tours (containing the depot node) and traditional subtours (without the depot node).

For each illegal tour detected, we seek to determine the maximal value of $V(N')$. Since the FRDP is an unpaired pickup and delivery routing problem, determining a tight bound for $V(N')$ may be as hard as solving the original problem. Instead, if the link cost matrix is symmetric *i.e.*, $t_{ij} = t_{ji}$ for all $(i,j) \in A$, we can impose the following bound (Laporte et al., 1984):

$$V(N') = \left\lceil \frac{\sum_{i,j\in N'} t_{ij}x_{ij}}{L} \right\rceil \tag{14}$$

Note that this bound is obtained immediately from the current solution and the resulting cutting-plane of the form (6) will eliminate both types of illegal tours. For asymmetric problems, a relaxed version of (14) should be used as discussed in Laporte et al. (1984).

Extensive testing indicates that solving $\mathcal{P}_R$ using a traditional cutting-plane algorithm to iteratively generate subtour elimination constraints of the form (6) may be computationally challenging for problems with 30 nodes or more. We next present an alternative approach which consist of further relaxing $\mathcal{P}_R$ in order to obtain a pure ILP formulation which may be more scalable.

3.3. Benders' Decomposition and Valid Inequalities for the FRDP

The formulation $\mathcal{P}_R$ for the FRDP can be further relaxed by decomposing the problem using Benders' method (Benders, 1962). Benders' decomposition approach has been receiving a growing attention of the past few decades and a comprehensive review of its application in logistics can be found in Rahmaniani et al. (2017).

The traditional application Benders' decomposition method to MILP consists of relaxing complicating constraints in the original problem, resulting in a relaxed *master* problem which provides a lower bound (for minimization problems) at each iteration. The complicating constraints are handled in a *sub*-problem which is solved iteratively for each relaxed solution of the master problem. Traditionally, all continuous variables are projected out of the master problem, hence the master is pure ILP and the sub problem is a LP. If the sub-problem is infeasible a feasibility cut is generated to prevent the master from producing this solution again. Otherwise, the resolution of the sub-problem is successful and its objective value can be added to that of the master to determine an upper bound. If the gap between the upper and lower bounds is not closed, an optimality cut can be derived to represent the cost of this solution in the master problem. Benders' method is guaranteed to converge to an exact solution since the number of extreme points (optimality cuts) and rays (feasibility cuts) of the dual of the sub-problem is finite.

The FRDP admits a natural Benders' decomposition formulation obtained by separating binary and continuous variables. Specifically, we can further relax $\mathcal{P}_R$ by omitting Constraints (7)-(12). Let $\bar{x}_{ij}$ denote the optimal solution of the current relaxed master problem, Benders' sub problem denoted

$\mathcal{SP}$ is summarized below.

$$\mathcal{SP}(\bar{\boldsymbol{x}}): \quad \min \quad Ew \tag{15a}$$

subject to

$$E \geqslant F_i - d_i \qquad \forall i \in N_d \tag{15b}$$

$$E \geqslant d_i - F_i \qquad \forall i \in N_d \tag{15c}$$

$$\sum_{i \in N_d} d_i = \sum_{i \in N_p} S_i \tag{15d}$$

$$\sum_{(i,j) \in A} l_{ij} - \sum_{(j,i) \in A} l_{ji} = S_i \qquad \forall i \in N_p \tag{15e}$$

$$\sum_{(i,j) \in A} l_{ij} - \sum_{(j,i) \in A} l_{ji} = -d_i \qquad \forall i \in N_p \tag{15f}$$

$$l_{0,j} = 0 \qquad \forall (0,j) \in A \tag{15g}$$

$$0 \leqslant l_{ij} \leqslant \bar{x}_{ij} C \qquad \forall (i,j) \in A \tag{15h}$$

$$0 \leqslant d_i \leqslant R_i \qquad \forall i \in N_d \tag{15i}$$

Let ϵ_i^+ and ϵ_i^- be the dual variables of (15b) and (15c); let ω^+ and ω^- be the dual variables of (15d) (one per inequality); let λ_i^+ and λ_i^- be the dual variables of (15e) (one per inequality); let γ_{ij} be the dual variable of (15h) and let ρ_i be the dual variable of (15i).

Let $z \geqslant 0$ be a dummy decision variable acting as a surrogate for the contribution of the projected continuous variables in the master problem. Solving the the sub-problem $\mathcal{SP}$ and obtaining the dual variables for its constraints, we can generate the following Benders' feasibility (16) and optimality (17) cuts.

$$0 \geqslant \sum_{i \in Nd} \epsilon_i^+ F_i - \epsilon_i^- F_i + (\omega^+ - \omega^-) \sum_{i \in Np} S_i + \sum_{i \in Np} \lambda_i^+ - \lambda_i^- - \sum_{(i,j) \in A} x_{ij} \gamma_{ij} C - \sum_{i \in Nd} \rho_i R_i \tag{16}$$

$$z \geqslant \sum_{i \in Nd} \epsilon_i^+ F_i - \epsilon_i^- F_i + (\omega^+ - \omega^-) \sum_{i \in Np} S_i + \sum_{i \in Np} \lambda_i^+ - \lambda_i^- - \sum_{(i,j) \in A} x_{ij} \gamma_{ij} C - \sum_{i \in Nd} \rho_i R_i \tag{17}$$

As observed by many researchers, Benders' decomposition can suffer from a number of implementation issues including sub-problem degeneracy, slow convergence, poor lower bounds; among others (Rahmaniani et al., 2017). To overcome these obstacles, several methods have been proposed to identify Pareto-optimal cuts (Magnanti & Wong, 1981), improve branching decisions (Rei et al., 2009) and generate multiple cuts (Birge & Louveaux, 1988; Saharidis et al., 2010). Further, it should be noted that Hernández-Pérez & Salazar-González (2007) solved the 1-PDTSP by projecting the continuous link flow variables onto the binary routing variables and efficiently separating Benders' cuts in a cutting-plane approach. In the 1-PDTSP, the continuous variables do not contribute to the objective

function since only routing cost is minimized. Hence, the Benders' sub-problem therein is a feasibility problem and the algorithm converges as soon as a load-compatible solution is found. However, the FRDP involves additional continuous decision variables to model the fair allocation component of the problem, hence the resulting sub-problem obtained after decomposing is an optimization problem which attempts to minimize the deviation to envy-freeness based on the current master problem solution.

In the context of the FRDP, extensive implementation suggests that the challenge lies in the convergence of the algorithm which may require a large number of iterations. To accelerate convergence, we can take advantage of the structure of the problem to generate valid inequalities in addition to or instead of the traditional Benders' cuts (16) and (17) at each iteration of the algorithm.

For a given solution $\bar{\boldsymbol{x}}$ of the master problem $\mathcal{MP}$, the sub-problem $\mathcal{SP}$ may be infeasible in one of two situations: the tour(s) in $\bar{\boldsymbol{x}}$ are not waste-free or vehicle load is violated. In the case of waste-inducing tours, we can derive additional cuts by identifying node visit sequences which leads to a non-zero waste.

Let $P = \{0, i_1, \ldots, i_k\}$ be a node sequence corresponding to a tour in the solution $\bar{\boldsymbol{x}}$, *i.e.*, $i_k \in N_d$ and $x_{i_k,0} = 1$. For any subpath $P' = \{i_j, \ldots, n_k\}$, $j \geqslant 1$ such that $\sum_{i \in P' \cap N_p} S_i > \sum_{i \in P' \cap N_d} R_i$. Observe that since the total supply is greater than the total request of the subpath P' and since $x_{i_k,0} = 1$, this node sequence will lead to a non-zero waste. Further, this is true for all permutations of node visits in P' as long as the last node visited in P' is connected to the depot node. This leads to the following proposition.

Proposition 3. *For any subset $N' \subset N$ such that $\sum_{i \in N' \cap N_p} S_i > \sum_{i \in N' \cap N_d} R_i$, the constraint:*

$$\sum_{i,j \in N' : i \neq j} x_{ij} + \sum_{i \in N' \cap N_d} x_{i,0} \leqslant |N'| - 1 \tag{18}$$

is a valid inequality for the FRDP.

Proof. We show that if the inequality (18) is violated by a solution then this solution is infeasible. Let $N' = \{i_1, \ldots, i_k\} \subset N$ be such that $\sum_{i \in N' \cap N_p} S_i > \sum_{i \in N' \cap N_d} R_i$. If $\sum_{i,j \in N' : i \neq j} x_{ij} > |N'| - 1$, then there is a subtour in N' and thus the solution is infeasible. If $\sum_{i,j \in N' : i \neq j} x_{ij} = |N'| - 1$ and there exists $j \in \{1, \ldots, k\}$ such that $x_{i_j,0} = 1$, then $\sum_{i \in N' \cap N_p} S_i - \sum_{i \in N' \cap N_d} R_i$ units of food are wasted and the solution is infeasible. If $\sum_{i,j \in N' : i \neq j} x_{ij} = |N'| - 1$ and $\sum_{i \in N' \cap N_d} x_{i,0} = 0$, or if $\sum_{i,j \in N' : i \neq j} x_{ij} < |N'| - 1$, then the inequality (18) is verified. □

A similar valid inequality can be derived for feasible subpaths with non-zero envy.

Proposition 4. *For any subset $N' \subset N$ such that $\sum_{i \in N' \cap N_p} S_i > \sum_{i \in N' \cap N_d} F_i$, the constraint:*

$$E \geqslant \left(\sum_{i \in N' \cap N_p} S_i - \sum_{i \in N' \cap N_d} F_i \right) \left(1 - |N'| + \sum_{i,j \in N' : i \neq j} x_{ij} + \sum_{i \in N' \cap N_d} x_{i,0} \right) \tag{19}$$

is a valid inequality for the FRDP.

Proof. The proof is similar to that of Proposition 3. Let $N' = \{i_1, \ldots, i_k\} \subset N$ be such that $\sum_{i \in N' \cap N_p} S_i > \sum_{i \in N' \cap N_d} F_i$. If $\sum_{i,j \in N' : i \neq j} x_{ij} > |N'| - 1$, then there is a subtour in N' and thus the solution is infeasible. If $\sum_{i,j \in N' : i \neq j} x_{ij} = |N'| - 1$ and there exists $j \in \{1, \ldots, k\}$ such that $x_{i_j,0} = 1$, then (19) becomes $E \geqslant \sum_{i \in N' \cap N_p} S_i - \sum_{i \in N' \cap N_d} F_i$. This inequality is valid since, in this case, $\boldsymbol{x}$ defines a subpath visiting all nodes in N' and finishing at the depot node. Since the total supply of this subpath exceeds the total fair share of the same, then the deviation from envy-freeness E must be at least $\sum_{i \in N' \cap N_p} S_i - \sum_{i \in N' \cap N_d} F_i$. If $\sum_{i,j \in N' : i \neq j} x_{ij} = |N'| - 1$ and $\sum_{i \in N' \cap N_d} x_{i,0} = 0$, or if $\sum_{i,j \in N' : i \neq j} x_{ij} < |N'| - 1$, then the inequality (19) is verified since its right-hand side is always non-positive. □

The valid inequalities (18) and (19) can be added to program $\mathcal{MP}$ to tighten the formulation and eliminate all waste-inducing tours. The formulation of the master problem $\mathcal{MP}$ is summarized below.

$$\begin{aligned}
\mathcal{MP}: \quad & \min \quad z + \sum_{(i,j) \in A} x_{ij} t_{ij} \\
& \text{subject to} \\
& \qquad (2) - (6), (16), (17), (19), (18) \\
& \qquad x_{ij} \in \{0,1\} \qquad\qquad \forall (i,j) \in A
\end{aligned}$$

Since the number of valid inequalities (16),(17),(19) and (18) is exponential, an efficient separation method should be used to identified. The implementation of Benders' decomposition and the separation of these valid inequalities is next detailed.

3.4. Cutting-plane Algorithm

The proposed Benders' decomposition approach can be implemented in a cutting-plane algorithm. The master problem $\mathcal{MP}$ is first fully relaxed *i.e.*, all subtours and length-violating tours elimination constraints (6), Benders' feasibility and optimality cuts (16)-(17), and valid inequalities (18)-(19) are initially omitted, and solved using an ILP solver. Cutting-planes of type (6) are first generated until a route-compatible solution is obtained. The resulting solution $\bar{\boldsymbol{x}}$ contains no subtours or length-violating tours and Benders' sub-problem $\mathcal{SP}(\bar{\boldsymbol{x}})$ is solved. If $\mathcal{SP}(\bar{\boldsymbol{x}})$ is infeasible, Benders' feasibility

cut (16) and/or valid inequality (18) are generated and added to $\mathcal{MP}$. If $\mathcal{SP}(\bar{\boldsymbol{x}})$ is feasible, Benders' optimality cut (17) and/or valid inequality (19) are generated and added to $\mathcal{MP}$.

To determine whether all valid inequalities (18) and (19) that could be derived from the current master problem solution $\bar{\boldsymbol{x}}$ should be generated, we examine each tour and identify the subpaths of these tours which are either waste-inducing (for (18)) or envy-inducing (for (19)). This can be done in linear time by iterating backward from the last node of each tour and adding each node to N' until $\sum_{i \in N' \cap N_p} S_i > \sum_{i \in N' \cap N_d} R_i$ (for (18)) or $\sum_{i \in N' \cap N_p} S_i > \sum_{i \in N' \cap N_d} F_i$ (for (19)).

For waste-inducing valid inequalities, low-cardinality subpaths will likely be the most efficient in cutting out infeasible solutions since they dominate larger subpaths. In our implementation, we search for the shortest waste-inducing subpath (in terms of number of nodes) connected to the last node of each tour and add the corresponding valid inequality (18) if such a subpath is found. Hence, our algorithm may generate as many waste-inducing valid inequalities as the number of tours in the current solution. If no waste-inducing subpath has been found, we generate the traditional Benders' feasibility cut (16); otherwise we omit generating this cut in order to control the size of the master problem.

