Possible and Necessary Allocations via Sequential Mechanisms

Haris Aziz  
NICTA and UNSW,  
Sydney 2033, Australia  
Haris.Aziz@nicta.com.au

Toby Walsh  
NICTA and UNSW,  
Sydney 2033, Australia  
toby.walsh@nicta.com.au

Lirong Xia  
RPI  
NY 12180, USA  
xial@cs.rpi.edu

Abstract

A simple mechanism for allocating indivisible resources is sequential allocation in which agents take turns to pick items. We focus on possible and necessary allocation problems, checking whether allocations of a given form occur in some or all mechanisms for several commonly used classes of sequential allocation mechanisms. In particular, we consider whether a given agent receives a given item, a set of items, or a subset of items for natural classes of sequential allocation mechanisms: balanced, recursively balanced, balanced alternation, and strict alternation. We present characterizations of the allocations that result respectively from the classes, which extend the well-known characterization by Brams and King [2005] for policies without restrictions. In addition, we examine the computational complexity of possible and necessary allocation problems for these classes.

1 Introduction

Efficient and fair allocation of resources is a pressing problem within society today. One important and challenging case is the fair allocation of indivisible items [Chevaleyre et al., 2006, Bouveret and Lang, 2008, Bouveret et al., 2010, Aziz et al., 2014b, Aziz, 2014]. This covers a wide range of problems including the allocation of classes to students, landing slots to airlines, players to teams, and houses to people. A simple but popular mechanism to allocate indivisible items is sequential allocation [Bouveret and Lang, 2011, Brams and Taylor, 1996, Kohler and Chandrasekaran, 1971, Levine and Stange, 2012]. In sequential allocation, agents simply take turns to pick the most preferred item that has not yet been taken. Besides its simplicity, it has a number of advantages including the fact that the mechanism can be implemented in a distributed manner and that agents do not need to submit cardinal utilities. Well-known mechanisms like serial dictatorship [Svensson, 1999] fall under the umbrella of sequential mechanisms.

The sequential allocation mechanism leaves open the particular order over the agents (the so called “policy”) [Kalinowski et al., 2013a, Bouveret and Lang, 2014]. Should it be a balanced policy i.e., each agent gets the same total number of rounds? Or should it be recursively balanced so that agents pick items in phases, and each agent gets one round per phase? Or perhaps it would be fairer to alternate but reverse the order of the agents in successive phases: $a_1 \succ a_2 \succ a_3 \succ a_4 \succ a_5 \succ \ldots$ so that agent $a_1$ takes the first and sixth round? This particular type of policy is used, for example, by the Harvard Business School to allocate courses to students [Budish and Cantillion, 2012] and is referred to as a balanced alternation policy. Another class of policies is strict alternation in which the same ordering is used in each round, such as $a_1 \succ a_2 \succ a_3 \succ a_4 \succ a_5 \succ \ldots$. The sets of balanced alternation and strict alternation policies are subsets of the set of recursively balanced policies which itself is a subset of the set of balanced policies.

We consider here the situation where a policy is chosen from a family of such policies. For example, at the Harvard Business School, a policy is chosen at random from the space of all balanced alternation policies. As a second example, the policy might be left to the discretion of the chair but, for fairness, it is restricted to one of the recursively balanced policies. Despite uncertainty in the policy, we might be interested in the possible or necessary outcomes. For example, can I get my three most preferred courses? Do I necessarily get my two most preferred courses? We examine the complexity of checking such questions. There are several high-stake applications for these results. For example, sequential allocation is used in professional sports ‘drafts’ [Brams and Straffin, 1979]. The precise policy chosen from among the set of admissible policies can critically affect which teams (read agents) get which players (read items).

The problems of checking whether an agent can get some item or set of items in a policy or in all policies is closely related to the problem of ‘control’ of the central organizer. For example, if an agent gets an item in all feasible policies, then it means that the chair cannot ensure that the agent does not get the item. Apart from strategic motivation, the problems we consider also have a design motivation. The central designer may want to consider all feasible policies uniformly at random (as is the case in random serial dictatorship [Aziz et al., 2013, Saban and Sethuraman, 2013]) and use them to find the probability that a certain item or set of items is given to an agent. The probability can be a suggestion of time sharing of an item. The problem of checking whether an agent gets a certain item or set of items in some policy is equiva-
Table 1: Complexity of possible and necessary allocation for sequential allocation. All possible allocation problems are NPC for $k = 1$. All necessary problems are in P for $k = 1$.

