Diversity-Aware Recommendation for Human Collectives

Diversity-Aware Recommendation for Human Collectives

Pavlos Andreadis and **Sofia Ceppi** and **Michael Rovatsos** and **Subramanian Ramamoorthy** [1]

Abstract.

Sharing economy applications need to coordinate humans, each of whom may have different preferences over the provided service. Traditional approaches model this as a resource allocation problem and solve it by identifying matches between users and resources. These require knowledge of user preferences and, crucially, assume that they act deterministically or, equivalently, that each of them is expected to accept the proposed match. This assumption is unrealistic for applications like ridesharing and house sharing (like airbnb), where user coordination requires handling of the diversity and uncertainty in human behaviour.

We address this shortcoming by proposing a diversity-aware recommender system that leaves the decision-power to users but still assists them in coordinating their activities. We achieve this through taxation, which indirectly modifies users' preferences over options by imposing a penalty on them. This is applied on options that, if selected, are expected to lead to less favourable outcomes, from the perspective of the collective. The framework we used to identify the options to recommend is composed by three optimisation steps, each of which has a mixed integer linear program at its core. Using a combination of these three programs, we are also able to compute solutions that permit a good trade-off between satisfying the global goals of the collective and the individual users' interests. We demonstrate the effectiveness of our approach with two experiments in a simulated ridesharing scenario, showing: (a) significantly better coordination results with the approach we propose, than with a set of recommendations in which taxation is not applied and each solution maximises the goal of the collective, (b) that we can propose a recommendation set to users instead of imposing them a single allocation at no loss to the collective, and (c) that our system allows for an adaptive trade-off between conflicting criteria.

1 Introduction

Sharing economy applications constitute an interesting domain for multi-agent resource allocation and coalition formation. In these applications, users act as producers and consumers of resources, aiming to find peers to share the resources with, while a platform supports them during peer discovery and resource sharing. These fundamental aspects of sharing applications highlight how the decisions of the *collective* of users lead to a globally desirable outcome, while the choices of a single user alone have no such power. However, the services the sharing applications provide should leave the decision-making power to each user in order to allow her to express her preferences and satisfy her individual needs. Consequently, instead of facing the problem of identifying a solution for the collective of users, the platform needs to help them in coordinating their individual choices in such a way that the goal of the collective can still be achieved. In this work, we tackle this issue by providing a recommender system that accounts for user preferences and facilitates the coordination among users, in scenarios where users perform *joint* tasks in subgroups consisting of the members of a larger collective. In the example of a ridesharing application, each user could be aiming to achieve the best fit between his schedule and the planned ride. However, since rides cannot be achieved without the collaboration of multiple users, the collective goal of facilitating as many users as possible will come into conflict with this individual preference.

Many multi-agent applications face the problem of coordinating autonomous agents that aim to share resources, which can be seen as a resource allocation problem. Traditional approaches to this problem typically express a degree of centralised control in order to provide functional, viable solutions to most, if not all, participating users. In particular, several algorithms have been designed that identify stable matches between users and resources [14, 18, 9]. However, users cannot affect the algorithm. The most flexible approaches proposed in the literature make use of sequential mechanisms that allow users to accept or reject the solution currently proposed to them [11, 1]. Finally, some of the existing approaches assume that the system knows the complete preference ordering of users over, e.g., other users [8]. The crucial drawback of this type of work is that it focuses on problems like how to assign children to schools, how to allocate students to shared rooms, and how to match donors with patients in the kidney exchange market. In these scenarios, the possibility that users might prefer not to be allocated, rather than be to allocated as prescribed by the algorithm, is not considered. However, this assumption makes the adoption of such algorithms unrealistic for several sharing applications, e.g. ridesharing and joint event planning.

Indeed, these approaches lack a crucial characteristic that systems that mediate between humans should have: the ability to model human diversity and consider the uncertainty of human behaviour. Indeed, human decision making is affected by multiple factors: social, cultural, psychological, personal, and available information [16], that are unique for each individual. These create variations *among* individuals in terms of preferences over given characteristics of the peers and resources, leading to diversity across users. Moreover, the variability of these factors adds complexity to the decision-making process *within* each individual, to the extent that near identical situations may lead to significantly different behaviour. This leads to uncertainty regarding user behaviour. Crucially, a sharing application that does not account for diversity and the uncertainty of human behaviour is likely to fail in supporting large-scale coordination within human collectives

[1] School of Informatics, University of Edinburgh, United Kingdom, email: p.andreadis@sms.ed.ac.uk, sceppi@inf.ed.ac.uk, mrovatso@inf.ed.ac.uk, sramamoo@staffmail.ed.ac.uk

effectively.

Ideally, a system could address user diversity and provide a very personalised service by eliciting information from users and understanding their general preferences over some defined characteristics of the services. In the literature, there is ample work providing techniques for learning user preferences in an accurate way [17, 7, 4, 12, 10, 5, 13, 21, 15] and that focuses on delivering personalised services [2, 10, 19, 6]. However, apart from the tricky task of eliciting information from users and understanding how any given factors affect user preferences, a system has to deal with the problem of understanding which factors affect human behaviour. This is a currently open problem that is attracting attention from researchers interested in, e.g., social computation and psychology. Given this, a sharing application should account for uncertainty in human behaviour. In particular, in designing such an application, the designer has to (i) pay attention to the type of interaction between the system and the users (both individually as a collective) and (ii) allow for flexibility such that it can adapt to unforeseen behaviour. In this work, in order to provide such flexibility, instead of offering users a single option computed with the techniques discussed above, we focus on the problem of recommending multiple options to users.

The allocation problem faced by sharing applications, whose users aim to find peers to share a resource or a task with, is of a combinatorial nature. As such, when a system offers multiple options, all the users assigned to a task have to agree to it, i.e. they have to choose the task for it to happen. Since there is no guarantee that users' independent choices are consistent with one another, the system has to provide a coordination mechanism. This problem can be seen as a coalition formation problem [20] in which incentives to stay in a suggested coalition may be provided to users who would otherwise reject it. Cost of stability [3] and taxation [23] are two techniques proposed to provide such incentives and achieve the desired effect by artificially modifying users' preferences. In this work, instead of using explicit coordination techniques that require communication with users, we provide an indirect coordination mechanism, based on the techniques used in coalition formation problems. More specifically, we introduce a taxation mechanism in the options computation process.

Note that a sharing application that aims to adapt to the user collective but also wants to account for the interests of individual users, faces a multi-criteria optimisation problem [17]. Indeed, the interest of the collective (that requires users' collaboration) is in conflict with the interest of individual users whose aim is to obtain what is the best option for themselves. The approach we propose allows the sharing application to specify to which extent it wants to account for individual users' interests and identify options that achieve the desired trade-off between conflicting interests.

The three main contribution of this work are:

- A formulation of the user coordination problem faced by sharing economy applications, in such a way that it allows for the explicit representation of the diversity and uncertainty in human behaviour;
- A diversity-aware system for the coordination of users in sharing economy applications that does not require communication between users;
- Experimental evidence for the necessity of taking human diversity and uncertainty into account when coordinating such applications. Specifically, we demonstrate that we can replace direct allocations with recommendation sets at no cost, while also allowing for adaptively trading-off between various criteria of optimality.

The remainder of this paper is organised as follows. In Section 2, we provide a formal description of the allocation problem that characterises sharing applications, propose a diversity-aware approach that aims to account for diversity and uncertainty of human behaviour, and describe our framework. Section 4 proposes a detailed description and formulation of the mixed integer linear programs used in our optimisation framework. The experimental evaluation and obtained results are described in Section 5. Finally, Section 6 concludes the paper.

2 Formal Model

In this section, we formalise the resource allocation problem that characterises sharing economy applications.