For envy-inducing valid inequalities, several of such inequalities could be derived for each tour. In practice, we find that focusing on subsets with a high deviation from envy-freeness provide the best compromise between overloading the master problem and tightening its formulation. Hence, we iterate backward from the last node of each tour and only add valid inequalities for which $\sum_{i \in N' \cap N_p} S_i - \sum_{i \in N' \cap N_d} F_i$ is greater or equal to that of the previous valid inequality derived from this tour (we denote δ this value in our implementation). Further, extensive testing suggests that generating traditional Benders' optimality cuts (17) helps in improving convergence, thus we systematically generate the corresponding cut if the current sub-problem $\mathcal{SP}$ is feasible.

The proposed cutting-plane algorithm is summarized in Algorithm 2 and henceforth referred to as CP-FRDP. To facilitate its presentation, we abuse notation and for each tour T in the current solution we denote $rev(T)$ the node sequence corresponding to tour T in reverse order. This exact solution algorithm will be used to benchmark the performance of the proposed heuristic algorithm for the FRDP.

Algorithm 2: CP-FRDP

Input: G, $\boldsymbol{S}$, $\boldsymbol{R}$, C, L, w
Output: $E^\star$, $R^\star$, $\boldsymbol{d}^\star$, $\boldsymbol{l}^\star$, $\boldsymbol{x}^\star$, $k^\star$
$\boldsymbol{F} \leftarrow$ EnvyFree (N_d, $\boldsymbol{R}$, $\boldsymbol{S}$)
while $UB - LB > 0$ **do**
  while *exists illegal tours* **do**
    $\bar{\boldsymbol{x}} \leftarrow$ Solve $\mathcal{MP}$
    for *each tour* $T \in \bar{\boldsymbol{x}}$ **do**
      $N' \leftarrow \{i \in T : i \neq 0\}$
      if $0 \in T$ *and* $\sum_{i,j \in N'} \bar{x}^\star_{ij} t_{ij} > L$ **then**
        Add (6) to $\mathcal{MP}$ to eliminate length-violating tour T
      if T *is a subtour* **then**
        Add (6) to $\mathcal{MP}$ to eliminate subtour T
  $LB \leftarrow z + \sum_{(i,j) \in A} \bar{x}_{ij} t_{ij}$
  $\bar{E} \leftarrow$ Solve $\mathcal{SP}(\bar{\boldsymbol{x}})$
  if $\mathcal{SP}(\bar{\boldsymbol{x}})$ *is feasible* **then**
    if $w\bar{E} + \sum_{(i,j) \in A} \bar{x}_{ij} t_{ij} < UB$ **then**
      $UB \leftarrow \sum_{(i,j) \in A} \bar{x}_{ij} t_{ij}$
      $E^\star, R^\star, \boldsymbol{d}^\star, \boldsymbol{l}^\star, \boldsymbol{x}^\star, k^\star \leftarrow$ Update best solution
    Add (17) to $\mathcal{MP}$
    for *each tour* $T \in \bar{\boldsymbol{x}}$ **do**
      $N' \leftarrow \{\}$, $\delta \leftarrow 0$
      for *each* $i \in rev(T)$ **do**
        $N' \leftarrow N' \cup \{i\}$
        if $\sum_{i \in N' \cap N_p} S_i - \sum_{i \in N' \cap N_d} F_i > \delta$ **then**
          Add (19) to $\mathcal{MP}$
          $\delta \leftarrow \sum_{i \in N' \cap N_p} S_i - \sum_{i \in N' \cap N_d} F_i$
  else
    cut $\leftarrow$ `False`
    for *each tour* $T \in \bar{\boldsymbol{x}}$ **do**
      $N' \leftarrow \{\}$
      for *each* $i \in rev(T)$ **do**
        $N' \leftarrow N' \cup \{i\}$
        if $\sum_{i \in N' \cap N_p} S_i > \sum_{i \in N' \cap N_d} R_i$ **then**
          Add (18) to $\mathcal{MP}$
          cut $\leftarrow$ `True`, **break**
    if cut = `False` **then**
      Add (16) to $\mathcal{MP}$

4. A Heuristic Algorithm for the FRDP

Most realistic food rescue and delivery operations cover large metropolitan areas with tens of pickup and delivery locations. In this section, we present a heuristic algorithm to solve the FRDP on medium to large-scale instances.

4.1. Presentation of Algorithm H-FRDP

Our proposed solution algorithm for the FRDP works by jointly seeking envy-free allocations and constructing a minimal number of tours to supply all delivery nodes. This is achieved by combining

greedy and local search techniques that aim to meet optimality conditions and construct balanced solutions. This heuristic solution algorithm is henceforth referred to as H-FRDP. Algorithm H-FRDP consists of four main procedures; namely, 1) pickup node clustering, 2) delivery nodes assignment, 3) tour planning and 4) final improvement; which are summarized in Algorithm 3.

Our approach works by iteratively solving the problem with a given number of vehicles (*i.e.*, clusters). Initially, we set the initial number of clusters $|K|$ equal to 1. At each iteration of the main loop of H-FRDP, we check the feasibility of each vehicle tour with regards to the tour length constraint as well as load and capacity constraints. We stop if all tours are feasible and increment the number of tours/clusters by one otherwise.

We next present each procedure of Algorithm H-FRDP in detail and we use a demonstration instance depicted in Figure 2 to illustrate the behavior of the proposed heuristic approach. The toy instance contains a depot node (labeled A), 10 delivery nodes (labeled from B to K) and 8 pickup nodes (labeled from L to S). The total supply is 106 and the total request is 143. The Euclidean distance between nodes is used as travel cost.

Algorithm 3: H-FRDP

Input: $G = (N, A)$, $\boldsymbol{S}$, $\boldsymbol{R}$, $\boldsymbol{t}$, C, L, ϵ, w
Output: $\boldsymbol{d}$, $\boldsymbol{\Pi}$
$|K| \leftarrow 1$
while *all tours are not feasible* **do**
  $K \leftarrow$ PickupCluster ($|K|$, N, $\boldsymbol{t}$)
  $K \leftarrow$ DeliAssign (K, N, $\boldsymbol{t}$, ϵ)
  for *each cluster* $c_k \in K$ **do**
    $\boldsymbol{d}_{c_k}, \Pi_{c_k} \leftarrow$ TourPlanning (c_k, w)
  K, $\boldsymbol{d}$, $\boldsymbol{\Pi} \leftarrow$ FinalImprovement (K, $\boldsymbol{d}$, $\boldsymbol{\Pi}$)
  if *exists infeasible tour* **then**
    $|K| \leftarrow |K| + 1$

4.2. Pickup nodes clustering

In the clustering procedure, our objective is to partition the set of pickup and delivery nodes where eventually each partition will be assigned to a vehicle. The clustering algorithm is a multi-objective procedure wherein we seek balanced clusters that 1) can provide envy-free allocations, 2) can be visited while respecting the tour length constraint and 3) minimize the total travel cost.

Several local search techniques are used within the clustering procedure and we next define some useful terminology and notation. Let c_k and $c_{k'}$ be two clusters, we denote $(c_k, i) \leftrightarrow (c_{k'}, i')$ the move consisting of *swapping* node $i \in c_k$ with $i' \in c_{k'}$. Let $\tau(c_k)$ be the total within distance of cluster c_k, *i.e.*, $\tau(c_k) = \sum_{i,j \in c_k} t_{ij}$. We say that the swap $(c_k, i) \leftrightarrow (c_{k'}, i')$ is *travel-improving* if this swap reduces the sum of $\tau(c_k)$ and $\tau(c_{k'})$. Further, we say that this swap is ϵ-*travel-improving* if

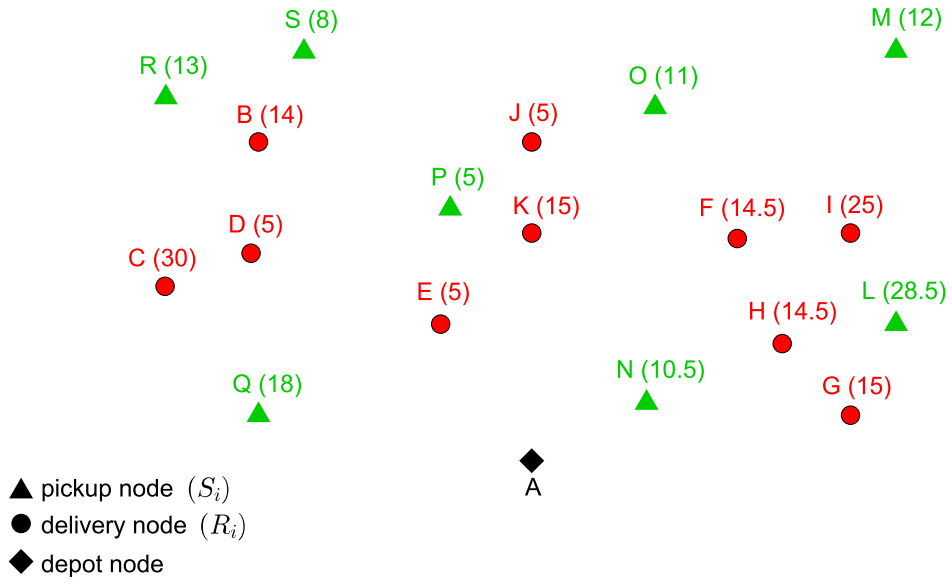


Figure 2: FRDP demonstration instance with node supply and request.

this swap reduces $\tau(c_k)$ (resp. $\tau(c_{k'})$) and $\tau(c_{k'})$ (resp. $\tau(c_k)$) increases by at most $\epsilon\%$. Finally, we define the *fair balance* of a cluster as $b(c_k) = \sum_{i \in c_k \cap N_p} S_i - \sum_{i \in c_k \cap N_d} F_i$ and we say that the swap $(c_k, i) \leftrightarrow (c_{k'}, i')$ is *balance-improving* if this swap reduces both $b(c_k)$ and $b(c_{k'})$.

To achieve the first objective, *i.e.*, find clusters with a potential to provide envy-free allocations, we build on the necessary optimality conditions for envy-free solutions given in Proposition 2 which state that to produce an envy-free solution, the total supply of each cluster must be equal to its total fair share. Since the existence of envy-free solutions is not guaranteed, we seek clusters that have the potential to minimize the deviation to envy-free allocations.

The initial partitioning of the pickup nodes is based on the postulate that the fair share of delivery nodes is near-uniform. While this may not be true, the fair share allocation given by Algorithm EnvyFree is likely to have a low variance since all high-request delivery nodes are allocated the same amount of rescued food. Using this assumption, we partition the pickup nodes such that the total supply of each partition is as close as possible to a multiple of the average fair share value. Let $g : 2^{N_p} \to \mathbb{R}$ be the *cluster-gap* function which calculates the gap between the supply of a cluster to a multiple of the average fair-share. Let $\bar{F} = \frac{1}{|N_d|} \sum_{i \in N_d} F_i$ be the average fair-share, g is defined as:

$$g(c) = \begin{cases} \frac{\sum_{i \in c} S_i}{\bar{F}} - \left\lfloor \frac{\sum_{i \in c} S_i}{\bar{F}} \right\rfloor & \text{if } \frac{\sum_{i \in c} S_i}{\bar{F}} - \left\lfloor \frac{\sum_{i \in c} S_i}{\bar{F}} \right\rfloor < \frac{1}{2} \\ \frac{\sum_{i \in c} S_i}{\bar{F}} - \left\lfloor \frac{\sum_{i \in c} S_i}{\bar{F}} \right\rfloor - 1 & \text{otherwise} \end{cases} \tag{20}$$

If the distribution of the requests of delivery nodes is uniform and if the total supply of each

partition is a multiple of the average fair share value, then an envy-free allocation can be found since the necessary optimality condition given in Proposition 2 are verified.

To ensure that the travel costs are balanced across clusters, we iterate over clusters and iteratively assign pickup nodes based on the above method. This aims to ensure that clusters have a similar cardinality. When assigning pickup nodes to clusters, we prioritize high-supply nodes to ensure that no cluster is under-supplied.

The initial pickup node partitioning may group nodes which are distant from each other, thus it may be necessary to spatially balance the clusters. After the initial partitioning of the pickup nodes, we search for *travel-improving* swaps among nodes of similar supply values. Swapping pickup nodes with similar supply values will maintain the total supply of clusters close to a multiple of the average fair share, thus preserving the potential fairness of the partitions.

This pickup clustering procedure is summarized in Algorithm 4, where the clustering procedure is henceforth referred to as PickupCluster and returns a partition K of the set of pickup nodes.

Algorithm 4: Pickup node clustering

Input: $|K|$, N, $\boldsymbol{t}$
Output: K
$N_p^o \leftarrow$ sort N_p by decreasing supply
for $k \in \{1, \ldots, |K|\}$ **do**
  Remove next node in N_p^o and assign to cluster c_k
while $|N_p^o| \neq 0$ **do**
  for $k \in \{1, \ldots, |K|\}$ **do**
    $i \leftarrow \arg\min\{g(c_k \cup \{i\}) : i \in N_p^o\}$
    $N_p^o \leftarrow N_p^o \setminus \{i\}$
    $c_k \leftarrow c_k \cup i$
$K \leftarrow c_1, \ldots, c_{|K|}$
for $c_k, c_{k'} \in K : k < k'$ **do**
  for $i \in c_k, i' \in c_{k'}$ **do**
    if $|S_i - S_{i'}| \leqslant stddev(N_p)$ **then**
      if $(c_k, i) \leftrightarrow (c_{k'}, i')$ *is travel-improving* **then**
        $c_k \leftarrow (c_k \cup \{i'\}) \setminus \{i\}$
        $c_{k'} \leftarrow (c_k \cup \{i\}) \setminus \{i'\}$

In the example, the average fair-share is $\bar{F} = 10.6$. In Figure 3, we illustrate the behavior of procedure PickupCluster. We assume that $|K| = 2$, hence the pickup nodes are partitioned into two clusters c_1 and c_2. The pickup nodes are first sorted by decreasing supply and the two largest supply nodes L (28.5) and Q (18) are assigned to cluster c_1 and c_2, respectively. The remaining of the pickup nodes are assigned based on the cluster-gap function $g(c)$ while iterating over clusters. The final assignment gives $c_1 = \{L, R, O, N\}$ and $c_2 = \{Q, P, S, M\}$ and the cluster-gap values are: $g(c_1) = -0.057$ and $g(c_2) = 0.057$. The swap $(c_1, R) \leftrightarrow (c_2, M)$ is travel-improving, thus since $|S_R - S_M|$ is less than the standard deviation of the supply values (6.75), nodes S and M are swapped.

