Group Activity Selection Problem

Andreas Darmann[1], Edith Elkind[2], Sascha Kurz[3], Jérôme Lang[4], Joachim Schauer[1], and Gerhard Woeginger[5]

[1] Universität Graz, Austria
[2] Nanyang Technological University, Singapore
[3] Universität Bayreuth, Germany
[4] LAMSADE – Université Paris-Dauphine, France
[5] TU Eindhoven, The Netherlands

Abstract We consider a setting where one has to organize one or several group activities for a set of agents. Each agent will participate in at most one activity, and her preferences over activities depend on the number of participants in the activity. The goal is to assign agents to activities based on their preferences. We put forward a general model for this setting, which is a natural generalization of anonymous hedonic games. We then focus on a special case of our model, where agents' preferences are binary, i.e., each agent classifies all pairs of the form "(activity, group size)" into ones that are acceptable and ones that are not. We formulate several solution concepts for this scenario, and study them from the computational point of view, providing hardness results for the general case as well as efficient algorithms for settings where agents' preferences satisfy certain natural constraints.

1 Introduction

There are many real-life situations where a group of agents is faced with a choice of multiple activities, and the members of the group have differing preferences over these activities. Sometimes it is feasible for the group to split into smaller subgroups, so that each subgroup can pursue its own activity. Consider, for instance, a workshop whose organizers would like to arrange one or more social activities for the free afternoon.[1] The available activities include a hike, a bus trip, and a table tennis competition. As they will take place simultaneously, each attendee can select at most one activity (or choose not to participate). It is easy enough to elicit the attendees' preferences over the activities, and divide the attendees into groups based on their choices. However, the situation becomes more complicated if one's preferences may depend on the number of other attendees who choose the same activity. For instance, the bus trip has a fixed transportation cost that has to be shared among its participants, which implies that, typically, an attendee i is only willing to go on the bus trip if the number of other participants of the bus trip exceeds a threshold ℓ_i. Similarly, i may only be willing to play table tennis if the number of attendees who signed up for the tournament does *not*

[1] Some of the co-authors of this paper had to deal with this problem when co-organizing a Dagstuhl seminar.

exceed a threshold u_i: as there is only one table, the more participants, the less time each individual spends playing.

Neglecting to take the number of participants of each activity into account may lead to highly undesirable outcomes, such as a bus that is shared by two persons, each of them paying a high cost, and a 48-participant table tennis tournament with one table. Adding constraints on the number of participants for each activity is a practical, but imperfect solution, as the agents' preferences over group sizes may differ: while some attendees (say, senior faculty) may be willing to go on the bus trip with just 4–5 other participants, others (say, graduate students) cannot afford it unless the number of participants exceeds 10. A more fine-grained approach is to elicit the agents' preferences over pairs of the form "(activity, group size)", rather than over activities themselves, and allocate agents to activities based on this information. In general, agents' preferences can be thought of as weak orders over all such pairs, including the pair "(do nothing, 1)", which we will refer to as the *void activity*. A simpler model, which will be the main focus of this paper, assumes that each agents classifies all pairs into ones that are acceptable to him and ones that are not, and if an agent views his current assignment as unacceptable, he prefers (and is allowed) to switch to the void activity (so the assignment is unstable unless it is acceptable to all agents).

The problem of finding a good assignment of agents to activities, which we will refer to as the *Group Activity Selection Problem* (GASP), may be viewed as a mechanism design problem (or, more narrowly, a voting problem) or as a coalition formation problem, depending on whether we expect the agents to act strategically when reporting their preferences. Arguably, in our motivating example the agents are likely to be honest, so throughout the paper we assume that the central authority knows (or, rather, can reliably elicit) the agents' true preferences, and its goal is to find an assignment of players to activities that, informally speaking, is stable and/or maximizes the overall satisfaction. This model is closely related to that of *anonymous hedonic games* [3], where, just as in our setting, players have to split into groups and each player has preferences over possible group sizes. The main difference between anonymous hedonic games and our problem is that, in our setting, the agents' preferences depend not only on the group size, but also on the activity that has been allocated to their group; thus, our model can be seen as a generalization of anonymous hedonic games. On the other hand, we can represent our problem as a general (i.e., non-anonymous) hedonic game [4,3], by creating a dummy agent for each activity and endowing it with suitable preferences (see Section 2.2 for details). However, our setting has useful structural properties that distinguish it from a generic hedonic game: for instance, it allows for succinct representation of players' preferences, and, as we will see, has several natural special cases that admit efficient algorithms for finding good outcomes.

In this paper, we initiate the formal study of GASP. Our goal is to put forward a model for this problem that is expressive enough to capture many real-life activity selection scenarios, yet simple enough to admit efficient procedures for finding good assignments of agents to activities. We describe the basic structure of the problem, and discuss plausible constraints of the number and type of available activities and the structure of agents' preferences. We show that even under a fairly simple preference model (where agents are assumed to approve or disapprove each available alternative) finding an assignment that maximizes the number of satisfied agents is computationally hard; however, we identify several natural special cases of the problem that admit efficient algorithms. We also briefly discuss the issue of stability in our setting.

We do not aim to provide a complete analysis of the group activity selection problem; rather, we view our work as a first step towards understanding the algorithmic and incentive issues that arise in this setting. We hope that our paper will lead to future research on this topic; to facilitate this, throughout the paper we highlight several possible extensions of our model as well as list some problems left open by our work.

2 Formal Model

Definition 1. *An instance of the* Group Activity Selection Problem (GASP) *is given by a set of* agents $N = \{1, \dots, n\}$*, a set of* activities $A = A^* \cup \{a_\emptyset\}$*, where* $A^* = \{a_1, \dots, a_p\}$*, and a* profile P*, which consists of* n votes *(one for each agent):* $P = (V_1, \dots, V_n)$*. The vote of agent* i *describes his preferences over the set of* alternatives $X = X^* \cup \{a_\emptyset\}$*, where* $X^* = A^* \times \{1, \dots, n\}$*; alternative* (a, k)*,* $a \in A^*$*, is interpreted as "activity* a *with* k *participants", and* $a_\emptyset$ *is the* void activity.

