Laboratoire d'Analyse et Modélisation de Systèmes pour l'Aide à la Décision
UMR 72

CAHIER DU LAMSADE

372

Février 2016

Constrained Object Allocation Problems

Laurent Gourvès, Carlos A.Martinhon, Jérôme Monnot

www.cnrs.fr

Constrained Object Allocation Problems

Laurent Gourvès[*], Carlos A. Martinhon[†], Jérôme Monnot[‡]

Abstract

The *Object Allocation Problem* (OAP) is a well studied problem which is about to allocate a set $\mathcal{X}$ of n objects to a set N of n agents. This paper deals with a generalization called *Constrained Object Allocation Problem* (COAP) where the set of objects allocated to the agents must satisfy a given feasibility constraint. The input is a set $\mathcal{X}$ of at least n elements and a collection $\mathcal{S}$ of subsets of $\mathcal{X}$, each of size n. Every $S \in \mathcal{S}$ defines a set of elements that the agents can collectively possess and such that every agent is allocated exactly one element.

In this article we first study the problem of a central authority who wants to maximize the social welfare defined in two ways: the sum of the agents' utility for the item they receive (utilitarian) or the utility of the poorest agent (egalitarian). This optimization problem is shown **NP**-hard for COAP in general but polynomial time solvable when $\mathcal{S}$ is the base set of a matroid (for the utilitarian social welfare and the egalitarian social welfare).

An allocation can be built by the agents without communicating their utilities to a central authority. They can use a mechanism like the famous *Serial Dictatorship* mechanism (SD). In SD, a permutation of the agents is given and, starting from scratch, the agents select in turn their most preferred element among the remaining items. We analyse the solutions produced by a version of SD adapted to COAP. There are instances of COAP such that SD fails to produce a socially optimal allocation, whatever the order on the agents. However, if $\mathcal{S}$ is the base set of a matroid, then we prove that SD produces a social optimum for at least one permutation (for the utilitarian social welfare and the egalitarian social welfare).

Then, we give tight worst case bounds on the ratio between the social utility under SD and the optimal social utility. These bounds are valid for both OAP and COAP.

We conclude with a proof showing that manipulating SD for inducing a socially good allocation in OAP is **NP**-hard even with 3-approval scores. Here, we retain two ways to indicate that an allocation is socially good: the sum of the agents' utilities is maximum and the minimum utility of an agent is maximum.

We end the paper by showing some special cases where manipulating is polynomial. In particular, we obtain a dichotomic complexity result of the manipulation problem for the egalitarian social welfare.

Keywords: object allocation, matroids, Egalitarian, Utilitarian, manipulation.

1 Introduction

The *Object Allocation Problem* (OAP) is about to match a set $\mathcal{X}$ of n objects (e.g. houses or jobs) to a set N of n agents. Every agent receives exactly one object so there are $n!$ possible *allocations*.

[*]CNRS UMR 7243 and Université Paris Dauphine, France (laurent.gourves@dauphine.fr)
[†]Fluminense Federal University, Department of Computer Science, Niterói, RJ, Brazil (mart@dcc.ic.uff.br)
[‡]CNRS UMR 7243 and Université Paris Dauphine, France (jerome.monnot@dauphine.fr)

This article deals with a generalization called *Constrained Object Allocation Problem* (COAP): $\mathcal{X}$ may contain more than n elements but each agent is allocated exactly one object and the sets of objects that the agents can collectively possess is prescribed. In concrete terms, we are given a collection $\mathcal{S}$ of subsets of $\mathcal{X}$, each of these subsets has cardinality n, and an allocation A is feasible if and only if $\{A(i) : i \in N\} \in \mathcal{S}$ where $A(i)$ is the object assigned to agent $i \in N$. Thus, $\mathcal{S}$ defines which sets of $2^{\mathcal{X}}$ the agents can collectively possess. But for any given $S \in \mathcal{S}$, there is no restriction on how S is distributed to the agents. We aim at studying some important features of OAP and see if they extend to COAP.

In both OAP and COAP the individual utility of an agent for a given allocation solely depends on the object he receives. The social welfare is mainly defined in two standard ways: utilitarian (total sum of the agents' individual utilities) and egalitarian (least individual utility of an agent).

For OAP, it is long known that a socially optimal allocation can be computed in polynomial time if the agents' utilities for the objects are given (maximum weight perfect matching). For example, these utilities can be collected by a central authority which computes the social optimum. Interestingly, we show that the problem is still polynomial-time solvable for COAP if $\mathcal{S}$ is the base set of a matroid defined on $\mathcal{X}$. This matroidal subcase of COAP, denoted by MOAP for *matroidal object allocation problem*, is well motivated and central to our work. For COAP in general, we prove that computing a socially optimal allocation is an **NP**-hard problem.

The remaining part of the article deals with *Serial Dictatorship* (SD), a famous mechanism for the greedy construction of solutions in the object allocation problem (no central authority is required). Given a permutation of N (also called policy), the first agent selects his *top item*, i.e. his most preferred object in $\mathcal{X}$, and removes it from $\mathcal{X}$. Then, the second agent selects his top item and removes it from $\mathcal{X}$, and so on. The allocation produced by SD is rarely a social optimum. However, it is known that for every instance of OAP, there must be a permutation of N such that SD ends up with a socially optimal allocation (a folk result).

In this article we provide an extension of SD to COAP: at each step, the agent who plays selects his most preferred element under the constraint that the set of currently selected elements can be completed in a set of $\mathcal{S}$. We show that for every instance of MOAP (the matroidal version of COAP), there must be at least one permutation of N such that SD ends up with a socially optimal allocation. For COAP in general, there is at least one instance such that no permutation of SD provides a socially optimal allocation according to the utilitarian or the egalitarian social welfare.

Finally we study the problem of a central authority who wishes that a socially optimal allocation emerges and, to do so, manipulates as little as possible the policy of SD. Here, assuming that the manipulation is measured by the number of agents who are imposed a choice of object by the central authority, we prove that manipulating SD for OAP is **NP**-hard even with approval scores and for several notions of social welfare (utilitarian, egalitarian, Pareto optimality). On the other side, we show that this problem becomes polynomial for simple scores like plurality, 2-approval or veto. In particular, we obtain the following dichotomic complexity result for the egalitarian social welfare of OAP with k-approval scores: k constant is polynomial if and only if $k \leq 2$.

This article is organized as follows. Related works are provided in Section 2. Formal definitions of COAP, MOAP and OAP, together with basic notions on matroids are given in Section 3. MOAP is new and central to our work so we motivate it with applications in Section 4. The problem of computing a social optimum is studied in Section 5. SD and its extension to COAP are investigated in Section 6. More precisely, see Section 6.1 for the existence of a permutation inducing a social optimum for MOAP. Finally, Section 7 deals with the manipulation of SD. A summary of our

<table>
<tr><th></th><th>central computation of a social optimum</th><th colspan="3">Serial Dictatorship (SD)</th></tr>
<tr><td>COAP</td><td>NP-hard</td><td>for some instances, no permutation induces a social optimum</td><td rowspan="3">approximation ratio of 0 in general. It is $O(n^{-2})$ under mild assumptions (*)</td><td rowspan="3">NP-hard to manipulate</td></tr>
<tr><td>MOAP</td><td rowspan="2">polynomial time solvable</td><td rowspan="2">at least one permutation induces a social optimum</td></tr>
<tr><td>OAP</td></tr>
</table>

Table 1: Contribution. Except (*), all these results hold for utilitarian and egalitarian social welfare.

contribution is given in Table 1.

2 Related work

Many real-world applications require to pair some entities: jobs and workers, houses and families, men and women, students and schools, etc. These well-studied problems are often called *markets* or *matchings*.

In a *two-sided market* there are two groups of agents and everyone has preferences over the members of the opposite group (e.g. men and women). A solution is a matching M that consists of pairs (one member of each group) and $M(x)$ denotes the agent matched with x under M. Given a two-sided market, the famous *stable marriage problem* is about to find a matching M such that no pair of agents (a, b), not matched together, satisfies "a prefers b to $M(a)$" and "b prefers a to $M(b)$". Such a stable matching always exists and Gale and Shapley provided an algorithm to build it [21].

Sometimes M has to satisfy some extra constraints. For example, a school may have bounds on the number of students that it can host. Schools can be classified according to their topic/location and there may be additional quotas, not on the schools directly, but on the groups on schools. In a different context, this is known as *laminar matroids* which are special cases of *matroids* [34, 29, 31]. Matroidal extensions of the stable matching problem have already been studied in [4, 19, 23, 27].

A *one-sided market* is also cut in two groups of agents but only one group has preferences over the other group (e.g. families and houses). In [35], Shapley and Scarf study a one-sided house market with endowments (each agent owns a house). They search for an allocation such that no coalition of agents can improve upon it. Such a stable allocation always exists and it is produced by Gale's *top trading cycle algorithm* (TTC) [35]. TTC is centralized and mainly based on reallocating resources along potentially long cycles of exchanges. In a recent paper [10], the authors propose to study this kind of algorithms with the restriction that only small cycles of exchanges are allowed (cycles involving at most 4 agents but most of the results concern bilateral exchange). For instance, the authors identified a domain where this procedure converges to a Pareto-optimal allocation, and they proved that the worst-case loss of welfare is as good as it can be under the assumption of individual rationality. They also show the **NP**-completeness of deciding whether an allocation resulting of swaps and maximizing utilitarian or egalitarian welfare is reachable.

This paper deals the *object allocation problem* (OAP), introduced in 1979 by Hylland and Zeckhauser [24] (see also [39, 37]). It is a one-sided market with no endowment; a set n items has to be allocated to a set N of n agents. One of the $n!$ possible allocations is chosen with a *mechanism*. A mechanism is deterministic if one specific allocation is returned with probability 1. Usually, a mechanism has to elicit the agents' private preferences but in that case, the agents may have incentive to strategize, i.e. to misreport their true preferences in order to influence the outcome of the mechanism. In a *strategy-proof* mechanism, reporting false preferences cannot be profitable. *Pareto optimality* is reached by a mechanism if the profile of the agents' utilities is not dominated by another utility vector.

Serial dictatorship (SD) is a well-studied deterministic mechanism for the object allocation problem. The agents play in turn according to a given permutation π. During his turn, an agent takes his most preferred item within the set of remaining items. We end up with an *allocation*, say A, where $A(i)$ designates the item allocated to agent i.

SD satisfies several valuable properties including Pareto optimality for strict preferences[1] and group-strategy-proofness (no group of agents can strategize).

Zhou [39] utilizes a random version of SD, called *random serial dictatorship* (RSD) which consists in choosing a permutation of N uniformly at random and then, SD is performed.If RSD is executed, then for every agent-object pair (i, x), agent i gets object x with probability P_{ix}. Saban and Sethuraman [32], together with Aziz, Brandt and Brill [1], have recently shown that computing the bi-stochastic matrix P is #P-complete.

In a recent work, Filos-Ratsikas *et al.* [18] study the approximation ratio of RSD for the object allocation problem. This is the worst case expected social utility of an allocation produced by RSD divided by the value of a social optimum. This ratio is $\Theta(n^{-1/2})$ where $n = |N|$.

3 Models and matroids

An instance of the *Constrained Object Allocation Problem* (COAP) consists of a set N of n agents and a structure $(\mathcal{X}, \mathcal{S})$ where $\mathcal{X}$ is a set of at least n elements and $\mathcal{S}$ is a collection of subsets of $\mathcal{X}$ where each $S \in \mathcal{S}$ is of size n.

We allow $\mathcal{S}$ to be defined explicitly (all its members are listed) or implicitly (we are equipped with a test which indicates whether $S \subseteq \mathcal{X}$ belongs to $\mathcal{S}$ and this test is polynomial in $|\mathcal{X}|$). However, for hardness results we assume, as it is mainly done in the literature, that $\mathcal{S}$ is given implicitly.

A *valid allocation* (or feasible solution) is a function $A : N \to \mathcal{X}$ satisfying $\bigcup_{i \in N} A(i) \in \mathcal{S}$. We say that $A(i)$ is the element *allocated* (or *assigned*) to agent i. In this article we assume that an agent cannot be allocated more than one object.