<table>
<thead>
<tr>
<th>Problems</th>
<th>Sequential Policy Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Any</td>
<td>Balanced</td>
</tr>
<tr>
<td>POSSIBLEITEM</td>
<td>in P</td>
</tr>
<tr>
<td>NECESSARYITEM</td>
<td>in P</td>
</tr>
<tr>
<td>POSSIBLESET</td>
<td>in P</td>
</tr>
<tr>
<td>NECESSARYSET</td>
<td>in P</td>
</tr>
<tr>
<td>Top-k POSSIBLESET</td>
<td>in P</td>
</tr>
<tr>
<td>Top-k NECESSARYSET</td>
<td>in P</td>
</tr>
<tr>
<td>POSSIBLESUBSET</td>
<td>in P</td>
</tr>
<tr>
<td>NECESSARYSUBSET</td>
<td>in P</td>
</tr>
<tr>
<td>POSSIBLEASSIGNMENT</td>
<td>in P</td>
</tr>
<tr>
<td>NECESSARYASSIGNMENT</td>
<td>in P</td>
</tr>
</tbody>
</table>

lent to checking whether an agent gets a certain item or set of items with zero utility to any agent. Similarly, the problem of checking whether an agent gets a certain item or set of items in all policies is equivalent to checking whether an agent gets a certain item or set of items with probability one.

We let $A = \{a_1, \ldots, a_n\}$ denote a set of $n$ agents, and $I$ denote the set of $m = kn$ items. Let $P = \{P_1, \ldots, P_n\}$ be the profile of agents’ preferences where each $P_i$ is a linear order over $I$. Let $M$ denote an assignment of all items to agents, that is, $M : I \rightarrow A$. We will denote a class of policies by $C$. Any policy $\pi$ specifies $|I|$ rounds of the agents. An agent picks her most preferred item that has not yet been allocated in her rounds.

Example 1. Consider the setting in which $A = \{a_1, a_2\}$, $I = \{b, c, d, e\}$, the preferences of agent $a_1$ are $a_1 b c d e$ and of agent $a_2$ are $b a_2 c d e$. Then for the policy $a_1 \triangleright a_2 \triangleright a_2 \triangleright a_1$, agent $a_1$ gets $\{b, e\}$ whilst agent $a_2$ gets $\{c, d\}$. There are four rounds divided into two phases. Agent $a_3$ picks item $e$ in the second phase.

We consider the following natural computational problems.

(i) POSSIBLEASSIGNMENT: Given $(A, I, P, M)$ and policy class $C$, does there exist a policy in $C$ which results in $M$? ;
(ii) NECESSARYASSIGNMENT: Given $(A, I, P, M)$, and policy class $C$, is $M$ the result of all policies in $C$? ;
(iii) POSSIBLEITEM: Given $(A, I, P, a_j, o)$ where $a_j \in A$ and $o \in I$, and policy class $C$, does there exist a policy in $C$ such that agent $a_j$ gets item $o$? ;
(iv) NECESSARYITEM: Given $(A, I, P, a_j, o)$ where $a_j \in A$ and $o \in I$, and policy class $C$, does agent $a_j$ get item $o$ for all policies in $C$? ;
(v) POSSIBLESET: Given $(A, I, P, a_j, I')$ where $a_j \in A$ and $I' \subseteq I$, and policy class $C$, does there exist a policy in $C$ such that agent $a_j$ gets exactly $I'$? ;
(vi) NECESSARYSET: Given $(A, I, P, a_j, I')$ where $a_j \in A$ and $I' \subseteq I$, and policy class $C$, does agent $a_j$ get exactly $I'$ for all policies in $C$? ;
(vii) POSSIBLESUBSET: Given $(A, I, P, a_j, I')$ where $a_j \in A$ and $I' \subseteq I$, and policy class $C$, does there exist a policy in $C$ such that agent $a_j$ gets $I'$? ;
(viii) NECESSARYSUBSET: Given $(A, I, P, a_j, I')$ where $a_j \in A$ and $I' \subseteq I$, and policy class $C$, does agent $a_j$ get $I'$ for all policies in $C$? ;

This is without loss of generality since we can add dummy items of zero utility to any agent.
When \( n = m \), serial dictatorship is a well-known mechanism in which there is an ordering of agents and with respect to that ordering agents pick the most preferred unallocated item in their turns [Svensson, 1999]. We note that serial dictatorship for \( n = m \) is a balanced, recursively balanced and balanced alternation policy.