Consider a set of tasks $J = \{1, \dots, |J|\}$. Each task $j \in J$ is associated with one and only one user who owns the task, for example, the user is the owner of the resource that will be shared, or whoever initiated a concrete sharing task. Since we assume that each owner has one and only one task associated with her, for the sake of simplicity, we interchangeably refer to $j \in J$ as both the task and its owner. Let $I = \{1, \dots, |I|\}$ be the set of users who do not own a task, and K the set of all users, i.e. $K = I \cup J$. Each owner $j \in J$ aims to find non-owners to share her task with and each non-owner $i \in I$ aims to find one task to join. All users have requirements and preferences about tasks and users to share a task with.

Let $\mathbf{x} = \{x_{1,1}, \dots, x_{1,|J|}, \dots, x_{|I|,1}, \dots x_{|I|,|J|}\} \in \mathbf{X}$ define which non-owner joins each task, i.e., $\mathbf{x}$ is an allocation of non-owners to tasks, where $\mathbf{X}$ is the set of allocations. In particular, if i is allocated to task j then $x_{i,j} = 1$, otherwise $x_{i,j} = 0$.

The preferences of each user $k \in K$ are represented by a utility function $u_k : \mathbf{X} \to \mathbb{R}$ which provides a complete ranking over potential allocations $\mathbf{x} \in \mathbf{X}$. Similarly, the system-level utility function is defined as $U_s : \mathbf{X} \to \mathbb{R}$.

Crucially, in sharing economy applications, single users and the collective of users have conflicting interests. While a user i aims to maximise her utility $u_i(\cdot)$, the interest of the collective of users, represented by the system-level utility function $U_s(\cdot)$, is related to the overall benefit the users can achieve. For example, $U_s(\cdot)$ may consider the sum of the user utility or the number of users that are allocated to tasks. Given this, it is obvious that the maximisation of $U_s(\cdot)$ provides no guarantee to individual users in terms of achieved utility. In order to provide such a guarantee, the application designer should, e.g., maximise the fairness of the solutions (i.e., minimise the difference between the utility achieved by every user) or maximise the minimum single user utility. However, in this case, no guarantee is given in terms of system-level utility.

The aim of the application is to aid users in finding compatible peers by suggesting allocations while accounting for this conflict of interest. However, it is fundamental to highlight that not all allocations are guaranteed to occur. Indeed, each user $k \in K$ selects an allocation from a set R or recommended solutions independently and without direct coordination with other users, according to a *user response model*. The three user response models typically used in the literature are [21]:

- *noiseless* response model: each user acts deterministically and always selects the solution that would maximise her utility. Formally, if $\mathbf{x} \in R$ is such that $u_k(\mathbf{x}) \geq u_k(\mathbf{x}'), \forall \mathbf{x}' \in R$ then $p_k(\mathbf{x}) = 1$ otherwise $p_k(\mathbf{x}) = 0$ for all $k \in K$, where $p_k(\mathbf{x})$ is the probability with which user $k \in K$ chooses solution $\mathbf{x}$.
- *constant noise* response model: each user selects the solution that would maximise her utility the majority of the time irrespective of the utility of other solutions. Each of the remaining solutions

is chosen with a equal small probability. Formally, if $\mathbf{x} \in R$ is such that $u_k(\mathbf{x}) \geq u_k(\mathbf{x}'), \forall \mathbf{x}' \in R$ then $p_k(\mathbf{x}) = \alpha$ otherwise $p_k(\mathbf{x}) = \beta$ with $\alpha >> \beta$ and $\sum_{\mathbf{x} \in R} p_k(\mathbf{x}) = 1$ for all $k \in K$.

- *logit* response model: each user selects an allocation from the set R proportionally to its utility value. Formally, $p_k(\mathbf{x}) = \frac{u_k(\mathbf{x})}{\sum_{\mathbf{x}' \in R} u_k(\mathbf{x}')}, \forall k \in K$.

Given that with each of these response models, every user selects a solution without reasoning about other users' choices but, rather, makes her decisions by exclusively considering her utility over each allocation, in order for a task to occur, all the users, owner and non-owners, allocated to that task in a given allocation $\mathbf{x}$ have to select $\mathbf{x}$. For example, if allocation $\mathbf{x}$ assigns to task j the subset of non-owners $\tilde{I} \subseteq I$, then, in order for task j to occur, all non-owners $i \in \tilde{I}$ must select allocation $\mathbf{x}$.

3 Diverse Aware Approach

The problem described in the previous section can be approached as a resource allocation problem, in which users are implicitly assumed to be compliant with any solution proposed to them, and are therefore not afforded any alternatives. The results of such an approach are constrained to that of a matching between users and resources. Consequently, there is no consideration of the inherent uncertainty in user behaviour, or the fact that users could simply refuse to participate in systems that do not satisfy their needs. A system that realistically addresses human diversity and the uncertainty in human behaviour requires an explicit representation of user preferences and their responses to different decision scenarios. Furthermore, the decision scenario needs to be formulated so that it allows for the recommendation of solutions to users, while accounting for their possible deviations from expected behaviour.

In this section, we will provide a detailed framework for the representation of the diversity and uncertainty in user behaviour, and outline our approach that focuses on the problems of *recommending* alternatives and facilitating the coordination of users. In particular, our system intends to present a set of allocations $R = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_{|R|}\}$ of fixed size $|R|$.

To achieve this goal we need to deal with two issues. The first of these is the multi-criteria optimisation problem in which we have to balance the conflicting interests of each single user (represented by her utility function $u_i(\cdot)$) and the interest of the collective of users (represented by the system-level utility function $U_s(\cdot)$). We overcome this problem by computing solutions that guarantee a minimum level of system utility and maximise, e.g. the fairness of the solution with respect to the allocated resources. The second is that the uncoordinated selection of allocations done by the users, along with their diversity in preferences, makes it unlikely that users will select allocations such that the task can actually occur. In order to help the system in the process of coordinating users' selections, we introduce a *taxation* mechanism, so as to influence user selection behaviour by artificially modifying the utility they have for the recommended solutions, i.e. by modifying their preferences. Effectively, taxation allows the system to impose a penalty on allocations users are better of not selecting. Generally, the tax imposed is different for each user and for each allocation, and must guarantee that users still have multiple options (e.g. the system cannot impose an infinite tax). We develop a different taxation mechanism for each of the user response models described in Section 2.

Crucially, these two problems must be tackled simultaneously, otherwise properties that are satisfied when the first issue is solved, e.g. fairness, may not hold anymore if taxation is applied in a separate step. The reason behind this is that both the problems and the solution to these problems are related to the function, i.e. user utility.

Now, we describe our approach that aims to optimise the recommendation set R while simultaneously dealing with the problems due to user's and collective's conflicting interests, and the lack of coordination in user selection. In order to handle this problem, we iteratively construct the recommendation set by sequentially executing three Mixed Integer Linear Programs (MILPs), each of them guaranteeing different solution properties. In this way we can deal with both the multi-criteria optimisation and the computational complexity of finding an exact solution.

In particular, in order to account for this conflict of interest, we initially construct a program called $MILP^{system}$ that aims to maximise the system utility $U_s(\cdot)$ and thus find a solution with the highest utility V^* that the system can achieve. A second program called $MILP^{first}$ takes V^* as input and guarantees that the computed solution achieves at least a give percentage of V^* in terms of system utility, while the objective function of the program is focused on maximising a different property, e.g. fairness. The advantage of this approach is that the application controls exactly to which extent the maximisation of the system-level utility and the fairness are satisfied. The alternative would have been to use a single MILP whose objective function accounts for both the system utility and the fairness. However, in this case *(i)* the best trade-off between the two factors would have been decided by the program and not by the application designer, and *(ii)* it would be possible to obtain solutions completely unbalanced towards one of the two factors.

To face the lack of coordination among users, we aim to modify their utility for the recommended allocations such that they all prefer the same solution, termed *sponsored* solution. This solution is the one computed by $MILP^{first}$ and, since is the one we want to sponsor, we do not alter the utility the users have for it. Instead, we apply taxation on all other recommended solutions. Since we need to solve the problem of identifying the solutions with the desired properties and apply taxation simultaneously (as explained in the previous section), we design a third program, $MILP^{others}$, dedicated to this. In particular, a solution obtained with this program aims to be similar to the one of $MILP^{first}$ in terms of users' utility, guarantees a minimum level of system utility, and has taxes computed on the basis of the specific user response model considered.

