Object Allocation Problems Under Constraints

Laurent Gourvès and Carlos A. Martinhon and Jérôme Monnot

Abstract

The *Object Allocation Problem* (OAP) is a well studied problem in which a set $\mathcal{X}$ of n objects is allocated 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 or the utility of the poorest agent. 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.

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.

1 Introduction

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

	central computation of a social optimum	Serial Dictatorship
COAP	**NP**-hard	for some instances, no permutation induces a social optimum
MOAP	polynomial time solvable	at least one permutation induces a social optimum

Table 1: Contribution. All these results hold for utilitarian and egalitarian social welfare.

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.

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. A summary of our contribution is given in Table 1.

2 Related work

In many real-world applications, one tries 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 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 an algorithm to build it was provided by Gale and Shapley [8].

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* [17, 14, 15]. Matroidal extensions of the stable matching problem have already been studied in [2, 6, 10, 13].

A *one-sided market* is also divided in two groups of entities but only one group has preferences over the other group (e.g. families and houses). In [18], 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) [18]. TTC is centralized and mainly based on reallocating resources along potentially long cycles of exchanges. In a recent paper [3], 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 with the *object allocation problem* (OAP), introduced in 1979 by Hylland and Zeckhauser [12] (see also [21, 19]). It is a one-sided market with no endowment; a set of 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 [21] utilizes a random version of SD, called *random serial dictatorship* (RSD) which consists of 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 [16], together with Aziz, Brandt and Brill [1], have recently shown that computing the bi-stochastic matrix P is #P-complete.

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.

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

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 this quantity is a nonnegative real. The *social welfare* of an allocation A is usually measured in two 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).

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 [17, 14, 15] for more details on matroid theory).

3.1 Matroids

This section contains basic notions of matroid theory. Quoting Dan Gusfield [11], *the reader unfamiliar with matroids, but familiar with graphs, can follow most of the paper by specializing the results to the minimum spanning tree problem, substituting graphs for matroids, edges for elements, spanning trees for bases, and cycles for circuits.*

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 $(E, \mathcal{F})$ [17]. 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 [4]. 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 $\sum_{e \in F} w(e)$ [7]. See also [17, 14] 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 $\Phi : T \rightarrow [1..k]$ satisfying

$t \in E_{\Phi(t)}$ for all $t \in T$. Then $(E, \mathcal{F})$ where $\mathcal{F} = \{T \in 2^E : T$ 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. Put differently, there is no loss of generality when we assume that the agents collectively possess a base and not just an independent set.

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 [9]. HC consists in deciding if a given graph has a Hamiltonian cycle. Given an instance $G = (V, E)$ of HC with vertex set $\{1, \ldots, n\}$, build an instance of COAP such that $N = \{1, \ldots, n\}$, $\mathcal{X} = \{(i, j) : 1 \leq i, j \leq n\}$ 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 a 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 $\mathcal{X} = \{x_1, \ldots, x_m\}$ and for every $k \in [m]$, let $Y_k = \{y_k^1, \ldots, y_k^n\}$ 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 $\mathcal{D} = \{D \subseteq \mathcal{Y} : (p(D) \in \mathcal{F}) \wedge (|D \cap Y_k| \leq 1,\, k = 1..m)\}$.

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 $D' \in \mathcal{D}$. $|D' \cap Y_k| \leq 1$ for all k implies $|D \cap Y_k| \leq 1$ 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 [17, 14].

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 Example 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

1 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)
2 Let π be a permutation on N (to be determined)
3 $N_0 \leftarrow N$
4 $j \leftarrow 1$
5 **while** $j \leq |N|$ **do**
6     $\langle \pi, j, N_0 \rangle \leftarrow$ Algorithm 2 $(N, (\mathcal{X}, \mathcal{F}), (\succ_i)_{i \in N}, \pi, j, N_0, \hat{A})$
7   **if** $N_0 \neq \emptyset$ **then**
8         $S' \leftarrow \{\hat{A}(i) : i \in N \setminus N_0\}$
9         $\hat{S} \leftarrow \{\hat{A}(i) : i \in N\}$
10   **if** $\exists i \in N_0$ *such that* $\hat{S} - \hat{A}(i) + top_i(S') \in \mathcal{F}$ **then**
11             $\hat{A}(i) \leftarrow top_i(S')$
12   **else**
13   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 $\mathcal{C}(\hat{S}, top_i(S')))$
14   Take a directed cycle C of G_{ex} of minimum length and let N'_0 be the node set of C
15   **foreach** $i \in N'_0$ **do**
16                 $\hat{A}(i) \leftarrow top_i(S')$
17 **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.*

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

Algorithm 2: Simulated SD

Data: N, a matroid $(\mathcal{X}, \mathcal{F})$ given by its independence oracle, $(\succ_i)_{i \in N}$, π, j, N_0, $\hat{A}$
Result: a permutation on N, an integer and a subset of N

$S' \leftarrow \{\hat{A}(i) : i \in N \setminus N_0\}$
$t \leftarrow j$
while *there exists* $i \in N_0$ *such that* $\hat{A}(i) = top_i(S')$ **do**
  $\pi(t) \leftarrow i$
  $N_0 \leftarrow N_0 - i$
  $t \leftarrow t + 1$
  $S' \leftarrow S' + \hat{A}(i)$
return $\langle \pi, t, N_0 \rangle$

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 [7] (see also [14]).

Theorem 4. *[7] 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 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

[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.

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. □

7 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 time (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 future works, it would be interesting to study other mechanisms than SD and also to consider other definitions of the social welfare (e.g. the ordered weighted averaging aggregation operators [20] because they extend both the utilitarian and the egalitarian social welfares). Another future direction is to study the case where the agents get more than one object. In this respect, we believe that the results of Section 5 extend to the case where the agents receive the same number of objects (see [5] for OAP with this constraint).

Acknowledgement: This work is supported by the project ANR-14-CE24-0007-01 CoCoRICo-CoDec. We thank the reviewers for their useful comments.

References

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

[2] 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.

[3] 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.

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

[5] Diodato Ferraioli, Laurent Gourvès, and Jérôme Monnot. On regular and approximately fair allocations of indivisible goods. In Ana L. C. Bazzan, Michael N. Huhns, Alessio Lomuscio, and Paul Scerri, editors, *International conference on Autonomous Agents and Multi-Agent Systems, AAMAS '14, Paris, France, May 5-9, 2014*, pages 997–1004. IFAAMAS/ACM, 2014.

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

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

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

[9] 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.

[10] 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.

[11] Dan Gusfield. Matroid optimization with the interleaving of two ordered sets. *Discrete Applied Mathematics*, 8(1):41 – 50, 1984.

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

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

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

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

[16] 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.

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

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

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

[20] Ronald R. Yager. On ordered weighted averaging aggregation operators in multicriteria decision making. *IEEE Transactions on Systems, Man and Cybernetics*, 18:183 – 190, 1988.

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

Laurent Gourvès
Université Paris-Dauphine, PSL Research University,
CNRS, UMR [7243], LAMSADE,
75016 PARIS, FRANCE
Email: `laurent.gourves@dauphine.fr`

Carlos A. Martinhon
Fluminense Federal University, Department of Computer Science,
Niterói, RJ, Brazil
Email: `mart@dcc.ic.uff.br`

Jérôme Monnot
Université Paris-Dauphine, PSL Research University,
CNRS, UMR [7243], LAMSADE,
75016 PARIS, FRANCE
Email: `jerome.monnot@dauphine.fr`