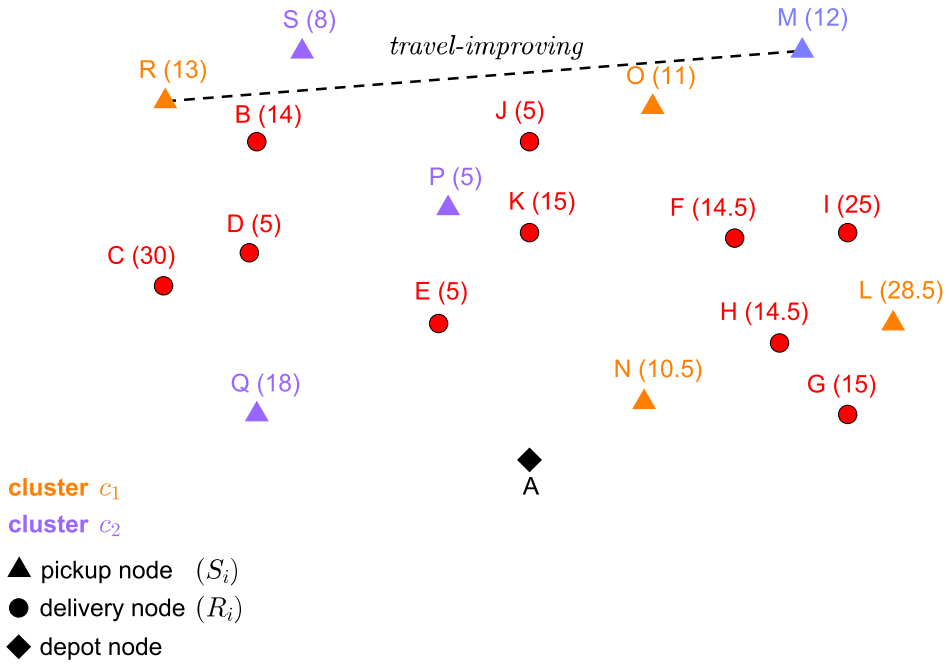


Figure 3: Illustration of Algorithm PickupCluster on the demonstration instance.

The final clusters are $c_1 = \{L, M, O, N\}$ and $c_2 = \{Q, P, S, R\}$ with total cluster-supply values of 62 and 44, respectively.

4.3. Delivery nodes assignment

Once pickup nodes have been partitioned, we assign delivery nodes by iterating over clusters and selecting the closest delivery node based on its distance to all nodes in the cluster. We then search for *balance-improving* and ϵ-*travel-improving* swaps among clusters. This local search selects the clusters with the minimal and maximal *fair balance* and explores all pairs of nodes within both clusters. If a swap is performed, clusters balance are updated and the search is repeated. We allow each pair of nodes to be swapped at most once. This delivery node assignment procedure is summarized in Algorithm 5, where the assignment procedure is henceforth referred to as DeliAssign and returns a partition K of the set of all customer nodes. We note that ϵ is an input to the procedure and a sensitivity analysis on the value of ϵ is provided in Section 5.3.

Figure 4 depicts the delivery node assignment procedure. The fair request of each node is shown besides its request. The delivery nodes are assigned to each cluster based on the total distance to the clusters' nodes while attempting to keep the total cluster fair-share below the total cluster supply. Nodes H, F, I, G, J and K are assigned to c_1 and nodes D, B, C and E are assigned to c_2. At this point clusters balance are $b(c_1) = -8$ and $b(c_2) = +8$. The move $(c_1, K) \leftrightarrow (c_2, E)$ is balance-improving since it leads to zero-balance clusters. For ϵ large enough, this move is ϵ-travel-improving

Algorithm 5: Delivery node assignment

Input: K, N, $\boldsymbol{t}$, ϵ
Output: K
$N'_d \leftarrow N_d$
$cond \leftarrow true$
while *cond* **do**
  $cond \leftarrow false$
  for $k \in \{1, \ldots, |K|\}$ **do**
    if $\sum_{i \in c_k \cap N_p} S_i > \sum_{i \in c_k \cap N_d} F_i$ **then**
      $i \leftarrow \arg\min\{\tau(c_k \cup \{i\}) : i \in N'_d\}$
      $c_k \leftarrow c_k \cup \{i\}$
      $N'_d \leftarrow N'_d \setminus \{i\}$
      $cond \leftarrow true$
if $|N'_d| \neq 0$ **then**
  for $i \in N'_d$ **do**
    $k \leftarrow \arg\min\{\tau(c_k \cup \{i\}) : k \in \{1, \ldots, |K|\}\}$
    $c_k \leftarrow c_k \cup \{i\}$
    $N'_d \leftarrow N'_d \setminus \{i\}$
$cond \leftarrow true$
while *cond* **do**
  $cond \leftarrow false$
  $\overline{k} \leftarrow \arg\max\{b(c_k) : k \in \{1, \ldots, |K|\}\}$
  $\underline{k} \leftarrow \arg\min\{b(c_k) : k \in \{1, \ldots, |K|\}\}$
  for $i \in c_{\overline{k}}, j \in c_{\underline{k}}$ **do**
    if $(c_{\overline{k}}, i) \leftrightarrow (c_{\underline{k}}, j)$ *is balance-improving* **then**
      if $(c_{\overline{k}}, i) \leftrightarrow (c_{\underline{k}}, j)$ *is* ϵ*-travel-improving* **then**
        $c_{\overline{k}} \leftarrow (c_{\overline{k}} \cup \{j\}) \setminus \{i\}$
        $c_{\underline{k}} \leftarrow (c_{\underline{k}} \cup \{i\}) \setminus \{j\}$
        $cond \leftarrow true$

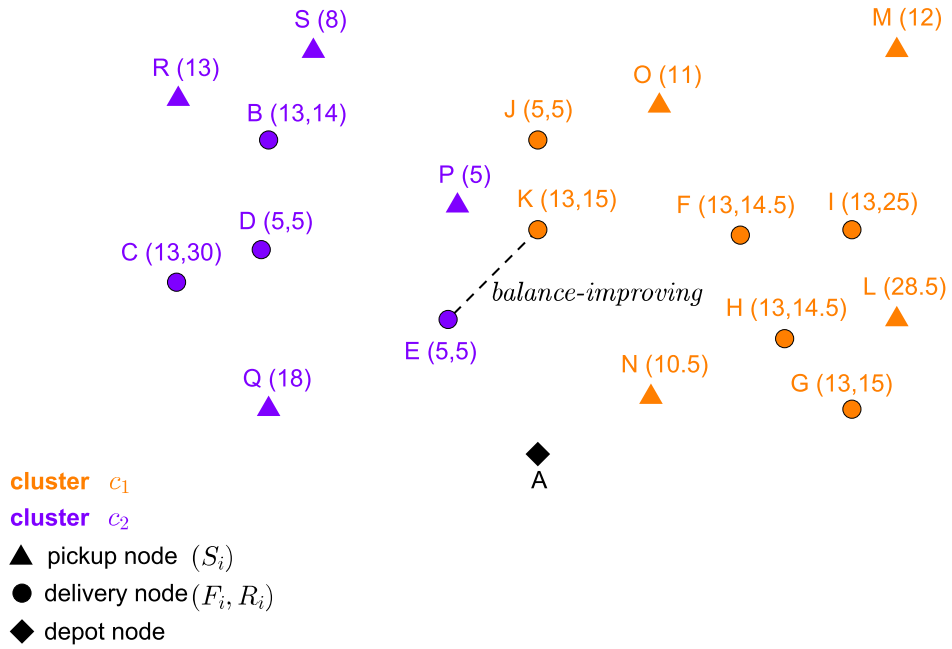


Figure 4: Delivery node assignment

and nodes K and E are swapped. The final clusters are then $c_1 = \{L, M, O, N, H, F, I, G, J, E\}$ and $c_2 = \{Q, P, S, R, D, B, C, K\}$ and both clusters have a zero balance.

4.4. The routing procedure

Once all pickup and delivery nodes have been partitioned into balanced clusters, we attempt to find a tour within each cluster respecting tour length and vehicle load constraints. Our approach is based on the first heuristic proposed by Hernández-Pérez & Salazar-González (2004b) which solve the 1-PDTSP using a combination of greedy and local search procedures. The algorithm proposed in Hernández-Pérez & Salazar-González (2004b) assumes fixed demands at all nodes and further assumes that the vehicle can leave the depot with any load (empty, partial, full) to facilitate routing operations. To adapt this algorithm to the FRDP, we start by fixing the delivery variables to their local fair-share values. Local fair-share values are determined by executing Algorithm EnvyFree for each cluster wherein the total supply available is equal to the sum of the pickup nodes' supply in the cluster and are denoted F_i' for each $i \in N_d$. In the FRDP, vehicles are assumed to leave the depot empty, hence, the heuristic proposed by Hernández-Pérez & Salazar-González (2004b) cannot be applied trivially. Our implementation consists in using a multi-start approach combined with three nearest-neighbor (greedy) searches, each of which using a different mapping of the link travel cost t_{ij} and a local optimization procedure. Upon completion of the routing procedure in each cluster, a tour-improvement procedure is executed in an attempt to improve the final tours by swapping nodes

among clusters.

Let q_i be the *demand* of node $i \in N$: $q_i = 0$ if $i = 0$, $q_i = S_i$ if $i \in N_p$ and $q_i = -F'_i$ if $i \in N_d$. The first greedy search uses the initial link costs t_{ij}, the second greedy search—based on Hernández-Pérez & Salazar-González (2004b)—uses a mapping $t^{=}_{ij}$ penalizing links with similar demands and the third greedy search uses a mapping $t^{\neq}_{ij}$ penalizing links with different demands. The link travel cost mappings $t^{=}_{ij}$ and $t^{\neq}_{ij}$ are defined as:

$$t^{=}_{ij} = t_{ij} + \frac{100}{\sum_{i \in N_p} S_i}(C - |q_i - q_j|) \tag{21}$$

$$t^{\neq}_{ij} = t_{ij} + \frac{100}{\sum_{i \in N_p} S_i}(C - |q_i + q_j|) \tag{22}$$

The motivation for these mappings is to promote tours which consists in either alternating pickup and delivery nodes in order to avoid vehicle load violations ($t^{=}_{ij}$) or visiting a maximum of pickup nodes before delivery nodes. Extensive experimentation indicates that all three mappings (nominal, similar, different) have a potential to find optimal solutions depending on the instance topology and demand distribution. As in Hernández-Pérez & Salazar-González (2004b), we say that a link (i, j) is infeasible if $q_i + q_j > C$ and we penalize infeasible links by setting its cost to $+\infty$.

Each greedy search starts from the depot or a pickup-node and attempts to find the nearest node—as determined by the mapping used—without violating vehicle load constraints. If such a node cannot be found the node minimizing the "infeasibility criterion" used in Hernández-Pérez & Salazar-González (2004b) is selected. A 2-opt procedure (Lin, 1965) minimizes the objective function (13) subject to load constraints is then applied in an attempt to find feasible tours. The shortest load-feasible tour found after executing the three greedy and local searches from each starting node is stored as the best tour for this cluster. Note that this tour may be infeasible—it can be load- or length-violating. In all our experiments, load-feasible tours are always found but such tours may violate the tour length constraints. The pseudo-code of this procedure is summarized in Algorithm 6 and is henceforth referred to as TourPlanning. For each cluster, TourPlanning returns a path $\Pi^{\star}_{c_k}$ which visit all nodes in c_k and provides local fair-share deliveries, *i.e.*, each tour is locally envy-free.

Note that although the proposed heuristic is designed to work for vehicles with homogeneous capacities, the routing procedure can be adapted to work with heterogeneous vehicles capacities since load-feasibility is verified for each vehicle independently.

Once all tours have been established, a final procedure attempts to improve the tours by exchanging nodes within the tours and re-computing the maximum deviation to envy-freeness and the total travel cost. This procedure iterates over pairs of clusters and considers exchanging pairs of pickup or delivery

Algorithm 6: TourPlanning

Input: c_k
Output: $\boldsymbol{d}^\star_{c_k}$, $\Pi^\star_{c_k}$, w
$\boldsymbol{d}^\star_{c_k} \leftarrow$ EnvyFree $(c_k, \boldsymbol{R}, \boldsymbol{S})$
$N_p^k \leftarrow$ Sort $c_k \cap N_p$ by decreasing supply
for $i \in \{0\} \cup N_p^k$ **do**
  $\Pi_{c_k} \leftarrow$ nearest neighbor (i, t_{ij})
  $\Pi_{c_k} \leftarrow$ 2-opt(Π_{c_k}, w)
  $\Pi^=_{c_k} \leftarrow$ nearest neighbor $(i, t^=_{ij})$
  $\Pi^=_{c_k} \leftarrow$ 2-opt$(\Pi^=_{c_k}, w)$
  $\Pi^\neq_{c_k} \leftarrow$ nearest neighbor $(i, t^\neq_{ij})$
  $\Pi^\neq_{c_k} \leftarrow$ 2-opt$(\Pi^\neq_{c_k}, w)$
  $\Pi^i_{c_k} \leftarrow$ least travel cost load-feasible path among $\Pi_{c_k}, \Pi^=_{c_k}, \Pi^\neq_{c_k}$
  if *travel cost of* $\Pi^i_{c_k}$ *improves on travel cost of* $\Pi^\star_{c_k}$ **then**
    $\Pi^\star_{c_k} \leftarrow \Pi^i_{c_k}$

nodes. Specifically, a pair of nodes of the same type (pickup or delivery) is swapped if the move is *Pareto-improving*, *i.e.*, at least one of the maximum deviation from envy-freeness or the total travel costs strictly decreases and the other is non-increasing. For every exchange considered, algorithm EnvyFree is executed on the newly obtained clusters to determine local fair-share values which are then used for deliveries. This final improvement procedure is summarized in Algorithm 7 and is henceforth referred to as FinalImprovement.

