Sustainability-Oriented Route Generation for Ridesharing Services ⋆

Mengya Liu[1], Vahid Yazdanpanah[2], Sebastian Stein[2], and Enrico Gerding[2]

[1] Smart Data Group, AI Lab, Lenovo Research at Lenovo
No 10 Courtyard, Xibeiwang, East Road, Haidian District, Beijing, China, 100094
myliu26@lenovo.com
[2] Agents, Interaction and Complexity Research Group, University of Southampton
SO17 1BJ, Southampton, UK
v.yazdanpanah@soton.ac.uk, ss2@ecs.soton.ac.uk, eg@ecs.soton.ac.uk

Abstract. Sustainability is the ability to maintain and preserve natural and man-made systems for the benefit of current and future generations. The three pillars of sustainability are social, economic, and environmental. These pillars are interdependent and interconnected, meaning that progress in one area can have positive or negative impacts on the others. This calls for smart methods to balance such benefits and find solutions that are optimal with respect to all the three pillars of sustainability. By using AI methods, in particular, genetic algorithms for multiobjective optimisation, we can better understand and manage complex systems in order to achieve sustainability. In the context of sustainability-oriented ridesharing, genetic algorithms can be used to optimise route finding in order to lower the cost of transportation and reduce emissions. This work contributes to this domain by using AI, specifically genetic algorithms for multiobjective optimisation, to improve the efficiency and sustainability of transportation systems. By using this approach, we can make progress towards achieving the goals of the three pillars of sustainability.

Keywords: Ridesharing, Multiobjective Algorithm, Mobility-on-demand, Sustainable Transportation, Evolutionary Computation.

1. Introduction

Mobility-On-Demand (MOD) services traditionally aim to reduce the economic and environmental cost of transportation [11,1]. Roughly speaking, MOD promises to utilise the mobility capacity in a more efficient way (leading to reductions in the collective and individual economic costs), while also reducing emissions caused (e.g., by avoiding single-driver journeys). Therefore, MOD systems and methods to support their implementation directly contribute to Sustainable Development Goal (SDG) 11 and 13 on *sustainable cities and communities* and *climate action*, respectively. If MOD is widely adopted, cities will enjoy its environmental and economic benefits and take a step towards mitigating climate change. However, if such services merely focus on what is optimal from the service providers' perspective and ignore users' requirements, it will be difficult to encourage users to move towards this service and ignore the comfort of using personal vehicles. Such

⋆ The initial idea behind this work is presented at ATT'22: Workshop Agents in Traffic and Transportation, July 25, 2022, Vienna, Austria [15].

a sustainability-oriented transition necessitates looking not only at operators' (economic) criteria, but also evaluating how a particular routing choice may affect individuals, e.g., via their total travel or waiting times. In this work, we argue that *sustainable ridesharing* needs to capture all the three pillars of economic, environmental, and social sustainability [19] and aim for routing solutions that are balanced with respect to all three aspects. The economic and environmental aspects call for minimising respective costs, both at a collective level, but also (to ensure fairness) for all the riders. Finally, the social aspect requires considering fairness in distributing tasks among drivers such that riders receive a balanced workload. Thus, addressing the routing problem in sustainable ridesharing requires a multiobjective approach that captures potential trade-offs among different aspects of sustainability for generating sustainable routing options.[3]

As discussed in related work, e.g., [21,1,25], the multidimensionality and complexity of the ridesharing routing problem, and, in our case, in view of the three pillars of sustainability, result in inapplicability of exact multiobjective optimisation techniques and justifies using genetic algorithms (GA). While [12] explored adopting reinforcement learning for multiobjective optimisation, this work considers GA to provide a diverse range of solutions (for a diverse set of users). Multiobjective GA allows capturing various objectives with fewer compromises regarding scalability [10]. In particular, we use a form of the Non-dominated Sorting Genetic Algorithm (NSGA) that is proven to be effective in various mobility settings [8].

Against this background, for the first time in this work, we capture the *social* and *environmental* aspects of sustainable routing in ridesharing and use genetic algorithms for generating routing options that consider all three pillars of sustainability. This is the first approach that integrates these two pillars of sustainability into traditional models of (purely) *economic* sustainability and generates routing solutions under six sustainable ridesharing objectives: travelling time, waiting time, overall/excess distance, travel cost, total emission, and working time balance. The list of sustainable routing options can be used in a "*user participation*" phase (e.g., in ridesharing services), where riders can select their desired route from a list of options. Using our approach, service providers can generate routing options that balance all the three pillars of economic, environmental, and social sustainability. In addition, they can provide routing options that reflect riders' preferences (e.g., by focusing on the environmental dimension). This is a step towards integrating equitability and user participation [2] into sustainable ridesharing practices.

In the remaining sections of this paper, we present a detailed explanation of the proposed ridesharing algorithm in Section, following which, in Section 3, is a brief introduction of the genetic algorithm (NAGA3) that we adopted; Section 4 explains the conducted experiments and illustrates the experiments results with respect to multiple aspects, and at the end, Section 5 concludes the contribution of this work.

2. Sustainable Ridesharing

The ridesharing routing problem is to allocate a given set of riders to vehicles with respect to different objectives. Each rider requires a ride from its starting point to its destination,

[3] In view of human-centred AI techniques and the need for developing trustworthy human-AI partnerships [20], we see sustainable ridesharing as an inherently sociotechnical problem and argue that its acceptance by society depends on the ability to capture all the three aspects of economic, environmental, and social sustainability.

along with a specified earliest departure time for taking a ride. Each vehicle can take a limited number of riders aboard at the same time, excluding the driver, which we refer to as its capacity. And the driving costs and capacity of a vehicle are associated with its type. A solution to the ridesharing routing problem is an arrangement that sends vehicles to pick up riders at their starting point, and then drops them off at their destination. The objectives evaluate the efficiency of a solution. In the following, we present the mathematical notations used for modelling the ridesharing problem and developing our approach.

2.1. Ride Requests

In the ridesharing routing problem, all the riders post their ride requests at the beginning. Let R be the set of posted requests and r represent a single request. To locate riders, we adopt a graph structure to model the real-world map, i.e., a map graph is $G = (N, E)$, where nodes in N represent the intersections on a real-world map and edges in E that link nodes together represent the roads between intersections. In general, to represent an intersection, a node is in the form of a tuple marking the latitude and longitude of the intersection. Thereby, let $r(s, t, u)$ denote a rider's request for a ride from node s to node t along with an earliest time for the rider to leave, u.

2.2. Features of Vehicles

To model the sustainability of the ridesharing routing problem from economic, environment and social aspects, this work considers 3 features of a vehicle as well as its location. Let V be the set of available vehicles and $v(s, t, p, e, c)$ denote a vehicle that starts working at node s, returns to node t at the end of its service with a physical capacity of p, emission level of e and a travelling cost per kilometre, c, including the driver wage, vehicle maintenance and fuel costs. Regardless of the difference of vehicle brands, there is a positive correlation between c and p. Hence, we assume that $c = \alpha \times p$ for a vehicle and α varies according to the type of vehicle. For a pessimistic estimation, we use $\alpha = 1$ in our experiments. Besides, the emission level of a vehicle e is a vector, and the ith element in the vector, e_i, denotes the emission rate of a vehicle when there are i riders aboard, since different numbers of riders aboard cause different emission rate. Specifically, in this work, the emission of a vehicle generally includes greenhouse gas (GHG) and air pollution. With respect to the reports from the UK's Department for Transport [16] and National Atmospheric Emission Inventory (NAEI) [23], and the EU standard vehicle emissions calculator, COPERT [9], the GHG and air pollution emission of a vehicle per kilometre are mainly related to the fuel and type of a vehicle, but the emission of GHG also depends on the number of passengers. Therefore, we model the emission level of a vehicle as a vector, where each element represents a emission rate associated with a number of riders on board. Notice that we also assume that all vehicles will drive at the same speed to simplify the problem. The is because the experimental data used in this work are city driving records which is more believed to be limited by the traffic condition instead of the type of vehicles. This setting can be simply extended to simulate a dynamic speed by varying the speed in a range of minimum to maximum urban/legal speed.