An allocation can be evaluated from the point of view of a single agent or from the point of view of the entire group of agents. The *individual utility* of agent i with respect to element $A(i) \in \mathcal{X}$ is denoted by $u_i(A(i))$ and $u_i(A(i))$ is a nonnegative real. The *social welfare* of an allocation A is usually measured in three standard ways:

- $\mathcal{U}(A) = \sum_{i \in N} u_i(A(i))$ (utilitarian social welfare);
- $\mathcal{E}(A) = \min_{i \in N} u_i(A(i))$ (egalitarian social welfare);
- *Pareto-optimality.* Allocation A is Pareto optimal if no other allocation A' satisfies $[\forall i \in N,\ u_i(A'(i)) \geq u_i(A(i))]$ and $[\exists j \in N,\ u_j(A'(j)) > u_j(A(j))]$.

[1]strict preference means $a \prec_i b$ iff $u_i(a) > u_i(b)$.

The *Object Allocation Problem* (OAP) is a special case of COAP where $\mathcal{X}$ is a set of $|N|$ objects and $\mathcal{S} = \{\mathcal{X}\}$. An intermediate case, called *Matroidal Object Allocation Problem* (MOAP), is defined on a *matroid*. Let us give some basic notions on matroids before MOAP is introduced (see [34, 29, 31] for more details on matroid theory).

3.1 Matroids

A matroid $(E, \mathcal{F})$ consists of a finite set E and a collection $\mathcal{F}$ of subsets of E such that:

(M1) $\emptyset \in \mathcal{F}$;

(M2) if $F_2 \subseteq F_1$ and $F_1 \in \mathcal{F}$, then $F_2 \in \mathcal{F}$;

(M3) if F_1, $F_2 \in \mathcal{F}$ such that $|F_1| < |F_2|$, then there exists $e \in F_2 \backslash F_1$ such that $F_1 \cup \{e\} \in \mathcal{F}$.

The elements of $\mathcal{F}$ are called *independent sets*. Inclusionwise maximal independent sets are called *bases*. A matroid can be defined by its set of bases, i.e. $(E, \mathcal{B})$, where $\mathcal{B}$ denotes the set of bases, is an alternative definition of $(E, \mathcal{F})$ [34]. The *rank* of $F \subseteq E$ is defined as $\max\{|G| : G \subseteq F, G \in \mathcal{F}\}$. All the bases of a matroid have the same cardinality, also called the rank of the matroid.

A subset of E that is not independent is *dependent*. Inclusionwise minimal dependent sets are called *circuits*. If for $F \in \mathcal{F}$ and $e \in E \setminus F$ we have $F \cup \{e\} \notin \mathcal{F}$ then $F \cup \{e\}$ contains a *unique* circuit denoted by $\mathcal{C}(F, e)$ and $\mathcal{C}(F, e)$ contains e.

The *independence oracle* of a matroid $(E, \mathcal{F})$ is a test for determining if a set $F \subseteq E$ belongs to $\mathcal{F}$. Usually, an algorithm does not manipulate a matroid directly but its independence oracle. In this article, we always assume that the time complexity of the independence oracle is polynomial in the size of E.

When every element $e \in E$ has a weight $w(e) \in \mathbb{R}$, a typical optimization problem consists in computing a base B that maximizes $\sum_{e \in B} w(e)$. This problem is solved in polynomial time by a greedy algorithm [11]. Given two matroids $(E,\ \mathcal{F}_1)$ and $(E,\ \mathcal{F}_2)$ and a weight $w(e) \in \mathbb{R}$ for every $e \in E$, there exist polynomial algorithms to find an independent set $F \in \mathcal{F}_1 \cap \mathcal{F}_2$ that maximizes $\sum_{e \in F} w(e)$ [20]. See also [34, 29] for the algorithms.

Let us finish this section with typical examples of matroids.

A *laminar* matroid is given by k (not necessarily disjoint) sets $E_1, \ldots, E_k$ and k nonnegative integers $b_1, \ldots, b_k$. For every pair of sets E_i, E_j, one of following cases occurs: $E_i \subseteq E_j$ or $E_i \subseteq E_j$ or $E_i \cap E_j = \emptyset$. A laminar matroid $(E, \mathcal{F})$ is such that $E := \bigcup_{i=1}^{k} E_i$ and $\mathcal{F} := \{F \subseteq E : |F \cap E_i| \leq b_i\}$. The *partition matroid* is a special case of laminar matroid in which the k sets are disjoint.

Given k (not necessarily disjoint) sets $E_1, \ldots, E_k$, subsets of a ground set E, a *partial transversal* is a set $T \subseteq E$ such that there exists an injective map $\Phi : T \to [1..k]$ satisfying $t \in X_{\Phi(t)}$ for all $t \in T$. Then $(E, \mathcal{F})$ where $\mathcal{F} = \{T \in 2^E : T \text{ is a partial transversal of } E\}$ is a *transversal matroid*.

If $\mathcal{F}$ denotes the set of forests of a multigraph $G = (V, E)$, then $(E, \mathcal{F})$ is called the *graphic matroid* of G. The *free matroid* is defined as $(E, 2^E)$, and E is its unique base.

3.2 MOAP

In this article we pay particular attention to MOAP — the matroidal version of OAP. For MOAP, $\mathcal{S}$ is the base set of a matroid $(\mathcal{X}, \mathcal{F})$ and each base has size $n = |N|$. Note that if we are given a matroid whose bases contain more than n elements, then we can restrict ourselves to $(\mathcal{X}, \mathcal{F}')$ such that $\mathcal{F}' = \{F \in \mathcal{F} : |F| \leq n\}$, which is also a matroid.

In the following, we interchangeably use $\mathcal{S}$ and the underlying matroid $(\mathcal{X}, \mathcal{F})$ for the input of MOAP. Notice that OAP corresponds to MOAP with the free matroid.

4 Motivation

Let us give some possible applications of MOAP and COAP.

Example 1. *Let $\mathcal{X}$ be a set of 75 offices composed of 10 units located in building A, 15 units located in building B and* 50 *units located in building C. There are* 60 *workers and we want to assign one office per worker. For financial reasons (e.g. offices in building A are more expensive than in the other buildings), at most* 8 *offices from building A can be allocated. Furthermore at least* 4 *offices of buildings A and B must be left free because of forthcoming recruitment.*

The situation depicted in Example 1 corresponds to a laminar matroid. We have $\mathcal{X}_A = \{x_1, \ldots, x_{10}\}$, $\mathcal{X}_{AB} = \{x_1, \ldots, x_{25}\}$, $\mathcal{X}_C = \{x_{26}, \ldots, x_{75}\}$ and $\mathcal{X} = \mathcal{X}_{AB} \cup \mathcal{X}_C$. Then $\mathcal{S}$ contains every set S satisfying $S \subseteq \mathcal{X}$, $|S| = 60$, $|S \cap \mathcal{X}_A| \leq 8$ and $|S \cap \mathcal{X}_{AB}| \leq 21$.

Example 2. *The researchers of a given institute can invite external colleagues for 1 month visits. Let $\mathcal{X}$ be the set of possible external researchers. We know during which months these possible guests can visit the institute. The problem is to assign one guest per internal researcher under the constraint that no two guests are invited at the same time. Internal researchers have utilities with respect to the external researchers but these values are independent of the visiting period. The next instance involves 5 possible guests and 3 months.*

	January	*April*	*June*
Dr. Red	*1*	*1*	*0*
Dr. Blue	*0*	*1*	*0*
Dr. Yellow	*1*	*0*	*1*
Dr. Pink	*0*	*1*	*0*
Dr. Brown	*1*	*0*	*1*

It is possible to invite Doctors Red, Blue and Yellow in January, April and June, respectively. However we cannot invite Doctors Red, Blue and Pink because none of them is available in June.

The situation depicted in Example 2 corresponds to a *transversal matroid.*

Example 3. *For the provisioning of the International Space Station (ISS) the 5 actors of the program (USA, Russia, EU, Japan, Canada) regularly send a cargo that is limited in space and weight. Let $\mathcal{X}$ denote the bundles of objects that an actor may wish to send to ISS. All possible sets of 5 bundles of objects satisfying the constraints of space and weight form $\mathcal{S}$.*

This last example falls in the case of COAP but not of MOAP.

5 Computing a socially optimal allocation

In this section we seek a good solution for the group of agents. Let $\hat{A}_{\mathcal{U}}$ and $\hat{A}_{\mathcal{E}}$ denote valid allocations that maximize the utilitarian social welfare $\mathcal{U}(\hat{A}_{\mathcal{U}}) = \sum_{i\in N} u_i(\hat{A}_{\mathcal{U}}(i))$ and the egalitarian social welfare $\mathcal{E}(\hat{A}_{\mathcal{E}}) = \min_{i\in N} u_i(\hat{A}_{\mathcal{E}}(i))$, respectively. We are going to see that computing $\hat{A}_{\mathcal{U}}$ or $\hat{A}_{\mathcal{E}}$ is **NP**-hard for COAP but polynomial for MOAP.

Proposition 1. *For COAP, computing $\hat{A}_{\mathcal{U}}$ or $\hat{A}_{\mathcal{E}}$ is* **NP***-hard.*

Proof. The reduction is done from *Hamiltonian Cycle* (HC in short) which is known to be **NP**-complete [22]. HC consists in deciding if a given graph has an Hamiltonian cycle. Given an instance $G = (V, E)$ of HC with vertex set $\{1, \dots, n\}$, build an instance of COAP such that $N = \{1, \dots, n\}$, $\mathcal{X} = \{(i, j) : 1 \le i, j \le n\}$ and $S \in \mathcal{S}$ if and only if S is a Hamiltonian cycle. Finally, $u_i(a, b) = 1$ if $(a, b) \in E$, otherwise $u_i(a, b) = 0$. Therefore $\mathcal{U}(\hat{A}_{\mathcal{U}}) = n$ (resp., $\mathcal{E}(\hat{A}_{\mathcal{E}}) = 1$) if and only if G has an Hamiltonian cycle. □

Notice that if $S \in \mathcal{S}$ is given, i.e. what the agents collectively receive is fixed, then finding an allocation A that maximizes $\mathcal{U}(A)$ (resp., $\mathcal{E}(A)$) under the constraint $\bigcup_{i\in N} A(i) = S$ can be done within polynomial time by using matching algorithms.

We are going to see that both $\hat{A}_{\mathcal{U}}$ and $\hat{A}_{\mathcal{E}}$ can be computed efficiently for MOAP (in a centralized manner). The input is $(N, (\mathcal{X}, \mathcal{S}))$ where $\mathcal{S}$ is the base set of a matroid $(\mathcal{X}, \mathcal{F})$.

The fact that $\hat{A}_{\mathcal{U}}$ and $\hat{A}_{\mathcal{E}}$ are polynomial time computable is shown after an intermediate result. Suppose $\mathcal{X} = \{x_1, \dots, x_m\}$ and for every $k \in [m]$, let $Y_k = \{y_k^1, \dots, y_k^n\}$ where each y_k^i can be seen as a copy of x_k associated with agent i. Let $\mathcal{Y} = \bigcup_{k=1}^m Y_k$. For any $D \subseteq \mathcal{Y}$, let $p(D) := \{x_k \in \mathcal{X} : |D \cap Y_k| > 0\}$ be the *projection of* D; note that $p(D)$ is not a multiset. Let $\mathcal{D} = \{D \subseteq \mathcal{Y} : (p(D) \in \mathcal{F}) \wedge (|D \cap Y_k| \le 1,\ k = 1..m)\}$.

Lemma 1. *If $(\mathcal{X}, \mathcal{F})$ is a matroid then $(\mathcal{Y}, \mathcal{D})$ is a matroid.*

Proof. We have to verify the three properties of a matroid.

(M1) $(\mathcal{X}, \mathcal{F})$ is a matroid, so $\emptyset \in \mathcal{F}$. Using $p(\emptyset) = \emptyset$ and $|\emptyset \cap Y_k| = 0$ for all k, we get that $\emptyset \in \mathcal{D}$.

(M2) Take D, D' such that $D \subset D' \subseteq \mathcal{Y}$ and $D' \in \mathcal{D}$. $|D' \cap Y_k| \le 1$ for all k implies $|D \cap Y_k| \le 1$ for all k. By the definition of p, $p(D)$ is a subset of $p(D')$. From $D' \in \mathcal{D}$ we know that $p(D') \in \mathcal{F}$. Because $(\mathcal{X}, \mathcal{F})$ is a matroid, any subset of $p(D')$ ($p(D)$ in particular) is in $\mathcal{F}$.