Given this, we can view our framework as composed of three steps. In the first one, $MILP^{system}$ is executed in order to identify the highest possible system utility achievable, in the second step $MILP^{first}$ is used to identify the sponsored solution, and in the third step all the remaining non-sponsored $|R| - 1$ solutions are computed by executing $MILP^{other}$ $|R| - 1$ times.

Note that, users may have other requirements that the system should satisfy, for example, they may have constraints regarding the characteristics of users they are willing to share a task with. Thus, all the MILPs must satisfy these requirements in order to compute a feasible solution. In the next section, we provide a description of the constraints needed to satisfy the properties our framework necessitates and the different type of user requirements.

4 Optimisation Problem Formulation

In this section we present the details of the Mixed Integer Linear Programs (MILPs) that compose the framework described in the previous section. For the sake of clarity and without loss of generality, we present the MILPs for the ridesharing scenario.

Ridesharing is a sharing application that can be modelled as specified above. Indeed, a set of passengers I and a set of drivers J aim to find people to share a ride with. Each driver is the owner of a task, i.e., a car, and passengers aim to join one car each. We assume that drivers impose their pick-up point, drop-off point, and time of pick-up on passengers they are sharing the ride with. Passengers, in turn, have preferences over the pick-up point and drop-off point, and their utility decreases with the distance between their preferences and what the driver they are assigned to imposes on them. The pick-up time does not affect users' utility, however the system imposes a threshold on the maximum difference between the pick-up time desired by a passenger and the one specified by the driver she is assigned to. Note that this is a hard requirement and thus no allocation that violates this can be recommended. Moreover, both passengers and drivers may require to be in a car without smokers, Finally, they may also require to either share the ride, or not to be in the same car as another specific user.

The requirements just described for the ridesharing scenario are examples of three different types of constraints that users of sharing applications may have. Thus, even if in the following formulations we focus on ridesharing, the constraints presented can be used for a wide range of applications. In order to illustrate the expressiveness of the MILP formulations and the requirements that can be captured, in what follows we describe the characteristic of each possible type of constraint and show how to formulate it by using an example.

4.1 Maximising collective-level objectives

The MILP presented in this section is used in the first step of our framework and aims to compute the maximum system utility achievable without violating any requirements.

We start by defining the utility function $u_i(\mathbf{x})$ of a passenger $i \in I$. Her utility function is affected by how much the allocation $\mathbf{x}$ satisfies her preferences. In particular, here we assume that users have preferences over two aspects that characterise a task: the pick-up point and the drop-off point.

Without loss of generality, assume the utility function $u_i(\mathbf{x})$ of each passengers $i \in I$ is a sum of partial utility functions $\alpha_i(\mathbf{x})$ and $\beta_i(\mathbf{x})$ as shown in Equation 1. In particular, $\alpha_i(\mathbf{x})$ is the contribution to the utility of agent i that depends on the difference between the pick-up point of i and the one of the driver assigned to her by allocation $\mathbf{x}$. Similarly, $\beta_i(\mathbf{x})$ depends on the difference between their drop-off point. We assume that these differences are divided into intervals and that all differences in the same interval affect the user's utility in the same way.

$$u_i(\mathbf{x}) = \alpha_i(\mathbf{x}) + \beta_i(\mathbf{x}) \tag{1}$$

The utility function of the system is a linear weighted combination as shown by Equation 2. In this specific case, w_1, w_2, and w_3 are the weights. The first weight multiplies the sum of passengers utility, i.e., the social welfare, the second the number of passengers that are allocated to a car, and the third the number of drivers. The idea is that the system cares about the sum of the utility achieved by passengers but also the number of users that have the possibility to get a ride.

$$U_s(\mathbf{x}) = w_1 \sum_{i \in I} u_i(\mathbf{x}) + w_2 \sum_{i \in I} \sum_{j \in J} x_{i,j} + w_3 \sum_{j \in J} 1 \tag{2}$$

We are now ready to define the objective function of $MILP^{system}$ that is to maximise the system's utility (Equation 3).

$$obj \quad max_{\mathbf{x} \in \mathbf{X}} U_s(\mathbf{x}) \tag{3}$$

We start the description of the constraints by focusing on the allocation variables $x_{i,j} \in [0,1], \forall i \in I, \forall j \in J$. Since each car $j \in J$ has a capacity c_j, we need to guarantee that no more than c_j passengers are allocated to j (Constraint 4). Finally, we need to guarantee that each passenger is allocated to at most one car (Constraint 5).

$$\sum_{i \in I} x_{i,j} \leq c_j, \forall j \in J \tag{4}$$

$$\sum_{j \in J} x_{i,j} \leq 1, \forall i \in I \tag{5}$$

We introduce a second set of variables $h_{i,i',j}$, one for each passenger $i \in I$, passenger $i' \in I$, and driver $j \in J$. These are binary variables indicating if two passengers are sharing the same car. In particular, Constraints 6 guarantee that $h_{i,i',j} = 1$ if passengers i and i' are both allocated to car j, and $h_{i,i',j} = 0$ otherwise.

$$\begin{aligned} h_{i,i',j} \leq & x_{i,j}, \forall i, i' \in I, \forall j \in J \\ h_{i,i',j} \leq & x_{i',j}, \forall i, i' \in I, \forall j \in J \\ h_{i,i',j} \geq & |x_{i,j} + x_{i',j}| - 1, \forall i, i' \in I, \forall j \in J \end{aligned} \tag{6}$$

We now move to describe the constraints needed to guarantee that the partial utility $\alpha_i(\mathbf{x})$ correctly reflects the distance between the pick-up preference of passenger i and what the driver she is assigned to imposes on her. Note that similar constraints are used to compute the partial utility $\beta_i(\mathbf{x})$. As mentioned before, we consider the possible pick-up distance as divided into $|T|$ intervals (where T is the set of intervals). For each interval $n \in T$, the parameter $\alpha_{i,n}$ indicates i's partial utility if the pick-up distance is in interval n, and parameters $\alpha_{n,lower}$ and $\alpha_{n,upper}$ denote the lower bound and the upper bound of interval n, respectively. A set of variables $k^{\alpha}_{i,n}$, one for each $n \in T$, is used to select the right interval. Note that $k^{\alpha}_{i,n}$ is a binary variable and that $k^{\alpha}_{i,n} = 0$ if the pick-up distance is in interval n and $k^{\alpha}_{i,n} = 1$ otherwise. Given this, the partial utility $\alpha_i(\mathbf{x})$ is given by Equation 7, and Constraints 8 guarantee the the only variable $k^{\alpha}_{i,n}$ that equals zero is the one of the interval n, to which the pick-up distance belongs to. Note that M is a very large number as typically used in the Big M method [22], while $\Delta^{\alpha}_i(\mathbf{x})$ measures the pick-up distance. In this particular case, we compute the distance by considering latitude ($p_{i,lat,pu}$ and $p_{j,lat,pu}$) and longitude ($p_{i,long,pu}$ and $p_{j,long,pu}$) of the pick-up points, and compute the Manhattan distance between them as shown in Equation 9. This equation is particularly interesting because it shows how we deal with imposing a zero partial utility to a passenger when she is not assigned to any car. In particular, in order to achieve this, a fictitious pick-up distance interval $n = |T|$ is introduced in the set T such that the large number M (bigger than any pick-up distance) is in interval $n = |T|$, $alpha_{i,|T|} = 0$, and $\Delta^{\alpha}_i(\mathbf{x}) = M$ if i is not allocated to any $j \in J$.

$$\alpha_i(\mathbf{x}) = \sum_{n \in T} \alpha_{i,n} \cdot (1 - k^{\alpha}_{i,n}), \forall i \in I \tag{7}$$

$$\begin{aligned} M \cdot k^{\alpha}_{i,n} + (\alpha_{n,upper} - \Delta^{\alpha}_i(\mathbf{x})) \geq & 0, \forall i \in I, \forall n \in T \\ M \cdot k^{\alpha}_{i,n} + (\Delta_i(\mathbf{x}) - \alpha_{n,lower}) \geq & 0, \forall i \in I, \forall n \in T \\ \sum_{n \in T} k^{\alpha}_{i,n} \leq |T| - 1, & \forall i \in I \end{aligned} \tag{8}$$

$$\Delta_i^{\alpha}(\mathbf{x}) = \sum_{j \in J} x_{i,j}(|p_{i,lat,pu} - p_{j,lat,pu}| + |p_{i,long,pu} - p_{j,long,pu}|) + (1 - \sum_{j \in J} x_{i,j}) \cdot M, \forall i \in I \quad (9)$$

In defining constraints due to requirements, we differentiate between *strict* constraints, *non-strict* constraints, and *potential* constraints.