The features of a vehicle, such as capacity, emission level and travelling cost, depend on its type. We consider 3 types of general passenger vehicles according to the vehicle

Table 1: Estimated Features for Different Types of Vehicle

	Vehicle Type	Fuel	Capacity	Vector of Emission Level	Cost
1	Mini Car	Petrol	1	$<1, 2 \times 0.9>$	1
2	Car	Electric	3	$<1/3.4, 2/3.4 \times 90\%, 3/3.4 \times 90\%^2, 4/3.4 \times 90\%^3>$	3
3	Medium Car	Petrol	6	$<1, 2\times 90\%, \cdots, 7\times 90\%^7>$	10
4	Large Car	Petrol	10	$<1, 2\times 90\%, \cdots, 11\times 90\%^{10}>$	10

categories specified on the UK Driving Licence Categories [22]: Small cars, medium-sized vehicles, large vehicles[4]. Table 1 lists their features. To simulate the emission level of different types of vehicles, we use the *car* type with a petrol engine and one passenger on board as the standard, and assume it emits 1 unit of greenhouse gas and air pollution per kilometre (e.g., 1 unit could be 100g of $\mathrm{CO_2}$) and costs 1 price unit per kilometer (e.g., \$1). According to the Transport and Environment Statistics [16], "an average petrol car emits around 4 times more per passenger than the equivalent journey by coach, or 3.4 times more per passenger emitted by the average electric car", and "maximising the number of people per vehicle can reduce emissions per person". Hence, we assume that the average emission per passenger reduces by 10% when the number of passengers aboard increases and the cost of vehicles are related with its capacity. The emission column in Table 1 lists the emission vector for each vehicle type where elements in the vector are emissions of a vehicle in the order of 0 passengers to its full capacity.

Our focus in this work is to demonstrate the impact of emissions of vehicles, and we are aware of the existence of other types of vehicles, such as motorcycles, and different types of emission calculators [24]. The types of vehicles and the emission estimation considered in this work are standard types and presented for the purpose of showing the performance of the approach.

2.3. Computational Complexity

To understand the complexity of ridesharing problem, we can analyse the number of arrangements and driving routes of all vehicles, First, the number of possible arrangements is $|V|^{|R|}$ since individual riders have $|V|$ options for their rides without considering feasibility, i.e. constrains such as the capacities of vehicles and the relative order of starting points and destinations to stop by. Regrading an arrangement, for each vehicle arranged to offer rider to τ passengers, there are $2\tau!$ different driving routes to stop by all the starting points and destinations of those riders. To simplify the calculation, assume the computational complexity of justifying whether a route is feasible and calculating objectives are $O(|V|)$. In addition, given multiple objectives, assume the best weights of multiple objectives for a ridesharing problem is preserved. Thereby, the computational complexity of ridesharing problem to find an optimal arrangement regrading the objectives is

$$O(R, V, Objectives) = O(|V| + |V|^{|R|} \times \prod_{\tau_1, \cdot, \tau_{|V|}} 2\tau_i!),$$

[4] we exclude minibuses and buses [6]. Since they have stable routes and we do not consider asking riders to change vehicles during their rider, they left no space for picking riders at their starting points.

where $\sum_{i=1}^{|V|} \tau_i = |R|$.

It will be expensive to find one optimal arrangement, especially when no prior knowledge is given about the method to balance multiobjectives. In the following sections, we will explain our design to filter out infeasible arrangements and adopt an genetic algorithm to efficiently generate diverse arrangements without requiring prior knowledge about multiobjectives.

2.4. Solution Design

Our intelligent routing approach is based on genetic algorithms [17]. First, we model an arrangement that a vehicle picks up and drops off a rider as a genetic chromosome: (vehicle, weight of picking up priority, weight of dropping off priority), and a solution to the ridesharing routing problem of arranging vehicles for m riders as:

$$\begin{bmatrix} r_1: & (v^1, w^{1[s]}, w^{1[t]}) \\ r_2: & (v^2, w^{2[s]}, w^{2[t]}) \\ \cdots & \cdots \\ r_m: & (v^m, w^{m[s]}, w^{m[t]}) \end{bmatrix} \tag{1}$$

For rider r_i, v^i denotes the vehicle that serves r_i a ride, $w^{i[s]}$ is a positive real number that represents the priority weight of picking up r_i, and $w^{i[t]}$ is a real number that indicates the priority weight of r_i to get off v^i at its destination. A higher priority weight implies a greater sense of urgency to start or finish a ride. Thus, for a vehicle that offers a ride to multiple riders, it will stop at nodes to pick up and drop off the riders with respect to their priority weights. In reality, the priority weight can be the time of a ride request, arriving time or even promotion tips.

2.5. Solution to Route

To calculate the routes for vehicles there are two factors to satisfy: feasibility and uniqueness. Feasibility requires that when a vehicle follows a route to pick up and drop off riders, the number of riders at any given time must not exceed the capacity of the vehicle. Uniqueness requires that, given a solution, one should be able to derive one and only one way to route the vehicles from it. The uniqueness criterion is necessary because it is the solutions that the genetic algorithm evaluates and optimises while objectives in the evaluation and optimisation are based on the routes for vehicles. Thus, to be able to evaluate a solution, we require a 1-1 correspondence with routes that the solution entails. We will explain our method to map a solution to a feasible and unique routing in the following, and introduce the objectives afterwards.

First, to capture the meaning of the priority weights in a solution, we use the following two rules when comparing the priority weights of different riders:

1. No consideration of dropping off priority for riders who are waiting for pick-up.
2. For riders with equal priority weights, the rider with a lower index number is prioritised, considering that the rider posted its ride request earlier.

Algorithm 1: Routing Algorithm:

Input: $Solution$, an array; V, vehicles; R, ride requests.
Output: $Routes$, an array

```
foreach v ∈ V do
    Routes[v] ← {v[s]} ;
    Board ← ∅;
    load ← Size(Board);
    Hold ← GetPassengers(v);
    while Board ≠ ∅ & Hold ≠ ∅ do
        rx, w_rx ← GetRiderWithGreatestPickupWeight(Hold);
        ry, w_ry ← GetRiderWithGreatestDropoffWeight(Board);
        if w_rx < w_ry then
            Board.remove(ry);
            load = load - 1;
            Routes[v].add(ry[t]);
        else
            if load < v[p] then
                Hold.remove(rx);
                Board.add(rx);
                load = load + 1;
                Routes[v].add(rx[s]);
            else
                Board.remove(ry);
                load = load - 1;
                Routes[v].add(ry[t]);
            end
        end
    end
    Routes[v].add(v[t]) ;
end
```

Regarding a solution, let $Passenger(v)$ denote the set of riders to whom vehicle v offers a ride, and $Passenger(v) = \{r_i | v^i = v\}$. The route of v is a sequence of nodes, $Path(v)$, that are either starting points or destinations of riders in $Passenger(v)$, and the path that a vehicle travels from a node to another is the shortest path calculated by Dijkstra's algorithm [7]. While a vehicle travels, let $Board(v, n)$ denote the riders that are on the vehicle v when it visits node n, and $Hold(v, n)$ be the riders that are still waiting for the vehicle for a pick-up.