The vote V_i *of an agent* $i \in N$ *is (also denoted by* $\succeq_i$*) is a weak order over* X^**; its induced strict preference and indifference relations are denoted by* $\succ_i$ *and* $\sim_i$*, respectively. We set* $S_i = \{(a, k) \in X^* \mid (a, k) \succ_i a_\emptyset\}$*; we say that voter* i approves *of all alternatives in* S_i*, and refer to the set* S_i *as the* induced approval vote *of voter* i*.*

Throughout the paper we will mostly focus on a special case of our problem where no agent is indifferent between the void activity and any other alternative (i.e., for any $i \in N$ *we have* $\{x \in X^* \mid x \sim_i a_\emptyset\} = \emptyset$*), and each agent is indifferent between all the alternatives in* S_i*; we denote this special case of our problem by* a-GASP.

It will be convenient to distinguish between activities that are unique and ones that exist in multiple copies. For instance, if there is a single tennis table and two buses, then we can organize one table tennis tournament, two bus trips (we assume that there is only one potential destination for the bus trip, so these trips are identical), and an unlimited number of hikes (again, we assume that there is only one hiking route). This distinction will be useful for the purposes of complexity analysis: for instance, some of the problems we consider are easy when we have k copies of one activity, but hard when we have k distinct activities. Formally, we say that two activities a and b are *equivalent* if for every agent i and every $j \in \{1, \dots, n\}$ it holds that $(a, j) \sim_i (b, j)$. We say that an activity $a \in A^*$ is k*-copyable* if A^* contains exactly k activities that are equivalent to a (including a itself). We say that a is *simple* if it is 1-copyable; if a is k-copyable for $k \geq n$, we will say that it is ∞*-copyable* (note that we would never need to organize more than n copies of any activity). If some activities in A^* are equivalent, A^* can be represented more succinctly by listing one representative of each equivalence class, together with the number of available copies. However, as long as we make the reasonable assumption that each activity exists in at most n copies, this representation is at most polynomially more succinct.

Our model can be enriched by specifying a set of *constraints* Γ. One constraint that arises frequently in practice is a *global cardinality* constraint, which specifies a bound

K on the number of activities to be organized. More generally, we could also consider more complex constraints on the set of activities that can be organized simultaneously, which can be encoded, e.g., by a propositional formula or a set of linear inequalities. We remark that there can also be external constraints on the number of participants for each activity: for instance, a bus can fit at most 40 people. However, these constraints can be incorporated into agents' preferences, by assuming that all agents view the alternatives that do not satisfy these constraints as unacceptable.

2.1 Special Cases

We now consider some natural restrictions on agents' preferences that may simplify the problem of finding a good assignment. We first need to introduce some additional notation.

Given a vote V_i and an activity $a \in A^*$, let $S_i^{\downarrow a}$ denote the projection of S_i onto $\{a\} \times \{1, \dots, n\}$. That is, we set $S_i^{\downarrow a} = \{k \mid (a, k) \in S_i\}$.

Example 1. Let $A^* = \{a, b\}$ and consider an agent i whose vote V_i is given by

$$(a, 8) \succ_i (a, 7) \sim_i (b, 4) \succ_i (a, 9) \succ_i (b, 3) \succ_i (b, 5) \succ_i (b, 6) \succ_i (a, 6) \succ_i a_\emptyset \succ_i \dots$$

Then $S_i = \{a\} \times [6, 9] \cup \{b\} \times [3, 6]$ and $S_i^{\downarrow a} = \{6, 7, 8, 9\}$.

We are now ready to define two types of restricted preferences for a-GASP that are directly motivated by our running example, namely, *increasing* and *decreasing* preferences. Informally, under increasing preferences an agent prefers to share each activity with as many other participants as possible (e.g., because each activity has an associated cost, which has to be split among the participants), and under decreasing preferences an agent prefers to share each activity with as few other participants as possible (e.g., because each activity involves sharing a limited resource). Of course, an agent's preferences may also be increasing with respect to some activities and decreasing with respect to others, depending on the nature of each activity. We provide a formal definition for a-GASP only; however, it can be extended to GASP in a straightforward way.

Definition 2. *Consider an instance (N, A, P) of* a-GASP. *We say that the preferences of agent i are* increasing (INC) *with respect to an activity $a \in A^*$ if there exists a threshold $\ell_i^a \in \{1, \dots, n+1\}$ such that $S_i^{\downarrow a} = [\ell_i^a, n]$ (where we assume that $[n+1, n] = \emptyset$). Similarly, we say that the preferences of agent i are* decreasing (DEC) *with respect to an activity $a \in A^*$ if there exists a threshold $u_i^a \in \{0, \dots, n\}$ such that $S_i^{\downarrow a} = [1, u_i^a]$ (where we assume that $[1, 0] = \emptyset$).*

We say that an instance (N, A, P) of a-GASP *is* increasing *(respectively,* decreasing*) if the preferences of each agent $i \in N$ are increasing (respectively, decreasing) with respect to each activity $a \in A^*$. We say that an instance (N, A, P) of* a-GASP *is* mixed increasing-decreasing (MIX) *if there exists a set $A^+ \subseteq A^*$ such that for each agent $i \in N$ his preferences are increasing with respect to each $a \in A^+$ and decreasing with respect to each $a \in A^- = A^* \setminus A^+$.*

For some activities, an agent may have both a lower and an upper bound on the acceptable group size: e.g., one may prefer to go on a hike with at least 3 other people, but does not want the group to be too large (so that it can maintain a good pace). In this case, we say that an agent has *interval* (INV) preferences; note that INC/DEC/MIX preferences are a special case of interval preferences.

Definition 3. *Consider an instance (N, A, P) of* a-GASP. *We say that the preferences of agent i are* interval (INV) *if for each $a \in A^*$ there exists a pair of thresholds $\ell_i^a, u_i^a \in \{1, \dots, n\}$ such that $S_i^{\downarrow a} = [\ell_i^a, u_i^a]$ (where we assume that $[i, j] = \emptyset$ for $i > j$).*

Other natural constraints on preferences include restricting the size of S_i (or, more liberally, that of $S_i^{\downarrow a}$ for each $a \in A^*$), or requiring agents to have similar preferences: for instance, one could limit the number of agent *types*, i.e., require that the set of agents can be split into a small number of groups so that the agents in each group have identical preferences. We will not define such constraints formally, but we will indicate if they are satisfied by the instances constructed in the hardness proofs in Section 4.1.