Strict constraints impose that every group of users sharing a car must have the same value for a given user's parameter. For example, in the ridesharing scenario, a strict constraint is imposed on the day of the ride and, thus, all users allocated to the same car must have the same value for the parameter p_i^{day}. In particular, if the two users are passengers, then the strict constraint is Constraint 10, while if the two users are a passenger and a driver, then Constraint 11 must be imposed.

$$|h_{i,i',j} p_i^{day} - h_{i,i',j} p_{i'}^{day}| \leq 0, \forall i, i' \in I, \forall j \in J \quad (10)$$

$$|x_{i,j} p_i^{day} - x_{i,j} p_j^{day}| \leq 0, \forall i, \in I, \forall j \in J \quad (11)$$

Non-strict constraints impose a threshold on how a passenger's requirement is satisfied. In the ridesharing scenario, this type of constraint is applied to the difference between the time of pick-up specified by the passenger, and the one of the driver she is sharing the car with. We formulate this type of constraint as shown by Constraint 12, where p_i^{time} and p_j^{time} are parameters that indicate the pick-up time of passenger i and driver j, respectively, and $p_{threshold}^{time}$ is the threshold.

$$\sum_{j \in J} x_{i,j}(|p_i^{time} - p_j^{time}|) \leq p_{threshold}^{time} \quad (12)$$

Finally, we discuss potential constraints. Constraints of this type do not always impose a condition that must be satisfied by a solution. Indeed, a user may have specific requirements about a characteristic of the users she is sharing the task with or she may be indifferent with respect to this characteristic. For example, in the ridesharing scenario, a user may require to be in a car without smokers, while another user, even if she is not a smoker, may not have such a requirement. In order to formulate the constraint that guarantees these requirements, we need to introduce a new binary variable v_j^{smoker}, one for each car $j \in J$. Constraints 13 impose that $v_j^{smoker} = 1$, if at least one user among the passengers and the driver sharing car j requires to be in a car without smokers. Parameter $p_i^{reqNoSmoke}$ indicates the "no smoker request" of a passenger $i \in I$. In particular if $p_i^{reqNoSmoke} = 1$ the passenger requires to be in a car with no smokers, while if $p_i^{reqNoSmoke} = 0$ the passenger has no preferences. $p_j^{reqNoSmoke}$ is similarly defined for driver $j \in J$. Constraint 14 guarantees that if none of the users in a car j has a "no smokers" requirement then $v_j^{smoker} = 0$. Now, if variable $v_j^{smoker} = 1$, then we need to impose that all the users in car j do not want to smoke during the ride. This is achieved by Constraint 15, where parameter $p_i^{NoSmoke} = 0$ if passenger $i \in I$ wants to smoke in the car and $p_i^{NoSmoke} = 1$ otherwise. $p_j^{NoSmoke}$ is similarly defined for driver $j \in J$.

$$\begin{aligned} v_j^{smoker} &\geq x_{i,j} p_i^{reqNoSmoke}, \forall j \in J, \forall i \in I \\ v_j^{smoker} &\geq p_j^{reqNoSmoke}, \forall j \in J \end{aligned} \quad (13)$$

$$p_j^{reqNoSmoke} + \sum_{i \in I} x_{i,j} p_i^{reqNoSmoke} \geq v_j^{smoker}, \forall j \in J \quad (14)$$

$$p_j^{NoSmoke} + \sum_{i \in I} x_{i,j} p_i^{NoSmoke} \geq (c_j + 1) v_j^{smoker}, \forall j \in J \quad (15)$$

To conclude, we consider the case in which the system allows a user to specify if she wants/does not want to share a task with another specific user. The constraints used to guarantee these requirements are strict constraints and are formalised as follows. Constraints 16 and 17 guarantee that passengers i and i' and passenger i and driver j are allocated to the same car, respectively. While Constraints 18 and 19 guarantee the opposite.

$$\sum_{j \in J} h_{i,i',j} \leq 0 \quad (16)$$

$$x_{i,j} \leq 0 \quad (17)$$

$$\sum_{j \in J} h_{i,i'',j} \geq 1 \quad (18)$$

$$x_{i,j} \geq 1 \quad (19)$$

4.2 Maximising fairness among users

$MILP^{first}$ constitutes the second step of our framework. The aim for this program is to identify the first solution that will be presented to users. Note that this is the solution we would like all users to choose among the ones in the recommendation set. The solution the program provides guarantees a minimum level of system utility while focusing on an objective function that is oriented to being beneficial to the users. In the following formulation we assume, without loss of generality, that the program aims to maximise user fairness. This translates into an objective function that minimises the difference between the utility of every pair of passengers $i, i' \in I$ (Equation 20).

$$obj \quad min_{\mathbf{x} \in \mathbf{X}} \sum_{i \in I} \sum_{i' \in I | i' > i} |u_i(\mathbf{x}) - u_{i'}(\mathbf{x})| \quad (20)$$

As mentioned before, the solution provided by this program must guarantee a minimum level of system utility. Given the maximum utility V^* the system can achieve (computed by $MILP^{system}$) and the parameter $d \in [0, 1]$, the required guarantee can be obtained by imposing Constraint 21.

$$U_s(\mathbf{x}) \geq V^* \cdot d \quad (21)$$

In addition to this, all the constraints described for $MILP^{system}$ must also hold for $MILP^{first}$.

4.3 Maximising user coordination

The last step of our framework aims to compute the remaining $|R| - 1$ solutions (one has already been identified by $MILP^{first}$). To achieve this, $MILP^{others}$ is executed $|R| - 1$ times.

A solution identified by $MILP^{others}$ has the following three characteristics: it guarantees a minimum level of system utility (as $MILP^{first}$ does), computes a solution that is different from the ones previously chosen, and artificially modifies the utility each passenger $i \in \bar{I}$ has for this solution such that all of them would prefer the solution $\mathbf{x}^*$ identified by $MILP^{first}$. Note, that the set $\bar{I} \subseteq I$ is composed by all passengers $i \in I$ such that $\sum_{j \in J} x_{i,j}^* = 1$, i.e., only the utility of passengers who are assigned to a driver in solution $\mathbf{x}^*$ is

artificially modified. This last characteristic depends on the response model of the users, and is achieved by using taxation.