Algorithm 7: FinalImprovement

Input: K, $\boldsymbol{d}$, $\boldsymbol{\Pi}$
Output: K, $\boldsymbol{d}$, $\boldsymbol{\Pi}$
for $c_k, c_{k'} \in K : k < k'$ **do**
  for $i \in c_k, i' \in c_{k'}$ **do**
    if $i, i' \in N_p$ *or* $i, i' \in N_d$ **then**
      if $(c_k, i) \leftrightarrow (c_{k'}, i')$ *is Pareto-improving* **then**
        Swap i and i' in $\Pi^\star_{c_k}$ and $\Pi^\star_{c'_k}$
        $\boldsymbol{d}^\star_{c_k} \leftarrow$ EnvyFree $(c_k, \boldsymbol{R}, \boldsymbol{S})$
        $\boldsymbol{d}^\star_{c_{k'}} \leftarrow$ EnvyFree $(c_{k'}, \boldsymbol{R}, \boldsymbol{S})$

5. Numerical Experiments

In this section, we present the numerical results for the FRDP. The results are organized in four experiments: we first explore the performance of Algorithm CP-FRDP on small to medium scale instances with regards to the maximum tour length L, vehicle capacity C and the trade-off parameter w. We next compare the performance of Algorithms CP-FRDP and H-FRDP on the same instances. We then examine the performance of Algorithm H-FRDP on larger instances with

varying node distributions. Finally, we consider a realistic case study on an instance representative of the operations of a food rescue organization—OzHarvest—in Sydney, Australia (OzHarvest, 2017); and conduct an in-depth analysis of the solution for this experiment. In all experiments, the real data gathered from OzHarvest's operations was used to inform the generation of artificial and realistic instances.

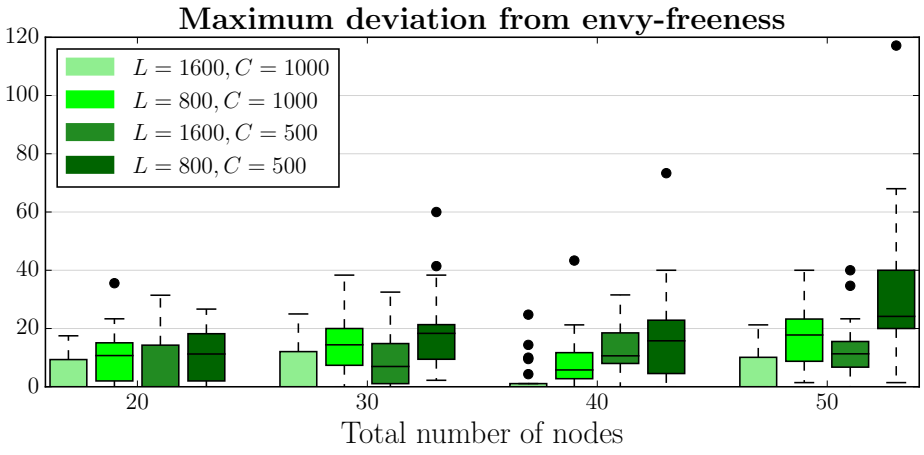
We use AMPL (Fourer et al., 2002) and CPLEX (CPLEX, 2009) to implement Algorithm CP-FRDP. We set time limits of 300s per solve and 1h per instance, and an optimality gap of 1%. Algorithm H-FRDP is implemented in Java. Both Algorithms CP-FRDP and H-FRDP are implemented on an iCore 5 machine with 16Gb of RAM and a CPU of 3.6GHz.

5.1. *Sensitivity analysis of Algorithm* CP-FRDP

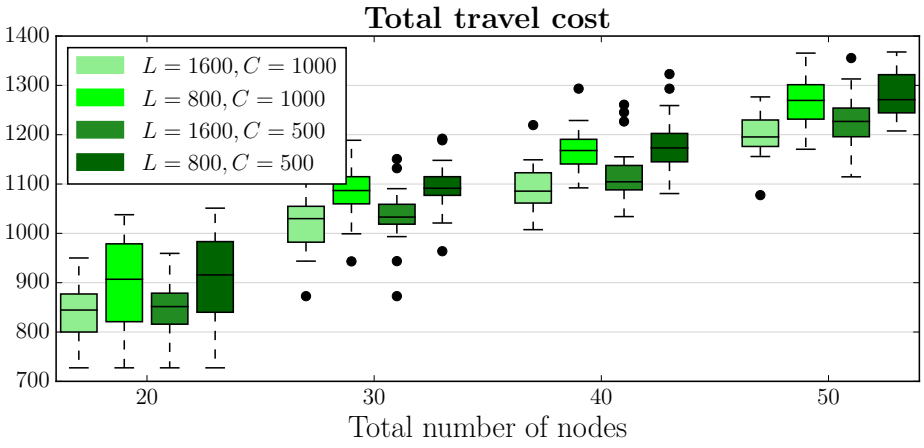
For this experiment, we use four sets of 20 artificial instances with a total of 20, 30, 40 and 50 nodes. In these instances, a region of 100x100 is used with the depot node at its center. The supply values and the request values are randomly generated in the range $[1, 200]$ and in such a way that the total supply is strictly less than the total request. We consider a base case scenario wherein $L = 1600$ and $C = 1000$ and we explore the impact of halving either or both of these values onto the algorithm performance. Hence a total of 240 FRDP instances are solved.

We first examine the performance of Algorithm CP-FRDP for a combination of maximum tour length (L) and vehicle capacity (C) values. The results of this sensitivity analysis are presented in Figure 5. Three metrics are reported using statistics for each group of 20 instances: the maximum deviation from envy-freeness (Figure 5a), the total travel cost (Figure 5b) and the computation time (Figure 5c). We find that the optimal deviation from envy-freeness only marginally increases with the number of nodes. For instances with 40 and 50 nodes, halving the value of L or C appears to considerably penalize the maximum deviation from envy-freeness compared to the base case ($L = 1600$, $C = 1000$). In contrast, on instances with 20 and 30 nodes, the effect of reducing C is less penalizing compared to that of reducing L. The total travel cost consistently increases with the number of nodes. Vehicle capacity reductions are found to negatively impact the total travel cost but by a lesser amount than reducing the maximum tour length. In terms of computation time, we observe that most instances with less than 40 nodes are solved in less than 5 minutes. Among the 50-node instances, the majority is solved in less than 15 minutes. Only a few instances, mainly under the most resource-constrained setting ($L = 800$, $C = 500$), are not fully solved within the available runtime (1h), however the gap to optimality upon termination was inferior to 0.1%. This highlights the ability of the proposed exact approach to efficiently tackle small to medium size food rescue logistics problems.

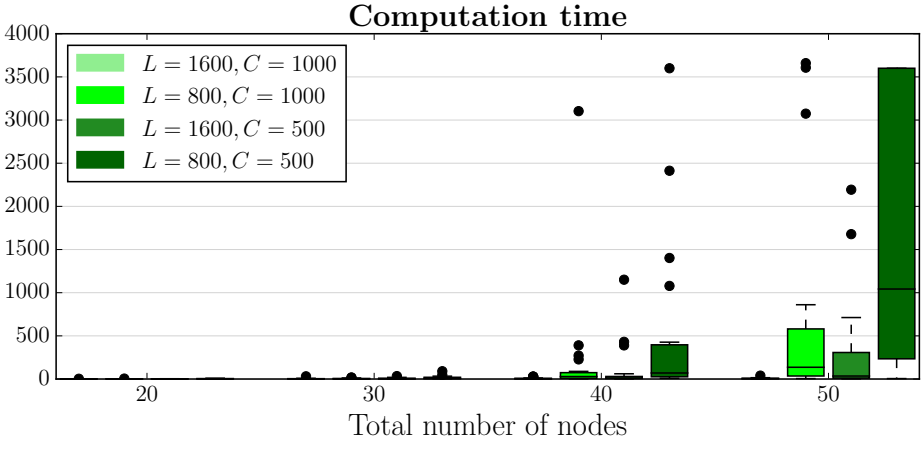
To provide insight into the effectiveness of valid inequalities (18) and (19) in reducing runtime, we reproduced this numerical experiment using a traditional Benders' decomposition approach wherein



(a)



(b)



(c)

Figure 5: Sensitivity of Algorithm CP-FRDP to the maximum tour length L and vehicle capacity C on small-scale instances with $|N| = 20$, 30, 40 and 50 nodes, and $w = 1$.

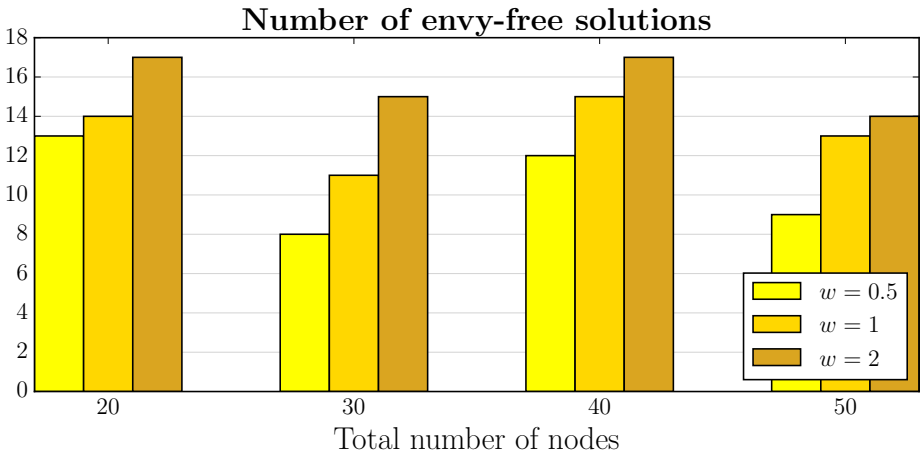


Figure 6: Sensitivity of Algorithm CP-FRDP to parameter w on instances with $|N| = 20$, 30, 40 and 50 nodes, $L = 1600$ and $C = 1000$.

only the feasibility cuts (16) and optimality cuts (17) are generated within Algorithm CP-FRDP—*i.e.*, no valid inequalities (18) or (19) are generated. We report that across all the instances solved, generating the valid inequalities (18) and (19) reduces runtime by 10.4%, on average, compared to a traditional Benders' decomposition approach. Further, we find that the proposed valid inequalities are particularly effective when solving larger instances: focusing on the 50-node problems solved, we observe an average runtime reduction of 22.9%, thus highlighting the potential of these cuts to tackle difficult problems.

We now examine the impact of the weight parameter w onto the number of envy-free solutions found. For this case study, we focus on the base case scenario wherein the vehicle capacity is set to $C = 1000$ and the maximum tour length is set to $L = 1600$. The results are depicted in Figure 6. Increasing w from 0.5 to 2 consistently increases the number of envy-free solutions found. For a given w-value, we observe that increasing the size of the instance often but not systematically reduces the number of envy-free solutions. For 20 nodes a maximum of 17 envy-free solutions are found for $w = 2$ whereas only 7 envy-free solutions are found among the 50-node tests with $w = 0.5$. Overall, $w = 1$ provides a balanced outcome with 11 (30-node) to 15 (40-node) envy-free solutions obtained.

5.2. Benchmark of Algorithms CP-FRDP *and* H-FRDP

We next benchmark Algorithms CP-FRDP and H-FRDP on the same instances used in the previous experiment *i.e.*, with 20 to 50 nodes. The tolerance parameter ϵ in Algorithm H-FRDP is set to 10% throughout the numerical experiments. For this benchmark, we set $w = 1$ and we focus on two extreme case settings: the base case setting wherein $L = 1600$ and $C = 1000$ and the most resource-constrained setting $L = 800$ and $C = 500$. In the former, the problem constraints are seldom

binding whereas, in the latter, tour length and capacity restrictions considerably restrict the solution space.

The numerical results are summarized in Tables 1-4 for 20- to 50-node instances, respectively. The columns of the tables are described from left to right. The first column is the instance ID, the second column indicates the values of L and C. The next eleven columns list the results of CP-FRDP: $E^{\star}_{CP}$ is the maximum deviation from envy-freeness, $R^{\star}_{CP}$ is the total route travel cost, $k^{\star}_{CP}$ is the number of tours, n_{it} is the number of iterations of the outer **while** loop in Algorithm CP-FRDP, $n_{\mathcal{MP}}$ is the number of iterations in the inner **while** loop in Algorithm CP-FRDP (number of times program $\mathcal{MP}$ is solved), n_L is the number of length-violating tours, n_S is the number of subtours, n_o is the number of valid inequalities (19) generated, n_f is the number of valid inequalities (18) generated, and Time indicates the computation time in seconds. The next five columns summarize the results of H-FRDP: Δ Obj. is the relative difference between the objective function in percent, $\Delta E^{\star}$ is the absolute difference between the maximum deviations from envy-free $\Delta E^{\star} = E^{\star}_{H} - E^{\star}_{CP}$ (where $E^{\star}_{H}$ is the maximal deviation from envy-free obtained with Algorithm H-FRDP), $\Delta R^{\star}$ is the relative difference between the route travel cost in percent $\Delta R^{\star} = (R^{\star}_{H} - R^{\star}_{CP})/R^{\star}_{H}$ where $R^{\star}_{H}$ is the total travel cost obtained with Algorithm H-FRDP), $\Delta k^{\star}$ is the absolute deviation between the number of tours $k^{\star}_{H} - k^{\star}_{CP}$ (where $k^{\star}_{H}$ is the number of vehicles obtained with Algorithm H-FRDP) and Time indicates the computation time of Algorithm H-FRDP in seconds.