(M3) Take D and D', two members of $\mathcal{D}$, such that $|D| < |D'|$. It follows that $|p(D)| < |p(D')|$. Since both $p(D)$ and $p(D')$ belong to $\mathcal{F}$, there must be $x_{k^*} \in p(D') \setminus p(D)$ such that $p(D) + x_{k^*} \in \mathcal{F}$ by property (M3). Let $y_{k^*}^{i^*}$ be the unique member of D' such that $p(\{y_{k^*}^{i^*}\}) = x_{k^*}$. $D \cap Y_{k^*}$ must be empty, otherwise $x_{k^*} \in p(D)$, a contradiction. Hence $|D + y_{k^*}^{i^*} \cap Y_{k^*}| = 1$. In conclusion, $y_{k^*}^{i^*}$ belongs to $D' \setminus D$ and $D + y_{k^*}^{i^*} \in \mathcal{D}$. □

Note that $(\mathcal{X}, \mathcal{F})$ and $(\mathcal{Y}, \mathcal{D})$ have the same rank. The independence oracle of $(\mathcal{X}, \mathcal{F})$ is, by hypothesis, polynomial in $|\mathcal{X}|$. Thus a polynomial independence oracle for $(\mathcal{Y}, \mathcal{D})$ is immediately derived. The interest of Lemma 1 is that $(\mathcal{Y}, \mathcal{D})$ carries more information than $(\mathcal{X}, \mathcal{F})$ (having y_k^i in a solution means that x_k is picked by agent i) and the properties of a matroid are preserved.

Theorem 1. *For MOAP, $\hat{A}_{\mathcal{U}}$ can be computed in polynomial time.*

Proof. We are going to see that a socially optimal allocation (utilitarian) corresponds to an independent set of maximum weight at the intersection of two matroids, which is a polynomial time solvable problem [34, 29].

Consider $(\mathcal{Y}, \mathcal{D})$, the matroid associated with $(\mathcal{X}, \mathcal{F})$ (see Lemma 1). For every $i \in N$, let Y'_i denote $\{y^i_1, y^i_2, \ldots, y^i_m\}$, i.e. the copies of $\mathcal{X}$ associated with agent i. We have a partition $Y'_1 \cup Y'_2 \cup \ldots \cup Y'_n$ of $\mathcal{Y}$, so $(\mathcal{Y}, \mathcal{G})$ where $\mathcal{G} = \{Z \subseteq \mathcal{Y} : |Z \cap Y'_i| \leq 1 \text{ for every } i \in N\}$ is a partition matroid. For every $y^i_k \in \mathcal{Y}$, define its weight $w(y^i_k)$ as $u_i(x_k)$ where $i \in N$ and $k \in [m]$. We claim that if $S \in \mathcal{D} \cap \mathcal{G}$ has maximum weight $w(S)$, then $p(S)$ is an optimum for the utilitarian social welfare $\mathcal{U}$. Indeed, for every $S \in \mathcal{D} \cap \mathcal{G}$ we know that $S \in \mathcal{D} \Rightarrow p(S) \in \mathcal{F}$ (definition of matroid and $S \in \mathcal{G}$ implies that no agent is associated with more than one element).

If $|S| < |N|$ then at least one agent, say i', is not associated with an element of S. Take any base B of $(\mathcal{X}, \mathcal{F})$. We have $|B| = |N|$ and $|p(S)| = |S| < |N|$ so by property M3, there exists $x_j \in B \setminus p(S)$ such that $x_j + p(S) \in \mathcal{F}$. It follows that $y^{i'}_j$, the copy of x_j associated with agent i', can be added to S, i.e. $y^{i'}_j \notin S$ and $y^{i'}_j + S \in \mathcal{D}$. We get that $w(y^{i'}_j + S) \geq w(S)$ by the non negativeness of $w(y^{i'}_j)$. Therefore, we can suppose w.l.o.g. that $|S| = |N|$. To conclude, $\mathcal{U}(p(S))$ is equal to $w(S)$. □

Theorem 2. *For MOAP, $\hat{A}_{\mathcal{E}}$ can be computed in polynomial time.*

Proof. The proof relies on the tools introduced for the previous proof. Let T be a threshold for which we are going to test if an allocation A satisfying $\mathcal{E}(A) \geq T$ can be built. Consider $(\mathcal{Y}, \mathcal{D})$, the matroid associated with $(\mathcal{X}, \mathcal{F})$. For every $i \in N$, let Y'_i denote $\{y^i_1, y^i_2, \ldots, y^i_m\}$. We have a partition matroid $(\mathcal{Y}, \mathcal{G})$ where $\mathcal{G} = \{Z \subseteq \mathcal{Y} : |Z \cap Y'_i| \leq 1 \text{ for every } i \in N\}$. For every $y^i_k \in \mathcal{Y}$, define its weight $w(y^i_k)$ as 1 if $u_i(x_k) \geq T$, otherwise $w(y^i_k) = 0$. Therefore, if $S \in \mathcal{D} \cap \mathcal{G}$ has weight $w(S) = n$, then $p(S)$ satisfies $\mathcal{E}(p(S)) \geq T$. With a binary search on T, we can guess the value $\mathcal{E}(\hat{A}_{\mathcal{E}})$. □

6 Serial Dictatorship

The previous section was dedicated to the centralized computation of a socially optimal allocation. This approach is relevant when there exists a sort of central authority who knows the agents' true utilities for the objects but it fails in the following situations:

- the agents are reluctant to disclose their true valuation for the objects because they do not trust the authority;
- the agents act strategically by misreporting their utilities;
- no central authority exists.

Therefore, other mechanisms for the construction of an allocation must be used. *Serial Dictatorship* (SD) is a well-studied deterministic mechanism for OAP. The agents play in turn according to a given permutation π. During his turn, an agent takes his most preferred item within the set of remaining items. We end up with an *allocation*, say A^π, where $A^\pi(i)$ designates the item allocated to agent i.

With SD no central authority is required and the agents need not disclose their true valuations. However an ordering on the agents is assumed. This ordering can be seen as an exogenous ranking of the agents. In Example 2, there can be an order of priority within the inviting researchers.

So far we assumed that the agents have utilities with respect to the objects and for SD we need to clarify which object is selected by an agent in case of a tie. We suppose that every agent i has his own total and strict order $\succ_i$ on $\mathcal{X}$. This order is compliant with u_i in the sense that $u_i(x) > u_i(y)$ implies $x \succ_i y$. If several available objects maximize the individual utility of an agent, then the element coming first in $\succ_i$ is selected by the agent.

Let us describe how SD extends to COAP with input $(N, (\mathcal{X}, \mathcal{S}))$. The agents play in turn according to a given permutation π on N. The allocation A^π, which is undefined at the beginning, is gradually built. At every step, the partial solution must be a subset of a member of $\mathcal{S}$.

When it is the turn of agent $\pi(i)$, the set of elements that are already assigned is $\bigcup_{j<i} A^\pi(\pi(j))$. The possible actions of agent $\pi(i)$ are to pick one element in $\{x \in \mathcal{X} \setminus \bigcup_{j<i} A^\pi(\pi(j)) : \exists S \in \mathcal{S}$ such that $S \supseteq x + \bigcup_{j<i} A^\pi(\pi(j))\}$. The element that agent $\pi(i)$ likes the most (according to $\succ_{\pi(i)}$) in this set is denoted by $top_{\pi(i)}$, or $top_{\pi(i)}(\bigcup_{j<i} A^\pi(\pi(j)))$ if the previously assigned elements need to be stressed. So $top_{\pi(i)}$ is allocated to agent $\pi(i)$. We sometimes say that it is *picked* by $\pi(i)$.

Let us emphasize a particularity of MOAP for SD. A^π is empty at the beginning and for $i = 1$ to n, agent $\pi(i)$ adds to $\{A^\pi(\pi(j)) : j < i\}$ the element x that he likes the most under the constraint that $\{A^\pi(\pi(j)) : j < i\} + x$ is an independent set. Because of M3 (see Section 3.1), as soon as adding x to $\{A^\pi(\pi(j)) : j < i\}$ preserves the independence of the partial solution, we know that $\{A^\pi(\pi(j)) : j < i\} + x$ can be completed in a base of the underlying matroid. Thus, no need to foresee if $\{A^\pi(\pi(j)) : j < i\} + x$ is the subset of some $S \in \mathcal{S}$.

6.1 Can SD induce a social optimum?

As a reminder, $\hat{A}_{\mathcal{U}}$ and $\hat{A}_{\mathcal{E}}$ designate an allocation maximizing the utilitarian and egalitarian social welfare, respectively. The allocation produced by SD under permutation π is denoted by A^π. We clearly have $\mathcal{U}(\hat{A}_{\mathcal{U}}) \geq \mathcal{U}(A^\pi)$ and $\mathcal{E}(\hat{A}_{\mathcal{E}}) \geq \mathcal{E}(A^\pi)$ for every permutation π. The best outcome if we restrict ourselves to the allocations produced by SD will be denoted by $A^{\pi^*_{\mathcal{U}}}$ and $A^{\pi^*_{\mathcal{E}}}$, respectively. That is, $\pi^*_{\mathcal{U}} = argmax_{\pi \in \mathcal{P}} \mathcal{U}(A^\pi)$ and $\pi^*_{\mathcal{E}} = argmax_{\pi \in \mathcal{P}} \mathcal{E}(A^\pi)$ where $\mathcal{P}$ denotes the set of all permutations on N.

It can be $\mathcal{U}(\hat{A}_{\mathcal{U}}) > \mathcal{U}(A^{\pi^*_{\mathcal{U}}})$ and $\mathcal{E}(\hat{A}_{\mathcal{E}}) > \mathcal{E}(A^{\pi^*_{\mathcal{E}}})$ because SD is sometimes unable to induce a social optimum. For instance, consider the instance of COAP described in Exemple 4.

Example 4. *There are two agents $N = \{1, 2\}$ (and thus, two possible permutations). The instance is described as follows:*

- $\mathcal{X} = \{l_1, l_2, l_3, r_1, r_2, r_3\}$;
- $\mathcal{S} = \{(l_1, r_2), (l_2, r_1), (l_3, r_3)\}$;
- $l_1 \succ_1 l_3 \succ_1 l_2 \succ_1 r_1 \succ_1 r_2 \succ_1 r_3$;
- $r_1 \succ_2 r_3 \succ_2 r_2 \succ_2 l_1 \succ_2 l_2 \succ_2 l_3$.

For the identity permutation, player 1 picks l_1 followed by player 2 who picks r_2. For the other permutation, player 2 picks r_1 followed by player 1 who picks l_2. Now we can find numerical values such that $\hat{A}_{\mathcal{U}} = \hat{A}_{\mathcal{E}} = \{(l_3, r_3)\}$ (agent 1 gets objet l_3 and the other agent has r_3), e.g. $u_1(l_1) = u_2(r_1) = 3$, $u_1(l_2) = u_2(r_2) = 0$ and $u_1(l_3) = u_2(r_3) = 2$. For the utilitarian social welfare, SD *produces a solution of value 3 whereas the optimum is 4. For the egalitarian social welfare,* SD *produces a solution of value 0 whereas the optimum is 2.*

Thus, SD may fail to produce a social optimum, whichever order on the agents is selected. This observation is made for COAP but if we consider MOAP then we are going to prove that for every instance, there exists a permutation of the agents such that SD produces a socially optimal allocation.

Let $(N, (\mathcal{X}, \mathcal{S}))$ be an instance of MOAP such that $\mathcal{S}$ is the base set of a matroid $(\mathcal{X}, \mathcal{F})$. We are going to compute a socially optimal solution $\hat{A}$ (depending on the context, $\hat{A} = \hat{A}_{\mathcal{U}}$ or $\hat{A} = \hat{A}_{\mathcal{E}}$) and a permutation π such that $A^{\pi} = \hat{A}$. The algorithm is described in Algorithm 1 and it uses Algorithm 2 as a subroutine.