When considering the noiseless response model and constant noise model, in order for a passenger $i \in I$ to select/prefer the solution computed by $MILP^{first}$, we need to guarantee that her utility for the solution $\mathbf{x}$ currently computed is lower than her utility for allocation $\mathbf{x}^*$, i.e. the solution computed by $MILP^{first}$. We achieve this by imposing a tax $\tau_i(\mathbf{x})$ on passenger i for allocation $\mathbf{x}$ that decreases the utility i has for $\mathbf{x}$. Constraint 23 guarantees that $\mathbf{x}^*$ is the preferred solution of passenger i. In this constraint, ϵ is a very small number used to express the fact that $u_i(\mathbf{x}^*)$ must be strictly higher than $u_i(\mathbf{x}) - \tau_i(\mathbf{x})$. However, this constraint imposes a lower bound to the tax $\tau_i(\mathbf{x})$ but no upper bound. Thus, potentially, the program can assign an infinite value to $\tau_i(\mathbf{x})$. This is not desirable because no real options would be effectively given to the passengers if all but one could be infinitely taxed. Moreover, we also want to avoid the case in which $\tau_i(\mathbf{x})$ is higher than the minimum tax required as the system should not aim to unnecessarily extract excessive utility from its participants. Thus, the upper bound $\tau_i(\mathbf{x})$ should be equal to its lower bound. We obtain this by using the Big M method [22] that involves changing the objective function as shown by Equation 22.

$$min \sum_{i \in I} |u_i(\mathbf{x}^*) - u_i(\mathbf{x}) + \tau_i(\mathbf{x})| + M\Big(\sum_{i \in I} \big(u_i(\mathbf{x}^*) - \epsilon - u_i(\mathbf{x}) + \tau_i(\mathbf{x})\big)\Big) \quad (22)$$

$$u_i(\mathbf{x}^*) - \epsilon \geq u_i(\mathbf{x}) - \tau_i(\mathbf{x}) \quad (23)$$

Similarly to the constant response model case, $MILP^{others}$ needs to impose a lower bound on the tax also for the logit response model (Constraint 25), and modify the objective function (Equation 24) such that the tax is the lowest possible. However, since in this case the probability with which a passenger selects solution $\mathbf{x}$ is proportional to the utility that $\mathbf{x}$ represents for that passenger, imposing a lower bound to the tax means imposing a lower bound to the selection probability of $\mathbf{x}^*$ for each passenger $i \in \bar{I}$. In Constraint 25, ψ is the minimum selection probability required for $\mathbf{x}^*$. Note that this constraint is linear because everything but $u_i(\mathbf{x})$ and $\tau_i(\mathbf{x})$ are parameters given as input to the program.

$$min \sum_{i \in I} |u_i(\mathbf{x}^*) - u_i(\mathbf{x}) + \tau_i(\mathbf{x})| + M\Big(\sum_{i \in I} \big(u_i(\mathbf{x}^*) - \psi \cdot \Big(\sum_{\mathbf{x}' \in R} \big(u_i(\mathbf{x}') - \tau_i(\mathbf{x}')\big) + u_i(\mathbf{x}) - \tau_i(\mathbf{x})\Big)\Big)\Big) \quad (24)$$

$$\frac{u_i(\mathbf{x}^*)}{\sum_{\mathbf{x}' \in R} \big(u_i(\mathbf{x}') - \tau_i(\mathbf{x}')\big) + u_i(\mathbf{x}) - \tau_i(\mathbf{x})\}} \geq \psi \quad (25)$$

Finally, $MILP^{others}$ needs to guarantee that solution $\mathbf{x}$ is different from the ones previously computed. Depending on the required degree of difference between two solutions, we can formulate the $MILP$ constraints as follows. Constraint 26 guarantees that the solution $\mathbf{x}$ differs from the ones already computed, i.e. the ones in set R, at least for the allocation of one passenger. While Constraint 27 requires that each ride (except the one with no passenger allocated to it) of solution $\mathbf{x}$ differs from the corresponding ride in solution $\mathbf{x}' \in R$, at least for the allocation of one passenger.

$$\sum_{i \in I} \sum_{j \in J} |x_{i,j} - x'_{i,j}| > 1, \forall \mathbf{x}' \in R \quad (26)$$

$$\sum_{i \in I} |x_{i,j} - x'_{i,j}| > 1, \forall \mathbf{x}' \in R, \forall j \in J \quad (27)$$

In addition to these, all the constraints described for $MILP^{system}$ and $MILP^{first}$ must also hold for $MILP^{others}$.

5 Experimental Evaluation

In order to demonstrate the effectiveness of our approach, we run two sets of simulated experiments. In the first set, we compare the recommended set of solutions generated with our diversity-aware approach, with a set of solutions that maximise the system's utility and provide no support for user coordination. Essentially, the benchmark set is produced without considering the need for consistency across users' selections. In this way, we will demonstrate that our diversity-aware approach is *strictly better* for the generation of *recommendation sets*.

In the second set of experiments, we compare our approach to that of allocating a single solution that maximises system utility. We assume users are characterised by a utility threshold of acceptance, unknown to the system. This latter set of experiments, will show how recommending a set of solutions through our approach can produce *results that are equally good* to what a *direct allocation* would have produced.

5.1 Experiment design

For both types of experiments we consider different configurations, each of which is characterised by the population of users, specifically the number of users and the percentage of drivers among them, the value of the threshold d used in $MILP^{first}$ and $MILP^{others}$, the user response model, and, for the logit model, the probability ϵ with which the sponsored allocation $\mathbf{x}^*$ is selected. In particular, we run experiments for 10 and 20 users, for each percentage of drivers among 20, 30, and 40 percent. We vary the utility threshold d for $MILP^{first}$ and $MILP^{others}$ between the values 0.5, 0.75, and 1. The user models evaluated are the constant noise model and the logit model, and, for the latter, the probability ϵ of users selecting the sponsored recommendation is either 60 or 80 percent. Every configuration is repeated 100 times. We choose not to evaluate the noiseless response model because, by construction, it is a special case of the the constant noise model with the best performance, i.e. with the probability of the most preferred option set to 1. Without loss of generality, we set the weights $w_1 = w_2 = w_3 = 1$ in the system-level utility function.

The metrics used for each experiment are *system utility*, *fairness* (computed as described in the previous section), *number of drivers with allocated passengers*, and *number of allocated passengers*. Note, that all evaluations are performed *after* user selections have been performed. Thus, rides that have not been chosen by all the users allocated to them are not considered in the performance evaluation. Finally, we highlight that the metrics proposed account for the taxation imposed on the solutions. That is, each user, given his final allocation, has had any taxation imposed on him deducted from his effective utility, which in turn affects the evaluation of the system-level utility (Eq. 2) and fairness of allocation (Eq. 20).

The procedure used to obtain the experimental results is the following: First we generate the desired number of users, divided into drivers and passengers as prescribed by the configuration. For each user, we randomly generate the latitude and longitude of the pick-up point and drop-off point (we restrict the variability of this coordinate to 50), the pick-up time, whether she allows smokers in the car, and whether she