All the instances solved with Algorithm CP-FRDP under the base case setting ($L = 1600$, $C = 1000$) only require a single vehicle. In turn, under the resource-constrained setting ($L = 800$, $C = 500$), all instances with 30 nodes or more require two vehicles. Only three 20-node instances (see Table 1, 20-2, 20-3 and 20-7) can be solved with a single vehicle under this problem configuration. The objective function value and the number of iterations (n_{it}) almost always increase when logistical resources are reduced, thus highlighting the potential hardness of the FRDP. While correlated with n_{it}, the number of master problem iterations ($n_{\mathcal{MP}}$) can vary significantly from an instance to another and may have a considerable impact on computation time. Overall, we find that if $n_{\mathcal{MP}}$ is relatively high (see Table 2, 30-10, 30-11, Table 3, 40-4, 40-7, 40-17, and Table 4, 50-3, 50-13, 50-15), for the base case setting, this will substantially increase for the resource-constrained setting. In this case, since the topology of the instance remains unchanged, the likelihood of finding illegal tours after solving $\mathcal{MP}$ remains high. Similarly, we observe that such difficult instances lead to a significant generation of valid inequalities (18), designed to eliminate waste-inducing tours, while less valid inequalities (19) are often generated.

Comparing the performance of the proposed heuristic algorithm with the optimal solution, we find that Algorithm H-FRDP scales effectively under the base case setting. In this context, across all instances, the maximum objective function gap remains under 12% and the average objective

		CP-FRDP											H-FRDP				
Instance	L, C	Obj. (Gap-%)	$E^{\star}_{CP}$	$R^{\star}_{CP}$	$k^{\star}_{CP}$	n_{it}	$n_{\mathcal{MP}}$	n_L	n_S	n_o	n_f	Time (s)	Δ Obj.	$\Delta E^{\star}$	$\Delta R^{\star}$(%)	$\Delta k^{\star}$	Time (s)
20-1	1600, 1000	918.7 (0.0)	0.0	918.7	1	0	12	0	19	0	4	0.7	2.2	0.0	2.2	0	1.7
	800, 500	1021.4 (1.0)	26.7	994.7	2	12	27	5	16	9	10	4.9	6.6	-6.7	7.3	0	1.8
20-2	1600, 1000	761.4 (0.0)	0.0	761.4	1	0	5	0	12	0	0	0.0	0.0	0.0	0.0	0	1.6
	800, 500	761.4 (0.0)	0.0	761.4	1	0	5	0	12	0	0	0.0	0.0	0.0	0.0	0	1.6
20-3	1600, 1000	783.5 (0.0)	0.0	783.5	1	2	10	0	16	4	0	0.8	0.0	0.0	0.0	0	1.6
	800, 500	783.5 (0.0)	0.0	783.5	1	2	10	0	16	4	0	2.0	0.0	0.0	0.0	0	1.7
20-4	1600, 1000	870.2 (0.0)	0.0	870.2	1	1	12	0	18	1	1	1.2	0.0	0.0	0.0	0	1.5
	800, 500	904.0 (0.0)	0.0	904.0	2	2	10	3	12	0	2	0.8	12.8	20.0	11.1	0	1.9
20-5	1600, 1000	842.6 (0.0)	0.0	842.6	1	0	8	0	17	0	0	0.0	2.5	0.0	2.5	0	1.4
	800, 500	919.9 (0.1)	10.0	909.9	2	1	12	2	18	1	0	1.6	13.0	-10.0	13.9	0	1.5
20-6	1600, 1000	846.5 (0.0)	0.0	846.5	1	0	6	0	10	0	0	0.0	0.0	0.0	0.0	0	1.5
	800, 500	948.1 (0.8)	18.6	929.6	2	6	14	2	11	9	1	2.2	0.0	1.4	-0.7	0	1.9
20-7	1600, 1000	727.4 (0.0)	0.0	727.4	1	0	3	0	9	0	0	0.0	0.0	0.0	0.0	0	1.6
	800, 500	727.4 (0.0)	0.0	727.4	1	0	3	0	9	0	0	0.0	0.0	0.0	0.0	0	1.6
20-8	1600, 1000	859.5 (1.0)	8.8	850.8	1	1	10	0	15	1	4	0.9	0.0	-8.8	0.1	0	1.4
	800, 500	947.7 (-0.5)	12.5	935.2	2	2	15	3	14	4	7	2.1	7.5	27.5	5.0	0	1.6
20-9	1600, 1000	937.4 (0.7)	17.5	919.9	1	2	8	0	14	4	1	0.7	0.0	-17.5	1.7	0	1.7
	800, 500	1017.5 (1.0)	17.5	1000.0	2	1	8	1	16	2	1	0.3	2.4	22.5	0.3	0	1.9
20-10	1600, 1000	806.5 (0.9)	13.3	793.1	1	3	18	0	21	5	3	2.2	0.0	-13.3	0.9	0	1.5
	800, 500	863.8 (0.3)	20.0	843.8	2	11	26	3	22	2	6	6.7	13.8	-20.0	15.7	0	1.5
20-11	1600, 1000	805.7 (0.0)	0.0	805.7	1	2	8	0	15	3	0	0.3	0.1	0.0	0.1	0	1.9
	800, 500	837.8 (0.5)	8.9	828.9	2	9	19	2	17	12	0	4.3	13.5	-8.9	14.4	0	2.1
20-12	1600, 1000	875.4 (0.8)	14.3	861.1	1	1	5	0	14	1	0	0.2	3.1	-14.3	4.7	0	1.8
	800, 500	1000.0 (0.9)	14.3	985.7	2	6	18	5	15	3	6	1.8	2.4	5.7	1.9	0	2.0
20-13	1600, 1000	944.0 (0.0)	0.0	944.0	1	1	20	0	19	1	7	2.2	0.0	0.0	0.0	0	1.7
	800, 500	1057.7 (0.6)	6.7	1051.1	2	18	46	8	16	6	25	9.5	14.2	13.3	13.4	0	1.9
20-14	1600, 1000	874.8 (0.0)	0.0	874.8	1	0	7	0	8	0	4	0.5	0.0	0.0	0.0	0	1.9
	800, 500	999.5 (0.9)	23.3	976.2	2	5	24	4	10	2	13	6.0	13.4	-3.3	13.9	0	1.7
20-15	1600, 1000	815.3 (0.9)	14.3	801.0	1	1	12	0	16	1	3	1.2	2.2	-14.3	3.9	0	1.4
	800, 500	936.0 (0.9)	14.3	921.7	2	7	24	1	18	4	10	7.2	4.2	25.7	1.6	0	1.4
20-16	1600, 1000	884.0 (0.0)	0.0	884.0	1	1	11	0	17	2	1	0.5	0.0	0.0	0.0	0	2.0
	800, 500	992.5 (0.9)	10.0	982.5	2	11	22	3	14	21	3	4.4	11.6	16.7	10.4	0	2.2
20-17	1600, 1000	797.0 (0.0)	0.0	797.0	1	0	4	0	10	0	0	0.0	0.0	0.0	0.0	0	1.8
	800, 500	797.0 (0.0)	0.0	797.0	1	0	4	0	10	0	0	0.0	0.0	0.0	0.0	0	1.8
20-18	1600, 1000	812.3 (0.0)	0.0	812.3	1	0	7	0	15	0	1	0.4	0.0	0.0	0.0	0	1.7
	800, 500	903.1 (0.3)	2.7	900.4	2	3	14	1	17	1	4	2.9	13.8	17.3	12.4	0	2.0
20-19	1600, 1000	838.6 (0.9)	11.1	827.5	1	4	25	0	23	5	10	3.1	2.1	-11.1	3.4	0	1.3
	800, 500	920.1 (0.4)	20.0	900.1	2	8	24	2	21	4	10	7.3	9.5	0.0	9.7	0	1.5
20-20	1600, 1000	950.1 (0.0)	0.0	950.1	1	0	9	0	14	0	1	0.6	0.0	0.0	0.0	0	1.8
	800, 500	1046.8 (0.4)	18.1	1028.7	2	5	24	6	17	2	5	4.4	4.6	-18.1	6.2	0	1.8

Table 1: Benchmark of Algorithms CP-FRDP and H-FRDP on 20-node instances

		CP-FRDP											H-FRDP				
Instance	L, C	Obj. (Gap-%)	$E^{\star}_{CP}$	$R^{\star}_{CP}$	$k^{\star}_{CP}$	n_{it}	$n_{\mathcal{MP}}$	n_L	n_S	n_o	n_f	Time (s)	Δ Obj.	$\Delta E^{\star}$	$\Delta R^{\star}(\%)$	$\Delta k^{\star}$	Time (s)
30-1	1600, 1000	965.7 (0.8)	22.0	943.7	1	3	15	0	29	7	3	1.9	0.4	-22.0	2.6	0	2.4
	800, 500	1043.0 (1.0)	22.0	1021.0	2	5	21	6	28	9	7	4.0	12.1	24.7	10.5	0	4.2
30-2	1600, 1000	1005.6 (1.0)	23.3	982.3	1	2	10	0	20	3	1	1.1	0.4	-23.3	2.8	0	2.9
	800, 500	1101.9 (1.0)	18.9	1083.0	2	11	20	4	23	1	1	4.7	15.6	51.1	12.3	0	4.7
30-3	1600, 1000	1035.6 (0.0)	0.0	1035.6	1	3	27	0	30	6	8	3.7	0.5	0.0	0.5	0	2.0
	800, 500	1097.2 (0.8)	17.3	1079.9	2	15	27	1	26	3	9	5.5	17.7	52.7	14.5	0	3.2
30-4	1600, 1000	1054.3 (0.0)	0.0	1054.3	1	1	12	0	27	1	1	1.3	0.1	0.0	0.1	0	2.5
	800, 500	1120.7 (0.2)	2.3	1118.4	2	3	14	4	27	0	2	3.0	14.9	57.7	11.0	0	4.6
30-5	1600, 1000	1039.4 (0.8)	7.8	1031.6	1	0	5	0	19	0	0	0.0	2.0	-7.8	2.7	0	2.4
	800, 500	1080.1 (0.7)	7.8	1072.3	2	2	15	7	23	2	1	2.9	7.4	32.2	4.8	0	3.9
30-6	1600, 1000	957.7 (-1.2)	11.4	946.3	1	1	9	0	26	2	0	0.3	1.2	-11.4	2.3	0	2.8
	800, 500	1131.3 (0.7)	41.4	1089.8	2	47	80	23	37	19	23	24.1	5.7	-21.4	7.6	0	3.9
30-7	1600, 1000	1082.5 (0.9)	10.0	1072.5	1	4	44	0	41	5	22	8.7	0.3	-10.0	1.2	0	1.8
	800, 500	1115.5 (-0.5)	10.0	1105.5	2	5	41	18	27	4	19	8.4	12.1	50.0	8.6	0	3.3
30-8	1600, 1000	1028.1 (0.0)	0.0	1028.1	1	0	6	0	19	0	0	0.0	2.3	0.0	2.3	0	2.9
	800, 500	1133.7 (0.9)	20.0	1113.7	2	10	36	12	26	15	8	7.9	8.2	0.0	8.4	0	4.0
30-9	1600, 1000	883.8 (0.7)	11.1	872.7	1	1	7	0	22	1	0	0.6	0.0	-11.1	0.8	0	2.2
	800, 500	974.9 (0.9)	11.1	963.8	2	19	73	18	34	12	31	33.1	8.2	-11.1	9.2	0	3.6
30-10	1600, 1000	1105.9 (0.0)	0.0	1105.9	1	7	103	0	64	11	52	31.4	0.8	0.0	0.8	0	2.6
	800, 500	1213.0 (1.0)	21.1	1191.9	2	42	147	54	40	1	69	90.5	17.8	38.9	15.8	0	4.4
30-11	1600, 1000	990.2 (0.8)	20.0	970.2	1	1	48	0	41	1	25	7.4	0.0	-20.0	1.8	0	3.1
	800, 500	1090.5 (0.9)	60.0	1030.5	2	36	106	22	36	10	54	46.5	12.6	-20.0	14.7	0	3.9
30-12	1600, 1000	989.8 (0.0)	0.0	989.8	1	0	10	0	31	0	1	0.7	0.1	0.0	0.1	0	1.9
	800, 500	1070.3 (0.6)	6.2	1064.1	2	8	41	14	40	2	11	8.4	7.9	58.8	3.0	0	3.2
30-13	1600, 1000	1070.4 (0.9)	14.0	1056.4	1	2	20	0	27	2	4	1.8	0.1	-14.0	1.4	0	3.1
	800, 500	1122.7 (0.7)	18.0	1104.7	2	2	11	1	20	1	2	2.5	13.8	2.0	13.9	1	4.0
30-14	1600, 1000	1018.5 (0.0)	0.0	1018.5	1	1	13	0	33	2	0	1.4	0.0	0.0	0.0	0	2.6
	800, 500	1099.1 (0.8)	18.7	1080.5	2	2	28	7	37	6	7	8.1	7.5	41.3	4.2	0	3.7
30-15	1600, 1000	981.7 (0.0)	0.0	981.7	1	1	22	0	25	1	8	2.7	0.3	0.0	0.3	0	2.6
	800, 500	1081.1 (0.2)	2.5	1078.6	2	9	39	9	28	5	19	11.5	5.7	57.5	0.7	0	4.0
30-16	1600, 1000	1157.1 (0.9)	25.0	1132.1	1	4	39	0	38	6	15	7.4	0.7	-25.0	2.8	0	2.3
	800, 500	1227.1 (1.0)	38.3	1188.7	2	7	61	37	28	1	21	18.8	10.6	41.7	8.1	1	3.9
30-17	1600, 1000	1015.1 (0.0)	0.0	1015.1	1	1	31	0	31	1	17	5.1	0.2	0.0	0.2	0	2.2
	800, 500	1128.8 (1.0)	20.0	1108.8	2	29	69	14	45	13	25	24.5	12.6	60.0	8.5	0	4.2
30-18	1600, 1000	1037.3 (0.0)	0.0	1037.3	1	0	9	0	24	0	0	0.0	0.6	0.0	0.6	0	2.9
	800, 500	1136.1 (1.0)	12.4	1123.7	2	12	42	13	28	8	11	10.0	11.5	-12.4	12.5	0	4.1
30-19	1600, 1000	1102.3 (0.0)	0.0	1102.3	1	1	9	0	25	2	1	0.6	1.4	0.0	1.4	0	3.1
	800, 500	1180.5 (0.3)	32.5	1148.0	2	2	14	5	25	3	2	3.3	6.1	32.5	3.7	0	5.0
30-20	1600, 1000	1034.5 (0.0)	0.0	1034.5	1	0	2	0	12	0	0	0.0	0.0	0.0	0.0	0	3.1
	800, 500	1095.3 (0.2)	2.2	1093.1	2	2	16	9	15	0	2	3.8	7.5	-2.2	7.7	0	4.5