2 Characterizations of Outcomes of Sequential Allocation

In this section we provide necessary and sufficient conditions for a given allocation to be the outcome of a balanced policy, recursively balanced policy, or balanced alternation policy. We first define conditions on an allocation \( M \). An allocation is **Pareto optimal** if there is no other allocation in which each item of each agent is replaced by at least as preferred an item and at least one item of some agent is replaced by a more preferred item.

**Condition 1.** \( M \) is Pareto Optimal.

**Condition 2.** \( M \) is balanced.

It is well-known that Condition 1 characterizes outcomes of all sequential allocation mechanisms (without constraints). Brams and King [2005] proved that an assignment is achievable via sequential allocation iff it satisfies Condition 1. The theorem of Brams and King [2005] generalized the characterization of Abdulkadiroğlu and Sonmez [1998] of Pareto optimal assignments as outcomes of serial dictatorships when \( m = n \). We first observe the following simple adaptation of the characterization of Brams and King [2005] to characterize possible outcomes of balanced policies:

**Remark 1.** Given a profile \( P \), an allocation \( M \) is the outcome of a balanced policy if and only if \( M \) satisfies Conditions 1 and 2.

Given a balanced allocation \( M \), for each agent \( a_j \in A \) and each \( 1 \leq j \leq k \), let \( p_j^i \) denote the item that is ranked at the \( i \)-th position by agent \( a_j \) among all items allocated to agent \( a_j \) by \( M \). The third condition requires that for all \( 1 \leq t < s \leq k \), no agent prefers the \( s \)-th ranked item allocated to any other agent to the \( t \)-th ranked item allocated to her.

**Condition 3.** For all \( 1 \leq t < s \leq k \) and all pairs of agents \( a_j, a_j' \), agent \( a_j \) prefers \( p_j^t \) to \( p_j^s \).

The next theorem states that Conditions 1 through 3 characterize outcomes of recursively balanced policies.

**Theorem 1.** Given a profile \( P \), an allocation \( M \) is the outcome of a recursively balanced policy if and only if it satisfies Conditions 1, 2, and 3.

**Proof.** To prove the “if” direction, clearly if \( M \) is the outcome of a recursively balanced policy then Condition 1 and 2 are satisfied. If Condition 3 is not satisfied, then there exists \( 1 \leq t < s \leq k \) and a pair of agents \( a_j, a_j' \) such that agent \( a_j \) prefers \( p_j^t \) to \( p_j^s \). We note that in the round when agent \( a_j \) is about to choose \( p_j^t \) according to \( M \), \( p_j^s \) is still available, because it is allocated by \( M \) in a later round. However, in this case agent \( a_j \) will not choose \( p_j^t \) because it is not her top-ranked available item, which is a contradiction.

To prove the “only if” direction, for any allocation \( M \) that satisfies the three conditions we will construct a recursively balanced policy \( \pi \). For each \( i \leq k = n/m \), we let phase \( i \) denote the \(((i-1)n+1)\)-th round through \( in \)-th round. It follows that for all \( i \leq k \), \( \{p_j^i : j \leq n\} \) are allocated in phase \( i \). Because of Condition 3, \( \{p_j^i : j \leq n\} \) is a Pareto optimal allocation when all items in \( \{p_j^i : i' < i, j \leq n\} \) are removed. Therefore there exists an order \( \pi \) over \( A \) that gives this allocation. Let \( \pi = \pi_1 \triangleright \pi_2 \triangleright \cdots \triangleright \pi_k \). It is not hard to verify that \( \pi \) is recursively balanced and \( M \) is the outcome of \( \pi \).

Given a profile \( P \) and an allocation \( M \) that is the outcome of a recursively balanced policy, that is, it satisfies the three conditions as proved in Theorem 1, we construct a directed graph \( G_M = (A, E) \), where the vertices are the agents, and we add the edges in the following way. For each even \( i \leq k \), we add a directed edge \( a_j \rightarrow a_j' \) if and only if agent \( a_j \) prefers \( p_j^i \) to \( p_j^i \) and the edge is not already in \( G_M \); for each even \( i \leq k \), we add a directed edge \( a_j \rightarrow a_j' \) if and only if agent \( a_j \) prefers \( p_j^i \) to \( p_j^i \) and the edge is not already in \( G_M \).