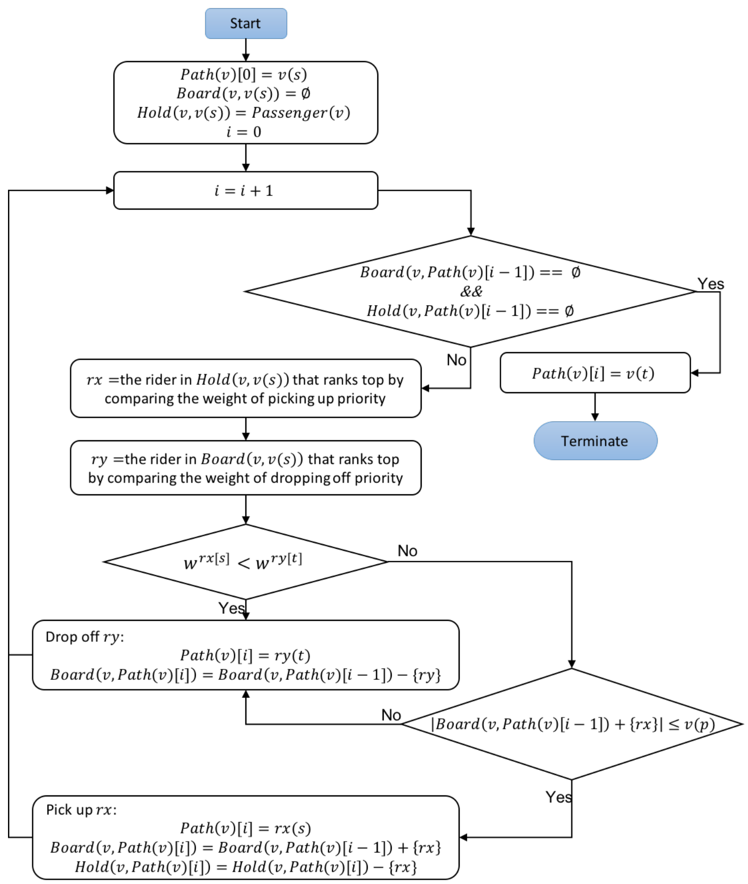


Fig. 1: Algorithm for Mapping Solution to Routes.

Figure 1 and Algorithm 1 illustrate the workflow and steps of our routing algorithm that maps a solution to the routes for a vehicle. [5] For each vehicle, the very first node in its route is its starting point (Line 1), and at that node, the boarding passengers is null (Lines3-4) and all the riders assigned to it are on hold (Line 5). Then, the next node in the

[5] For a complete implementation of our routing algorithm, please refer to `https://github.com/Miya-Liu/equitable-ridesharing`.

route of the vehicle depends on the priority weights of the aboard and waiting riders. First, we compare the waiting riders' weights of picking up priority and get the top one waiting rider (Line 7), and then compare the aboard riders' weights of dropping off priority and get the top one aboard rider (Line 8). If the aboard rider's weight of dropping off priority is greater than the waiting rider's weight of picking up priority (Line 9), the next node in the route of the vehicle is the destination of the aboard rider (Lines 10-12). Otherwise, we check whether picking up the waiting rider violates the vehicle's capacity constraint (Line 14). If not, the vehicle will travel to the starting point of the waiting rider and pick it up (Lines 15-18). If yes, the vehicle still needs to drop off the aboard rider first (Line 20-22). Until there is no rider aboard or waiting (Line 6-25), the vehicle will travel to its destination (Line 26) and terminates its route.

This routing algorithm guarantees the feasibility of the travel paths of vehicles generated from a solution and the uniqueness of the generation dynamically. Note that, regardless of the uniqueness, it is still possible that different solutions generate the same routes for one or even more vehicles. This is because the different weights of either picking up priority or dropping off priority can result in the same ranking of the riders in the algorithm. Note that the potential redundancies are left to be resolved in the genetic algorithm.

2.6. Objectives

Regarding the travel paths of vehicles and their loaded riders, we introduce and minimise 6 objectives from 3 aspects of sustainability: economic, environmental and social. The economic aspect evaluates the efficiency of the routes generated from a solution with respect to travelling time, waiting time, travel cost and excess distance. Then, the environmental aspect of sustainability considers the impact of vehicles' emission and tries to minimise the total emission of rides. Finally, the social aspect concerns the working time of drivers and aims at reducing the differences among the working time of all drivers.

Economic - Travelling Time (ET): This objective measures the total travelling time of individual riders. The measurement of the travelling time for one rider is the time that it takes from the moment the rider gets on a vehicle until the vehicle drops off the rider at her destination. This includes the time that the vehicle travels and waits to pick up and drop off other riders while the rider is on board. The waiting time of a vehicle includes time periods when the vehicle stops at a starting point of a rider to pick her up. Such a waiting takes place when a vehicle arrive (too) early, i.e., when the arriving time of the vehicle is earlier than the earliest leaving time of the rider. Less travelling time means that the riders entails a more efficient trip.

Let $wait(v, n)$ denote the time that a vehicle v waits for picking riders up at node n along its route. Let $d(n_i, n_j)$ be the shortest distance between node n_i and n_j, and $arrive(v, n)$ represent the time that the vehicle arrives at node n along its route. Hence, $wait(v, n_0) = 0$, $arrive(v, n_0) = 0$, and

$$arrive(v, n_i) = arrive(v, n_{i-1}) + wait(v, n_{i-1}) + \frac{d(n_{i-1}, n_i)}{speed},$$

where $n_i = Path(v)[i]$, and $speed$ is a given average speed. Assume that v will pick up k riders at n_i, thus, $wait(v, n_i) = \max\{0, r_x(u) - arrive(v, n_i)\}$, where $r_x(u)$ is the greatest earliest leaving time among the k riders. Therefore,

$$ET = \sum_{r} \big(arrive(v^r, r(t)) - arrive(v^r, r(s)\big). \tag{2}$$

Economic - Waiting Time (EW): This objective is to evaluate the waiting time for all riders before vehicles pick them up with respect to their earliest leaving time.

$$EW = \sum_{r} \max\{0, arrive(v^r, r(s)) - r(u)\} \tag{3}$$

Economic - Excess Distance (ED): This objective measures the extra distance that a vehicle travels when it needs to pick up and drop off riders compared to the distance of directly driving from its starting point to the destination. Let n_i be the ith node in a route,

$$ED = \sum_{v} \Big(\sum_{i=0}^{|Path(v)|-1} d(n_i, n_{i+1}) \quad - d(v(s), v(t)) \Big) \tag{4}$$

Economic - Travel Cost (EC): This objective measures the cost of all rides in total.

$$EC = \sum_{v} \Big(v(c) \times \sum_{i=0}^{|Path(v)|-1} d(n_i, n_{i+1}) \Big) \tag{5}$$

Environmental - Emission (SE): This objective is designed to measure the emission of all the vehicles. By minimising this objective, sharing a ride can reduce pollution. Recall that the emission rate of a vehicle is related to the number of passengers on the vehicle. Therefore, the emission of all the vehicles regarding one solution is

$$SE = \sum_{v} \sum_{i=0}^{|Path(v)|-1} v(e)[l(v, n_i)] \times d(n_i, n_j). \tag{6}$$

where $l(v, n) = |Board(v, n)|$.

Social - Working Time (SW): The working time is calculated from the moment a vehicle leaves its starting point until it arrives at its destination, which is $arrive(v, v(t))$. This objective demonstrates the workload of a vehicle. Regarding the social sustainability, this work tries to balance the workload among all drivers and ensure a sustainable ridesharing service that is fairness-aware. Hence, this objective is defined as the Gini coefficient [5] of all vehicles' working time, $W = \{arrive(v_1, v_1(t)), arrive(v_2, v_2(t)), \cdots, arrive(v_m, v_m(t))\}$ as follow.

$$SW = Gini(W). \tag{7}$$

With the above-defined multiobjectives, we will later explain our algorithm that generates multiple routing options that balance the six objectives of all three pillars of sustainability.