Table 2: Benchmark of Algorithms CP-FRDP and H-FRDP on 30-node instances

		CP-FRDP											H-FRDP				
Instance	L, C	Obj. (Gap-%)	$E^{\star}_{CP}$	$R^{\star}_{CP}$	$k^{\star}_{CP}$	n_{it}	$n_{\mathcal{MP}}$	n_L	n_S	n_o	n_f	Time (s)	Δ Obj.	$\Delta E^{\star}$	$\Delta R^{\star}$(%)	$\Delta k^{\star}$	Time (s)
40-1	1600, 1000	1075.7 (0.0)	0.0	1075.7	1	0	22	0	42	0	7	2.5	5.5	80.0	-1.6	0	4.7
	800, 500	1181.3 (0.9)	40.0	1141.3	2	29	83	28	44	6	37	55.0	11.5	20.0	10.5	0	4.4
40-2	1600, 1000	1119.4 (0.0)	0.0	1119.4	1	0	27	0	41	0	9	4.7	4.1	60.0	-1.1	0	2.8
	800, 500	1196.6 (0.9)	15.6	1181.0	2	20	80	35	49	2	21	26.7	15.3	88.4	9.8	0	4.6
40-3	1600, 1000	1158.7 (0.8)	9.5	1149.1	1	1	13	0	37	1	2	2.0	2.1	19.0	0.5	0	2.9
	800, 500	1259.6 (1.0)	22.9	1236.7	2	24	139	64	55	26	48	95.1	11.5	57.1	7.9	0	2.8
40-4	1600, 1000	1157.5 (0.9)	10.0	1147.5	1	3	28	0	45	3	9	5.3	7.0	90.0	-0.3	0	2.6
	800, 500	1311.3 (1.0)	18.0	1293.3	2	272	519	233	55	30	142	1402	11.4	-18.0	12.6	0	4.4
40-5	1600, 1000	1063.0 (0.0)	0.0	1063.0	1	0	17	0	43	0	3	1.8	9.6	85.7	2.4	0	3.4
	800, 500	1156.6 (0.4)	4.8	1151.9	2	4	118	64	56	0	39	75.2	10.5	75.2	5.0	0	3.2
40-6	1600, 1000	1052.3 (0.0)	0.0	1052.3	1	0	6	0	37	0	0	0.0	0.4	0.0	0.5	0	3.0
	800, 500	1127.4 (0.5)	5.8	1121.6	2	5	34	24	46	2	0	8.5	19.3	74.2	14.9	0	5.3
40-7	1600, 1000	1139.0 (0.0)	0.0	1139.0	1	5	67	0	65	1	33	24.3	7.2	100.0	-1.0	0	4.8
	800, 500	1299.1 (1.0)	40.0	1259.1	2	241	466	164	71	17	218	2412	11.2	0.0	11.5	0	5.1
40-8	1600, 1000	1032.3 (1.0)	24.8	1007.5	1	6	40	0	45	9	16	12.4	4.5	75.2	-2.6	0	4.4
	800, 500	1103.6 (0.9)	22.9	1080.7	2	14	60	18	49	3	28	26.2	9.7	17.1	8.6	0	4.4
40-9	1600, 1000	1047.7 (0.0)	0.0	1047.7	1	0	3	0	20	0	0	0.0	0.0	0.0	0.0	0	4.0
	800, 500	1149.5 (0.3)	3.9	1145.6	2	26	59	23	35	3	28	24.6	10.0	20.1	8.5	0	4.9
40-10	1600, 1000	1120.8 (0.0)	0.0	1120.8	1	8	68	0	64	13	25	27.6	2.1	100.0	-7.2	0	4.5
	800, 500	1224.7 (0.0)	0.0	1224.7	2	46	176	55	49	31	90	427.	5.3	100.0	-2.6	0	4.2
40-11	1600, 1000	1082.5 (0.0)	0.0	1082.5	1	2	19	0	42	3	3	2.8	4.9	73.3	-1.6	0	3.3
	800, 500	1166.4 (0.1)	1.0	1165.5	2	19	74	30	44	5	29	37.0	13.9	72.4	9.0	0	5.2
40-12	1600, 1000	1103.1 (1.0)	14.4	1088.7	1	2	14	0	40	4	1	2.1	3.1	39.4	-0.4	0	3.8
	800, 500	1191.6 (-0.3)	21.2	1170.4	2	60	295	127	50	87	131	1078	11.2	47.6	8.1	0	3.4
40-13	1600, 1000	1092.5 (0.0)	0.0	1092.5	1	2	17	0	38	3	3	2.9	5.4	64.0	-0.1	0	3.5
	800, 500	1145.3 (0.2)	2.0	1143.3	2	10	34	15	39	7	5	9.0	15.1	98.0	8.5	0	4.0
40-14	1600, 1000	1114.9 (0.0)	0.0	1114.9	1	5	38	0	54	17	9	7.7	4.4	40.0	1.0	0	3.2
	800, 500	1176.9 (-0.2)	1.0	1176.0	2	25	73	25	50	10	22	23.8	3.5	64.0	-1.8	0	3.8
40-15	1600, 1000	1133.4 (0.4)	4.3	1129.1	1	7	71	0	54	17	33	25.8	5.8	87.3	-1.6	0	3.1
	800, 500	1223.7 (1.0)	38.3	1185.4	2	44	226	124	44	28	93	386.	17.3	48.3	14.9	0	3.3
40-16	1600, 1000	1080.2 (0.0)	0.0	1080.2	1	4	17	0	45	9	1	2.5	0.0	0.0	0.0	0	2.9
	800, 500	1150.6 (0.8)	21.5	1129.1	2	25	50	20	48	7	4	16.1	15.9	33.5	14.1	0	3.6
40-17	1600, 1000	1219.5 (0.0)	0.0	1219.5	1	5	78	0	74	8	28	28.9	6.7	80.0	0.7	0	4.8
	800, 500	1396.3 (3.2)	73.3	1323.0	2	108	496	236	73	29	218	3674	4.1	13.3	3.3	0	3.8
40-18	1600, 1000	1041.9 (0.0)	0.0	1041.9	1	0	7	0	23	0	1	0.5	6.4	68.0	0.3	0	4.2
	800, 500	1211.2 (1.0)	16.0	1195.2	2	34	185	105	51	4	55	192.	15.0	4.0	15.0	0	5.1
40-19	1600, 1000	1072.2 (0.0)	0.0	1072.2	1	0	5	0	31	0	0	0.0	0.0	0.0	0.0	0	4.0
	800, 500	1188.2 (0.7)	8.0	1180.2	2	42	127	64	41	19	44	97.1	10.2	92.0	3.5	0	4.2
40-20	1600, 1000	1056.6 (0.0)	0.0	1056.6	1	1	10	0	38	1	0	0.9	1.1	10.0	0.1	0	4.0
	800, 500	1158.2 (0.9)	10.0	1148.2	2	25	79	35	52	17	24	64.6	17.6	72.9	13.2	0	4.2

Table 3: Benchmark of Algorithms CP-FRDP and H-FRDP on 40-node instances

		CP-FRDP											H-FRDP				
Instance	L, C	Obj. (Gap-%)	$E^{\star}_{CP}$	$R^{\star}_{CP}$	$k^{\star}_{CP}$	n_{it}	$n_{\mathcal{MP}}$	n_L	n_S	n_o	n_f	Time (s)	Δ Obj.	$\Delta E^{\star}$	$\Delta R^{\star}(\%)$	$\Delta k^{\star}$	Time (s)
50-1	1600, 1000	1175.1 (0.2)	2.8	1172.3	1	2	32	0	65	3	9	8.4	6.0	52.8	1.9	0	9.1
	800, 500	1268.5 (1.0)	44.4	1224.1	2	26	171	109	75	13	37	291.	12.7	47.6	10.1	0	10.7
50-2	1600, 1000	1229.8 (0.0)	0.0	1229.8	1	2	22	0	59	4	0	4.5	8.5	86.2	2.2	0	8.6
	800, 500	1348.7 (1.0)	27.2	1321.5	2	100	341	198	81	55	88	1433	14.4	52.8	11.7	0	13.8
50-3	1600, 1000	1261.5 (0.0)	0.0	1261.5	1	0	21	0	52	0	6	5.6	11.2	117.1	3.2	0	9.2
	800, 500	1484.8 (6.7)	117.1	1367.7	2	66	528	409	75	3	92	3626	10.4	-0.0	11.2	0	9.2
50-4	1600, 1000	1182.3 (0.0)	0.0	1182.3	1	0	16	0	40	0	5	2.7	5.5	100.0	-2.7	0	13.6
	800, 500	1256.2 (0.9)	20.0	1236.2	2	12	57	22	53	6	19	23.2	17.9	80.0	13.6	0	14.4
50-5	1600, 1000	1210.7 (0.8)	10.0	1200.7	1	0	21	0	43	0	9	3.6	7.1	63.3	2.3	0	8.5
	800, 500	1271.9 (1.0)	15.6	1256.4	2	7	62	38	42	3	19	18.3	9.5	34.4	7.3	0	9.5
50-6	1600, 1000	1190.6 (0.9)	10.4	1180.1	1	4	41	0	63	8	8	11.5	4.5	80.2	-2.1	0	10.2
	800, 500	1259.9 (1.0)	22.7	1237.3	2	121	188	57	65	35	88	349.	17.4	97.3	12.0	0	12.2
50-7	1600, 1000	1174.8 (0.0)	0.0	1174.8	1	3	25	0	50	8	5	3.9	12.6	120.0	4.1	0	9.7
	800, 500	1241.1 (0.9)	20.0	1221.1	2	5	28	11	51	2	3	5.0	17.1	80.0	12.6	0	10.8
50-8	1600, 1000	1097.5 (0.0)	20.0	1077.5	1	1	8	0	46	2	0	1.0	4.6	40.0	1.2	0	10.0
	800, 500	1231.0 (0.5)	23.3	1207.6	2	7	34	19	55	4	2	5.6	17.3	56.7	14.3	0	11.9
50-9	1600, 1000	1197.1 (0.0)	0.0	1197.1	1	0	4	0	30	0	0	0.0	8.3	100.0	0.7	0	10.0
	800, 500	1325.6 (1.0)	20.0	1305.6	2	37	221	142	62	12	50	322.	13.1	60.0	9.6	0	11.3
50-10	1600, 1000	1237.4 (0.0)	0.0	1237.4	1	1	32	0	59	2	12	9.0	6.8	105.3	-1.3	0	9.9
	800, 500	1349.8 (1.4)	34.7	1315.1	2	51	421	290	68	21	118	3710	20.7	71.0	17.6	1	12.4
50-11	1600, 1000	1241.3 (0.9)	11.4	1229.9	1	0	14	0	44	0	1	2.2	8.0	107.1	0.0	0	9.9
	800, 500	1349.9 (0.0)	1.4	1348.5	2	38	330	227	77	23	67	1288	15.7	108.6	9.6	1	11.7
50-12	1600, 1000	1260.1 (0.0)	0.0	1260.1	1	2	56	0	75	3	17	15.3	6.1	100.0	-1.5	0	10.7
	800, 500	1345.9 (1.0)	12.7	1333.1	2	45	271	172	71	2	75	797.	17.4	87.3	12.8	0	12.9
50-13	1600, 1000	1213.8 (0.9)	20.0	1193.8	1	5	25	0	47	14	6	4.7	3.8	80.0	-2.7	0	11.0
	800, 500	1342.3 (1.4)	20.0	1322.3	2	59	549	334	82	2	213	3657	20.3	80.0	16.5	1	12.8
50-14	1600, 1000	1276.6 (0.0)	0.0	1276.6	1	7	83	0	81	21	30	39.5	7.0	80.0	1.2	0	13.2
	800, 500	1398.8 (2.4)	40.0	1358.8	2	62	432	283	68	31	140	3623	2.5	60.0	-1.8	0	12.9
50-15	1600, 1000	1201.0 (0.0)	0.0	1201.0	1	2	22	0	64	2	3	4.5	6.2	80.0	-0.0	0	12.2
	800, 500	1354.4 (3.9)	68.0	1286.4	2	75	544	159	94	9	339	3601	11.3	-24.0	13.2	0	10.2
50-16	1600, 1000	1187.0 (0.0)	0.0	1187.0	1	0	33	0	54	0	11	9.0	8.3	120.0	-1.1	0	9.5
	800, 500	1293.8 (1.0)	32.9	1261.0	2	128	358	175	76	8	129	1651	13.4	83.8	8.4	0	10.6
50-17	1600, 1000	1155.8 (0.0)	0.0	1155.8	1	1	35	0	61	2	13	9.1	6.5	75.0	0.4	0	13.4
	800, 500	1297.9 (1.4)	25.0	1272.9	2	111	431	285	69	3	98	3830	20.3	95.0	15.6	0	13.7
50-18	1600, 1000	1178.1 (0.9)	21.2	1156.8	1	4	22	0	52	6	2	5.1	6.9	47.5	3.3	0	11.0
	800, 500	1264.3 (0.4)	4.6	1259.7	2	49	197	102	66	43	50	429.	12.1	95.4	5.8	0	12.6
50-19	1600, 1000	1176.6 (0.0)	0.0	1176.6	1	3	9	0	35	3	0	2.2	2.3	20.0	0.6	0	9.1
	800, 500	1314.2 (1.3)	45.0	1269.2	2	149	483	285	64	36	119	3605	11.9	28.3	10.5	0	11.6
50-20	1600, 1000	1206.6 (0.0)	0.0	1206.6	1	1	26	0	48	4	9	6.6	5.0	100.0	-3.2	0	13.2
	800, 500	1286.6 (0.9)	40.0	1246.6	2	4	105	48	60	2	37	61.8	10.7	-40.0	13.5	0	15.9

Table 4: Benchmark of Algorithms CP-FRDP and H-FRDP on 50-node instances

function gap is approximately 1.5%. Further, the maximum routing gap $\Delta R^\star$ remains under 6% and the number of vehicle required always matches that of the optimal solution. This shows that the proposed heuristic is highly competitive for single-vehicle problems. In turn, under a resource-constrained setting, Algorithm H-FRDP occasionally requires an additional vehicle *i.e.*, $\Delta k^\star = 1$ (see Table 2, 30-13, 30-16, and Table 4, 50-10, 50-11, 50-13). In some instances, Algorithm H-FRDP improves on the total travel cost or maximum deviation from envy-freeness provided by Algorithm CP-FRDP, *i.e.*, $\Delta R^\star < 0$ or $\Delta E^\star < 0$ (see, among other, Table 1, 20-9, 20-10, Table 2, 30-6, 30-13, Table 3, 40-4, 40-10, Table 4, 50-15, 50-20). This is due to the inherent trade-off between the two criteria optimized. The maximum objective function gap is 20.7% and the maximum routing gap is 17.6% (see Table 4, 50-10). Although this contrasts with the performance of Algorithm H-FRDP in the base case setting, the average performance of the heuristic reveals an objective function gap of 10.8% and routing gap of 8.5%. Finally, we observe that the performance of Algorithm H-FRDP is not heavily penalized when the network size increases, even under the resource-constrained setting. Specifically, the optimality gaps increase sub-linearly with the number of nodes of the instances.