**Condition 4.** Suppose \( M \) is the outcome of a recursively balanced policy. There is no cycle in \( G_M \).

**Theorem 2.** An allocation \( M \) is achievable by a balanced alternation policy if and only if it satisfies Conditions 1–4.

**Proof.** The “only if” direction: Suppose \( M \) is achievable by a balanced alternation policy \( \pi \). Let \( \pi' \) denote the suborder of \( \pi \) from turn 1 to turn \( n \). Let \( G_{\pi'} = (A, E') \) denote the directed graph where the vertices are the agents and there is an edge \( a_j \rightarrow a_j' \) if and only if \( a_j' \triangleright_{\pi'} a_j \). It is easy to see that \( G_{\pi'} \) is acyclic and complete. We claim that \( G_M \) is a subgraph of \( G_{\pi'} \). For the sake of contradiction suppose there is an edge \( a_j \rightarrow a_j' \) in \( G_M \) but not in \( G_{\pi'} \). If \( a_j \rightarrow a_j' \) is added to \( G_M \) in an odd round \( i \), then it means that agent \( j' \) prefers \( p_j^i \) to \( p_j^i \). Because \( a_j \rightarrow a_j' \) is not in \( G_{\pi'} \), \( a_j' \triangleright_{\pi'} a_j \). This means that right before \( a_j' \) choosing \( p_j^i \) in \( M \), \( p_j^i \) is still available, which contradicts the assumption that \( a_j' \) chooses \( p_j^i \) in \( M \). If \( a_j \rightarrow a_j' \) is added to \( G_M \) in an even round, then following a similar argument we can also derive a contradiction. Therefore, \( G_M \) is a subgraph of \( G_{\pi'} \), which means that \( G_M \) is acyclic.

The “if” direction: Suppose the four conditions are satisfied. Because \( G_M \) has no cycle, we can find a linear order \( \pi' \) over \( A \) such that \( G_{\pi'} \) is a subgraph of \( G_M \). We next prove that \( M \) is achievable by the balanced alternation policy \( \pi \) whose first \( n \) rounds are \( \pi' \). For the sake of contradiction suppose this is not true and let \( t \) denote the earliest round that the allocation in \( \pi \) differs the allocation in \( M \). Let \( a_j \) denote the agent at the \( t \)-th round of \( \pi \), let \( p_j^t \) denote the item she gets at round \( t \) in \( \pi \), and let \( p_j^t \) denote the item that she is supposed to get according to \( M \). Due to Condition 3, \( i' \leq i \). If \( i' < i \) then agent \( a_j \) did not get item \( p_j^t \) in a previous round, which contradicts the selection of \( t \). Therefore \( i' = i \). If \( i \) is odd, then there is an edge \( a_j' \rightarrow a_j \) in \( G_M \), which means that \( a_j' \triangleright_{\pi'} a_j \). This means that \( a_j \) would have chosen \( p_j^i \) in a previous round, which is a contradiction. If \( i \) is even, then a similar contradiction can be derived. Therefore \( M \) is achievable by \( \pi \). □
Given a profile $P$ and an allocation $M$ that is the outcome of a recursively balanced policy, that is, it satisfies the three conditions as proved in Theorem 1, we construct a directed graph $H_M = (A, E)$, where the vertices are the agents, and we add the edges in the following way. For each $j \leq n$ and $i \leq k$, we let $p_i^j$ denote the item that is ranked at the $i$-th position among all items allocated to agent $j$. For each $i \leq k$, we add a directed edge $a_{j'} \rightarrow a_j$ if $j'$ prefers $p_{j'}^i$ to $p_j^i$ if the edge is not already there.

**Condition 5.** If $M$ is the outcome of a recursively balanced policy, then there is no cycle in $H_M$.

**Theorem 3.** An allocation $M$ is achievable by a strict alternation policy if and only if satisfies Condition 1, 2, 3, 4, 5.

**Proof.** The “only if” direction: If $M$ is an outcome of a recursively balanced policy but does not satisfy 5, then this means that there is a cycle in $H_M$. Let agents $a_i$ and $a_j$ be in the cycle. This means that $a_i$ is before $a_j$ in one phase and $a_j$ is before $a_i$ in some other phase.

The “if” direction: Now assume that $M$ is an outcome of a recursively balanced policy but is not alternating. This means that there exist at least two agents $a_i$ and $a_j$ such that $a_i$ comes before $a_j$ in one phase and $a_j$ comes before $a_i$ in some other phase. But this means that there is cycle $a_i \rightarrow a_j \rightarrow a_i$ in graph $H_M$. 