Let us give an overview of the algorithms. First a socially optimal solution $\hat{A}$ is computed. The current solution S' is initially empty and the current position j in the permutation is initially 1. Every agent is unassigned (N_0 is the set of unassigned agents). SD is simulated by Algorithm 2: As long as there exists an unassigned agent i such that $top_i(S') = \hat{A}(i)$, agent i is put on position j of the permutation, $\hat{A}(i)$ is added to S', j is incremented by 1, and agent i is removed from the set of unassigned agents. If the set of unassigned agents is empty then we are done. Otherwise, $\hat{A}$ must be modified in order to continue the construction of the permutation.

Algorithm 1:

Data: N, a matroid $(\mathcal{X}, \mathcal{F})$ given by its independence oracle, $(\succ_i)_{i \in N}$, $(u_i)_{i \in N}$

Result: a permutation π on N such that A^{π} is a social optimum for $\mathcal{U}$ or $\mathcal{E}$, depending on the context

Build a social optimum $\hat{A} = \hat{A}_{\mathcal{U}}$ (or $\hat{A} = \hat{A}_{\mathcal{E}}$, depending on the context) for the instance (see Theorem 1 or Theorem 2)

Let π be a permutation on N (to be determined)

$N_0 \leftarrow N$

$j \leftarrow 1$

while $j \leq |N|$ **do**

    $\langle \pi, j, N_0 \rangle \leftarrow$ Algorithm 2 $(N, (\mathcal{X}, \mathcal{F}), (\succ_i)_{i \in N}, \pi, j, N_0, \hat{A})$

    if $N_0 \neq \emptyset$ **then**

        $S' \leftarrow \{\hat{A}(i) : i \in N \setminus N_0\}$

        $\hat{S} \leftarrow \{\hat{A}(i) : i \in N\}$

        if $\exists i \in N_0$ *such that* $\hat{S} - \hat{A}(i) + top_i(S') \in \mathcal{F}$ **then**

            $\hat{A}(i) \leftarrow top_i(S')$

        else

            Create an exchange digraph $G_{ex} = (N_0, E_{ex})$ such that $(i, i') \in E_{ex}$ if and only if $\hat{A}(i') \in \mathcal{C}(\hat{S}, top_i(S'))$ (see Section 3.1 for the definition of $\mathcal{C}(\hat{S}, top_i(S'))$)

            Take a directed cycle C of G_{ex} of minimum length and let N'_0 be the node set of C

            foreach $i \in N'_0$ **do**

                $\hat{A}(i) \leftarrow top_i(S')$

return π

Theorem 3. *For every matroid $(\mathcal{X}, \mathcal{F})$, Algorithm 1 provides a permutation π such that $\mathcal{U}(A^{\pi}) = \mathcal{U}(\hat{A}_{\mathcal{U}}) = \mathcal{U}(\hat{A})$ or $\mathcal{E}(A^{\pi}) = \mathcal{E}(\hat{A}_{\mathcal{E}}) = \mathcal{E}(\hat{A})$, depending on the definition of the social welfare.*

Algorithm 2: Simulated SD

```
Data: N, a matroid (𝒳, ℱ) given by its independence oracle, (≻_i)_{i∈N}, π, j, N_0, Â
Result: a permutation on N, an integer and a subset of N
S' ← {Â(i) : i ∈ N \ N_0}
t ← j
while there exists i ∈ N_0 such that Â(i) = top_i(S') do
    π(t) ← i
    N_0 ← N_0 − i
    t ← t + 1
    S' ← S' + Â(i)
return ⟨π, t, N_0⟩
```

Proof. We prove the result simultaneously for the utilitarian and the egalitarian social welfare, because the proofs are similar. The case $N_0 = \emptyset$ (see line 7 of Algorithm 1) is direct since the permutation π is fully determined. Let us consider the case $N_0 \neq \emptyset$. The current solution is $S' := \{\hat{A}(\ell) : \ell \in N \setminus N_0\}$ and for every $i \in N_0$ it holds that $top_i(S') \neq \hat{A}(i)$ and $\exists S \in \mathcal{S}$ such that $S \supset (top_i(S') + S')$.

At line 10 of Algorithm 1 we check if an unassigned agent i can replace $\hat{A}(i)$ by $top_i(S')$ so that the new allocation remains valid. If it is possible then $\hat{A}$ is modified accordingly. Otherwise we use an *exchange digraph* $G_{ex} = (N_0, E_{ex})$ such that $(i, i') \in E_{ex}$ if and only if $\hat{A}(i') \in \mathcal{C}(\hat{S}, top_i(S'))$ (see Section 3.1 for the definition of $\mathcal{C}(\hat{S}, top_i(S'))$).

Property 1. *G_{ex} admits a directed cycle if $\forall i \in N_0$, $(\hat{S} - \hat{A}(i) + top_i(S'))$ is not independent.*

Proof. For every $i \in N_0$ there exists a base S of $\mathcal{F}$ such that $S \supset (top_i(S')+S')$. Thus, $top_i(S')+S'$ is independent. However $top_i(S') \neq \hat{A}(i)$ and $\hat{S}$ is a base so $top_i(S')+\hat{S}$ contains a circuit $\mathcal{C}(\hat{S}, top_i(S'))$ and this circuit must contain at least one element of $\{\hat{A}(j) : j \in N_0 - i\}$[2], say $\hat{A}(i')$. By construction, arc (i, i') belongs to E_{ex}. Therefore, for each node i of N_0, there is at least one arc to another node i' of N_0. As a consequence, G_{ex} admits a directed cycle. □

Property 1 indicates that the directed cycle mentioned at line 14 of Algorithm 1 must exist. We shall use a theorem taken from [20] (see also [29]).

Theorem 4. *[20] Let $(E, \mathcal{F})$ be a matroid and $F \in \mathcal{F}$. Let $x_1, \ldots, x_s \in F$ and $y_1, \ldots, y_s \notin F$ with*

(a) $x_k \in \mathcal{C}(F, y_k)$ for $k = 1, \ldots, s$ and

(b) $x_j \notin \mathcal{C}(F, y_k)$ for $1 \leq j < k \leq s$.

Then $(F \setminus \{x_1, \ldots, x_s\}) \cup \{y_1, \ldots, y_s\} \in \mathcal{F}$.

Let us denote the members of N_0' by $\{1, \ldots, s\}$ such that the directed cycle mentioned at line 14 of Algorithm 1 is $\{(k, k+1) : 1 \leq k \leq s-1\} \cup \{(s, 1)\}$. Since N_0' are the nodes of a minimum

[2] If not, this circuit is included in $(top_i(S') + S' + \hat{A}(i))$. Now, since $(top_i(S') + S')$ and $\hat{S}$ are independent, axiom M3 of matroids implies that we can add all the elements of $\hat{S} \setminus S'$ except one to $(top_i(S') + S')$. By hypothesis, it is not $\hat{A}(i)$) and then $(\hat{S} + top_i(S') - \hat{A}(i))$ is a base which is a contradiction with the initial assumption.

directed cycle C, we must have that (N_0', C) is an induced subgraph of G_{ex} or equivalently C is chordless.

Because of line 13 of Algorithm 1, item (a) of Theorem 4 is satisfied if we let $F = \hat{S}$, $y_k = top_k(S')$ and $x_k = \hat{A}(k+1)$ for $k = 1, \ldots, s$ (with the convention $s + 1 = 1$). Indeed $(i, i+1) \in E_{ex}$ if and only if $x_i = \hat{A}(i+1) \in \mathcal{C}(\hat{S}, top_i(S')) = \mathcal{C}(F, y_i)$. In words, y_k is agent k's top object, y_k can be added to $\hat{S}$ if x_k is removed and x_k is initially assigned to agent $k+1$.

Now we consider item (b) of Theorem 4. The case where an agent $i \in N_0$ can replace $\hat{A}(i)$ by $top_i(S')$ is treated at line 10 of Algorithm 1 so we can consider that $|N_0'| \geq 2$. If item (b) does not hold then there exists j and k such that $x_j \in \mathcal{C}(F, y_k)$ and $1 \leq j < k \leq s$. This is equivalent to $x_j = \hat{A}(j+1) \in \mathcal{C}(\hat{S}, top_k(S'))$. In others words, E_{ex} contains arc $(k, j+1)$. If $j + 1 = k$ then we get a contradiction with the fact that no agent $i \in N_0$ can replace $\hat{A}(i)$ by $top_i(S')$. If $j + 1 < k$ then we get a contradiction with the minimality of N_0' since there is a directed cycle on $N_0' \setminus \{j\}$, i.e. consecutive arcs from $j + 1$ to k and one arc from k to $j + 1$.

Therefore, we can apply Theorem 4 and state that $\{\hat{A}(i) : i \in N \setminus N_0'\} \cup \{top_i(S') : i \in N_0'\}$ is independent. It is, of course, a base because it has the same size as $\hat{S}$. At line 16 of Algorithm 1, $\hat{A}(i)$ is replaced by $top_i(S')$ for every $i \in N_0'$. The social utility of $\{\hat{A}(i) : i \in N \setminus N_0'\} \cup \{top_i(S') : i \in N_0'\}$ is as good as $\mathcal{U}(\hat{A}_{\mathcal{U}})$ (resp., $\mathcal{E}(\hat{A}_{\mathcal{E}})$) because $u_i(top_i(S')) \geq u_i(\hat{A}_{\mathcal{U}}(i))$ (resp., $u_i(top_i(S')) \geq u_i(\hat{A}_{\mathcal{E}}(i))$) for every $i \in N_0'$.

Finally, the construction of π can be resumed by the use of Algorithm 2 (line 6 of Algorithm 1). The termination of Algorithm 1 is due to the fact that we can always find $i \in N_0$ such that $top_i = \hat{A}(i)$, until all the agents are assigned. This concludes the proof of Theorem 3. □

6.2 How bad is SD?

In this section and the next sections, we suppose without loss of generality that π is the identity so we denote by A, instead of A^π, the outcome of SD. If there is no restriction on the agents' utilities then next example shows that only the trivial lower bound of 0 can be stated.

Example 5. *Consider the following instance of OAP. There are two agents $\{1, 2\}$ and two items $\{a, b\}$. Agent 1 has utility ϵ, with $\epsilon > 0$, for both items whereas agent 2 has utility $1 + \epsilon$ for a and 0 for b. Moreover $a \succ_i b$ for every $i \in \{1, 2\}$. In the assignment A induced by* SD*, agents 1 and 2 get items a and b, respectively. Thus, $\mathcal{U}(A) = \epsilon$ and $\mathcal{E}(A) = 0$ while $\mathcal{U}(\hat{A}_{\mathcal{U}}) = 1 + 2\epsilon$ and $\mathcal{E}(\hat{A}_{\mathcal{E}}) = \epsilon$. So $\mathcal{U}(A)/\mathcal{U}(\hat{A}_{\mathcal{U}})$ tends to 0 and $\mathcal{E}(A)/\mathcal{E}(\hat{A}_{\mathcal{E}}) = 0$.*

As we will see, this negative result can be mitigated for the utilitarian social welfare if natural assumptions on the agents' utilities are made for utilitarian welfare.

Let $S^i \in \mathcal{S}$ be the outcome that agent $i \in N$ would prefer if he were allocated all the elements, i.e. S^i maximizes $\sum_{x \in S^i} u_i(x)$. Let $\{x_{i1}, \ldots, x_{in}\}$ be the elements of S^i and let α_{ij} be a simplified notation for $u_i(x_{ij})$. We suppose w.l.o.g. that

$$\alpha_{i1} \geq \alpha_{i2} \geq \ldots \geq \alpha_{in} \geq 0, \quad \forall i \in N \tag{1}$$

In addition $\alpha_{i1} > 0$, $\forall i \in N$. We make a *normalization assumption* stating that the utility of agent i for S^i is equal to a positive constant U_{tot}.

$$\sum_{j=1}^{n} \alpha_{ij} = U_{tot}, \quad \forall i \in N \tag{2}$$

This is similar to the *unit-sum* representation used in [18], where an agent's utility for an item is between 0 and 1, and the sum of an agent's utilities for the entire set of items must be 1.

Proposition 2. *Under the normalization assumption, the approximation ratio of COAP with the utilitarian social welfare is* $\frac{1}{n(n-1)+1}$.