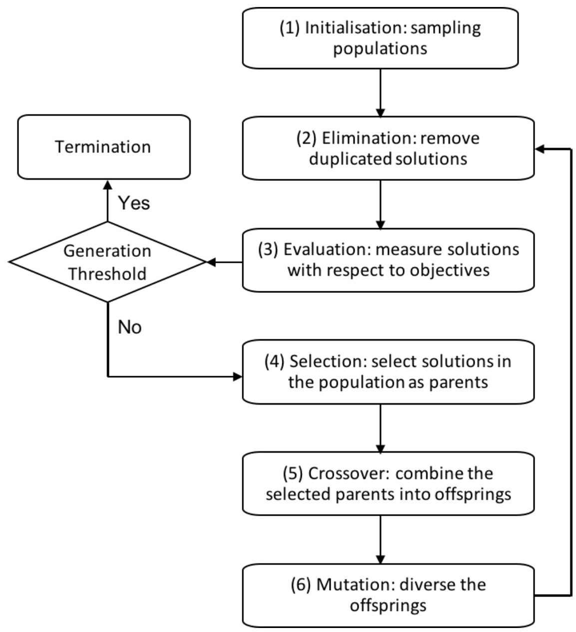


Fig. 2: Modified NSGA3 for Sustainable Ridesharing.

3. NSGA3 for Sustainable Ridesharing

This work adopts an existing genetic algorithm called Non-dominated Sorting Genetic Algorithm 3 (NSGA3) [13] for dynamic routing in the sustainable ridesharing problem. Figure 2 shows the workflow of the NSGA3 with modifications for the ridesharing setting.

NSGA3 requires no configurations of the importance or weights of multiple objectives in the optimisation, but balance them automatically. The optimisation procedure includes:

1. Sampling: Given the number of riders, vehicles and sample size, it generates an initial sample population. In this step, for each solution in the population, we randomly assign vehicles to serve riders, and assign random values as the weights of the picking up and dropping off priority of riders.
2. Elimination: It deletes duplicate solutions in population. And if the size of the current population after elimination is smaller than the initial population, the following introduced crossover process is repeated until the desired number of offspring is fulfilled.
3. Evaluation: For individual solution, it calls the routing algorithm first for a feasible ridesharing arrangement of the solution and evaluates the solutions according to the predefined objectives or any other customised metrics or constraints.
4. Selection: It selects some solutions as parents for generating offspring in next generation. NSGA3 uses a reference points [4] based selection operator. As we applying this genetic algorithm for multiobjective optimisation, this selection is ideal as it is guided by specifying a set of well-maintain diversity in the population regarding different objectives.

5. Crossover: It combines the selected parent solutions to generate offspring solutions. We define the crossover as two parent solutions generating one offspring. The pattern to generate an offspring for each pair of parents is to use the first half chromosomes from a parent and the second half chromosomes from the other parent to generate an offspring solution. Note that our implementation supports splitting both parents into any number of slices and then selecting the same number of slices to generate an offspring.
6. Mutation: It mutates offspring to increase the diversity of the current population. The modified mutation is: for each offspring, we select half riders and change the value of its corresponding chromosomes by (1) changing the vehicle assigned to a rider; (2) increasing its weight of picking up priority by a positive number; (3) increasing its weight of dropping off priority by a random positive number.

Generally speaking, the parameters, including the size of population, number of parents, crossover method, mutation method, number of generations and termination threshold, would affects the performance of the genetic algorithm. After observing the impact of those parameters on small test data, we choose the above-mentioned parameters in our algorithm settings, and the threshold of the number of generations as the condition to terminate the optimisation while attempted to vary the threshold in a wide range to reduce the impact of other static parameters.

This approach will automatically generate multiple routing solutions when we set the size of population greater than 1. In addition, the routing solutions are feasible and balance the economic, environmental and social sustainability, while the algorithm optimise the six objectives. Note that we did not consider the objectives when calculating the shortest path among all nodes of a map graph. This is because the objectives are defined and calculated based on the paths of all the vehicles that they will drive, pick up, and drop off riders. For instance, we cannot get the override distance just for the edge between two nodes.

4. Experimental Evaluation

The main goal of this work is to demonstrate the impact of sustainable objectives in ridesharing routing and the efficiency of our GA-based routing approach. We present 5 groups of experiments with various numbers of riders, vehicles, objectives and generations to illustrate the effectiveness of our approach. This section evaluates the performance with respect to the standard metrics in the field of vehicle routing [14,3] and includes social and environmental metrics such as the waiting time of a rider to start a ride.

4.1. Data Sources

We use the Cargo benchmark dataset [18], which takes data from the ridesharing company Didi. The instances have maximum 65,500 riders and 50,000 vehicles over a long time horizon and a scale of 876km^2 area. Since we focus on one-shot routing, we take slices from the dataset for our evaluation. Note that in practice, new routes can be calculated as more requests come in.

– Road Map: We use the road map of Manhattan from Cargo [18]. It has 12,320 nodes, 15,722 edges in an area of 59km^2.

Table 2: Information about Group Instances.

Parameters	Variation in Instances
# Riders	[53, 48, 24, 16, 16]
Riders' Location	[Original, Noise in s, Noise in t, Noise in Journey]
# Vehicles	[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Vehicle Type	[1, 3, 6, 10]

Table 3: Algorithm Parameter Settings

Parameters	Value/Range
Population	10
Offset	5
Generation	[100, 200, 300, ..., 1000]
Objectives	[EW, ET, ED, EC, SE, SW]

– Instances: We design 5 groups of riders, and experiments with instances as detailed in Table 2. The variation column shows the range of parameters that we vary in the experiments.
  - Riders: For each group, there is a fixed number of riders. To discovery the possibility of riders to share a vehicle, we first construct a graph using the starting points and destination of original 5,033 riders as nodes and their journeys as edges, and then select a set of nodes whose clustering coefficient is not zero. With the selected node set, we extract riders whose starting points or destination fall into the set. Last, we put those riders into 5 clusters by K-means clustering method, which are the original riders in each group of experiments. In addition, we generate another 3 variations of the original riders by adding noises to either the riders' starting points, or destination, or both of them. The noise is added by randomly change the target nodes to one of its neighbors on the road map.
  - Vehicles: For each group, the number of vehicles range from 1 to 10 by 1 and location of vehicles are randomly picked from Cargo. At each fixed number of vehicles, we allow all vehicles to be either one type of vehicles from Table 1.

With above constructed instances, we can examine the impact of number and location of riders, and number and type of vehicles on the performance of ridesharing routing, by varying the above parameters in the experiments.

4.2. Benchmark - Greedy Routing

An greedy-based routing algorithm is designed and implemented as the baseline for ridesharing performance evaluation. For a group of riders, the greedy routing algorithm first sorts all ride requests by their posting time. Then, for each ride request in the rank, it searches over all the vehicles to find one that is closet to the starting point of the request at its early leaving time. The selection is based on the distance from the current location of a vehicle, to its travelling destination if it has passenger on board and to the starting point of the request. After that, the selected vehicle will travel to the starting point of the request and directly drive the rider to its destination. Note that the algorithm will not change the arrangement made before receiving the request of new riders. At the end, we have the routes of all vehicles and evaluate the routes with respect to the same objectives used in our algorithm.

In the following discussion, all the displayed results regarding the six objectives is the relative reduction in comparison with the ones of greedy routing algorithm.