3 General Complexity Results

Before we delve into the complexity results, we observe the following reductions between various problems.

**Lemma 1.** Fixing the policy class to be one of \{all, balanced policies, recursively balanced policies, balanced alternation policies\}, there exist polynomial-time many-one reductions between the following problems: \text{POSSIBLESET to POSSIBLESUBSET}; \text{POSSIBLEITEM to POSSIBLESUBSET}; \text{Top-k POSSIBLESET to POSSIBLESET}; \text{NECESSARYSET to NECESSARYSUBSET}; \text{NECESSARYITEM to NECESSARYSET}; and \text{Top-k NECESSARYSET to NECESSARYSET}.

A polynomial-time many-one reduction from problem $Q$ to problem $Q'$ means that if $Q$ is NP-(coNP)-hard then $Q'$ is also NP-(coNP)-hard, and if $Q'$ is in P then $Q$ is also in P. We also note the following. For $n = 2$, \text{POSSIBLESET and POSSIBLEITEM} are equivalent for any type of policies. Since $n = 2$, the allocation of one agent completely determines the overall assignment.

For $m = n$, checking whether there is a serial dictatorship under which each agent gets exactly one item and a designated agent $a_j$ gets item $o$ is NP-complete [Theorem 2, Saban and Sethuraman, 2013]. They also proved that for $m = n$, checking if for all serial dictators, agent $a_j$ gets item $o$ is in P. Hence, we get the following statements.

**Theorem 4.** \text{POSSIBLEITEM and POSSIBLESET} is NP-complete for balanced, recursively balanced as well as balanced alternation policies.

Theorem 4 does not necessarily hold if we consider the top element or the top $k$ elements. Therefore, we will especially consider top-$k$ \text{POSSIBLESET}. Similarly, we get that for $m = n$, \text{NECESSARYITEM and NECESSARYSET} is polynomial-time solvable for balanced, recursively balanced, and balanced alternation policies.

For arbitrary policies, we first observe that \text{POSSIBLEITEM, NECESSARYITEM and NECESSARYSET} are trivial: \text{POSSIBLEITEM} always has a yes answer (just give all the turns to that agent) and \text{NECESSARYITEM and NECESSARYSET} always have a no answer (just don’t give the agent any turn). Similarly, \text{NECESSARYASSIGNMENT} always has a no answer.

**Theorem 5.** \text{POSSIBLEASSIGNMENT} is polynomial-time solvable for arbitrary policies.

**Proof.** By the characterization of Brams and King [2005], all we need to do is to check whether the assignment is Pareto optimal. It can be checked in polynomial time $O(|I|^2)$ whether a given assignment is Pareto optimal via an extension of a result of Abraham et al. [2005].

There is also a polynomial-time algorithm for \text{POSSIBLESET} for arbitrary policies via a greedy approach.

**Theorem 6.** \text{POSSIBLESET} is polynomial-time solvable for arbitrary policies.

4 Balanced Policies

In contrast to arbitrary policies, \text{POSSIBLEITEM, NECESSARYITEM, NECESSARYSET, and NECESSARYASSIGNMENT} are more interesting for balanced policies since we may be restricted in allocating items to a given agent to ensure balance. Before we consider them, we get the following corollary of Remark 1.

**Corollary 1.** \text{POSSIBLEASSIGNMENT} for balanced assignments is in P.

Note that an assignment is achieved via all balanced policies iff the assignment is the unique balanced assignment that is Pareto optimal. This is only possible if each agent gets his top $k$ items. Hence, we obtain the following.

**Theorem 7.** \text{NECESSARYASSIGNMENT} for balanced assignments is in P.

Compared to \text{NECESSARYASSIGNMENT}, the other ‘necessary’ problems are intractable.

**Theorem 8.** \text{NECESSARYITEM and NECESSARYSUBSET} for balanced policies where $k$ is not fixed is coNP-complete.

**Proof.** Membership in coNP is obvious. By Lemma 1 it suffices to prove that \text{NECESSARYITEM} is coNP-hard, which we will prove by a reduction from \text{POSSIBLEITEM} for $k = 1$, which is NP-complete [Saban and Sethuraman, 2013]. Let $(A, I, P, a_1, o)$ denote an instance of the possible allocation problem for $k = 1$, where $A = \{a_1, \ldots, a_n\}$, $I = \{o_1, \ldots, o_n\}$, $o \in I$, $P = (P_1, \ldots, P_n)$ is the preference profile of the $n$ agents, and we are asked whether it is possible for agent $a_1$ to get item $o$ in some sequential allocation. Given $(A, I, P, a_1, o)$, we construct the following \text{NECESSARYITEM} instance.