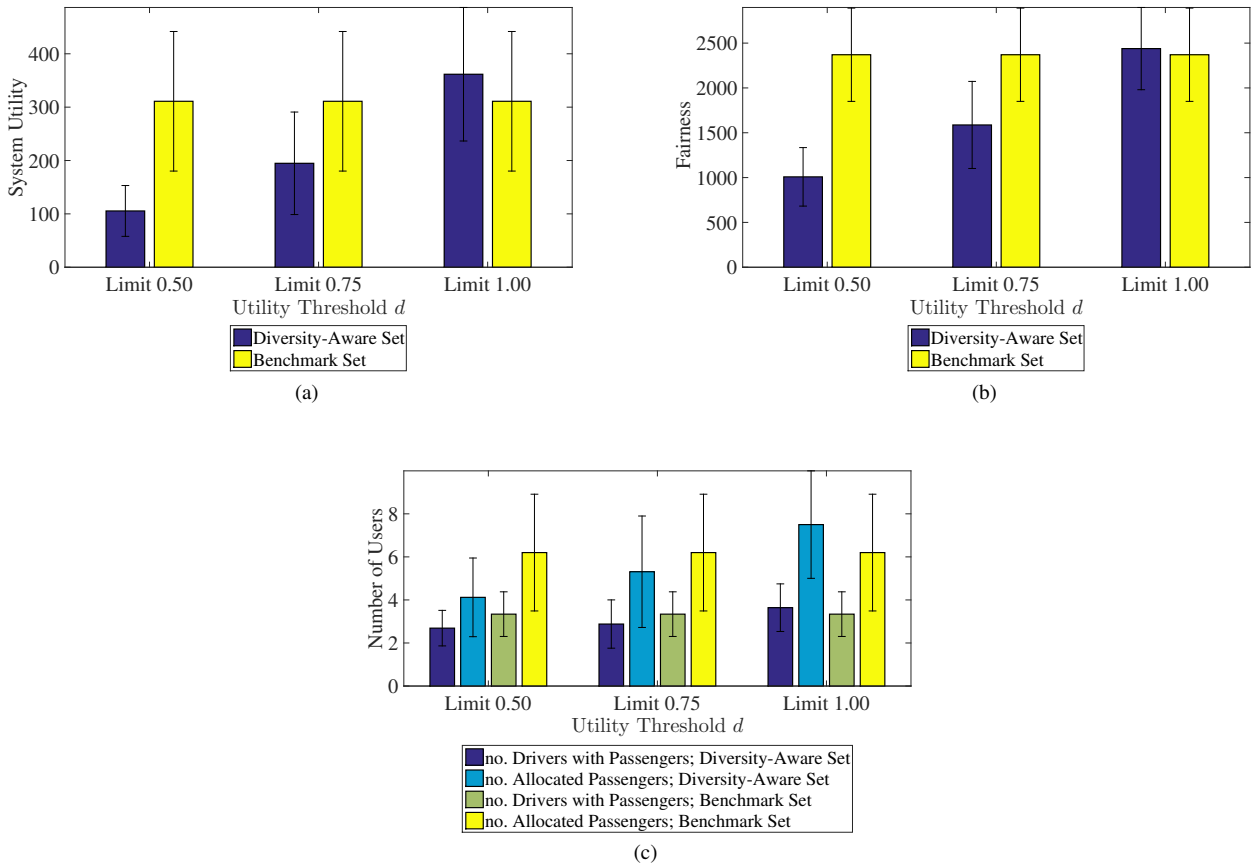

Figure 1: Results on *System Utility* (a), *Fairness* (b), and the number of *Allocated Passengers* and *Drivers with Passengers* (c) for the experiments with a recommendation set benchmark, with 20 users, 30% drivers, and constant noise.

wants to smoke. Then we generate $|R| = 7$ solutions (where possible) both for the case without coordination support in which the goal of the application is solely to maximise the system's utility, and following our approach. Finally, we simulate user behaviour according to the respective user response model and compute the metrics listed above.

This final step of our evaluation changes slightly depending on the benchmark we are comparing our approach to. As mentioned before, we consider two benchmarks: a benchmark with a recommendation set and a benchmark where a single allocation is proposed to the users. Note that in both cases, the solutions computed aim to maximise system-level utility.

When we consider the benchmark with a recommendation set, we compute the two sets of $|R|$ solutions computed as described above. Then, both for the benchmark case and for our approach, we recommend to each passenger the rides she is allocated to by these solutions. Each passenger then, independently, and without knowledge of other passengers' behaviour, selects a solution in accordance with her utility over each option, and her response model. Given these independent choices, we identify which rides have been selected by all the users allocated to them and, on the basis of this, we evaluate the performance of both approaches.

In the comparison with the benchmark with a single allocation, we assume that passengers select rides that satisfy a minimum level of user utility, i.e. there is a threshold over the user utility and solutions that do not satisfy this threshold cannot be selected. For example, in the case of ridesharing, we can assume that if the utility of a passenger for a ride is lower than her utility for taking the train, then she chooses not to join that ride. Given this, we consider the set $|R|$ of solutions computed using our diversity-aware approach and, for the benchmark, the solution computed by $MILP^{system}$. We assume that users apply this utility threshold and thus, all the rides that do not satisfy the threshold are removed from the ones that can be selected. This can be understood as users implicitly *rejecting* these solutions. We therefore make use of this label in corresponding figures. After this, each user selects one of the remaining options (note that in the benchmark case each user has either one ride or no option available). Now, as in the case of the previous benchmark, given these independent choices, we identify which rides have been selected by all users allocated to them and, on the basis of this, we evaluate the performance of both approaches.

A key point to remember, is that our diversity-aware approach influences users' utility over allocations through the use of taxation, and that this directly impacts on selection behaviour. We expect that this will aid in aligning user selections, and therefore lead to greater performance in terms of allocated passengers and drivers with passengers and, consequently, in terms of system-level utility.

5.2 Results

Below we present and discuss the results of our experiments for the case of 20 users with 30% drivers as representative of all experiments. The results on other population sizes and driver percentages were qualitatively equivalent. We analyse each set of experiments (recommendation set and single allocation) separately and present the

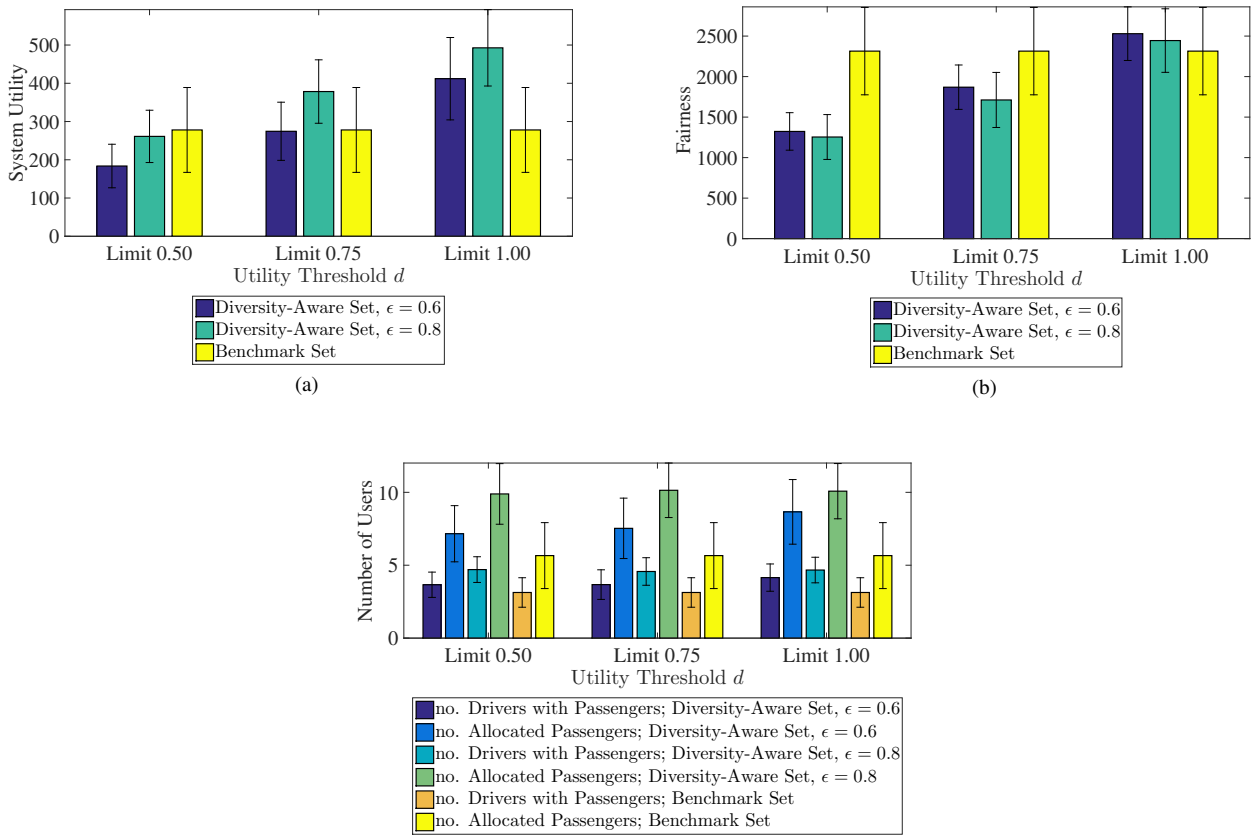

Figure 2: Results on *System Utility* (a), *Fairness* (b), and the number of *Allocated Passengers* and *Drivers with Passengers* (c) for the experiments with a recommendation set benchmark, with 20 users, 30% drivers, and logit noise.

constant and logit noise cases for each of them.