Proof. Here, we set $\hat{A}_{\mathcal{U}} = \hat{A}$ because we only focus on the utilitarian social welfare. If for A, every agent i is allocated x_{i1}, or another element for which he has utility α_{i1}, then $\mathcal{U}(A)/\mathcal{U}(\hat{A}) = 1$. Otherwise, there is an agent i' such that $u_{i'}(\hat{A}(i')) \leq \alpha_{i'2}$.

$$\mathcal{U}(\hat{A}) \leq \left(\sum_{i \in N \setminus \{i'\}} \alpha_{i1} \right) + \alpha_{i'2} \tag{3}$$

We have

$$\mathcal{U}(A) \geq \alpha_{11} \tag{4}$$

because agent 1 can always pick his top element. By Inequalities (1) and (2), we know that $n\alpha_{11} \geq \sum_{j=1}^{n} \alpha_{1j} = U_{tot}$. We deduce that

$$\begin{aligned} n(n-1)\alpha_{11} &\geq (n-1)U_{tot} \\ (n(n-1)+1)\mathcal{U}(A) &\geq \alpha_{11} + (n-1)U_{tot} \end{aligned} \tag{5}$$

where (4) is used for the last inequality. Let us show that $\left(\sum_{i \in N \setminus \{i'\}} \alpha_{i1}\right) + \alpha_{i'2} \leq \alpha_{11} + (n-1)U_{tot}$. It holds for $i' = 1$ because $\alpha_{i1} \leq \sum_{j=1}^{n} \alpha_{i'j} = U_{tot}$ by (2) and $\alpha_{i'2} = \alpha_{12} \leq \alpha_{11}$ by (1). If $i' \neq 1$, then $\left(\sum_{i \in N \setminus \{1,i'\}} \alpha_{i1}\right) + \alpha_{11} + \alpha_{i'2} \leq \alpha_{11} + (n-1)U_{tot}$.

Combine $\left(\sum_{i \in N \setminus \{i'\}} \alpha_{i1}\right) + \alpha_{i'2} \leq \alpha_{11} + (n-1)U_{tot}$ with (3) and (5) to get that $(n(n-1)+1)\mathcal{U}(A) \geq \mathcal{U}(\hat{A})$.

Finally, observe that the bound is tight for the following family of instances of OAP ($N = \mathcal{X} = \{1, \ldots, n\}$). The first agent has utility $1/n$ for every item. For any other agent $i > 1$, his most preferred item is $i-1$ and $u_i(i-1) = 1$. Agent i's utility for the any other item is 0. Therefore $A(i) = i$ and the social utility is equal to $1/n$. The optimal allocation $\hat{A}$ consists in giving item n to agent 1 and item $i-1$ to agent i for $i > 1$. The optimal social utility is $(n-1) + 1/n$. In all, the ratio $\mathcal{U}(A)/\mathcal{U}(\hat{A}) = 1/\left(n(n-1)+1\right)$ for this instance. □

Now suppose there is a value α such that $\alpha_{i1} = \alpha$ for all $i \in N$. We still consider the normalization assumption (2). This is similar to the *unit-range* representation used in [18], where an agent's utility for an item is between 0 and 1, and every agent's maximum and minimum utility for an item must be 1 and 0, respectively.

Proposition 3. *Under the normalized assumption and if* $\alpha_{i1} = \alpha$ *for all* $i \in N$*, the approximation ratio of COAP for the utilitarian social welfare is* $\frac{1}{n-1}$.

Proof. If for A, every agent i is allocated x_{i1}, or another element for which he has utility α_{i1}, then $\mathcal{U}(A)/\mathcal{U}(\hat{A}) = 1$. Otherwise, there is an agent i' such that $u_{i'}(A(i')) \leq \alpha_{i'2}$. It follows that $\mathcal{U}(\hat{A}) \leq \left(\sum_{i \in N \setminus \{i'\}} \alpha_{i1}\right) + \alpha_{i'2}$ where $\alpha_{i1} = \alpha$ and $\alpha_{i'2} \leq \min\{\alpha, U_{tot} - \alpha\}$.

$$\mathcal{U}(\hat{A}) \leq (n-1)\alpha + \min\{\alpha, U_{tot} - \alpha\} \tag{6}$$

Agent i has utility at least α_{ii} for $A(i)$ so $\mathcal{U}(A) \geq \alpha_{11} + \alpha_{22}$ where $\alpha_{11} = \alpha$ and $\alpha_{22} \geq \frac{U_{tot}-\alpha}{n-1}$.

$$(n-1)\mathcal{U}(A) \geq (n-2)\alpha + U_{tot} \tag{7}$$

If $\min\{\alpha, U_{tot} - \alpha\} = \alpha$ then $U_{tot} \geq 2\alpha$ and $(n-1)\mathcal{U}(A) \geq n\alpha \geq \mathcal{U}(\hat{A})$. Otherwise $U_{tot} \leq 2\alpha$ and $\mathcal{U}(\hat{A}) \leq (n-1)\alpha + U_{tot} - \alpha \leq (n-1)\mathcal{U}(A)$. Thus, $\mathcal{U}(A)/\mathcal{U}(\hat{A}) \geq (n-1)$ in any case.

The bound is reached for every n by generalizing the next instance of OAP. $N = \mathcal{X} = \{1, 2, 3, 4\}$ and the agents' preferences are as follows.

$$\begin{array}{ccccccc}
1 & \succ_1 & 4 & \succ_1 & 3 & \succ_1 & 2 \\
1 & \succ_2 & 2 & \succ_2 & 4 & \succ_2 & 3 \\
2 & \succ_3 & 1 & \succ_3 & 3 & \succ_3 & 4 \\
3 & \succ_4 & 2 & \succ_4 & 1 & \succ_4 & 4
\end{array}$$

Suppose $\alpha_{i1} = 1$ and any other score is equal to 0. Agent i takes his i-th most preferred item in A while in $\hat{A}$ every agent gets his most preferred item, excepting the first agent who gets his second choice. □

Notice that the upper bounds given in the proofs of Propositions 2 and 3 hold for OAP so there is no hope for better approximation ratios for OAP or MOAP. For the *egalitarian* social welfare, results like Propositions 2 and 3 cannot be derived (for example, take $\epsilon = 1$ in Example 5).

7 Manipulation of the serial dictatorship mechanism

In the previous section we focused on SD for MOAP from both the centralized and the decentralized viewpoints. The crucial point of SD is the turn order of the agents, i.e. the permutation π. In Section 6.1 we proved that if a central authority controls the choice of π then it can always find a turn order inducing an allocation with optimal social welfare. On the opposite, when the order of turn is fixed, we showed in Section 6.2 that the social welfare of the allocation induced by SD is, in the worst case, far from the optimum. Now, assuming that π is given in advance, the goal of this part is to know if the central authority, who is aware of the agents' true utilities for the objects, can *manipulate* in order to reach a socially good allocation. We explore the possibility for the central authority to force some agents, during their turns, to choose an element which is not their most preferred one. The goal of this manipulation is to induce a socially optimal allocation. Since the decentralized nature of SD is weakened, we consider the optimization problem of minimizing the number of forced agents, provided that a socially optimal allocation emerges from the modified SD. Before giving a formal description of the manipulation problem under consideration, let us mention some related problems.

7.1 Works related to manipulation

Simple sequential allocation procedures for sharing indivisible goods between agents in which agents take turns, according to a *policy* or mechanism, to pick items have been studied in [25, 26, 5, 38].

For instance, supposing additive utilities and independence between the agents, it is shown in [25] that the expected utilitarian social welfare[3] is maximized when the agents take alternate turns.

[3]It is the sum of the agents' individual utilities.

In this section, the order (i.e. the policy) by which the agents make their decision is fixed and it is the identity permutation (i.e. $\pi = Id_N$ is the identity).

Usually, there are several ways to measure the manipulation in procedures for allocating indivisible goods: the number of flips made on the order of turn, the addition/deletion of agents and the control of some agents' action. Dealing with manipulation or *bribery* in resource allocation, to our best knowledge, only a few results appeared in the literature [5, 26] while some exist in judgment argumentation [3, 13] and many others in voting theory (see surveys [6, 9, 16, 15]). For instance, in voting theory, the complexity of electoral control has been extensively studied for constructive/destructive control by adding/deleting/partitioning candidates/voters [7, 8, 17]. In [12], different models of bribery are studied where the price of each vote depends on the amount of change (also known as *swap bribery* or *shift bribery*) that the voter is asked to implement.

In game theory, the notion of *Stackelberg strategy* is somehow related to manipulation [36]. The set of players is partitioned in two groups: leaders and followers. The leaders play first and keep their strategies fixed. Then the followers play in response to the leaders' strategy in order to maximize their individual payoffs. A Stackelberg equilibrium is reached when no follower can change his strategy and improve his payoff. The goal of the leaders is to guide the game to a desirable outcome (e.g. one that is good for the entire group of players). In algorithmic game theory, Stackelberg strategies are used to induce a socially optimal outcome and this problem is known as the *price of optimum* [30, 28, 14].

7.2 Two manipulation problems

In this section we study the following manipulation problem related to COAP. The problem is called MFA(P) where $P \in \{\mathcal{U}, \mathcal{E}\}$ and the goal is to find a set of forced agents W and an allocation A such that:

- If $i \notin W$, then agent i picks $A(i) = top_i(\bigcup_{j<i} A(j))$ which is i's top item during turn i;
- If $i \in W$, then agent i is forced to pick $A(i)$ which is not i's top item during turn i;
- $W \subseteq N$;
- A is a socially optimal allocation with respect to P (depending on the context, $\mathcal{U}(A) = \mathcal{U}(\hat{A}_{\mathcal{U}})$ or $\mathcal{E}(A) = \mathcal{E}(\hat{A}_{\mathcal{E}})$);
- $|W|$ is minimum.

Because computing a socially optimal allocation is difficult for COAP (see Proposition 1), MFA(P) for COAP is immediately difficult, $P \in \{\mathcal{U}, \mathcal{E}\}$. Our contribution is that MFA(P) is computationally difficult, even in a restricted case of OAP for $P \in \{\mathcal{U}, \mathcal{E}\}$. We focus on instances where the agents' utilities are expressed with *scores*. The allocation of indivisible goods with scores has been studied, for instance, in [2].

An instance of OAP is said to be with k-*approval scores* if every agent i partitions $\mathcal{X}$ into two equivalence classes $\mathcal{X}_i^1$ and $\mathcal{X}_i^0$ such that $\mathcal{X}_i^1 = \{x \in \mathcal{X} : u_i(x) = 1\}$, $\mathcal{X}_i^0 = \{x \in \mathcal{X} : u_i(x) = 0\}$ and $|\mathcal{X}_i^1| = k$. Equivalently, the linear order $\succ_i$ of every agent i satisfies $[\mathcal{X}_i^1 \succ_i \mathcal{X}_i^0]$ and $k = |\mathcal{X}_i^1|$. The special cases $k = 1$ and $k = n - 1$ are named *plurality* and *veto*, respectively. An object x is said to be *approved* by agent i if and only if $x \in \mathcal{X}_i^1$. Note that an item can be approved by several agents.

Let us give an illustration of MFA$(\mathcal{E})$ with 2-approval scores.

Example 6. *Let $\mathcal{X} = \{a, b, c, d\}$ be a set of 4 objects and 3 agents $N = \{1, 2, 3, 4\}$. The preference of the agents are:*

1. $a \succ_1 c \succ_1 b \succ_1 d$
2. $b \succ_2 d \succ_2 c \succ_2 a$
3. $b \succ_3 a \succ_3 c \succ_3 d$
4. $a \succ_4 b \succ_4 c \succ_4 d$

We consider 2-approval scores which means that $\mathcal{X}_1^1 = \{a, c\}$, $\mathcal{X}_2^1 = \{b, d\}$ and $\mathcal{X}_3^1 = \mathcal{X}_4^1 = \{a, b\}$. Obviously, $\mathcal{E}(\hat{A}) = 1$ where for instance $\hat{A}(1) = c$, $\hat{A}(2) = d$, $\hat{A}(3) = a$ and $\hat{A}(4) = b$.