4.3. Experiment Setting

The implementation of the GA algorithm in the Python programming language (using pymoo[6]) allows configuring the population size, number of generations, the offspring rate, and muting/enabling different objectives. Table 3 displays the settings of algorithm parameters. To evaluate our approach, we set the population size and offspring size to 10 and 5, respectively. In the experiments, the generation varies from 100 to 100 by 100 to briefly demonstrate the effects of large generations. Additionally, to evaluate the effectiveness of our approach with respect to the 6 different objectives from 3 aspects, we set up experiments with either one objective or the other left 5 objectives to compare the optimised routes with those after optimising the all 6 objectives. [7] Therefore, in the rest of this section, we will illustrate our experiment results and answer the questions: How the generation affects the performance, how the number of vehicles affects the performance and how the setting of objectives affects the performance and the sharing rate of each vehicle.

4.4. Experiment Results

This section examines and illustrates our experiment results regrading the sharing rate, optimisation and balance of objectives and diversity among the generated ridesharing routes. The sharing rate measures the efficiency of the ridesharing routing algorithm in promoting multiple riders in one vehicles during their trips. With respect to the 6 different objectives, we will compare the efficiency of the ridesharing routing algorithm in optimising each individual objectives against the all objectives and their balance, in order to have a further view of the relationship among the 6 different objectives. At the end, we evaluate the diversity of the generated solutions by the GA-based ridesharing routing algorithms which can be an ideal voting pool for what we call participatory route selection (see Section 5).

Sharing Rate To measure the efficiency of our routing algorithm in promoting ride sharing, we define a sharing rate metric, Sharing Rate Over Time (SROT), as follow:

$$SROT = \sum_{i=0}^{|Path(v)|} |Board(v, Path(v)[i])| * d(Path(v)[i], Path(v)[i+1])/speed.$$

For one vehicle, this metric calculates the average number of riders who share the vehicle along its complete travelling path.

Figure 3 uses boxplot to illustrate the sharing rate of the generated routes for 53 riders, 5 vehicles with varied capacities while setting up the 6 objectives and 1000 generations. The 4 subplots correspond to the 4 variations of riders' location. Comparing to the results of vehicles with a capacity of 1, the sharing rate of greater capacities is higher. In total, the capacity of 3, 6, 10 increase the sharing rate by 61.63%, 96.35% and 110.19% on average,

[6] `https://pymoo.org/`

[7] To comply with the anonymity requirements of the track, we excluded the link to the repository in this version. The instance and algorithm implementation files will be provided in the next version. For the complete experiment results please refer to the support material.

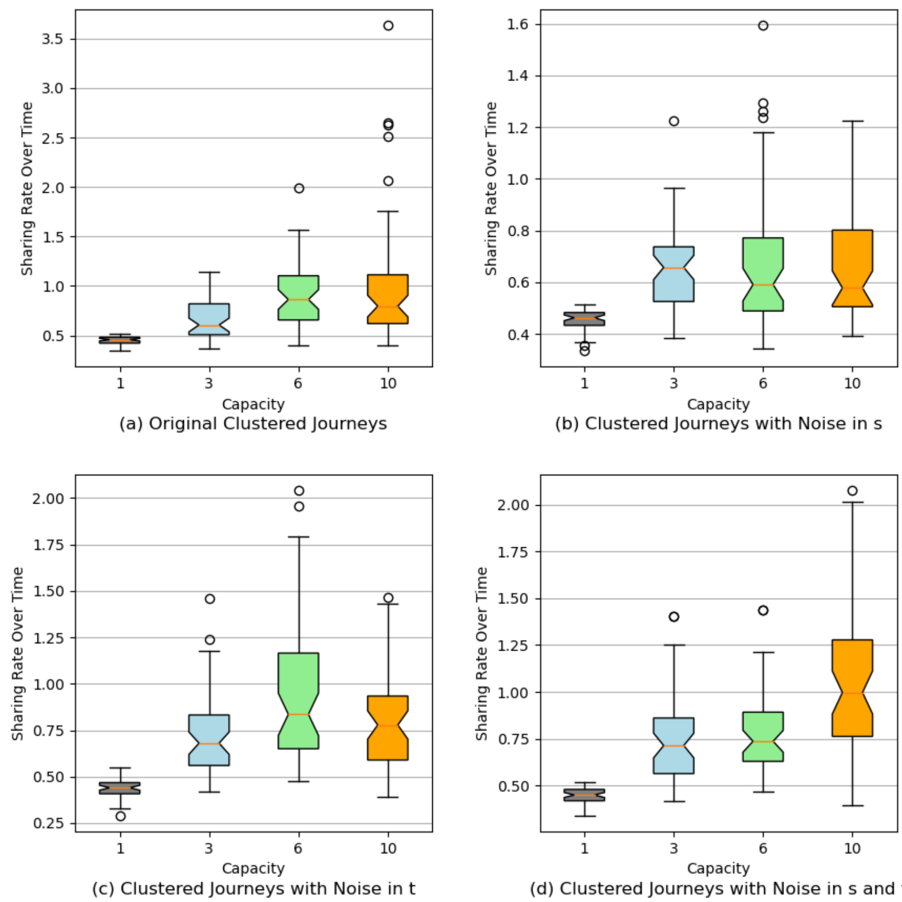


Fig. 3: Sharing Rate of Arranged Routes for Riders in Group 1 with 5 Vehicles after Optimising 1000 Generations for 6 Objectives

respectively. In addition, comparing the mean sharing rate of 4 different vehicle capacities when the riders' locations vary, the vehicles with capacity of 6 outperform the others in Figure 3(a) and (c), while capacities of 3 and 10 have greatest mean sharing rate in Figure 3(b) and (d), respectively. The standard deviation of the 4 different capacities when the locations of riders vary, can represent the impact of riders' locations on the ridesharing efficiency of our routing algorithm, which are 0.0096, 0.0493, 0.1255 and 0.1928. This infers that when the capacity of vehicles increases, their sharing rates are more likely to be affected by the locations of riders.

Furthermore, Figure 4 shows the average sharing rate of vehicles when the number of riders and vehicles changes. The darker area in the bottom right corner of Figure 4 marks the sharing rate lower than 0.5. In these area, capacities of all available vehicles are greater than the number of riders, which indicates there is not enough riders to share

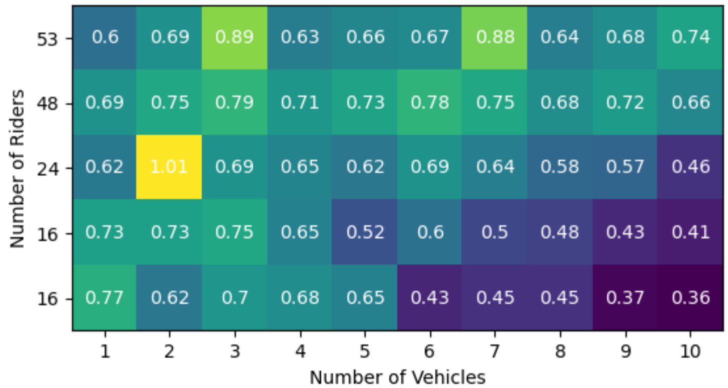


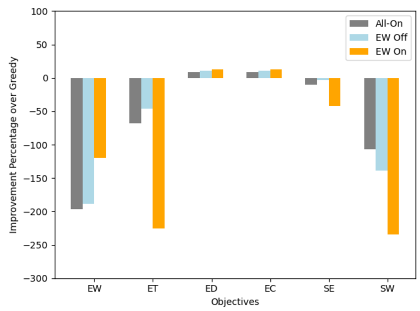
Fig. 4: Average Sharing Rate of Vehicles with Capacity of 3

vehicles. Considering the optimising objectives, SW and EW, which aim to balance the working hour of all vehicles and reduce the waiting time of riders, the sharing rate of vehicles will reduce when the number of riders decrease to be smaller than the overall capacities of vehicles. The brighter cells in Figure 4 are in the top and left corner when there are more riders for the algorithm to arrange ridesharing routes for vehicles.