2.2 GASP and Hedonic Games

Recall that a *hedonic game* [3,4] is given by a set of agents N, and, for each agent $i \in N$, a weak order $\geq_i$ over all coalitions (i.e., subsets of N) that include i. That is, in a hedonic game each agent has preferences over coalitions that he can be a part of. A coalition S, $i \in S$, is said to be *unacceptable* for player i if $\{i\} \geq_i S$. A hedonic game is said to be *anonymous* if each agent is indifferent among all coalitions of the same size that include him, i.e., for every $i \in N$ and every $S, T \subseteq N \setminus \{i\}$ such that $|S| = |T|$ it holds that $S \cup \{i\} \geq_i T \cup \{i\}$ and $T \cup \{i\} \geq_i S \cup \{i\}$.

At a first glance, it may seem that the GASP formalism is more general than that of hedonic games, since in GASP the agents care not only about their coalition, but also about the activity they have been assigned to. However, we will now argue that GASP can be embedded into the hedonic games framework.

Given an instance of the GASP problem (N, A, P) with $|N| = n$, where the i-th agent's preferences are given by a weak order $\succeq_i$, we construct a hedonic game $H(N, A, P)$ as follows. We create $n + p$ players; the first n players correspond to agents in N, and the last p players correspond to activities in A^*. The last p players are indifferent among all coalitions. For each $i = 1, \dots, n$, player i ranks every non-singleton coalition with no activity players as unacceptable; similarly, all coalitions with two or more activity players are ranked as unacceptable. The preferences over coalitions with exactly one activity player are derived naturally from the votes: if S, T are two coalitions involving player i, x is the unique activity player in S, and y is the unique activity player in T, then i weakly prefers S to T in $H(N, A, P)$ if and only if $(x, |S| - 1) \succeq_i (y, |T| - 1)$, and i weakly prefers S to $\{i\}$ in $H(N, A, P)$ if and only if $(x, |S| - 1) \succeq_i a_\emptyset$. We emphasize that the resulting hedonic games are not anonymous. Further, while this embedding allows us to apply the standard solution concepts for hedonic games without redefining them, the intuition behind these solution concepts is not always preserved (e.g., because activity players never want to deviate). Therefore, in Section 3, we will provide formal definitions of the relevant hedonic games solution concepts adapted to the setting of a-GASP.

We remark that when A^* consists of a single ∞-copyable activity (i.e., there are n activities in A^*, all of them equivalent to each other), GASP become equivalent to anonymous hedonic games. Such games have been studied in detail by Ballester [2], who provides a number of complexity results for them. In particular, he shows that finding an outcome that is core stable, Nash stable or individually stable (see Section 3 for the definitions of some of these concepts in the context of a-GASP) is NP-hard. Clearly, all these complexity results also hold for GASP. However, they do not directly imply similar hardness results for a-GASP.

3 Solution Concepts

Having discussed the basic model of GASP, as well as a few of its extensions and special cases, we are ready to define what constitutes a solution to this problem.

Definition 4. *An* assignment *for an instance (N, A, P) of* GASP *is a mapping $\pi : N \to A$; $\pi(i) = a_\emptyset$ means that agent i does not participate in any activity. Each assignment naturally partitions the agents into at most $|A|$ groups: we set $\pi^0 = \{i \mid \pi(i) = a_\emptyset\}$ and $\pi^j = \{i \mid \pi(i) = a_j\}$ for $j = 1, \ldots, p$. Given an assignment π, we define a* coalition structure *CS_π over N as follows:*

$$CS_\pi = \left\{\pi^j \mid j = 1, \ldots, p, \pi^j \neq \emptyset\right\} \cup \left\{\{i\} \mid i \in \pi^0\right\}.$$

Clearly, not all assignments are equally desirable. As a minimum requirement, no agent should be assigned to a coalition that he deems unacceptable. More generally, we prefer an assignment to be stable, i.e., no agent (or group of agents) should have an incentive to change its activity. Thus, we will now define several *solution concepts*, i.e., classes of desirable assignments. We will state our definitions for a-GASP only, though all of them can be extended to the more general case of GASP in a natural way. Given the connection to hedonic games pointed out in Section 2.2, we will proceed by adapting the standard hedonic game solution concepts to our setting; however, this has to be done carefully to preserve intuition that is specific to our model.

The first solution concept that we will consider is *individual rationality*.

Definition 5. *Given an instance (N, A, P) of* a-GASP*, an assignment $\pi : N \to A$ is said to be* individually rational *if for every $j > 0$ and every agent $i \in \pi^j$ it holds that $(a_j, |\pi^j|) \in S_i$.*

Clearly, if an assignment is not individually rational, there exists an agent that can benefit from abandoning his coalition in favor of the void activity. Further, an individually rational assignment always exists: for instance, we can set $\pi(i) = a_\emptyset$ for all $i \in N$. However, a benevolent central authority would usually want to maximize the number of agents that are assigned to non-void activities. Formally, let $\#(\pi) = |\{i \mid \pi(i) \neq a_\emptyset\}|$ denote the number of agents assigned to a non-void activity. We say that π is *maximum individually rational* if π is individually rational and $\#(\pi) \geq \#(\pi')$ for every individually rational assignment π'. Further, we say that π is *perfect*[2] if $\#(\pi) = n$. We denote

[2] The terminological similarity with the notion of perfect partition in a hedonic game [1] is not a coincidence; there a perfect partition assigns each agent to her preferred coalition; here a perfect assignment assigns each agent to one of her equally preferred alternatives.

the size of a maximum individually rational assignment for an instance (N, A, P) by $\#(N, A, P)$. In Section 4, we study the complexity of computing a perfect or maximum individually rational assignment for a-GASP, both for the general model and for the special cases defined in Section 2.1.

Besides individual rationality, there is a number of solution concepts for hedonic games that aim to capture stability against individual or group deviations, such as Nash stability, individual stability, contractual individual stability, and (weak and strong) core stability (see, e.g., [5]). In what follows, due to lack of space, we only provide the formal definition (and some results) for Nash stability. We briefly discuss how to adapt other notions of stability to our setting, but we leave the detailed study of their algorithmic properties as a topic for future work.