This benchmark highlights the capabilities of both Algorithms CP-FRDP and H-FRDP, and points out the computational challenges in solving the FRDP to optimality. This experiment shows that Algorithm H-FRDP is able to provide competitive solutions compared to CP-FRDP. In particular, both algorithms are able to find non-trivial solutions envy-free solutions (involving one or two vehicles). Further, as a heuristic approach, it is expected that H-FRDP finds more expensive routes than CP-FRDP.

5.3. Performance of Algorithm H-FRDP *on large-scale instances*

We now implement Algorithm H-FRDP on three sets of artificial instances with a total of 100, 200 and 300 nodes. We use the same region of 100x100 than in the former experiments. For this experiment, we use a month of real data from OzHarvest to fit two distributions for nodes' supply and request. The best fit for the distribution of pickup nodes' supply values is found to be a log-normal distribution with a mean of 4.09 and a variance of 1.12. For delivery nodes' request values, we obtain a best fit with a gamma distribution with shape and scale parameters of 1.53 and 61.13, respectively. In all instances generated, the node supply and request values are generated by sampling from these distributions and the ratio of the total supply to the total request is in the range $[0.5, 0.9]$. As in the previous experiment, the ratio of pickup nodes to delivery nodes is set to $|N_p|/|N_d| = 1.25$.

We consider two spatial distribution of pickup and delivery nodes: uniform and non-uniform. To model these spatial distributions, we divide the region into 25 square cells of edge length 20. Recall that the ratio of pickup to delivery nodes is set to 1.25. For both uniform and non-uniform cases, we attempt to respect this ratio when generating the large-scale instances (the ratio may no exact due

to the discreteness of the nodes) and ensure that the distribution of each type of node (pickup and delivery) follows the same node density pattern. Specifically, for the uniform case, we ensure that each cell receives the approximately the same amount of delivery and pickup nodes. For the non-uniform case, for each instance, we randomly select three high-density cells and impose that their node density is at least five times greater than that of other cells.

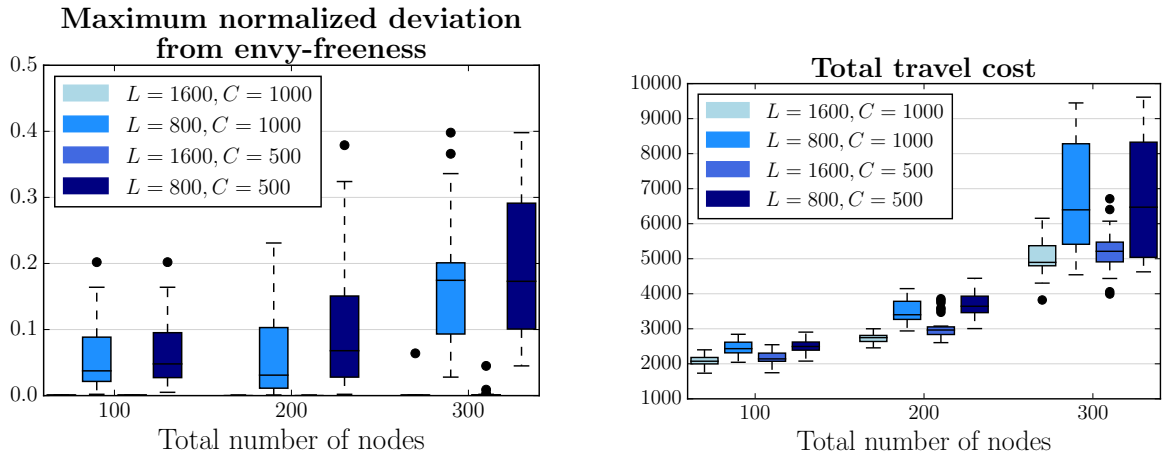
We set the vehicle capacity to 1000—this value is used within OzHarvest operations and is consistent with the magnitude of node supply and requests values—and the maximum tour length to 1600. We generate 30 instances of each node density (uniform and non-uniform) and of each set (100, 200 and 300) nodes, hence a total of 180 instances divided into 6 scenarios are solved. For each instance, we implement Algorithm H-FRDP for all combinations of maximum tour length $L = 1600$, 800 and vehicle capacity $C = 1000$, 500.

For each delivery node $i \in N_d$, we calculate the normalized deviation from envy-freeness defined as $\max\{(F_i - d_i^\star)/F_i, 0\}$. This normalized deviation takes a value of 1 if node i is allocated nothing and 0 if node i receives at least its fair request. The numerical results are summarized in Figures 7 (uniform node densities) and 8 (non-uniform node densities), both of which are composed of four sub-figures.

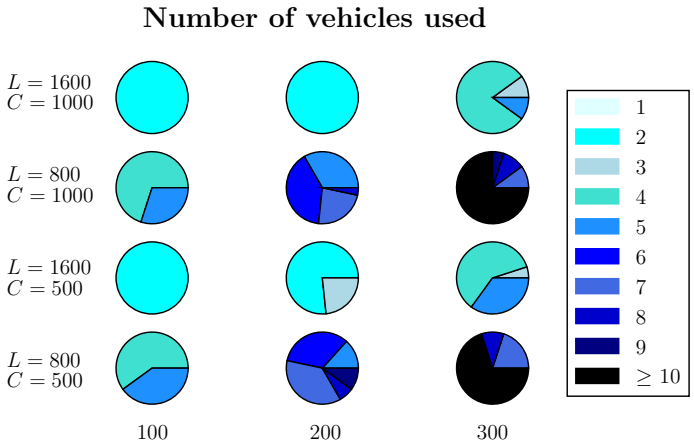
Figures 7a and 8a show the maximum normalized deviation from envy-freeness for uniform and non-uniform node densities, respectively. We find the that reducing the maximum tour length L is much more impactful on the fairness of the allocation compared to reducing vehicle capacity C. This is alignment with the trend observed for the tests conducted on instances with 40 and 50 nodes. Further, we observe that algorithm H-FRDP is able to find several envy-free solutions, even on 300-node instances.

The total travel cost is seen to increase with the number of nodes but also with the reduction of L (see Figures 7b and (Figure 8b)). This is due to the additional vehicles required when the tour length is restricted to $L = 800$ (see Figures 7c and (Figure 8c)). The number of vehicles used is on average slightly greater in uniform (Figure 7c) compared to non-uniform (Figure 8c) instances. For 100-node instances, 2 to 5 vehicles are always used in uniform instances whereas several of the instances require only a single vehicle and a maximum of 4 vehicles are used in non-uniform instances. In turn, up to 9 (resp. 7) vehicles are required for 200-node instances for uniform (resp. non-uniform) instances. In both node densities, for 300-node instances, reducing L to 800 leads to the deployment of more than 10 vehicles. Finally, reducing vehicle capacity only marginally impacts the number of vehicles required.

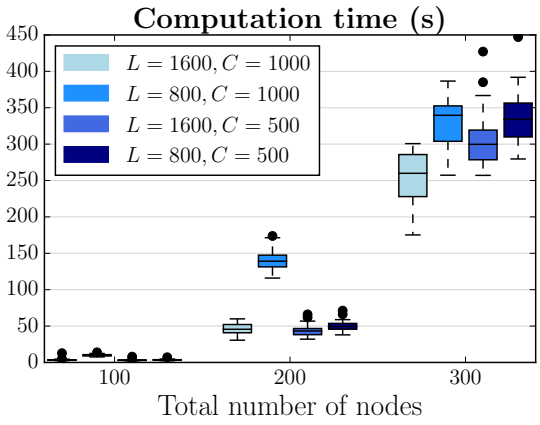
The computational performance of Algorithm H-FRDP is depicted in Figures 7d and 8d. The



(a) (b)

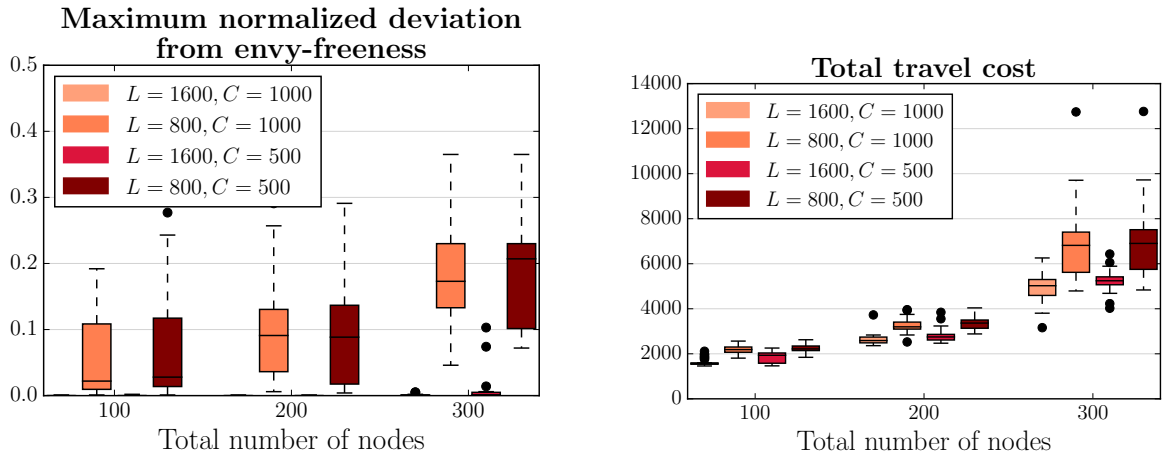


(c)

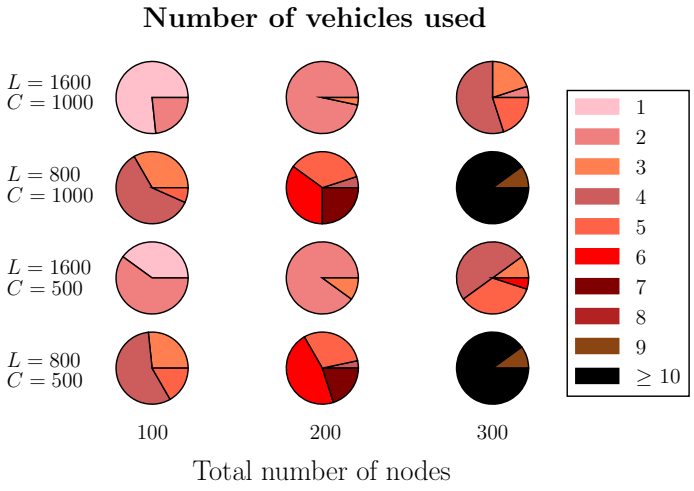


(d)

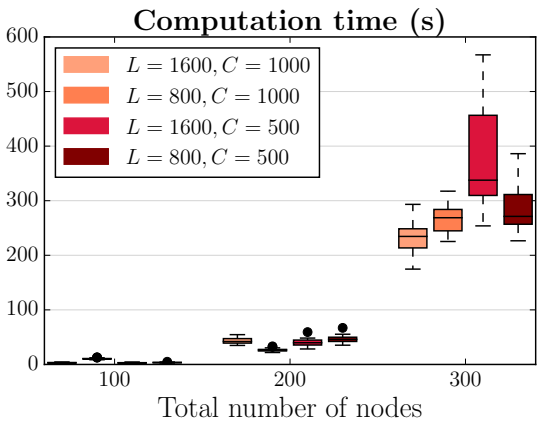
Figure 7: Results from the experiments conducted on the large-scale instances with $|N| = 100$, 200 and 300 for $L = 1600$, 800 and $C = 1000$, 500 on instances with uniform node densities.



(a) (b)



(c)



(d)

Figure 8: Results from the experiments conducted on the large-scale instances with $|N| = 100$, 200 and 300 for $L = 1600$, 800 and $C = 1000$, 500 on instances with non-uniform node densities.

runtime is seen to increase approximately quadratically with the total number of nodes which is a direct consequence of the 2-opt operator. The 100-node instances are solved in a few seconds, whereas 200-node instances require on average 70s (uniform) and 65s (non-uniform). The 300-node instances have a higher variance with a maximum of 450s (uniform) and 580s (non-uniform).

The overall performance of Algorithm H-FRDP is seen to be robust to the node density distribution (*i.e.*, uniform and non-uniform), thus demonstrating the robustness of the proposed approach. This experiment underlines the computational sustainability of the heuristic which is able to scale efficiently to large instances with both uniform and non-uniform node densities while providing balanced solutions in terms of food allocation and travel cost. Further, this experiment highlights the capability of the proposed heuristic to find envy-free solutions in large-scale instances.