**Agents:** $A' = A \cup \{a_{n+1}\}$. 

471
Items: \( I' = I \cup D \cup F_1 \cup \cdots \cup F_n \), where \(|D| = n - 1\) and for each \( a_j \in A \), \(|F_j| = n - 2\). We have \(|I'| = (n + 1)(n-1)\) and \( k = n - 1\).

Preferences:
- The preferences of \( a_1 \) is \([F_1 \succ P_1 \succ \text{others}]\).
- For any \( j \leq n \), the preferences of \( a_j \) are obtained from \([F_j \succ P_j]\) by replacing \( o \) by \( D \), and then add \( o \) to the bottom position.
- The preferences for \( a_{n+1} \) is \([o \succ \text{others}]\).

We are asked whether agent \( a_{n+1} \) always gets item \( o \).

If \((A, I, P, a_1, o)\) has a solution \( \pi \), we show that the \text{NECESSARY ITEM} instance is a “No” instance by considering \( \pi \succ \cdots \succ \pi \succ a_{n+1} \succ \cdots \succ a_n \). In the first \((n - 2)n\)

\[ n-1 \]

\[ n-1 \]

rounds all \( F_j \)'s are allocated to agent \( a_j \)'s. In the following \( n \) rounds \( o \) is allocated to \( a_1 \), which means that \( a_{n+1} \) does not get \( o \).

Suppose the \text{NECESSARY ITEM} instance is a “No” instance and agent \( n + 1 \) does not get \( o \) in a balanced policy \( \pi' \). Because agent \( a_2 \) through \( a_n \) rank \( o \) in their bottom position, \( o \) must be allocated to agent \( a_1 \). Clearly in the first \( n - 2 \) times when agent \( a_1 \) through \( a_n \) choose items, they will choose \( F_1 \) through \( F_n \) respectively. Let \( \pi \) denote the order over which agents \( a_1 \) through \( a_n \) choose items for the last time. We obtain another order \( \pi'' \) over \( A \) from \( \pi \) by moving all agents who choose an item in \( D \) after agent \( a_1 \) while keeping other orders unchanged. It is not hard to see that the outcomes of running \( \pi \) and \( \pi'' \) are the same from the beginning until agent \( a_1 \) gets \( o \). This means that \( \pi'' \) is a solution to \((A, I, P, a_1, o)\).

The problems becomes easier when \( k \) is constant or we are concerned about top \( k \) items.

\textbf{Theorem 9.} For any constant \( k \), \text{NECESSARY SET} and \text{NECESSARY SUBSET} for balanced policies are in \( P \).

\textbf{Proof.} W.l.o.g. given a \text{NECESSARY SET} instance \((A, I, P, a_1, I')\), if \( I' \) is not the top-ranked \( k \) items of agent \( a_1 \) then it is a “No” instance because we can simply let agent \( a_1 \) choose items in the first \( k \) rounds. When \( I' \) is top-ranked \( k \) items of agent \( a_1 \), \((A, I, P, a_1, I')\) is a “No” instance if and only if \((A, I, P, a_1, o)\) is a “No” instance for some \( o \in I' \), which can be checked in polynomial time by Theorem 10. A similar algorithm works for \text{NECESSARY SUBSET}.

\textbf{Theorem 10.} For any constant \( k \), \text{NECESSARY ITEM} for balanced policies is in \( P \).

\textbf{Proof.} Given a \text{NECESSARY ITEM} instance \((A, I, P, a_1, o)\), if \( o \) is ranked below the \( k \)-th position by agent \( a_1 \) then we can return “No”, because by letting agent \( a_1 \) choose in the first \( k \) rounds she does not get item \( o \). Suppose \( o \) is ranked at the \( k \)-th position by agent \( a_1 \) with \( k' \leq k \), the next claim provides an equivalent condition to check whether the \text{NECESSARY ITEM} instance is a “No” instance.