Definition 6. *Given an instance (N, A, P) of* a-GASP*, an assignment $\pi : N \to A$ is said to be* Nash stable *if it is individually rational and for every agent $i \in N$ such that $\pi(i) = a_\emptyset$ and every $a_j \in A^*$ it holds that $(a_j, |\pi^j| + 1) \notin S_i$.*

If π is not Nash stable, then there is an agent assigned to the void activity who wants to join a group that is engaged in a non-void activity, i.e., he would have approved of the size of this group and its activity choice if he was one of them. Note that a perfect assignment is Nash stable. The reader can verify that our definition is a direct adaptation of the notion of Nash stability in hedonic games: if an assignment is individually rational, the only agents who can profitably deviate are the ones assigned to the void activity. The requirement of Nash stability is much stronger than that of individual rationality, and there are cases where a Nash stable assignment does not exist.

Proposition 1. *For each $n \geq 2$, there exists an instance (N, A, P) of* a-GASP *with $|N| = n$ that does not admit a Nash stable assignment. This holds even if $|A^*| = 1$ and all agents have interval preferences.*

Proof. Consider an instance (N, A, P) of a-GASP with $A^* = \{a\}$ and induced approval votes given by $S_1 = \{(a, 1)\}$, $S_2 = \{(a, 2)\}$ and $S_i = \emptyset$ for all $i \geq 3$; note that all approved sets are intervals. Whichever assignment π is chosen, either π is not individually rational or agent 2 wants to join a. □

In Definition 6 an agent is allowed to join a coalition even if the members of this coalition are opposed to this. In contrast, the notion of *individual stability* only allows a player to join a group if none of the existing group members objects. We remark that if all agents have increasing preferences, individual stability is equivalent to Nash stability: no group of players would object to having new members join.

A related hedonic games solution concept is *contractual individual stability*: under this concept, an agent is only allowed to move from one coalition to another if neither the members of his new coalition nor the members of his old coalition object to the move. However, for a-GASP contractual individual stability is equivalent to individual stability. Indeed, in our model no agent assigned to a non-void activity has an incentive to deviate, so we only need to consider deviations from singleton coalitions.

The solution concepts discussed so far deal with individual deviations; resistance to group deviations is captured by the notion of the *core*. One typically distinguishes between *strong* group deviations, which are beneficial for each member of the deviating group, and *weak* group deviations, where the deviation should be beneficial for at least one member of the deviating group and non-harmful for others; these notions of deviation correspond to, respectively, *weak* and *strong* core. We note that in the context of a-GASP strong group deviations amount to players in π^0 forming a coalition in order to engage in a non-void activity. This observation immediately implies that every instance of a-GASP has a non-empty weak core, and an outcome in the weak core can be constructed by a natural greedy algorithm; we omit the details due to space constraints.

4 Computing Good Outcomes

In this section, we consider the computational complexity of finding a "good" assignment for a-GASP. We mostly focus on finding perfect or maximum individually rational assignments; towards the end of the section, we also consider Nash stability. Besides the general case of our problem, we consider special cases obtained by combining constraints on the number and type of activities (e.g., unlimited number of simple activities, a constant number of copyable activities, etc.) and constraints on voters' preferences (INC, DEC, INV, etc.). Note that if we can find a maximum individually rational assignment, we can easily check if a perfect assignment exists, by looking at the size of our maximum individually rational assignment. Thus, we will state our hardness results for the "easier" perfect assignment problem and phrase our polynomial-time algorithms in terms of the "harder" problem of finding a maximum individually rational assignment.

4.1 Individual Rationality: Hardness Results

We start by presenting four NP-completeness results, which show that finding a perfect assignment is hard even under fairly strong constraints on preferences and activities. We remark that this problem is obviously in NP, so in what follows we will only provide the hardness proofs. The full proofs of all results in this section can be found in Appendix A.

Our first hardness result applies when all activities are simple and the agents' preferences are increasing.

Theorem 1. *It is* NP*-complete to decide whether* a-GASP *admits a perfect assignment, even when all activities in* A^* *are simple and all agents have increasing preferences.*

Proof (sketch). We provide a reduction from EXACT COVER BY 3-SETS (X3C). Recall that an instance of X3C is a pair $\langle X, \mathcal{Y} \rangle$, where $X = \{1, \dots, 3q\}$ and $\mathcal{Y} = \{Y_1, \dots, Y_p\}$ is a collection of 3-element subsets of X; it is a "yes"-instance if X can be covered by exactly q sets from $\mathcal{Y}$, and a "no"-instance otherwise. Given an instance $\langle X, \mathcal{Y} \rangle$ of X3C, we construct an instance of a-GASP as follows. We set $N = \{1, \dots, 3q\}$ and $A^* = \{a_1, \dots, a_p\}$. For each agent i, we define his vote V_i so that the induced approval vote S_i is given by $S_i = \{(a_j, k) \mid i \in Y_j, k \geq 3\}$, and let $P = (V_1, \dots, V_n)$. Clearly, (N, A, P) is an instance of a-GASP with increasing preferences. It is not hard to check that $\langle X, \mathcal{Y} \rangle$ is a "yes"-instance of X3C if and only if (N, A, P) admits a perfect assignment. □

Our second hardness result applies to simple activities and decreasing preferences, and holds even if each agent is willing to share each activity with at most one other agent.

Theorem 2. *It is* NP*-complete to decide whether* a-GASP *admits a perfect assignment, even when all activities in A^* are simple, all agents have decreasing preferences, and, moreover, for every agent $i \in N$ and every alternative $a \in A^*$ we have $S_i^{\downarrow a} \subseteq \{1, 2\}$.*

Proof (sketch). Consider the following restricted variant of the problem of scheduling on unrelated machines. There are n jobs and p machines. An instance of the problem is given by a collection of numbers $\{p_{ij} \mid i = 1, \ldots, n, j = 1, \ldots, p\}$, where p_{ij} is the running time of job i on machine j, and $p_{ij} \in \{1, 2, +\infty\}$ for every $i = 1, \ldots, n$ and every $j = 1, \ldots, p$. It is a "yes"-instance if there is a mapping $\rho : \{1, \ldots, n\} \to \{1, \ldots, p\}$ assigning jobs to machines so that the makespan is at most 2, i.e., for each $j = 1, \ldots, p$ it holds that $\sum_{i:\rho(i)=j} p_{ij} \leq 2$. This problem is known to be NP-hard (see the proof of Theorem 5 in [7]).