5.4. Realistic scenario analysis

We next examine the behavior of Algorithm H-FRDP on a realistic instance re-constructed from actual pickup and deliveries in the region of Sydney, Australia. This instance contains 201 nodes, including a depot node, 102 pickup nodes and 98 delivery nodes. The total supply is 8585 and the total request is 9627. In OzHarvest operations, vehicles with a capacity of 1000 are typically used. Further, vehicles' tour length is limited by workday hours which is set to 6h (including breaks and pre-trip operations). Arc weights are representative of vehicle travel times assuming a speed of 60km/h, which is in line with Sydney's urban and peri-urban road speed limits.

To explore the influence of vehicle capacity and maximum tour length on the solution, we conduct a sensitivity analysis by also considering the cases with a reduced vehicle capacity ($C = 500$) and with a shorter maximum tour length ($L = 3h$, $L = 4h$, $L = 5h$). In the objective function, travel time is expressed in seconds and we use $w = 100$ to compromise between fairness and routing costs.

Table 5 summarizes the numerical results of this experiment. The bottom row can be viewed as the benchmark for this experiment since it corresponds to OzHarvest actual logistics, *i.e.*, $L = 6h$ and $C = 1000$, is referred to as the *base case*. In the base case, Algorithm H-FRDP finds a solution with 2 vehicles and total travel cost of 9.9h, thus corresponding to an average tour of 4.95h. For the same capacity, reducing the maximum tour length to $L = 5h$ requires an additional vehicle. For $L = 4h$, the maximum deviation to envy-free remains identical to the base case but 4 vehicles are required and this translates into a total travel cost of 11.2h. Halving the base tour length to $L = 3h$ results in using 7 vehicles with an average tour length of 2.0h and a slight increase in the maximum deviation to envy-free. Halving the base capacity to $C = 500$ has a drastic impact on the objective function which increases by 12.3% compared to the base case setting ($L = 6h$). In this context, however, a lower maximum deviation from envy-freeness is achieved. If the maximum tour length is set to $L = 3h$, the objective function is increased by 4.8% and, in this context, the same level of fairness in the food

C	$L(h)$	Obj.	$E_H^\star$	$R_H^\star$	$k_H^\star$	$\frac{R^\star}{k_H^\star}$	Time (s)
500	3	123,385	180.4	14.6	7	2.1	56.6
	4	110,609	179.0	12.8	5	2.6	57.7
	5	106,501	178.5	12.3	4	3.1	53.3
	6	101,547	140.0	12.1	3	4.1	51.5
1000	3	117,399	180.4	13.8	7	2.0	47.5
	4	98,715	179.0	11.2	4	2.8	43.9
	5	97,442	179.0	11.0	3	3.7	42.0
	6	89,047	179.0	9.9	2	4.9	38.7

Table 5: Numerical results on the realistic instance with different vehicle capacity and tour length constraints.

allocation is achieved compared to the setting with $C = 1000$, *i.e.*, only the routing cost is penalized.

6. Conclusion

In this paper, we have presented a novel approach for the Food Rescue and Delivery Problem encountered in food relief logistics. The aim of the FRDP is to find envy-free allocation and travel cost-effective routes to pick-up and re-distribute rescued food to welfare agencies. The problem is bi-objective and we proved that the FRDP is NP-hard. We introduced an exact cutting-plane algorithm (CP-FRDP) for its solution which combines a traditional illegal tour elimination procedure with a Benders' decomposition approach and the separation of valid inequalities. In addition, we present a novel tailored heuristic algorithm (H-FRDP) which combines greedy and local search strategies that attempt to meet optimality conditions. We evaluated the proposed solution algorithms through numerical experiments and explored the trade-offs between improving the level fairness in the allocation and travel cost savings. In addition, we examined the performance of the proposed heuristic approach on large-scale instances with varying node densities and conducted a sensitivity analysis on a realistic instance re-constructed from actual operations conducted within a food rescue organization in Sydney, Australia.

The numerical experiments show that Algorithm CP-FRDP is capable to find optimal solutions efficiently and that Algorithm H-FRDP is able to find competitive and balanced solutions for the FRDP. The proposed solutions methods were designed to solve FRDP with homogeneous vehicle capacity and tour length constraints. In this context, we find that using less vehicles is likely to help in improving the fairness of the allocation. In particular, a solution with a single vehicle may yield an envy-free allocation (granted that vehicle capacity is sufficiently large) since the vehicle will be able to collect enough supply to deliver the exact fair request at each delivery node. While this is an intuitive behavior, our experiments show that longer vehicle deadlines will likely have a more significant impact

on the distribution of envy across welfare agencies than increasing vehicle capacity. This can be used to inform food rescue organizations' logistical operations.

Future work will be focused on exploring alternative fair allocation schemes and incorporating richer operational constraints, such as time windows and heterogeneous vehicle capacities. Another extension of this work will investigate the multi-commodity case in which welfare agencies have preferences over commodities.

Acknowledgements

- We would like to express our sincere appreciation to Gopi Krishnan, national general manager OzHarvest for their contribution towards this research. This research has been supported by grant ARC LP150101266 from Australian Research Council.

References

Azadian, F., Murat, A., & Chinnam, R. B. (2017). An unpaired pickup and delivery problem with time dependent assignment costs: Application in air cargo transportation. *European Journal of Operational Research*, .

Balcik, B., Iravani, S., & Smilowitz, K. (2014). Multi-vehicle sequential resource allocation for a nonprofit distribution system. *IIE Transactions*, *46*, 1279–1297.

Bartholdi III, J. J., Platzman, L. K., Collins, R. L., & Warden III, W. H. (1983). A minimal technology routing system for meals on wheels. *Interfaces*, *13*, 1–8.

Benders, J. F. (1962). Partitioning procedures for solving mixed-variables programming problems. *Numerische mathematik*, *4*, 238–252.

Birge, J. R., & Louveaux, F. V. (1988). A multicut algorithm for two-stage stochastic linear programs. *European Journal of Operational Research*, *34*, 384–392.

Campbell, A. M., Vandenbussche, D., & Hermann, W. (2008). Routing for relief efforts. *Transportation Science*, *42*, 127–145.

Carotenuto, P., Giordani, S., & Ricciardelli, S. (2007). Finding minimum and equitable risk routes for hazmat shipments. *Computers & Operations Research*, *34*, 1304–1327.

Chen, Q., Li, K., & Liu, Z. (2014). Model and algorithm for an unpaired pickup and delivery vehicle routing problem with split loads. *Transportation Research Part E: Logistics and Transportation Review*, *69*, 218–235.

CPLEX, I. I. (2009). *V12 Users Manual for CPLEX*.

Dantzig, G., Fulkerson, R., & Johnson, S. (1954). Solution of a large-scale traveling-salesman problem. *Journal of the operations research society of America*, *2*, 393–410.

Davis, L. B., Sengul, I., Ivy, J. S., Brock, L. G., & Miles, L. (2014). Scheduling food bank collections and deliveries to ensure food safety and improve access. *Socio-Economic Planning Sciences*, *48*, 175–188.

Feeding India (2017). https://www.feedingindia.org/.

Foley, D. K. (1967). Resource allocation and the public sector. *YALE ECON ESSAYS, VOL 7, NO 1, PP 45-98, SPRING 1967. 7 FIG, 13 REF.*, .

FoodBank (2017). https://www.foodbank.org.au/.

Fourer, R., Gay, D. M., & Kernighan, B. W. (2002). *AMPL: A Modeling Language for Mathematical Programming*. (2nd ed.). Brooks/Cole.

Gorr, W., Johnson, M., & Roehrig, S. (2001). Spatial decision support system for home-delivered services. *Journal of geographical systems*, *3*, 181–197.

Haghani, S.-C. O. A. (1997). Testing and evaluation of a multi-commodity multi-modal network flow model for disaster relief management. *Journal of Advanced Transportation*, *31*, 249–282.

Hernández-Pérez, H., Rodríguez-Martín, I., & Salazar-González, J. J. (2009). A hybrid grasp/vnd heuristic for the one-commodity pickup-and-delivery traveling salesman problem. *Computers & Operations Research*, *36*, 1639–1645.

Hernández-Pérez, H., & Salazar-González, J.-J. (2004a). A branch-and-cut algorithm for a traveling salesman problem with pickup and delivery. *Discrete Applied Mathematics*, *145*, 126–139.

Hernández-Pérez, H., & Salazar-González, J.-J. (2004b). Heuristics for the one-commodity pickup-and-delivery traveling salesman problem. *Transportation Science*, *38*, 245–255.

Hernández-Pérez, H., & Salazar-González, J.-J. (2007). The one-commodity pickup-and-delivery traveling salesman problem: Inequalities and algorithms. *Networks*, *50*, 258–272.

Huang, M., Smilowitz, K., & Balcik, B. (2012). Models for relief routing: Equity, efficiency and efficacy. *Transportation research part E: logistics and transportation review*, *48*, 2–18.

Ichoua, S. (2010). Relief distribution networks: design and operations. *Supply Chain Optimization, Design, and Management: Advances and Intelligent Methods*, (pp. 171–186).

Laporte, G., Desrochers, M., & Nobert, Y. (1984). Two exact algorithms for the distance-constrained vehicle routing problem. *Networks*, *14*, 161–172.

Leket Israel (2017). www.leket.org/.

Lien, R. W., Iravani, S. M., & Smilowitz, K. R. (2014). Sequential resource allocation for nonprofit operations. *Operations Research*, *62*, 301–317.

Lin, S. (1965). Computer solutions of the traveling salesman problem. *The Bell System Technical Journal*, *44*, 2245–2269. doi:10.1002/j.1538-7305.1965.tb04146.x.

Luis, E., Dolinskaya, I. S., & Smilowitz, K. R. (2012). Disaster relief routing: Integrating research and practice. *Socio-economic planning sciences*, *46*, 88–97.

Magnanti, T. L., & Wong, R. T. (1981). Accelerating benders decomposition: Algorithmic enhancement and model selection criteria. *Operations research*, *29*, 464–484.

Martinovic, G., Aleksi, I., & Baumgartner, A. (2009). Single-commodity vehicle routing problem with pickup and delivery service. *Mathematical Problems in Engineering*, *2008*.

Mladenović, N., Urošević, D., Ilić, A. et al. (2012). A general variable neighborhood search for the one-commodity pickup-and-delivery travelling salesman problem. *European Journal of Operational Research*, *220*, 270–285.

Nair, D., Grzybowska, H., Fu, Y., & Dixit, V. (2017a). Scheduling and routing models for food rescue and delivery operations. *Socio-Economic Planning Sciences*, *in press*.

Nair, D., Grzybowska, H., Rey, D., & Dixit, V. (2016). Food rescue and delivery: Heuristic algorithm for periodic unpaired pickup and delivery vehicle routing problem. *Transportation Research Record: Journal of the Transportation Research Board*, *2548*, 81–89.

Nair, D. J., Rey, D., & Dixit, V. V. (2017b). Fair allocation and cost-effective routing models for food rescue and redistribution. *IISE Transactions*, *49*, 1172–1188.

Nolz, P. C., Doerner, K. F., Gutjahr, W. J., & Hartl, R. F. (2010). A bi-objective metaheuristic for disaster relief operation planning. In *Advances in multi-objective nature inspired computing* (pp. 167–187). Springer.

Orgut, I., Ivy, J., & Uzsoy, R. (2017). Modeling for the equitable and effective distribution of food donations under stochastic receiving capacities. *IISE Transactions*, *49*, 567–578.

Orgut, I. S., Ivy, J., Uzsoy, R., & Wilson, J. R. (2016). Modeling for the equitable and effective distribution of donated food under capacity constraints. *IIE Transactions*, *48*, 252–266.

OzHarvest (2017). http://www.ozharvest.org/.

Rahmaniani, R., Crainic, T. G., Gendreau, M., & Rei, W. (2017). The benders decomposition algorithm: A literature review. *European Journal of Operational Research*, *259*, 801–817.

Rawls, J. (2009). *A theory of justice*. Harvard University Press.

Rei, W., Cordeau, J.-F., Gendreau, M., & Soriano, P. (2009). Accelerating benders decomposition by local branching. *INFORMS Journal on Computing*, *21*, 333–345.

Saharidis, G. K., Minoux, M., & Ierapetritou, M. G. (2010). Accelerating benders method using covering cut bundle generation. *International Transactions in Operational Research*, *17*, 221–237.

SecondBite (2017). https://www.secondbite.org/.

SecondHarvest (2017). http://secondharvest.ca/.

Solak, S., Scherrer, C., & Ghoniem, A. (2014). The stop-and-drop problem in nonprofit food distribution networks. *Annals of Operations Research*, *221*, 407–426.

Steinhaus, H. (1948). The problem of fair division. *Econometrica*, *16*.

Tzeng, G.-H., Cheng, H.-J., & Huang, T. D. (2007). Multi-objective optimal planning for designing relief delivery systems. *Transportation Research Part E: Logistics and Transportation Review*, *43*, 673–686.

Varian, H. R. (1974). Equity, envy, and efficiency. *Journal of economic theory*, *9*, 63–91.

Wong, D. W., & Meyer, J. W. (1993). A spatial decision support system approach to evaluate the efficiency of a meals-on-wheels program. *The Professional Geographer*, *45*, 332–341.

Xu, D., Li, K., Zou, X., & Liu, L. (2017). An unpaired pickup and delivery vehicle routing problem with multi-visit. *Transportation Research Part E: Logistics and Transportation Review*, *103*, 218–247.

Yildiz, H., Johnson, M. P., & Roehrig, S. (2012). Planning for meals-on-wheels: algorithms and application. *Journal of the Operational Research Society*, *64*, 1540–1550.

Zhao, F., Li, S., Sun, J., & Mei, D. (2009). Genetic algorithm for the one-commodity pickup-and-delivery traveling salesman problem. *Computers & Industrial Engineering*, *56*, 1642–1648.