On the other hand, if A^{Id_N} is the allocation returned by SD*, then $A^{Id_N}(1) = a$, $A^{Id_N}(2) = b$, $A^{Id_N}(3) = c$ and $A^{Id_N}(4) = d$. Hence, we get $\mathcal{E}(A^{Id_N}) = 0$. Now, we have to impose the items assigned to the agents in, at least, $W = \{1, 2\}$ with $A(1) = c$ and $A(2) = d$ in order to obtain $\mathcal{E}(A) = 1$.*

An instance of MFA(P) for OAP, with k-approval scores, can be modeled as a bipartite graph $B = (N \cup \mathcal{X}, E)$ where $E = \bigcup_{i \in N}\{(i, x) : x \in \mathcal{X}_i^1\}$. Since $k \geq 1$, the degree of every $i \in N$ in B is at least one. We assume that at least one agent must be forced. For a subset of agents N', $A(N')$ designates $\{A(i) : i \in N'\}$. In particular, dealing with MFA($\mathcal{E}$), we assume that the bipartite graph B has a perfect matching since otherwise, in any allocation A, the poorest agent does not approve its item and then $\mathcal{E}(\hat{A}) = 0$. In that case, we do not force any agent ($W = \emptyset$).

Theorem 5. *$\forall P \in \{\mathcal{U}, \mathcal{E}\}$,* MFA($P$) *is **NP**-hard for 3-approval instances of OAP.*

Proof. The **NP**-completeness result is proved from a reduction of MAX CUT.

MAX CUT	
Input:	A simple graph $G = (V, E)$ of r vertices and an integer k
Solution:	A partition of V into two sets V_1, V_2 called cut $\langle V_1, V_2 \rangle$
Goal:	Deciding whether there is a cut $\langle V_1, V_2 \rangle$ such that the number of edges $\lvert\langle V_1, V_2 \rangle\rvert$ of the cut is at least k.

MAX CUT is known to be **NP**-complete (problem [GT16] page 210 in [22]).

Take an instance (G, k) of MAX CUT in which $V = \{v_1, \ldots, v_r\}$ and $E = \{e_1, \ldots, e_m\}$ (of course $k \leq m$). Build an instance $(N, \mathcal{X}, (\succ_i)_{i \in N})$ of OAP where each agent approves a subset of exactly 3 objects[4]:

- $N = \{a_1, a_2, a_3\} \cup N_V \cup N'_V \cup N_E \cup N'_E$ is the set of $2r+2m+3$ agents where $N_V = \{b_1, \ldots, b_r\}$, $N'_V = \{b'_1, \ldots, b'_r\}$ $N_E = \{c_1, \ldots, c_m\}$ and $N'_E = \{c'_1, \ldots, c'_m\}$. Both b_i and b'_i correspond to the same vertex v_i of V. Both c_q and c'_q correspond to the same edge e_q of E. These agents are ordered in the following manner (permutation π): $b_1, \ldots, b_r$, then $c_1, \ldots, c_m$, then $c'_1, \ldots, c'_m$, then $b'_1, \ldots, b'_r$ and finally a_1, a_2, a_3. The set of objects $\mathcal{X}$ is equal to $V \cup V' \cup E \cup E' \cup \{x_1, x_2, x_3\}$ where $V' = \{v'_1, \ldots, v'_r\}$ and $E' = \{e'_1, \ldots, e'_m\}$.
- The preferences of agents a_1, a_2 and a_3 are $x_1 \succ x_2 \succ x_3$.

[4]We only explain the preference of the first 3 objects for each agent.

- For every vertex $v_i \in V$, the preferences of $b_i \in N_V$ and $b'_i \in N'_V$ are $x_1 \succ v_i \succ v'_i$ and $x_1 \succ v'_i \succ v_i$, respectively.
- For every edge $e_q = (v_i, v_j) \in E$ where $1 \leq i < j \leq r$, the preferences of $c_q \in N_E$ and $c'_q \in N'_E$ are $v_i \succ v'_j \succ e_q$ and $v'_i \succ v_j \succ e'_q$, respectively.

The instance of OAP is clearly built in polynomial time and the corresponding bipartite graph B admits a perfect matching $\{(a_i, x_i) : i = 1, 2, 3\} \cup \{(b_i, v_i), (b'_i, v'_i) : i = 1, \ldots, r\} \cup \{(c_q, e_q), (c'_q, e'_q) : q = 1, \ldots, m\}$, which means that $\mathcal{U}(\hat{A}_{\mathcal{U}}) = |N|$ and $\mathcal{E}(\hat{A}_{\mathcal{E}}) = 1$.

We claim that there is a cut of size at least k in G if and only if, we can force at most $2r+2m-k$ agents for MFA(P) with $P \in \{\mathcal{U}, \mathcal{E}\}$, i.e. $|W| \leq 2r + 2m - k$.

If $\langle V_1, V_2 \rangle$ is a cut of G with $|\langle V_1, V_2 \rangle| \geq k$, then by considering the allocation A given by $A(a_i) = x_i$ for $i = 1, 2, 3$, $[A(b_i) = v_i$ and $A(b'_i) = v'_i$ if $v_i \in V_1]$ and $[A(b_i) = v'_i$ and $A(b'_i) = v_i$ if $v_i \in V_2]$, for $i = 1, \ldots, r$ and finally, $A(c_q) = e_q$ and $A(c'_q) = e'_q$ for $q = 1, \ldots, m$, we have that $\langle W, A \rangle$ is a feasible solution with $|W| = 2r + 2m - |\langle V_1, V_2 \rangle| \leq 2r + 2m - k$.

Conversely, suppose $\langle W, A \rangle$ is a feasible solution with $|W| \leq 2r + 2m - k$ for MFA(P) with $P \in \{\mathcal{U}, \mathcal{E}\}$. We have $\mathcal{U}(\hat{A}_{\mathcal{U}}) = |N|$ and $\mathcal{E}(\hat{A}_{\mathcal{E}}) = 1$ because the bipartite graph B admits a perfect matching. Thus $\{(x, A(x)) : x \in N\}$ must be a perfect matching of B. Observe that every perfect matching of B satisfies:

- $\{a_1, a_2, a_3\}$ is matched with $\{x_1, x_2, x_3\}$;
- $N_E \cup N'_E$ is matched with $E \cup E'$ because only agent c_q approves item e_q (resp., only agent c'_q approves item e'_q);
- $N_V \cup N'_V$ is matched with $V \cup V'$ because they are the remaining vertices.

We claim that if $V_1 := A(N_V) \setminus V'$, then $\langle V_1, V \setminus V_1 \rangle$ is a cut of G of size at least k.

It must be $N_V \cup N'_V \subseteq W$ in order to prevent these agents to pick x_1 during their turn. Agent $c_q \in N_E$ (resp., $c'_q \in N'_E$) is matched with e_q (resp., e'_q) which comes on third position of the agent's preference order. Since the items on first and second positions belong to $V \cup V'$, either c_q (resp., c'_q) is not forced (the first two items are already allocated) or he is forced (one of the first two items is available). Therefore, the only way to have agent c_q (resp., c'_q) out of W is that v_i and v'_j (resp., v'_i and v_j), where $e_q = (v_i, v_j)$, are already allocated. Since N_V and N'_V appear before and after $N_E \cup N'_E$ in the permutation, respectively, c_q (resp., c'_q) is not forced to pick e_q (resp., e'_q) if and only if v_i and v'_j (resp., v'_i and v_j) were previously allocated to agents of N_V.

Concerning agents a_1, a_2 and a_3 who come last in the permutation, none of them needs to be forced because a_1 ranks x_1 first whereas only a_2 approves x_2 and only a_3 approves x_3.

Therefore, the solution to MFA(P) under consideration, which forces (at most) $2m + 2r - k = |N_V \cup N'_V \cup N_E \cup N'_E| - k$ agents, does not force (at least) k agents of $N_E \cup N'_E$. This set of unforced agents are associated with edges of the form (s, t) where $s \in A(N_V) \cap V$ and $t \in A(N_V) \cap V'$.

Thus, $\langle V_1, V \setminus V_1 \rangle$ where $V_1 := A(N_V) \setminus V'$ is a cut of G and its size is at least k. $\square$

In conclusion, MFA(P) is **NP**-hard even for 3-approval instances of OAP where $P \in \{\mathcal{U}, \mathcal{E}\}$. It is interesting to stress out that the negative result of Theorem 5 is also valid for Pareto optimality.

Next subsections are polynomial special cases with approval scores. We solve MFA$(\mathcal{U})$ instances with *plurality* or *veto* and MFA$(\mathcal{E})$ for *plurality* and 2-approval scores. Hence, dealing with MFA$(\mathcal{E})$, we obtain a dichotomic complexity result with respect to k-approval scores with constant k. It is polynomial for $k \leq 2$ and **NP**-complete for $k \geq 3$. All procedures described below are based on the same technique: find a socially optimal matching M of B and then decide which agent is forced.

The bipartite graph $B = (N \cup \mathcal{X}, E)$ is such that $(i, x) \in E$ if, and only if, agent $i \in N$ approves item x (thus, $x \in \mathcal{X}_i^1$ and $u_i(x) = 1$). Given a bipartite graph B, $(\succ_i)_{i \in N}$ and a prescribed maximum matching M of B, PRE-MFA consists of finding a set of forced players W of minimum size such that SD produces an allocation A satisfying $M \subseteq \{(i, A(i)) : i \in N\}$. Note that M is a subset of E, but M is not necessarily a complete allocation. Actually, PRE-MFA is a variant of MFA(P) for $P \in \{\mathcal{U}, \mathcal{E}\}$ where a specified maximum matching is given in advance. Clearly, the optimal resolution of PRE-MFA for all maximum matchings of B provides an optimal solution to MFA(P). Unfortunately, a slight modification of the proof of Theorem 5 allows us to show that PRE-MFA remains **NP**-hard for approval scores.

Given a bipartite graph B and a prescribed maximum matching M of B, Algorithm 3 outputs a set of forced players W and an assignment A such that SD produces A and $M \subseteq \{(i, A(i)) : i \in N\}$. This algorithm runs in linear time. By construction, $W' = W \cap N'$ where N' denotes the subset of players saturated[5] by M and $\mathcal{X}''$ is the set of items defined at Step 14.

Algorithm 3:

Data: A bipartite graph $B = (N \cup \mathcal{X}, E)$ and a target maximum matching
$M = \{(i, M_i) : i \in N'\}$ of B where N' is the subset of agents saturated by M
Result: A set of forced agents $W := W' \cup W''$ and an allocation A

1 $W'' \leftarrow \emptyset$ and $W' \leftarrow \emptyset$; `/* W'' is a waiting list */`
2 $\mathcal{X}' \leftarrow \mathcal{X}$; `/* X' is the set of not yet allocated items */`
3 **for** $i = 1$ *to* n **do**
4   Let j be agent i's most preferred item in $\mathcal{X}'$;
5   **if** $i \in N'$ **then**
6   **if** $j \neq M_i$ **then**
7             $W' \leftarrow W' \cup \{i\}$;
8         $A(i) \leftarrow M_i$ and $\mathcal{X}' \leftarrow \mathcal{X}' \setminus \{M_i\}$;
9   **else**
10   **if** j *is not saturated by* M **then**
11             $A(i) \leftarrow j$ and $\mathcal{X}' \leftarrow \mathcal{X}' \setminus \{j\}$;
12   **else**
13             $W'' \leftarrow W'' \cup \{i\}$;
14 $\mathcal{X}'' \leftarrow \mathcal{X}'$;
15 Complete A by any allocation of W'' to $\mathcal{X}''$;
16 **return** $\langle W := W' \cup W'', A \rangle$

Let us see that Algorithm 3 produces a feasible solution to PRE-MFA.

Property 2. *The following properties hold for the solution returned by Algorithm 3.*

(*i*) $\forall i \notin W$, $A(j) \succ_i A(i)$ *implies* $j < i$;

[5]A node x of B is saturated by M if an edge of M is incident to x.

(ii) $\mathcal{X}''$ *is a subset of items not saturated by* M*.*

Proof. For (i). By contradiction, there exists an agent $i \notin W$ such that $A(j) \succ_i A(i)$ and $j > i$. By construction, $A(j)$ is not allocated during the turn of agent i because $j > i$. If $(i, A(i)) \in M$, then agent i would be forced to pick item $A(i)$, contradiction with $i \notin W$. If $(j, A(j)) \in M$, then agent i would be put in W'' which is a subset of W, contradiction. If neither $(i, A(i))$ nor $(j, A(j))$ belongs to M, then i is assigned his most preferred item, which cannot be $A(i)$ because $A(j) \succ_i A(i)$, contradiction.