Given an instance $\{p_{ij} \mid i = 1, \ldots, n, j = 1, \ldots, p\}$ of this problem, we construct an instance of a-GASP as follows. We set $N = \{1, \ldots, n\}$, $A^* = \{a_1, \ldots, a_p\}$. Further, for each agent $i \in N$ we construct a vote V_i so that the induced approval vote S_i satisfies $S_i^{\downarrow a_j} = \{1\}$ if $p_{ij} = 2$, $S_i^{\downarrow a_j} = \{1, 2\}$ if $p_{ij} = 1$, and $S_i^{\downarrow a_j} = \emptyset$ if $p_{ij} = +\infty$. Clearly, these preferences satisfy the constraints in the statement of the theorem, and it can be shown that a perfect assignment for (N, A, P) corresponds to a schedule with makespan of at most 2, and vice versa. □

Our third hardness result also concerns simple activities and decreasing preferences. However, unlike Theorem 2, it holds even if each agent approves of at most 3 activities. The proof proceeds by a reduction from MONOTONE 3-SAT.

Theorem 3. *It is* NP*-complete to decide whether* a-GASP *admits a perfect assignment, even when all activities in A^* are simple, all agents have decreasing preferences, and, moreover, for every agent $i \in N$ it holds that $|\{a \mid S_i^{\downarrow a} \neq \emptyset\}| \leq 3$.*

Our fourth hardness result applies even when there is only one activity, which is ∞-copyable, and every agent approves at most two alternatives; however, the agents' preferences constructed in our proof do not satisfy any of the structural constraints defined in Section 2.1. The proof proceeds by a reduction from X3C.

Theorem 4. *It is* NP*-complete to decide whether* a-GASP *admits a perfect assignment, even when all activities in A^* are equivalent and for every $i \in N$ (i.e., A^* consists of a single ∞-copyable activity a) and for every $i \in N$ we have $|S_i^{\downarrow a}| \leq 2$.*

4.2 Individual Rationality: Easiness Results

The hardness results in Section 4.1 imply that if A^* contains an unbounded number of distinct activities, finding a maximum individually rational assignment is computationally hard, even under strong restrictions on agents' preferences (such as INC or DEC). Thus, we can only hope to develop an efficient algorithm for this problem if we assume that the total number of activities is small (i.e., bounded by a constant) or, more liberally, that the number of pairwise non-equivalent activities is small, and the agents' preferences satisfy additional constraints. We will now consider both of these settings, starting with the case where $p = |A^*|$ is bounded by a constant.

Theorem 5. *There exist an algorithm that given an instance of* a-GASP *finds a maximum individually rational assignment and runs in time* $(n+1)^p \text{poly}(n)$.

Proof. We will check, for each $r = 0, \dots, n$, if there is an individually rational assignment π with $\#(\pi) = r$, and output the maximum value of r for which this is the case. Fix an $r \in \{0, \dots, n\}$. For every vector $(n_1, \dots, n_p) \in \{0, \dots, n\}^p$ that satisfies $n_1 + \dots + n_p = r$ we will check if there exists an assignment of agents to activities such that for each $j = 1, \dots, p$ exactly n_j agents are assigned to activity a_j (with the remaining agents being assigned to the void activity), and each agent approves of the resulting assignment. Each check will take $\text{poly}(n)$ steps, and there are at most $(n+1)^p$ vectors to be checked; this implies our bound on the running time of our algorithm.

For a fixed vector $(n_1, \dots, n_p)$, we construct an instance of the network flow problem as follows. Our network has a source s, a sink t, a node i for each player $i = 1, \dots, n$, and a node a_j for each $a_j \in A^*$. There is an arc of unit capacity from s to each agent, and an arc of capacity n_j from node a_j to the sink. Further, there is an arc of unit capacity from i to a_j if and only if $(a_j, n_j) \in S_i$. It is not hard to see that an integral flow F of size r in this network corresponds to an individually rational assignment of size r. It remains to observe that it can be checked in polynomial time whether a given network admits a flow of a given size. □

Moreover, when A^* consists of a single simple activity a, a maximum individually rational assignment can be found by a straightforward greedy algorithm.

Proposition 2. *Given an instance* (N, A, P) *of* a-GASP *with* $A^* = \{a\}$*, we can find a maximum individually rational assignment for* (N, A, P) *in time* $O(s \log s)$*, where* $s = \sum_{i \in N} |S_i|$.

Proof. Clearly, (N, A, P) admits an individually rational assignment π with $\#(\pi) = k$ if and only if $|\{i \mid (a,k) \in S_i\}| \geq k$. Let $\mathcal{R} = \{(i,k) \mid (a,k) \in S_i\}$; note that $|\mathcal{R}| = s$. We can sort the elements of $\mathcal{R}$ in descending order with respect to their second coordinate in time $O(s \log s)$. Now we can scan $\mathcal{R}$ left to right in order to find the largest value of k such that $\mathcal{R}$ contains at least k pairs that have k as their second coordinate; this requires a single pass through the sorted list. □

Now, suppose that A^* contains many activities, but most of them are equivalent to each other; for instance, A^* may consist of a single k-copyable activity, for a large value of k. Then the algorithm described in the proof of Theorem 5 is no longer efficient, but this setting still appears to be more tractable than the one with many distinct activities. Of course, by Theorem 4, in the absence of any restrictions on the agents' preferences, finding a maximum individually rational assignment is hard even for a single ∞-copyable activity. However, we will now show that this problem becomes easy if we additionally assume that the agents' preferences are increasing or decreasing.

Observe first that for increasing preferences having multiple copies of the same activity is not useful: if there is an individually rational assignment where agents are assigned to multiple copies of an activity, we can reassign these agents to a single copy of this activity without violating individual rationality. Thus, we obtain the following easy corollary to Theorem 5.

Corollary 1. *Let* (N, A, P) *be an instance of* a-GASP *with increasing preferences where* A^* *contains at most* K *activities that are not pairwise equivalent. Then we can find a maximum individually rational assignment for* (N, A, P) *in time* $n^K\text{poly}(n)$.