5.2.1 *Experiments with set recommendation benchmark*

This subsection discusses the experimental results from the set of experiments where the benchmark also presents the passengers with a set recommendation. Focusing first on the *constant noise* case (Fig. 1) we notice that there is a near linear trade-off between fairness and system utility (Fig. 1a, 1b). A trade-off that can be modulated by change of the utility threshold d. We also note that when $d = 1$, which forces MIP^{first} and MIP^{others} optimisations to prioritise system utility maximising solutions, the diversity-aware approach slightly outperforms the benchmark set, in terms of system utility, number of allocated passengers, and number of drivers with passengers (Fig. 1a, 1c). Reducing the value of parameter d offers better results on fairness (Fig. 1b), in comparison to the benchmark, but at a cost of reduced system utility (Fig. 1a).

Moving on to the *logit noise* case (Fig. 2), we notice once more the role of the utility threshold parameter d in trading off system utility and fairness (Fig. 2a, 2b). Further, the only scenario in which we perform slightly worse than the benchmark, in terms of user allocation and system utility, is for $\epsilon = 0.6$ and $d = 0.5$, i.e. when we optimise mostly for fairness and where we try not to influence user decisions too much. In terms of fairness, we only under-perform for $d = 1$, a result emerging from the large number of users with 0 utility, as results from the benchmark (Fig. 2a, 2c). Otherwise, we notice that the diversity-aware procedure significantly outperforms the set recommendation benchmark, in terms of system utility, number of allocated passengers, and number of drivers with passengers (Fig. 2a, 2c).

Summarising the results of this subsection, we note the significant improvement in performance afforded by our diversity-aware system. This signifies how important it is to be aware of the diversity among users when coordinating in sharing economy applications. These results increase in significance once we consider that there is no coordination present between users, and all the improvement is the result of implicit system interactions with each individual user; users that are free to choose amongst recommended alternatives. We conclude that making set recommendations by simply listing a set of system-optimal alternatives is a significantly sub-optimal procedure.

5.2.2 *Experiments with single allocation benchmark, and rejection*

This subsection discusses the experimental results from the set of experiments involving a diversity-aware set recommendation and a benchmark single allocation, while considering passengers that can reject rides. Focusing first on the *constant noise* case (Fig. 3), we notice that our system under-performs in terms of system utility (Fig. 3a), a loss it gains in its increased performance in terms of fairness (Fig. 3b). We further notice, as above, that there is a near linear trade-off between fairness and system utility, which can be modulated by change of the utility threshold d (Fig. 3a, 3b).

Finally, in the *logit noise* case for the second set of experiments (Fig. 4), we notice once more the role of the utility threshold parameter d

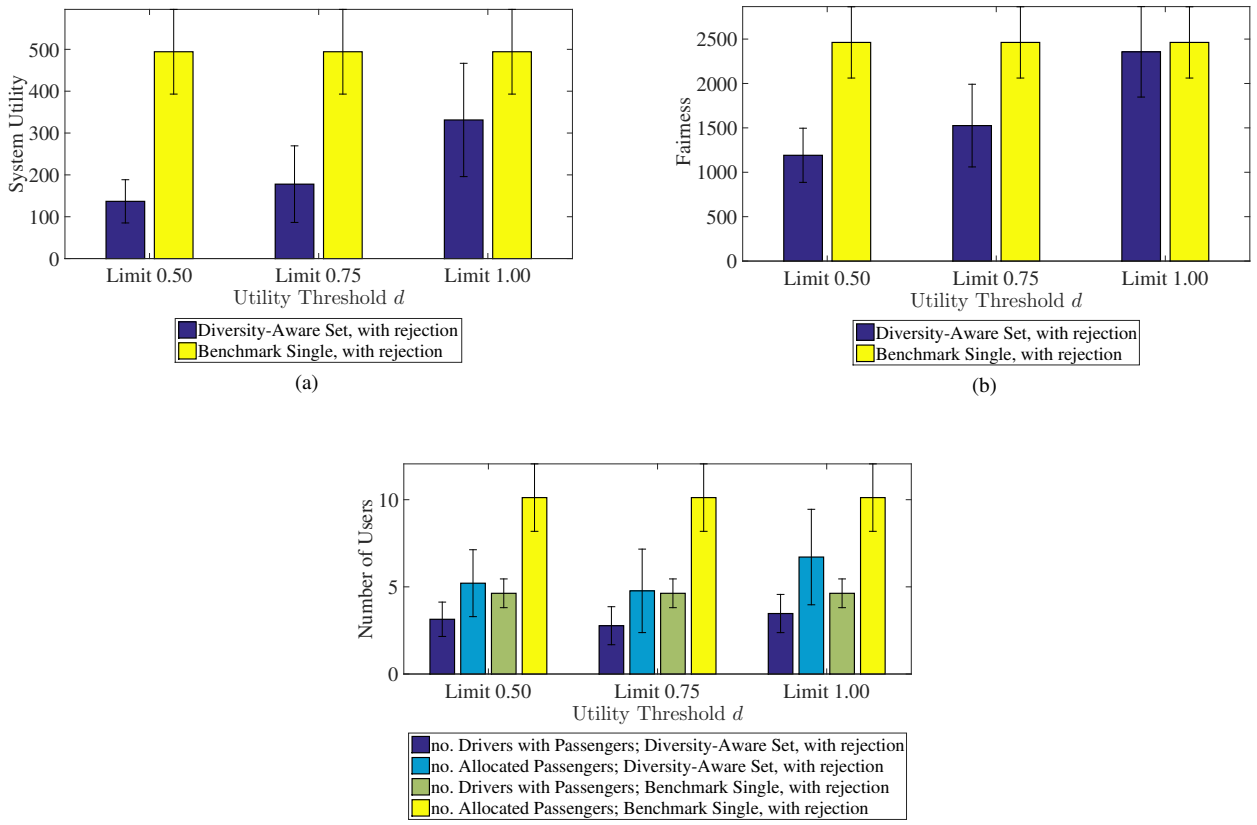

Figure 3: Results on *System Utility* (a), *Fairness* (b), and the number of *Allocated Passengers* and *Drivers with Passengers* (c) for the experiments with a single allocation benchmark, and rejection, with 20 users, 30% drivers, and constant noise.

in trading off system utility and fairness (Fig. 4a, 4b). Importantly, we note that our diversity-aware approach, which recommends a set, allowing for users to freely choose their preferred option, matches the performance of the benchmark, which allocates a single solution to each user (Fig. 4a, 4c). These results hold for when we do not wish to emphasise fairness, i.e in the $d = 1$ scenario.

The results of this subsection show that our diversity-aware set recommendation system, can consistently provide results that are equivalent to those of allocating a single item to each user. This shows, that providing users with options can be essentially *free*, in terms of system utility, even without considering any other beneficial effects that could result from allowing a system to recommend rather than allocate solutions to its user base. Moreover, we are afforded additional options in trading-off system utility with fairness.

6 Conclusion

We presented a methodology for the coordination of user collectives, in the absence of communication among agents. Our diversity-aware approach significantly outperforms the system utility maximising procedure, demonstrating that the recommendation of sets of solutions in sharing applications requires explicitly handling the uncertainty over user behaviour. Furthermore, we showed how our procedure can match the performance of a direct allocation of users to resources. This significant result demonstrates that we can allow users to *have a choice* in their alternatives, at no loss to the system. Lastly, our procedure allows for the adaptive trade-off between system-level utility and fairness of final allocation.

Future work will examine handling beliefs over user preferences in the context of recommending a set of options for sharing economies. Specifically, we are studying the inclusion of *active learning* procedures in the mixed integer linear program formulations. Further, we are interested in studying the robustness of our procedures to varying degrees of incorrect assumptions.

REFERENCES

[1] Haris Aziz, Markus Brill, Felix A. Fischer, Paul Harrenstein, Jérôme Lang, and Hans Georg Seedig, 'Possible and necessary winners of partial tournaments', *J. Artif. Intell. Res. (JAIR)*, **54**, 493–534, (2015).