\textbf{Claim 1.} Suppose \( o \) is ranked at the \( k' \)-th position by agent \( a_1 \) with \( k' \leq k \), the \text{NECESSARY ITEM} instance \((A, I, P, a_1, o)\) is a “No” instance if and only if there exists a balanced policy \( \pi \) such that (i) agent \( a_1 \) picks items in the first \( k' \) rounds and the last \( k - k' + 1 \) rounds, and (ii) agent \( a_1 \) does not get \( o \).

Let \( I' \) denote agent \( a_1 \)'s top \( k' \) items. In light of the claim above, to check whether the \((A, I, P, a_1, o)\) is a “No” instance, it suffices to check for every set of \( k - k' + 1 \) items ranked below the \( k' \)-th position by agent \( a_1 \), denoted by \( I' \), whether it is possible for agent \( a_1 \) to get \( I' \) and \( I' \) by a balanced policy where agent \( a_1 \) picks items in the first \( k' \) rounds and the last \( k - k' + 1 \) rounds. To this end, for each \( I' \subseteq I - I^- - \{o\} \) with \(|I'| = k - k' + 1 \), we construct the following maximum flow problem \( F_{I'} \), which can be solved in polynomial-time by e.g. the Ford-Fulkerson algorithm.

\textbf{Algorithm 1:} \text{NECESSARY ITEM} for balanced policies.

\textbf{Input:} A \text{NECESSARY ITEM} instance \((A, I, P, a_1, o)\).

\begin{enumerate}
  \item if \( o \) is ranked below the \( k \)-th position by agent \( a_1 \) then \( \text{return} \ “\text{No}” \).
  \item end
  \item Let \( I' \) denote agent \( a_1 \)'s top \( k' \) items.
  \item for \( I' \subseteq I - I^- - \{o\} \) with \(|I'| = k - k' + 1 \) do \[\text{Algorithm 1:} \text{NECESSARY ITEM} for balanced policies.\]
  \item if \( F_{I'} \) has a solution \( \text{then} \)
  \item \( \text{return} \ “\text{No}” \)
  \item end
  \item end
  \item return \ “Yes”.
\end{enumerate}

\textbf{Theorem 11.} \text{NECESSARY SET} and top-\( k \) \text{NECESSARY SET} for balanced policies are in \( P \) even when \( k \) is not fixed.

\textbf{Proof.} Given an instance of \text{NECESSARY SET}, if the target set is not top-\( k \) then the answer is “No” because we can simply let the agent choose \( k \) items in the first \( k \) rounds. It remains to show that top-\( k \) \text{NECESSARY SET} for balanced policies is in \( P \). That is, given \((A, I, P, a_1)\), we can check in polynomial time whether there is a balanced policy \( \pi \) for which agent \( a_1 \) does not get exactly her top \( k \) items.

For \text{NECESSARY SET}, suppose agent \( a_1 \) does not get her top-\( k \) items under \( \pi \). Let \( \pi' \) denote the order obtained from \( \pi \) by moving all agent \( a_1 \)'s rounds to the end while keeping the other orders unchanged. It is easy to see that agent \( a_1 \) does
not get her top- \( k \) items under \( \pi' \) either. Therefore, \( \text{NECESSARYSET} \) is equivalent to checking whether there exists an order \( \pi \) where agent \( a_1 \) picks item in the last \( k \) rounds so that agent \( a_1 \) does not get at least one of her top- \( k \) items.

We consider an equivalent, reduced allocation instance where the agents are \( \{a_1, a_2, \ldots, a_n\} \), and there are \( k(n-1)+1 \) items \( I' = (I-I^*) \cup \{c\} \), where \( I^* \) is agent \( a_1 \)'s top- \( k \) items. Agent \( a_j \)'s preferences over \( I' \) are obtained from \( P_j \) by replacing the first occurrence of items in \( I^* \) by \( c \), and then removing all items in \( I^* \) while keeping the order of other items the same. We are asked whether there exists an order \( \pi \) where agent \( a_1 \) is the last to pick and \( a_1 \) picks a single item, and each other agents picks \( k \) times, so that agent \( a_1 \) does not get item \( c \). This problem can be solved by a polynomial-time algorithm based on maximum flows that is similar to the algorithm for \( \text{NECESSARYITEM} \) in Theorem 10.

\( \Box \)

5 Recursively Balanced Policies

From Theorem 1, we get the following corollary.

**Corollary 2.** \( \text{POSSIBLEASSIGNMENT} \) for recursively balanced policies is in \( P \).

We also report computational results for problems other than \( \text{POSSIBLEASSIGNMENT} \). The following algorithm works via a greedy approach.