If all preferences are decreasing, we can simply eliminate all ∞-copyable activities. Indeed, consider an instance (N, A, P) of a-GASP where some activity $a \in A^*$ is ∞-copyable. Then we can assign each agent $i \in N$ such that $(a, 1) \in S_i$ to his own copy of a; clearly, this will only simplify the problem of assigning the remaining agents to the activities.

It remains to consider the case where the agents' preferences are decreasing, there is a limited number of copies of each activity, and the number of distinct activities is small. While we do not have a complete solution for this case, we can show that in the case of a single k-copyable activity a natural greedy algorithm succeeds in finding a maximum individually rational assignment.

Theorem 6. *Given a decreasing instance* (N, A, P) *of* a-GASP *where* A^* *consists of a single* k*-copyable activity (i.e.,* $A^* = \{a_1, \ldots, a_k\}$*, and all activities in* A^* *are pairwise equivalent), we can find a maximum individually rational assignment in time* $O(n \log n)$.

Proof. Since all activities in A^* are pairwise equivalent, we can associate each agent $i \in N$ with a single number $u_i \in \{0, \ldots, n\}$, which is the maximum size of a coalition assigned to a non-void activity that he is willing to be a part of. We will show that our problem can be solved by a simple greedy algorithm. Specifically, we sort the agents in non-increasing order of u_is. From now on, we will assume without loss of generality that $u_1 \geq \cdots \geq u_n$. To form the first group, we find the largest value of i such that $u_i \geq i$, and assign agents $1, \ldots, i$ to the first copy of the activity. In other words, we continue adding agents to the group as long as the agents are happy to join. We repeat this procedure with the remaining agents until either k groups have been formed or all agents have been assigned to one of the groups, whichever happens earlier.

Clearly, the sorting step is the bottleneck of this procedure, so the running time of our algorithm is $O(n \log n)$. It remains to argue that it produces a maximum individually rational assignment. To show this, we start with an arbitrary maximum individually rational assignment π and transform it into the one produced by our algorithm without lowering the number of agents that have been assigned to a non-void activity. We will assume without loss of generality that π assigns all k copies of the activity (even though this is is not necessarily the case for the greedy algorithm).

First, suppose that $\pi(i) = a_\emptyset$, $\pi(j) = a_\ell$ for some $i < j$ and some $\ell \in \{1, \ldots, k\}$. Then we can modify π by setting $\pi(i) = a_\ell$, $\pi(j) = a_\emptyset$. Since $i < j$ implies $u_i \geq u_j$, the modified assignment is individually rational. By applying this operation repeatedly, we can assume that the set of agents assigned to a non-void activity forms a contiguous prefix of $1, \ldots, n$.

Next, we will ensure that for each $\ell = 1, \ldots, k$ the group of agents that are assigned to a_ℓ forms a contiguous subsequence of $1, \ldots, n$. To this end, let us sort the coalitions in π according to their size, from the largest to the smallest, breaking ties arbitrarily. That is, we reassign the k copies of our activity to coalitions in π so that $\ell < r$ implies $|\pi^\ell| \geq |\pi^r|$. Now, suppose that there exist a pair of players i, j such that $i < j$, $\pi(i) = a_\ell$, $\pi(j) = a_r$, and $\ell > r$ (and hence $|\pi^\ell| \leq |\pi^r|$). We have $u_j \geq |\pi^r| \geq |\pi^\ell|$, $u_i \geq u_j \geq |\pi^r|$, so if we swap i and j (i.e., modify π by setting $\pi(j) = a_\ell$, $\pi(i) = a_r$), the resulting assignment remains individually rational. Observe that every such swap increases the quantity $\Sigma = \sum_{t=1}^{k} \sum_{s \in \pi^t} (s \cdot t)$ by at least 1: prior to the swap, the contribution of i and j to Σ is $i\ell + jr$, ans after the swap it is $ir + j\ell > i\ell + jr$. Since for any assignment we have $\Sigma \leq kn(n+1)/2$, eventually we arrive to an assignment where no such pair (i, j) exists. At this point, each π^ℓ, $\ell = 1, \dots, k$, forms a contiguous subsequence of $1, \dots, n$, and, moreover, $\ell < r$ implies $i \leq j$ for all $i \in \pi^\ell$, $j \in \pi^r$.

Now, consider the smallest value of ℓ such that π^ℓ differs from the ℓ-th coalition constructed by the greedy algorithm (let us denote it by γ^ℓ), and let i be the first agent in $\pi^{\ell+1}$. The description of the greedy algorithm implies that π^ℓ is a strict subset of γ^ℓ and agent i belongs to γ^ℓ. Thus, if we modify π by moving agent i to π^ℓ, the resulting allocation remains individually rational (since i is happy in γ^ℓ). By repeating this step, we will gradually transform π into the output of the greedy algorithm (possibly discarding some copies of the activity). This completes the proof. □

The algorithm described in the proof of Theorem 6 can be extended to the case where we have one k-copyable activity a and one simple activity b, and the agents have decreasing preferences over both activities. For each $s = 1, \dots, n$ we will look for the best solution in which s players are assigned to b; we will then pick the best of these n solutions. For a fixed s let $N_s = \{i \in N \mid (b, s) \in S_i\}$. If $|N_s| < s$, no solution for this value of s exists. Otherwise, we have to decide which size-s subset of N_s to assign to b. It is not hard to see that we should simply pick the agents in N_s that have the lowest level of tolerance for a, i.e., we order the agents in N_s by the values of u_i^a from the smallest to the largest, and pick the first s agents. We then assign the remaining agents to copies of a using the algorithm from the proof of Theorem 6. Indeed, any assignment can be transformed into one of this form by swapping agents so that the individual rationality constraints are not broken. It would be interesting to see if this idea can be extended to the case where instead of a single simple activity b we have a constant number of simple activities or a single k'-copyable activity.

We conclude this section by giving an $O(\sqrt{n})$-approximation algorithm for finding a maximum individually rational assignment in a-GASP with a single ∞-copyable activity.

Theorem 7. *There exists a polynomial-time algorithm that given an instance* (N, A, P) *of* a-GASP *where* A^* *consists of a single* ∞*-copyable activity* a*, outputs an individually rational assignment* π *with* $\#(\pi) = \Theta(\frac{1}{\sqrt{n}})\#(N, A, P)$.