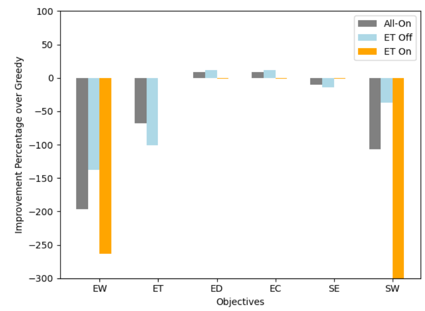
To summarise, when the capacity of vehicles increases, the sharing rate will increase on average. However, it does not imply higher capacity will always result in greater sharing rate. First, sharing rate is related to the location of riders. When riders are randomly located at their starting point but are travelling to the same direction, vehicles with a capacity of 3 would gain better sharing rate. When riders start their journeys from close location but travel to different destinations, the vehicles with a capacity of 6 performs better. In addition, the number of riders and vehicles have impact on the sharing rate of vehicles. When the number of riders is far greater than the capacities of all vehicles, the sharing rate of vehicles is more likely to be higher. And knowing the number of riders is small, sending out less vehicles with smaller capacity can help increase their sharing rate.

Objectives To evaluate the performance of our routing algorithm with respect to individual objectives, we use the greedy routing algorithm as the baseline. Figure 5 shows the percentage reduction of the generated solutions to arrange routes for 53 riders, 3 vehicles with capacity of 3. Each sub figure displays the percentage reduction of individual objectives when optimising all objectives (All-On), one objective (EW/ET/EC/ED/SE/SG On) and the other objectives (EW/ET/EC/ED/SE/SG Off). Throughout all charts in Figure 5, we can find that all objectives are not independent. For instance, as Figure 5e shows, when only minimising emission, the routing solution turns out to have the better performance regrading the objective EW and ET. In addition, the switch optimisation results of ET and SW affect each other the most, as Figure 5b and 5f showed, i.e. when social inequality improves most the travelling time increases most and vice versa.

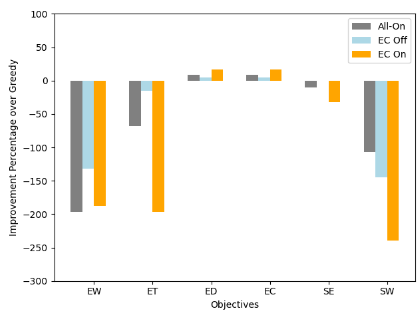
Moreover, the percentage reductions at objective ED and EC are always at the same pace over the six group of comparison. And our ridesharing algorithm outperforms greedy routing in optimising ED and EC by 8.9% while optimising all objectives, 16.9% if only



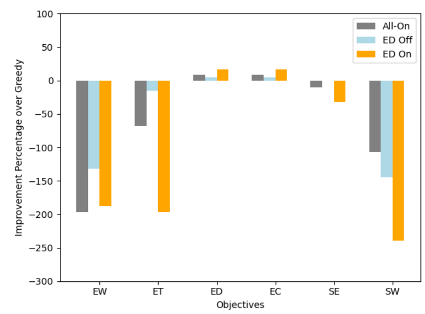
(a) EW



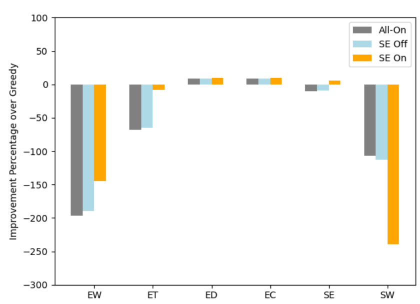
(b) ET



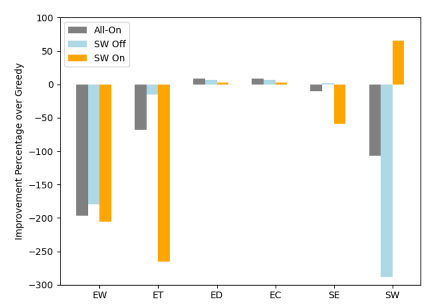
(c) EC



(d) ED



(e) SE



(f) SW

Fig. 5: Percentage Reduction in Comparison of Optimising One Objective, the Other Objectives and All Objectives with 53 Riders, 3 Vehicles with Capacity of 3.

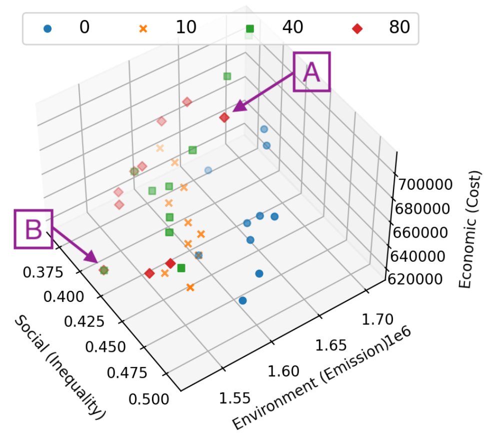


Fig. 6: Evolution of GA Solutions.

optimising those two, and at lest 5.2% in other cases. The ridesharing routes generated by our algorithm constantly require higher results in EW and ET, because greedy routing always try to let each rider get on a vehicle as quick as possible and directly drive individual riders from their starting points to destinations, which results in minimum ET and the better performance of reducing EW. As to SW, the ridesharing routes contribute to unbalanced working hours among vehicles excepting when only optimising SW and improved it by 65.7%. Hence, the ridesharing algorithm is capable to optimising individual objectives as well as dependent multiple objectives.

Notably, the improved percentage over greedy regarding the 6 objectives falls in varied scales, especially over -100% of EW in 5a and 16.9% of ED in 5d. The reason is that the primer task of ridesharing to fulfil is to drive all riders from their distinct starting points to their destinations, leaving limited space for improvements of ED, EC and SE. The public transportation, such as underground, that has numeric capacity and riders get on and off it at stable stops, are believed to be more economic and environment friendly. In our future work, public transportation will be considered in ridesharing, and our approach will try to deliver suggested routes involve public transportation with an estimated reduction of economic cost and emission.

To further understand the balancedness of ridesharing routing solutions with respect to the economic, environmental and social aspects of sustainability, we plot the results for 53 riders and 10 vehicles with capacities of 3 in Figure 7. In general, as the optimisation proceeds, the environmental and economic sustainability improves while the social inequality fluctuates between -1.5 and -2.5. In addition, we plotted the routing results in 4 sampling generations for 150 riders and 100 vehicles with capacity of 1 aiming at optimising 6 objectives. Figure 6 presents the evolution of the populations at generations 0, 10, 40 and 80. The solutions in each population have diverse effectiveness against the three aspects of sustainability. Basically, as the optimisation proceeds, the social inequality decreases. The initial generation (in blue) has greater social inequality than other generations while the 80*th* generation has the lowest inequality. However, from the perspective of economics and environment, the costs and emissions do not improve for all solutions. Although, the