For (ii). All the items are initially in $\mathcal{X}'$. During the turn of an agent i saturated by M, M_i is assigned to i and M_i is removed from $\mathcal{X}'$. Thus, in the end of the for loop, no item saturated by M is in $\mathcal{X}'$, and $\mathcal{X}'' \leftarrow \mathcal{X}'$ at this moment. □

Note that if B admits a perfect matching, then PRE-MFA can be solved optimally and the unique optimal solution is produced by Algorithm 3. The proof is obvious, so it is omitted.

Proposition 4. *For every perfect matching* M *of* $B = (N \cup \mathcal{X}, E)$*, Algorithm 3 forces a minimum number of agents such that* $M = \{(i, A(i)) : i \in N\}$*.*

7.3 MFA($\mathcal{U}$) for plurality and veto

7.3.1 Plurality

Here, we suppose that for every agent $i \in N$ there exists exactly one item $f(i)$ such that $u_i(f(i)) = 1$ and $u_i(j) = 0$ for every $j \in \mathcal{X} \setminus \{f(i)\}$. Hence $\mathcal{X}_i^1 = \{f(i)\}$. Agent i only approves alternative $f(i)$ but $f(i)$ can be approved by several agents. An item j is said to be *approved* if there exists $i \in N$ such that $j = f(i)$. According to the next lemma, a social optimum has a special structure in the plurality case.

Lemma 2. *A matching* M *of* B *is of maximum cardinality if, and only if,* M *saturates every approved item.*

Proof. ($\Leftarrow$) The number of approved items is an upper bound on the size of a matching.
($\Rightarrow$) Take a maximum cardinality matching M^* and suppose, by contradiction, that an item j approved by an agent, say i, is not matched. Since i only approves j, i is unmatched and we can increase $|M^*|$ by adding (i, j), contradiction. □

Algorithm 4:

Data: B
Result: A matching M of B

```
M ← ∅ ;
for every j ∈ X do
    if j is approved by at least one agent then
        M ← M ∪ {(i, j)} where i is the agent with minimum index who approves j ;
return M
```

By Lemma 2, Algorithm 4 produces a maximum matching M, and by Property 2, Algorithm 3 with input M produces a feasible solution to $\textsc{mfa}(\mathcal{U})$. Besides, the two algorithms are clearly polynomial in the size of the input.

Given an allocation A, let $\mathcal{M}(A)$ and $\mathcal{M}_1(A)$ denote $\{(i, A(i)) : i \in N\}$ and $\{(i, A(i)) : i \in N \text{ and } u_i(A(i)) = 1\}$, respectively.

Lemma 3. *There exists an optimal solution $\langle W^*, A^* \rangle$ to $\textsc{mfa}(\mathcal{U})$ such that $\mathcal{M}_1(A^*) = M$, where M is the matching returned by Algorithm 4.*

Proof. The proof consists of picking one optimal solution $\langle W^*, A^* \rangle$ and showing that if $\mathcal{M}_1(A^*) \neq M$, then another optimal solution $\langle W^{**}, A^{**} \rangle$ satisfies $|\mathcal{M}_1(A^*) \cap M| < |\mathcal{M}_1(A^{**}) \cap M|$.

Suppose $\mathcal{M}_1(A^*) \neq M$. There exists an item j_r matched with two distinct agents in $\mathcal{M}_1(A^*)$ and M. Let i_1 and i_r be the agents satisfying $(i_r, j_r) \in \mathcal{M}_1(A^*)$ (i.e., $A^*(i_r) = j_r$) and $(i_1, j_r) \in M$. We know that both i_1 and i_r approve j_r. Since in Algorithm 4, j_r is assigned to the agent with smallest index who approves it, we get that $i_1 < i_r$. Let $\{i_2, \ldots, i_{r-1}\}$ be the agents who are between i_1 and i_r in the permutation. We denote by j_k the item allocated to i_k in A^*, i.e. $A^*(i_k) = j_k$, for $1 \leq k \leq r$.

Let $\langle W^{**}, A^{**} \rangle$ be a solution which is identical to $\langle W^*, A^* \rangle$ for every agent and item out of $\{i_1, \ldots, i_r\} \cup \{j_1, \ldots, j_r\}$. Since i_1 approves j_r but he does not get it in A^*, i_1 must be forced to take j_1, i.e. $i_1 \in W^*$ and $(i_1, j_1) \in \mathcal{M}(A^*)$. In $\langle W^{**}, A^{**} \rangle$, i_1 is left free to pick j_r so $i_1 \notin W^{**}$.

For $k = 2$ to $k = r - 1$, we complete $\langle W^{**}, A^{**} \rangle$ as follows. If $i_k \in W^*$, then $i_k \in W^{**}$ and $A^{**}(i_k) = A^*(i_k) = j_k$. Otherwise, i_k is left free to choose his top item. Note that in this case, the top item must be in $\{j_1, \ldots, j_{r-1}\}$. Moreover, an agent i_k who gets his approved item j_k in A^* still gets it in A^{**}.

After the turn of i_{r-1}, only one item in $\{j_1, \ldots, j_{r-1}\}$ remains available (it is not necessarily j_1). Let j' denote this item and force i_r to take it, i.e. $(i_r, j') \in \mathcal{M}(A^{**})$. If j' were the top item of i_r, then $|W^{**}| < |W^*|$, contradicting the optimality of $\langle W^*, A^* \rangle$.

In conclusion, $\langle W^{**}, A^{**} \rangle$ is a feasible solution to $\textsc{mfa}(\mathcal{U})$, it is optimal because $|W^{**}| = |W^*|$ and $|\mathcal{M}_1(A^{**}) \cap M| > |\mathcal{M}_1(A^*) \cap M|$ because of (i_1, j_r). $\square$

Theorem 6. *Algorithm 3 applied to the target matching M returned by Algorithm 4 produces an optimal solution to plurality instances of $\textsc{mfa}(\mathcal{U})$.*

Proof. We know from Lemma 2 that Algorithm 4 returns a matching of maximum cardinality. Therefore, Algorithm 3 returns a solution $\langle W^*, A^* \rangle$ which induces a social optimum M. Lemma 3 states that an optimal solution to $\textsc{mfa}(\mathcal{U})$ inducing M exists. In order to prove that Algorithm 3 is optimal, it remains to prove that W^* is of minimum size within the solutions inducing M.

In Algorithm 3, the set of forced agents W is partitioned in W' and W''. Note that for plurality instances, W' must be empty. Indeed, if $i \in N'$ (see Step 5 of Algorithm 3), then i is matched with $f(i)$, the only item that he approves. $f(i)$ comes first in $\succ_i$ and $f(i)$ cannot be already allocated to another agent because, according to Algorithm 4, $f(i)$ is allocated to the agent with minimum index, within the set of agents who approve $f(i)$. Therefore, only the members of W'' are forced (see Step 13 of Algorithm 3). An agent is put in W'' if, during his turn, his top item is an approved item assigned to another agent whose turn comes later. Thus, every agent of W'' must be forced in any solution inducing M. $\square$

7.3.2 Veto

Let us finish the analysis of polynomial cases for MFA($\mathcal{U}$) by studying *veto* scores, also known as *anti-plurality* or *disapproval*. This case is equivalent to ($|N|-1$)-approval, meaning that every agent approves all items but one (only the last choice in an agent's linear preference is not approved). Recall that during step i of SD, agent i picks his favorite item among the remaining items.

Let $j^* \in \mathcal{X}$ be the item disapproved by agent n, i.e. the linear order of agent n is $[\mathcal{X}\setminus\{j^*\} \succ_n j^*]$. If j^* is disapproved by the entire set of agents, then the optimal social utility is $n-1$ (e.g. $A^*(i) = i$ for $i = 1, \dots, n-1$). In that case, we can run SD without forcing any agent ; we get an allocation with optimal social utility $n-1$. This is true because at least one approved item is available for every agent $i \in [1..n-1]$ during turn i.

Now, suppose j^* is approved by at least one agent. There exists an allocation with social utility n because no item is disapproved by the entire set of agents.

Let i^* be the agent with largest index who approves item j^* ($1 \leq i^* \leq n-1$). Run SD without forcing any agent. If the resulting allocation A has social utility n, then forcing no agent ($W = \emptyset$) is optimal. The last case is when the social utility of the allocation induced by SD is $n-1$ whereas the maximum social utility is n. In any feasible solution, at least one agent is forced, and we claim that it suffices to force agent i^* to pick item j^*. Indeed, during the turn of agent $i \neq i^*$, at least one item approved by i is available.

Proposition 5. MFA($\mathcal{U}$) *is polynomial time solvable for veto scores.*

Note that $|W^*| \leq k$ for k-veto scores. Hence, we can produce a polynomial algorithm (running time of $O(k!n^{2k})$ by exhaustive search) of MFA($\mathcal{U}$) for k-veto instances, where k-veto means that each agent approves exactly $n-k$ objects.

7.4 MFA($\mathcal{E}$) for plurality and 2-approval

Dealing with plurality, we always have $W = \emptyset$ for MFA($\mathcal{E}$) because if $\mathcal{E}(\hat{A}_{\mathcal{E}}) = 0$, then trivially $W = \emptyset$. If $\mathcal{E}(\hat{A}_{\mathcal{E}}) \neq 0$, then $\mathcal{E}(\hat{A}_{\mathcal{E}}) = 1$ because we use approval scores. In that case, each agent approves a distinct object since otherwise a same object is liked by 2 agents or more and then $\mathcal{E}(\hat{A}_{\mathcal{E}}) = 0$. In all cases, $W = \emptyset$. Now, we deal with *2-approval* (i.e. each agent approves exactly two items) for MFA($\mathcal{E}$). Actually, 2-approval means that the bipartite graph $B = (N \cup \mathcal{X}, E)$ satisfies $d_B(i) = 2$ for every agent $i \in N$ where $d_B(v)$ denotes the degree of node v in graph B. Moreover, we can assume that each item is approved at least once, i.e., $\forall j \in \mathcal{X}$, $d_B(j) \geq 1$ since otherwise $\mathcal{E}(\hat{A}_{\mathcal{E}}) = 0$ and then, $W = \emptyset$.Suppose the bipartite graph B has p connected components $B_j = (N_j \cup \mathcal{X}_j, E_j)$, for $j = 1, \dots, p$.

We are going to prove several properties that hold because we can assume $\mathcal{E}(\hat{A}_{\mathcal{E}}) = 1$: (1) $\forall j = 1, \dots, p$, $|N_j| = |\mathcal{X}_j|$, (2) B_i is 2-regular and thus it only admits two perfect matchings and (3) solving MFA($\mathcal{E}$) on $(N, \mathcal{X}, (\succ_i)_{i\in N})$ is equivalent to solve MFA($\mathcal{E}$) independently on each component $(N_j, \mathcal{X}_j, (\succ_i)_{i\in N_j})$ for $j = 1, \dots, p$. Hence, by applying twice Algorithm 3 for PRE-MFA on each bipartite graph B_j, we obtain the expected result.

Property (1) is quite straightforward. If at least one connected component B_j does not contain any perfect matching, then B has no perfect matching and $\mathcal{E}(\hat{A}_{\mathcal{E}}) = 0$.

For (2), consider a connected component B_j of B. First, we apply a preprocessing (Algorithm 5) to show that we can restrict ourselves to a connected 2-regular bipartite graph B'_j, that is a subgraph of B_j.

Algorithm 5:

Data: A connected bipartite graph $B_j = (N_j \cup \mathcal{X}_j, E_j)$ where $|N_j| = |\mathcal{X}_j|$ and $\forall i \in N_j$, $d_{B_j}(i) = 2$

Result: A connected 2-regular bipartite graph $B'_j = (N'_j \cup \mathcal{X}'_j, E'_j)$ and a matching M'_j.