Proof. We say that an agent i is *active* in π if $\pi(i) \neq a_\emptyset$; a coalition of agents is said to be *active* if it is assigned to a single copy of a. We construct an individually rational assignment π iteratively, starting from the assignment where no agent is active. Let $N^* = \{i \mid \pi(i) = a_\emptyset\}$ be the current set of inactive agents (initially, we set $N^* = N$). At each step, we find the largest subset of N^* that can be assigned to a single fresh copy of a without breaking the individual rationality constraints, and append this assignment to π. We repeat this step until the inactive agents cannot form another coalition.

Now we compare the number of active agents in π with the number of active agents in a maximum individually rational assignment π^*. To this end, let us denote the active coalitions of π by $B_1, \dots, B_s$, where $|B_1| \geq \dots \geq |B_s|$. If $|B_1| \geq \sqrt{n}$, we are done, so assume that this is not the case. Note that since B_1 was chosen greedily, this implies that $|C| \leq \sqrt{n}$ for every active coalition C in π^*.

Let $\mathcal{C}$ be the set of active coalitions in π^*. We partition $\mathcal{C}$ into s groups by setting $\mathcal{C}^1 = \{C \in \mathcal{C} \mid C \cap B_1 \neq \emptyset\}$ and $\mathcal{C}^i = \{C \in \mathcal{C} \mid C \cap B_i \neq \emptyset, C \notin \mathcal{C}^j \text{ for } j < i\}$ for $i = 2, \dots, s$. Note that every active coalition $C \in \pi^*$ intersects some coalition in π: otherwise we could add C to π. Therefore, each active coalition in π^* belongs to one of the sets $\mathcal{C}^1, \dots, \mathcal{C}^s$. Also, by construction, the sets $\mathcal{C}^1, \dots, \mathcal{C}^s$ are pairwise disjoint. Further, since the coalitions in $\mathcal{C}^i$ are pairwise disjoint and each of them intersects B_i, we have $|\mathcal{C}^i| \leq |B_i|$ for each $i = 1, \dots, s$. Thus, we obtain

$$\begin{aligned}
\#(\pi^*) = \sum_{i=1,\dots,s} \sum_{C \in \mathcal{C}^i} |C| &\leq \sum_{i=1,\dots,s} \sum_{C \in \mathcal{C}^i} \sqrt{n} \\
&\leq \sum_{i=1,\dots,s} |\mathcal{C}^i| \sqrt{n} \leq \sum_{i=1,\dots,s} |B_i| \sqrt{n} \leq \#(\pi)\sqrt{n}.
\end{aligned}$$

□

4.3 Nash Stability

We have shown that a-GASP does not not always admit a Nash stable assignment (Proposition 1). In fact, it is difficult to determine whether a Nash stable assignment exists; we relegate the proof to Appendix B due to space constraints.

Theorem 8. *It is* NP*-complete to decide whether* a-GASP *admits a Nash stable assignment.*

However, if agents' preferences satisfy INC, DEC, or MIX, a Nash stable assignment always exists and can be computed efficiently (see Appendix B for the proof).

Theorem 9. *If (N, A, P) is an instance of* a-GASP *that is increasing, decreasing, or mixed increasing-decreasing, a Nash stable assignment always exists and can be found in polynomial time.*

Moreover, the problem of finding a Nash stable assignment that maximizes the number of agents assigned to a non-void activity admits an efficient algorithm if A^* consists of a single simple activity.

Theorem 10. *There exist a polynomial-time algorithm that given an instance (N, A, P) of* a-GASP *with $A^* = \{a\}$ finds a Nash stable assignment maximizing the number of agents assigned to a non-void activity, or decides that no Nash stable assignment exists.*

Proof. For each $k = n, \dots, 0$, we will check if there exists a Nash stable assignment π with $\#(\pi) = k$, and output the largest value of k for which this is the case.

For each $i \in N$, let $S'_i = S_i^{\downarrow a}$. For $k = n$ a Nash stable assignment π with $\#(\pi) = n$ exists if and only if $n \in S'_i$ for each $i \in N$. Assigning every agent to $a_\emptyset$ is Nash stable if and only if $1 \notin S'_i$ for each $i \in N$. Now we assume $1 \leq k \leq n-1$ and set $U_1 = \{i \in N \mid k \in S'_i, k+1 \notin S'_i\}$, $U_2 = \{i \in N \mid k \notin S'_i, k+1 \in S'_i\}$, and $U_3 = \{i \in N \mid k \in S'_i, k+1 \in S'_i\}$. If $|U_1| + |U_3| < k$, there does not exist an individually rational assignment π with $\#(\pi) = k$. If $U_2 \neq \emptyset$, no Nash stable assignment π with $\#(\pi) = k$ can exist, since each agent from U_2 would want to switch. If $|U_3| > k$, no Nash stable assignment π with $\#(\pi) = k$ can exist, since at least one agent in U_3 would not be assigned to a and thus would be unhappy. Finally, if $|U_1| + |U_3| \geq k$, $|U_3| \leq k$, $U_2 = \emptyset$, we can construct a Nash stable assignment π by assigning all agents from U_3 and $k - |U_3|$ agents from U_1 to a. Since we have $\pi(i) = a_\emptyset$ for all i with $k \notin S'_i$ and $\pi(i) \neq a_\emptyset$ for all i with $k+1 \in S'_i$, no agent is unhappy. □

5 Conclusions and Future Work

We have defined a new model for the selection of a number of group activities, discussed its connections with hedonic games, defined several stability notions, and, for two of them, we have obtained several complexity results. A number of our results are positive: finding desirable assignments proves to be tractable for several restrictions of the problem that are meaningful in practice. Interesting directions for future work include exploring the complexity of computing other solution concepts for a-GASP and extending our results to the more general setting of GASP.

Acknowledgments This research was supported by National Research Foundation (Singapore) under Research Fellowship NRF2009-08, by the project ComSoc (ANR-09-BLAN-0305-01), and by the Austrian Science Fund (P23724-G11 and P23829-N13). This project was initiated during the Dagstuhl seminar 12101 "Computation and Incentives in Social Choice", and the authors are very grateful to Dagstuhl for providing a great research environment and inspiration for this work. Part of this work was done when the second author was visiting Université Paris-Dauphine.