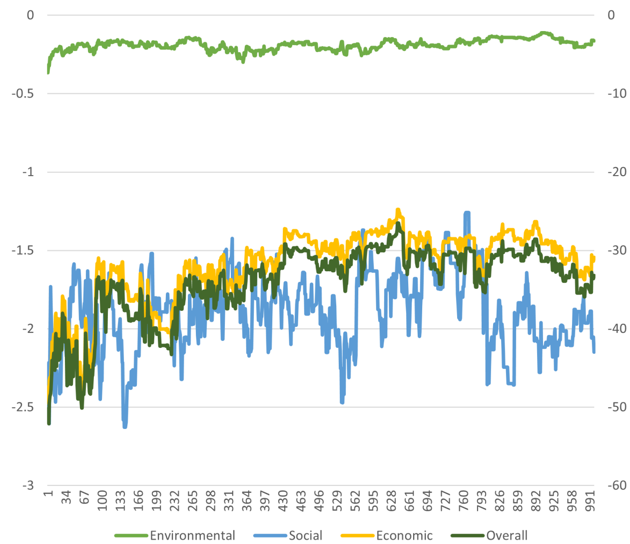
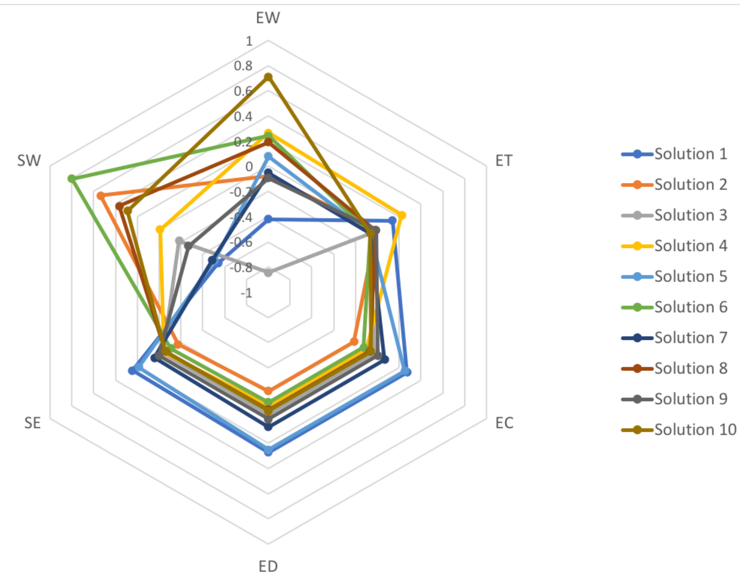


Fig. 7: Performance on Optimising Objectives over Generations

ridesharing routing algorithm can balance and improve the overall objectives, but optimising SW separately would result in lower social inequality.

Furthermore, it is observable that we have a more diverse set of solutions. For instance, solution A (in the 80th generation) has a low economic cost and social inequality which compensates for its high environmental emission level. The other notable solution is labelled with a boxed B which performs well against all the three dimensions. We will further discussion the solution diversity in the following section.

Solution Diversity To understand and illustrate the diversity of populations, we plot the 10 solutions to routing 16 riders and 10 vehicles with capacity of 3 in Figure 8. It presents the radar plot of effectiveness of each solution with respect to the 6 objectives, after normalising the results by the mean value of 10 solutions. The smaller number of an objective implies a better performance of a solution on that objective. Among these 10 solutions, the effectiveness regarding social sustainability and riders' waiting time vary greater than the others. This is because the changes in allocating riders from a vehicle to another directly affects the working time of the vehicles and their waiting time. The population offers diverse solutions that improve social inequality between 0.13 to 0.52 whereas riders' waiting time between 2768 to 254. We expect to take advantage of populations' diversity in the next phase of our work on sustainable mobility and allow riders to vote on routes with respect to their concerns, such as economic costs, environment emissions, or social inequality.



Solution	EW	ET	EC	ED	SE	SW
1	940	25026.50	488337	162779	59578.75	0.13
2	1491	21725	301746	100582	39623.52	0.44
3	254	20593	371610	123870	46123.18	0.23
4	2051	26893	345294	115098	45876.63	0.28
5	1749	20593	480135	160045	56762.88	0.14
6	2009	20593	335763	111921	42608.76	0.52
7	1535	20593	410178	136726	49815.50	0.15
8	1930	21545	358578	119526	44954.98	0.39
9	1470	21725	386103	128701	47976.06	0.21
10	2768	20593	361275	120425	45021.09	0.37
Avg	1619.70	21987.95	383901.90	127967.30	47834.14	0.29
Max	2768	26893	488337	162779	59578.75	0.52
Min	254	20593	301746	100582	39623.52	0.13

Fig. 8: Diversity of 10 Solutions w.r.t. Each Objective

5. Conclusions

In this work, we presented a multiobjective evolutionary approach based on GA algorithm for generating routing options in sustainable ridesharing. Although there are well-studied multiobjective optimisation methods [12], a GA algorithm generates a diverse set of solutions naturally for the ridesharing problem. Our method is not only sustainability-aware but also establishes a foundation for explainable, participatory, and dynamic ridesharing services.

Explainability for Riders, Drivers, and Operators: In comparison to data-driven techniques with black-box optimisation components, in our approach, stakeholders can be provided with visualisations to see how different objectives (e.g., minimising emissions) affect routing solutions. For instance, they can be presented using graphs as in Figure 5 and with explanations on how waiting a bit more (in comparison to using private rides) can benefit the environment or the fairness of the service for drivers.

Promoted sustainability: With the awareness of sustainability, our approach provides a feasible solution to promote ridesharing with respect to varied type of vehicles, and highlights the factors associated with the sharing rate when promoting ridesharing is possible. We demonstrate that the average sharing rate among riders can be improved when the capacity of vehicles increases, although higher capacity can not always result in greater sharing rate. In addition, the locations and numbers of riders also affect the sharing rate the number of riders and vehicles have impact on the sharing rate of vehicles. For numerous riders starting from close locations, vehicles with a greater capacity would contribute to a higher sharing rate, where riders from random locations but heading to close destinations can share vehicles with a lower capacity for a higher sharing rate.

Participatory Route Selection: Building on this approach, ridesharing operators can present routing options to riders (or autonomous agents that represent riders) and allow voting among them. This way, users can directly participate in the route selection process and opt for the most collectively equitable route. Our diverse set of GA solutions are not ranked. Thus, a set of riders may prefer one over another and to allow that, we aim to extend our work by adding a preference/vote-based route ranking module in the future.

Dynamic Fine-Tuning: Our approach allows dynamic fine-tuning over time. Users and service operators can inspect routing solutions, evaluate if they are realistic and feasible, and participate in fine-tuning the route generation algorithm and the objective weights to set trade-offs. One can use focus groups for such a tuning over time—e.g., as a city and its citizens change—to enable dynamic fine-tuning of sustainable ridesharing services.

We aim to extend our work by integrating a participatory route selection process and allowing users to vote over a diverse set of routing solutions with all the objectives and then also muting one or two objectives to provide solutions that match diversity in users' preferences. With a better understanding of users' preferences, we aim to explore other methods for multiobjective optimisation in the context of ridesharing service. For example, we can define the assignment of a rider to a vehicle as a move and evaluate the move with respect to the multiple objectives, and then adopt reinforcement learning for this problem. Moreover, we plan to test the efficacy of our approach in larger datasets and investigate simulation-based methods to analyse how different map structure and spatio-temporal properties of requests affect the optimality and equitability of solutions.

Data access statement. This study was a reanalysis of data that are publicly available from the the Cargo benchmark dataset [18]. Implementations and data derived through

the reanalysis undertaken in this study are available from the public GitHub repository at https://github.com/Miya-Liu/equitable-ridesharing.

Acknowledgements. We thank the anonymous reviewers and participants at the ATT'22: Workshop on Agents in Traffic and Transportation, for their incisive comments that were most useful in revising this paper. This work was supported by the UK Engineering and Physical Sciences Research Council (EPSRC) through a Turing AI Fellowship (EP/V022067/1) on Citizen-Centric AI Systems (https://ccais.ac.uk/) and the platform grant entitled "AutoTrust: Designing a Human-Centred Trusted, Secure, Intelligent and Usable Internet of Vehicles" (EP/R029563/1). For the purpose of open access, the author has applied a creative commons attribution (CC BY) licence to any author accepted manuscript version arising.