[2] Yoram Bachrach, Sofia Ceppi, Ian A. Kash, Peter Key, Filip Radlinski, Ely Porat, Michael Armstrong, and Vijay Sharma, 'Building a personalized tourist attraction recommender system using crowdsourcing', in *Proceedings of the 2014 International Conference on Autonomous Agents and Multi-agent Systems*, pp. 1631–1632, (2014).

[3] Yoram Bachrach, Edith Elkind, Reshef Meir, Dmitrii Pasechnik, Michael Zuckerman, Jörg Rothe, and Jeffrey S. Rosenschein, *The Cost of Stability in Coalitional Games*, 122–134, Springer Berlin Heidelberg, 2009.

[4] C. Boutilier, 'A POMDP formulation of preference elicitation problems', in *Proceedings of the 18th National Conference on Artificial Intelligence*, pp. 239–246, (2002).

[5] D. Braziunas, 'Computational approaches to preference elicitation', Technical report, University of Toronto, (2006).

[6] Sofia Ceppi and Ian Kash, 'Personalized payments for storage-as-a-service', *SIGMETRICS Perform. Eval. Rev.*, **43**(3), 83–86, (2015).

[7] Urszula Chajewska, Daphne Koller, and Ronald Parr, 'Making rational decisions using adaptive utility elicitation', in *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, pp. 363–369. AAAI Press, (2000).

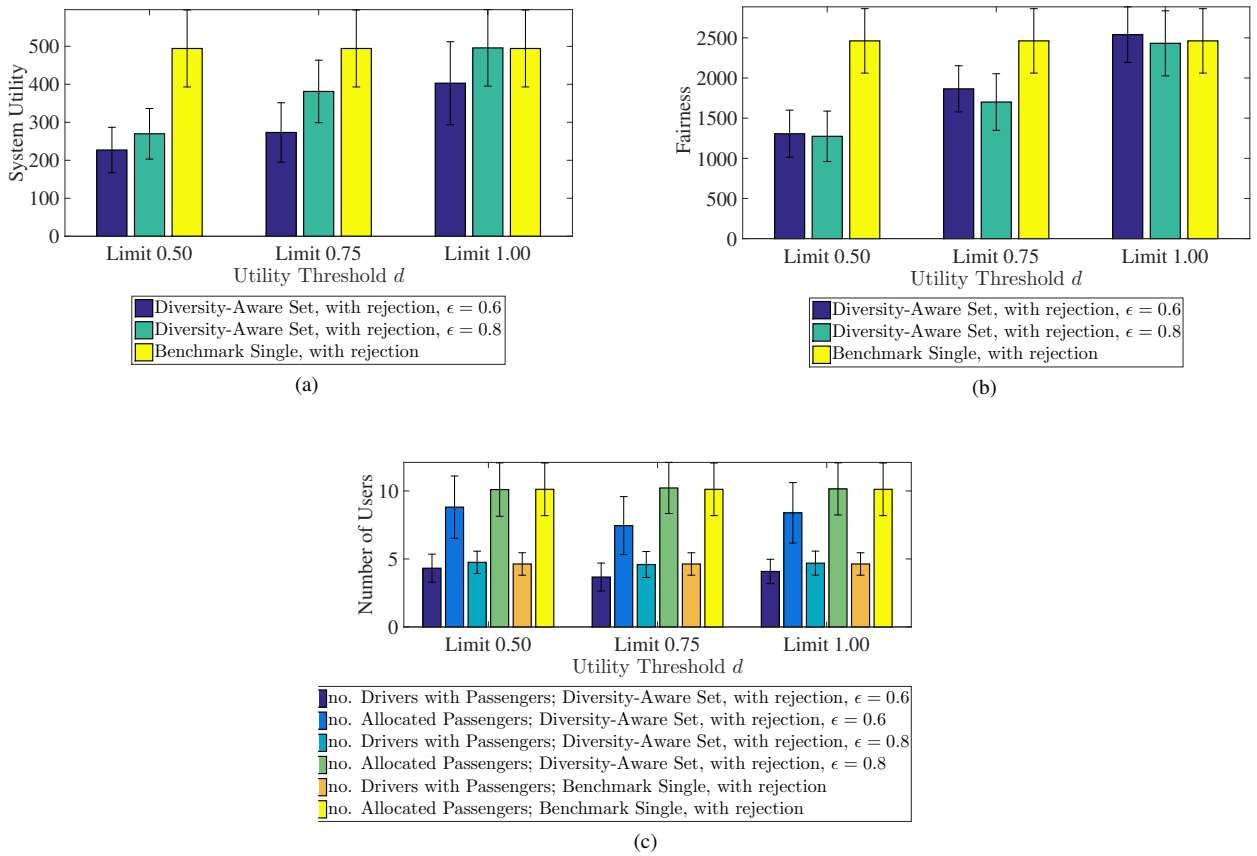

Figure 4: Results on *System Utility* (a), *Fairness* (b), and the number of *Allocated Passengers* and *Drivers with Passengers* (c) for the experiments with single allocation benchmark, and rejection, with 20 users, 30% drivers, and logit noise.

[8] Kim-Sau Chung, 'On the existence of stable roommate matchings', *Games and Economic Behavior*, **33**(2), 206 – 230, (2000).

[9] John P. Dickerson, Ariel D. Procaccia, and Tuomas Sandholm, 'Optimizing kidney exchange with transplant chains: theory and reality', in *International Conference on Autonomous Agents and Multiagent Systems, AAMAS*, pp. 711–718, (2012).

[10] Krzysztof Gajos and Daniel S. Weld, 'Preference elicitation for interface optimization', in *Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology*, UIST '05, pp. 173–182, New York, NY, USA, (2005). ACM.

[11] D. Gale and L. S. Shapley, 'College admissions and the stability of marriage', *The American Mathematical Monthly*, **69**(1), 9–15, (1962).

[12] Christophe Gonzales and Patrice Perny, 'GAI networks for utility elicitation.', *KR*, **4**, 224–234, (2004).

[13] Shengbo Guo and Scott Sanner, 'Real-time multiattribute bayesian preference elicitation with pairwise comparison queries', in *International Conference on Artificial Intelligence and Statistics*, pp. 289–296, (2010).

[14] Dan Gusfield and Robert W. Irving, *The Stable Marriage Problem: Structure and Algorithms*, MIT Press, Cambridge, MA, USA, 1989.

[15] Neil Houlsby, Ferenc Huszar, Zoubin Ghahramani, and Jose M Hernández-lobato, 'Collaborative Gaussian processes for preference learning', in *Advances in Neural Information Processing Systems*, pp. 2096–2104, (2012).

[16] Daniel Kahneman and Amos Tversky, 'Choices, values, and frames.', *American psychologist*, **39**(4), 341, (1984).

[17] R. L. Keeney and H. Raiffa, *Decisions with Multiple Objectives: Preferences and Value Trade-Offs*, Cambridge University Press, 1993.

[18] Michel Balinski Mourad Baou, 'The stable allocation (or ordinal transportation) problem', *Mathematics of Operations Research*, (3), 485–503, (2002).

[19] Filip Radlinski and Susan Dumais, 'Improving personalized web search using result diversification', in *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 691–692, (2006).

[20] Onn Shehory and Sarit Kraus, 'Methods for task allocation via agent coalition formation', *Artificial Intelligence*, **101**(1), 165 – 200, (1998).

[21] Paolo Viappiani and Craig Boutilier, 'Optimal bayesian recommendation sets and myopically optimal choice query sets', in *Advances in Neural Information Processing Systems*, pp. 2352–2360, (2010).

[22] Wayne L. Winston, *Introduction to Mathematical Programming: Applications and Algorithms*, Duxbury Resource Center, 2003.

[23] Yair Zick, Maria Polukarov, and Nicholas R. Jennings, 'Taxation and stability in cooperative games', in *Proceedings of the 2013 International Conference on Autonomous Agents and Multi-agent Systems*, pp. 523–530, (2013).