**Theorem 12.** \( \text{NECESSARYASSIGNMENT} \) for recursively balanced policies is in \( P \).

The other ‘necessary problems’ turn out to be computationally intractable.

**Theorem 13.** For all \( k \geq 2 \), \( \text{NECESSARYITEM} \), \( \text{NECESSARYSET} \), \( \text{top-} k \text{-NECESSARYSET} \), and \( \text{NECESSARYSUBSET} \) for recursively balanced policies are coNP-complete.

On the other hand, Top-2 \( \text{POSSIBLESET} \) is easy via a reduction to maximum matching.

**Theorem 14.** Top-2 \( \text{POSSIBLESET} \) for recursively balanced policies is in \( P \) for \( k = 2 \).

Finally, top- \( k \)-\( \text{POSSIBLESET} \) is NP-complete iff \( k \geq 3 \).

**Theorem 15.** For all \( k \geq 3 \), top- \( k \)-\( \text{POSSIBLESET} \) for balanced policies is NP-complete.

6 Strict Alternation Policies

Since there are \( n! \) possible strict alternation policies, so if \( n \) is constant, then all problems can be solved in polynomial time by brute force search. Note that such an argument does not apply to recursively balanced policies. As a result of our characterization of strict alternation outcomes (Theorem 3), we get the following.

**Corollary 3.** \( \text{POSSIBLEASSIGNMENT} \) for strict alternation policies is in \( P \).

We also present other computational results.

**Theorem 16.** \( \text{NECESSARYASSIGNMENT} \) for strict alternation policies is in \( P \).

**Theorem 17.** Top- \( k \)-\( \text{POSSIBLESET} \) for strict alternation policies is in \( P \) for \( k = 2 \).

For Theorem 17, the polynomial-time algorithm is similar to the algorithm for Theorem 14. The next theorems state that the remaining problems are hard to compute. Both theorems are proved by reductions from \( \text{POSSIBLEITEM} \).

**Theorem 18.** For all \( k \geq 3 \), \( \text{POSSIBLESET} \) is NP-complete for strict alternation policies.

**Theorem 19.** For all \( k \geq 2 \), \( \text{NECESSARYITEM} \), \( \text{NECESSARYSET} \), \( \text{top-} k \text{-NECESSARYSET} \), and \( \text{NECESSARYSUBSET} \) are coNP-complete for strict alternation policies.

7 Balanced Alternation Policies

If \( n \) is constant, then all problems can be solved in polynomial time by brute force search. As a result of our characterization of balanced alternation outcomes (Theorem 2), we get the following.

**Corollary 4.** \( \text{POSSIBLEASSIGNMENT} \) for balanced alternation policies is in \( P \).

\( \text{NECESSARYASSIGNMENT} \) can be solved efficiently as well.

**Theorem 20.** \( \text{NECESSARYASSIGNMENT} \) for balanced alternation policies is in \( P \).

We already know that for \( k = m/n = 1 \), top- \( k \) possible and necessary problems can be solved in polynomial time. The next theorems state that for any other \( k \), they are NP-complete for balanced alternation policies. Theorem 21 is proved by a reduction from the EXACT 3-COVER problem and Theorem 22 is proved by a reduction from the \( \text{POSSIBLEITEM} \) problem.

**Theorem 21.** For all \( k \geq 2 \), top- \( k \)-\( \text{POSSIBLESET} \) is NP-complete, \( \text{NECESSARYITEM} \) is coNP-complete, and \( \text{NECESSARYSUBSET} \) is coNP-complete for balanced alternation polices.

**Theorem 22.** For all \( k \geq 2 \), top- \( k \)-\( \text{NECESSARYSET} \) for balanced alternation policies is coNP-complete.

8 Conclusions

We have studied sequential allocation mechanisms where the policy has not been fixed or has been fixed but not announced. We have characterized the allocations achievable with common classes of policies. We have also identified the computational complexity of identifying the possible or necessary items, set or subset of items to be allocated to an agent when using one of the policy classes. There are interesting future directions including considering other common classes of policies, as well as other properties of the outcome like the possible or necessary welfare.

Acknowledgments

We thank anonymous reviewers of AAAI-14 and IJCAI-15 and participants of MATCHUP-15 for helpful comments. Xia acknowledges NSF CAREER under award number IIS-1453542 for support. NICTA is funded by the Australian Government through the Department of Communications and the Australian Research Council through the ICT Centre of Excellence Program.
References