References

1. Atasoy, B., Ikeda, T., Song, X., Ben-Akiva, M.E.: The concept and impact analysis of a flexible mobility on demand system. Transportation Research Part C: Emerging Technologies 56, 373–392 (2015)
2. Bardaka, E., Hajibabai, L., Singh, M.P.: Reimagining ride sharing: Efficient, equitable, sustainable public microtransit. IEEE Internet Computing 24(5), 38–44 (2020)
3. Ben Cheikh-Graiet, S., Dotoli, M., Hammadi, S.: A tabu search based metaheuristic for dynamic carpooling optimization. Computers & Industrial Engineering 140, 106217 (2020), https://www.sciencedirect.com/science/article/pii/S0360835219306862
4. Blank, J., Deb, K., Roy, P.C.: Investigating the normalization procedure of nsga-iii. In: EMO. pp. 229–240 (2019)
5. Catalano, M.T., Leise, T.L., Pfaff, T.J.: Measuring resource inequality: The gini coefficient. numeracy 2 (2): Article 4 (2009)
6. Colorado Department of Transportation: Overview of transit vehicles (2022), https://www.codot.gov/programs/innovativemobility/assets/commuterchoices/documents/trandir_transit.pdf
7. Cormen, T.H.: Section 24.3: Dijkstra's algorithm. Introduction to algorithms pp. 595–601 (2001)
8. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: Nsga-ii. IEEE transactions on evolutionary computation 6(2), 182–197 (2002)
9. EMISIA SA Corporate: Copert the industry standard emissions calculator (2018), https://www.emisia.com/utilities/copert/
10. Fonseca, C.M., Fleming, P.J.: Multiobjective genetic algorithms. In: IEE colloquium on genetic algorithms for control systems engineering. pp. 6–1. Iet (1993)
11. Furuhata, M., Dessouky, M., Ordóñez, F., Brunet, M.E., Wang, X., Koenig, S.: Ridesharing: The state-of-the-art and future directions. Transportation Research Part B: Methodological 57, 28–46 (2013)
12. Hayes, C.F., Rădulescu, R., Bargiacchi, E., Källström, J., Macfarlane, M., Reymond, M., Verstraeten, T., Zintgraf, L.M., Dazeley, R., Heintz, F., et al.: A practical guide to multi-objective reinforcement learning and planning. Autonomous Agents and Multi-Agent Systems 36(1), 1–59 (2022)
13. Jain, H., Deb, K.: An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part ii: Handling constraints and extending to an adaptive approach. IEEE Transactions on evolutionary computation 18(4), 602–622 (2013)
14. Laporte, G.: Fifty years of vehicle routing. Transportation Science 43(4), 408–416 (nov 2009)

15. Liu, M., Yazdanpanah, V., Stein, S., Gerding, E.: Multiobjective routing in sustainable mobility-on-demand. In: ATT'22: Workshop Agents in Traffic and Transportation, July 25, 2022, Vienna, Austria: Part of IJCAI 2022 (23/07/22 - 29/07/22). pp. 47–61 (July 2022)
16. Millen, S., Page, E.: Transport and environment statistics 2021 annual report (2021), `https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/984685/transport-and-environment-statistics-2021.pdf`
17. Murata, T., Ishibuchi, H., et al.: Moga: multi-objective genetic algorithms. In: IEEE international conference on evolutionary computation. vol. 1, pp. 289–294. IEEE Piscataway, NJ, USA (1995)
18. Pan, J.J., Li, G., Hu, J.: Ridesharing: simulator, benchmark, and evaluation. Proceedings of the VLDB Endowment 12(10), 1085–1098 (2019)
19. Purvis, B., Mao, Y., Robinson, D.: Three pillars of sustainability: in search of conceptual origins. Sustainability science 14(3), 681–695 (2019)
20. Ramchurn, S.D., Stein, S., Jennings, N.R.: Trustworthy human-ai partnerships. iScience 24(8), 102891 (2021)
21. Simonin, G., O'Sullivan, B.: Optimisation for the ride-sharing problem: a complexity-based approach. In: ECAI 2014, pp. 831–836. IOS Press (2014)
22. UK Government: Driving licence categories (2022), `https://www.gov.uk/driving-licence-categories`
23. UK NAEI - National Atmospheric Emissions Inventory: Primary NO2 emission factors for road vehicles (2022), `https://naei.beis.gov.uk/resources/Primary_NO2_Emission_Factors_for_Road_Vehicles_NAEI_Base_2021_v3.pdf`
24. Weigel, B.A., Southworth, F., Meyer, M.D.: Calculators to estimate greenhouse gas emissions from public transit vehicles. Transportation research record 2143(1), 125–133 (2010)
25. Yu, Y., Wu, Y., Wang, J.: Bi-objective green ride-sharing problem: Model and exact method. International Journal of Production Economics 208, 472–482 (2019)

A. NSGA3 Implementation

A.1. Sampling

Algorithm 2: Sampling(n_riders, $n\_vehicles$, $n\_samples$)

```
Population ← ∅ ;
for 1 ≤ s ≤ n_samples do
    solution ← ∅;
    for 1 ≤ i ≤ n_riders do
        chore['vehicle'] ← random(n_vehicles);
        chore['pickup_weight'] ← random(100 * n_riders);
        chore['dropoff_weight'] ← random(100 * n_riders);
        solution.append(chore);
    end
    Population.append(solution);
end
return Population;
```

A.2. Elimination

Algorithm 3: Elimination(Population, $n\_samples$)

```
foreach s ∈ Population do
    foreach s' ∈ Population - {s } do
        if s equals to s' then
            Population ← Population - {s' };
        end
    end
end
if Population.size < n_samples then
    Population ← Crossover();
end
return Population;
```

A.3. Evaluation

Algorithm 4: Evaluation(V, $solution$, obj_list)

```
basic_objs ← [EW,ET,EC,ED,SE,SW];
Routes ← Routing(solution) ;
res ←;
foreach obj ∈ basic_objs do
    if obj ∈ obj_list then
        res.append(Cal(obj, Routes))
    end
end
return res;
```

A.4. Selection

Algorithm 5: Selection(Population, $n\_selection$, $n\_parents$)

```
count ← n_selection × n_parents;
multi ← ⌈ count / Population.size ⌉ ;
selection ← pymoo.util.misc.random_permuations(multi,
  Population.size)[0: count] ;
return selection;
```

A.5. Crossover

Algorithm 6: Crossover(Parents, $n\_offspring$, n_slices)

```
parent_A ← Parents[1];
parent_B ← Parents[2];
offsprings ← ∅ ;
for 1 ≤ o ≤ n_offspring do
    one_offspring ← ∅ ;
    num ← parent_A[o].size÷n_slices + 1;
    for 1 ≤ j ≤ num do
        from_slice ← num ×j;
        to_slice = num ×j + num ;
        if to_slice > parent_A[o].size then
            to_slice ← parent_A[o].size;
        end
        if j mod 2 = 0 then
            one_offspring.concat(parent_A[o][from_p:to_p]) ;
        else
            one_offspring.concat(parent_B[o][from_p:to_p]) ;
        end
    end
    offsprings.append(one_offspring);
end
return offsprings;
```

A.6. Mutation

Algorithm 7: Mutation(One_Offspring)

```
mutation_items ← [1 to One_Offspring.size/2];
mutated_offpsring ← ∅ ;
foreach i ∈ mutation_items do
    item ← One_Offspring[i];
    item[i]['vehicle'] ← random(n_vehicles);
    item[i]['pickup_weight'] ← random(10 ∗ n_riders);
    item[i]['dropoff_weight'] ← random(10 ∗ n_riders) + 1;
    mutated_offpsring.append(item);
end
return mutated_offpsring;
```