$M'_j \leftarrow \emptyset$, $N'_j \leftarrow N_j$ and $\mathcal{X}'_j \leftarrow \mathcal{X}_j$;

Let B'_j be the subgraph induced by $N'_j \cup \mathcal{X}'_j$;

while *there is* $q \in \mathcal{X}'_j$ *such that* $d_{B'_j}(q) = 1$ **do**

  Let i_q be the unique neighbor of q ;

  $N'_j \leftarrow N'_j \setminus \{i_q\}$, $\mathcal{X}'_j \leftarrow \mathcal{X}'_j \setminus \{q\}$ and $M'_j \leftarrow M'_j \cup \{(q, i_q)\}$;

  Let B'_j be the subgraph induced by $N'_j \cup \mathcal{X}'_j$;

return B'_j and M'_j

Lemma 4. *Algorithm 5 returns a connected 2-regular bipartite graph B'_j. Moreover, M'_j is a matching in the subgraph induced by $(N_j \setminus N'_j) \cup (\mathcal{X}_j \setminus \mathcal{X}'_j)$.*

Proof. By construction, M'_j is a matching in the subgraph induced by $(N_j \setminus N'_j) \cup (\mathcal{X}_j \setminus \mathcal{X}'_j)$. Now, an inductive proof easily shows that at each step the subgraph B'_j induced by $N'_j \cup \mathcal{X}'_j$ is connected, $|N'_j| = |\mathcal{X}'_j|$ and the number of leaves decreases by one unit. Thus, at the end, we obtain a connected bipartite graph B'_j where $d_{B'_j}(i) = 2$ for all $i \in N'_j$ and $d_{B'_j}(q) \geq 2$ for all $q \in \mathcal{X}'_j$. This implies that $d_{B'_j}(q) = 2$ for all $q \in \mathcal{X}'_j$ and then B'_j is a connected 2-regular bipartite graph. □

For (3), we consider the projection of B on B_j which means that the ordering and the preference of the agents of N_j are the restriction of B to N_j and $\mathcal{X}_j$, respectively. For instance, if $N_j = \{j_1, \ldots, j_r\}$ with $j_1 < \cdots < j_r$, then agent j_1 appears first, and then it is the turn of j_2, and so on until the appearance of the last agent j_r. In particular, this implies that any item of $\mathcal{X}_{j'}$ is disapproved by agents of N_j, for $j' \neq j$. Thus, the choice of agents of N_j and the allocation only depends on the bipartite graph B_j whatever the given allocation outside of B_j.

Theorem 7. MFA$(\mathcal{E})$ *is polynomial-time solvable for 2-approval.*

Proof. Using Lemma 4, we know that B'_j is a connected 2-regular bipartite graph and then it can be decomposed into two perfect matchings M^j_1 and M^j_2. Thus, $M^j_1 \cup M'_j$ and $M^j_2 \cup M'_j$ are perfect matchings of B_j. Now, using a simple inductive proof, we can show that any perfect matching of B_j contains M'_j. In conclusion, $M^j_1 \cup M'_j$ and $M^j_2 \cup M'_j$ are the only perfect matchings of B_j. Finally, using Proposition 4, we obtain a polynomial time algorithm by running Algorithm 3 with the two perfect matchings $M^j_1 \cup M'_j$ and $M^j_2 \cup M'_j$, and retain the best solution W^*_j. Finally, by (3), we know that we can solve separately each connected bipartite graph B_j for $j \leq p$ and the final solution is the concatenation of the partial solutions, i.e.. $W^* = \cup_{j=1}^{p} W^*_j$. □

8 Concluding remarks

An extension of the well studied OAP was proposed in this article. We have shown that two important features of OAP extend to MOAP: a social optimum can be computed in polynomial (provided that the agents' utilities for the objects are known) and for every instance, there always exists a permutation such that SD induces a social optimum. Therefore it is natural to ask if these results can be extended to a problem that is more general than MOAP. We conjecture that the existence of an underlying matroid is necessary for these properties to hold.

As a future work, it would be interesting to study other mechanisms than SD.

References

[1] Haris Aziz, Felix Brandt, and Markus Brill. The computational complexity of random serial dictatorship. *Economics Letters*, 121(3):341 – 345, 2013.

[2] Dorothea Baumeister, Sylvain Bouveret, Jérôme Lang, Nhan-Tam Nguyen, Trung Thanh Nguyen, and Jörg Rothe. Scoring rules for the allocation of indivisible goods. In Schaub et al. [33], pages 75–80.

[3] Dorothea Baumeister, Gábor Erdélyi, Olivia Johanna Erdélyi, and Jörg Rothe. Complexity of manipulation and bribery in judgment aggregation for uniform premise-based quota rules. *Mathematical Social Sciences*, 76:19–30, 2015.

[4] Péter Biró, Tamás Fleiner, Robert W. Irving, and David Manlove. The college admissions problem with lower and common quotas. *Theor. Comput. Sci.*, 411(34-36):3136–3153, 2010.

[5] Sylvain Bouveret and Jérôme Lang. Manipulating picking sequences. In Schaub et al. [33], pages 141–146.

[6] Felix Brandt, Vincent Conitzer, and Ulle Endriss. Computational social choice. *Multiagent systems*, pages 213–283, 2012.

[7] Laurent Bulteau, Jiehua Chen, Piotr Faliszewski, Rolf Niedermeier, and Nimrod Talmon. Combinatorial voter control in elections. *Theor. Comput. Sci.*, 589:99–120, 2015.

[8] Jiehua Chen, Piotr Faliszewski, Rolf Niedermeier, and Nimrod Talmon. Elections with few voters: Candidate control can be easy. In Blai Bonet and Sven Koenig, editors, *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA.*, pages 2045–2051. AAAI Press, 2015.

[9] Yann Chevaleyre, Ulle Endriss, Jérôme Lang, and Nicolas Maudet. A short introduction to computational social choice. In Jan van Leeuwen, Giuseppe F. Italiano, Wiebe van der Hoek, Christoph Meinel, Harald Sack, and Frantisek Plasil, editors, *SOFSEM 2007: Theory and Practice of Computer Science, 33rd Conference on Current Trends in Theory and Practice of Computer Science, Harrachov, Czech Republic, January 20-26, 2007, Proceedings*, volume 4362 of *Lecture Notes in Computer Science*, pages 51–69. Springer, 2007.

[10] Anastasia Damamme, Aurélie Beynier, Yann Chevaleyre, and Nicolas Maudet. The power of swap deals in distributed resource allocation. In Gerhard Weiss, Pinar Yolum, Rafael H. Bordini, and Edith Elkind, editors, *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2015, Istanbul, Turkey, May 4-8, 2015*, pages 625–633. ACM, 2015.

[11] J. Edmonds. Matroids and the greedy algorithm. *Mathematical programming*, 1(1):127–136, 1971.

[12] Edith Elkind, Piotr Faliszewski, and Arkadii M. Slinko. Swap bribery. In Marios Mavronicolas and Vicky G. Papadopoulou, editors, *Algorithmic Game Theory, Second International Symposium, SAGT 2009, Paphos, Cyprus, October 18-20, 2009. Proceedings*, volume 5814 of *Lecture Notes in Computer Science*, pages 299–310. Springer, 2009.

[13] Ulle Endriss, Umberto Grandi, and Daniele Porello. Complexity of judgment aggregation. *J. Artif. Intell. Res. (JAIR)*, 45:481–514, 2012.

[14] Bruno Escoffier, Laurent Gourvès, and Jérôme Monnot. The price of optimum in a matching game. In Giuseppe Persiano, editor, *Proceedings of SAGT 2011*, volume 6982 of *LNCS*, pages 81–92. Springer, 2011.

[15] P Faliszewski and J Rothe. Control and bribery in voting. *Handbook of Computational Social Choice, chapter 7 in press*, 2015.

[16] Piotr Faliszewski, Edith Hemaspaandra, and Lane A. Hemaspaandra. Using complexity to protect elections. *Commun. ACM*, 53(11):74–82, 2010.

[17] Piotr Faliszewski, Edith Hemaspaandra, and Lane A. Hemaspaandra. Weighted electoral control. *J. Artif. Intell. Res. (JAIR)*, 52:507–542, 2015.

[18] Aris Filos-Ratsikas, Søren Kristoffer Stiil Frederiksen, and Jie Zhang. Social welfare in one-sided matchings: Random priority and beyond. In Ron Lavi, editor, *Proceedings of SAGT 2014*, volume 8768 of *LNCS*, pages 1–12. Springer, 2014.

[19] Tamás Fleiner and Naoyuki Kamiyama. A matroid approach to stable matchings with lower quotas. In *Proceedings of SODA 2012*, pages 135–142, 2012.

[20] András Frank. A weighted matroid intersection algorithm. *Journal of Algorithms*, 2(4):328–336, 1981.

[21] David Gale and Lloyd S. Shapley. College admissions and the stability of marriage. *The American Mathematical Monthly*, 69(1):9–15, 1962.

[22] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1979.

[23] Masahiro Goto, Naoyuki Hashimoto, Atsushi Iwasaki, Yujiro Kawasaki, Suguru Ueda, Yosuke Yasuda, and Makoto Yokoo. Strategy-proof matching with regional minimum quotas. In *Proceedings of AAMAS 2014*, pages 1225–1232, 2014.

[24] Aanund Hylland and Richard Zeckhauser. The efficient allocation of individuals to positions. *Journal of Political Economy*, 87(2):293 – 314, 1979.

[25] Thomas Kalinowski, Nina Narodytska, and Toby Walsh. A social welfare optimal sequential allocation procedure. In Francesca Rossi, editor, *IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013*. IJCAI/AAAI, 2013.

[26] Thomas Kalinowski, Nina Narodytska, Toby Walsh, and Lirong Xia. Strategic behavior when allocating indivisible goods sequentially. In Marie desJardins and Michael L. Littman, editors, *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence, July 14-18, 2013, Bellevue, Washington, USA*. AAAI Press, 2013.

[27] Yuichiro Kamada and Fuhito Kojima. Efficient matching under distributional constraints: Theory and applications. *American Economic Review*, 105(1):67–99, 2015.

[28] Alexis C. Kaporis and Paul G. Spirakis. The price of optimum in stackelberg games on arbitrary single commodity networks and latency functions. *Theor. Comput. Sci.*, 410(8-10):745–755, 2009.

[29] Bernhard Korte and Jens Vygen. *Combinatorial Optimization: Theory and Algorithms*. Springer Publishing Company, Incorporated, 4th edition, 2007.

[30] Marios Mavronicolas, Vicky Papadopoulou, and Paul G. Spirakis. Algorithmic game theory and applications. In Amiya Nayak and Ivan Stojmenović, editors, *Handbook of Applied Algorithms: Solving Scientific, Engineering, and Practical Problems*, pages 287–315. Wiley-IEEE Press, 2008.

[31] James G. Oxley. *Matroid Theory*. Oxford University Press, 1992.

[32] Daniela Sabán and Jay Sethuraman. The complexity of computing the random priority allocation matrix. In Yiling Chen and Nicole Immorlica, editors, *Proceedings of WINE 2013*, volume 8289 of *LNCS*, page 421. Springer, 2013.

[33] Torsten Schaub, Gerhard Friedrich, and Barry O'Sullivan, editors. *ECAI 2014 - 21st European Conference on Artificial Intelligence, 18-22 August 2014, Prague, Czech Republic - Including Prestigious Applications of Intelligent Systems (PAIS 2014)*, volume 263 of *Frontiers in Artificial Intelligence and Applications*. IOS Press, 2014.

[34] Alexander Schrijver. *Combinatorial Optimization: Polyhedra and Efficiency*. Springer-Verlag Berlin and Heidelberg GmbH & Co., 2003.

[35] Lloyd Shapley and Herbert Scarf. On cores and indivisibility. *Journal of Mathematical Economics*, 1(1):23 – 28, 1979.

[36] Marwaan Simaan and Jose B. Cruz. On the stackelberg strategy in nonzero-sum games. *Journal of Optimization Theory and Applications*, 11(5):533–555, 1973.

[37] Lars-Gunnar Svensson. Queue allocation of indivisible goods. *Social Choice and Welfare*, 11(4):323–330, 1994.

[38] Lirong Xia. Assigning indivisible and categorized items. In *International Symposium on Artificial Intelligence and Mathematics, ISAIM 2014, Fort Lauderdale, FL, USA, January 6-8, 2014*, 2014.

[39] Lin Zhou. On a conjecture by gale about one-sided matching problems. *Journal of Economic Theory*, 52(1):123 – 135, 1990.