References

1. H. Aziz, F. Brandt, and P. Harrenstein. Pareto optimality in coalition formation. In *SAGT*, pages 93–104, 2011.
2. C. Ballester. NP-competeness in hedonic games. *Games and Economic Behavior*, 49:1–30, 2004.
3. S. Banerjee, H. Konishi, and T. Sönmez. Core in a simple coalition formation game. *Social Choice and Welfare*, 18:135–153, 2001.
4. A. Bogomolnaia and M. O. Jackson. The stability of hedonic coalition structures. *Games and Economic Behavior*, 38:201–230, 2002.
5. G. Chalkiadakis, E. Elkind, and M. Wooldridge. *Computational Aspects of Cooperative Game Theory*. Morgan and Claypool, 2011.
6. M. R. Garey and D. S. Johnson. *Computers and Intractibility*. W. H. Freeman and Company, 1979.
7. J. Lenstra, D. Shmoys, and É. Tardos. Approximation algorithms for scheduling unrelated parallel machines. *Math. Program.*, 46:259–271, 1990.

A Proofs for Section 4.1

Theorem 1. *It is* NP*-complete to decide whether* a-GASP *admits a perfect assignment, even when all activities in A^* are simple and all agents have increasing preferences.*

Proof. We provide a reduction from EXACT COVER BY 3-SETS (X3C). Recall that an instance of X3C is a pair $\langle X, \mathcal{Y} \rangle$, where $X = \{1, \dots, 3q\}$ and $\mathcal{Y} = \{Y_1, \dots, Y_p\}$ is a collection of 3-element subsets of X; it is a "yes"-instance if X can be covered by exactly q sets from $\mathcal{Y}$, and a "no"-instance otherwise. Given an instance $\langle X, \mathcal{Y} \rangle$ of X3C, we construct an instance of a-GASP as follows. We set $N = \{1, \dots, 3q\}$ and $A^* = \{a_1, \dots, a_p\}$. For each agent i, we define his vote V_i so that the induced approval vote S_i is given by $S_i = \{(a_j, k) \mid i \in Y_j, k \geq 3\}$, and let $P = (V_1, \dots, V_n)$. Clearly, (N, A, P) is an instance of a-GASP with increasing preferences. We claim that $\langle X, \mathcal{Y} \rangle$ is a "yes"-instance of X3C if and only if (N, A, P) admits a perfect assignment.

Indeed, suppose that there is a set of indices $I \subseteq \{1, \dots, p\}$ such that $|I| = q$ and $\cup_{i \in I} Y_i = X$. Define the assignment π by $\pi(i) = a_j$, where j is such that (1) $i \in Y_j$ and (2) $j \in I$. As $\{Y_i\}_{i \in I}$ is a partition of X, this assignment is well-defined and each agent is assigned to a non-void activity. Moreover, for each agent $i \in N$ we have $(\pi(i), 3) \in S_i$ and the number of agents assigned to $\pi(i)$ in π is exactly 3, so π is individually rational. Hence, π is a perfect assignment.

Conversely, assume that π is a perfect assignment for (N, A, P). Consider an alternative $a_j \in A^*$ with $\pi^j \neq \emptyset$. Since a_j is approved by exactly 3 agents, we have $|\pi^j| \leq 3$. On the other hand, since no agent approves of a_j if $\pi^j < 3$, it has to be the case that $|\pi^j| = 3$. Now, since π is a perfect assignment, it follows that there are exactly q activities a_j with $|\pi^j| = 3$. Therefore, the collection of subsets $\{Y_j \mid |\pi^j| = 3\}$ is an exact cover for X. □

Theorem 2. *It is* NP*-complete to decide whether* a-GASP *admits a perfect assignment, even when all activities in A^* are simple, all agents have decreasing preferences, and, moreover, for every agent $i \in N$ and every alternative $a \in A^*$ we have $S_i^{\downarrow a} \subseteq \{1, 2\}$.*

Proof. Consider the following restricted variant of the problem of scheduling on unrelated machines. There are n jobs and p machines. An instance of the problem is given by a collection of numbers $\{p_{ij} \mid i = 1, \dots, n, j = 1, \dots, p\}$, where p_{ij} is the running time of job i on machine j, and $p_{ij} \in \{1, 2, +\infty\}$ for every $i = 1, \dots, n$ and every $j = 1, \dots, p$. It is a "yes"-instance if there is a mapping $\rho : \{1, \dots, n\} \to \{1, \dots, p\}$ assigning jobs to machines so that the makespan is at most 2, i.e., for each $j = 1, \dots, p$ it holds that $\sum_{i:\rho(i)=j} p_{ij} \leq 2$. This problem is known to be NP-hard (see the proof of Theorem 5 in [7]).

Given an instance $\{p_{ij} \mid i = 1, \dots, n, j = 1, \dots, p\}$ of this problem, we construct an instance of a-GASP as follows. We set $N = \{1, \dots, n\}$, $A^* = \{a_1, \dots, a_p\}$. Further, for each agent $i \in N$ we construct a vote V_i so that the induced approval vote S_i satisfies $S_i^{\downarrow a_j} = \{1\}$ if $p_{ij} = 2$, $S_i^{\downarrow a_j} = \{1, 2\}$ if $p_{ij} = 1$, and $S_i^{\downarrow a_j} = \emptyset$ if $p_{ij} = +\infty$. Clearly, these preferences satisfy the constraints in the statement of the theorem.

Suppose there is a mapping ρ that assigns jobs to machines so that the makespan is at most 2, and consider an assignment π given by $\pi(i) = a_j$ if and only if $\rho(i) =$

j. Clearly, under this assignment each agent is assigned to a non-void activity, and, moreover, $\pi(i) = a_j$ implies $S_i^{\downarrow a_j} \neq \emptyset$ (since otherwise the completion time of machine j under ρ would be $+\infty$). Now, consider an agent i with $\pi(i) = a_j$. Since under ρ each machine can be assigned at most 2 jobs, we have $|\pi^j| \leq 2$. Thus, if $S_i^{\downarrow a_j} = \{1, 2\}$, agent i is clearly satisfied. Further, if $S_i^{\downarrow a_j} = \{1\}$, we have $p_{ij} = 2$. Therefore, under ρ no job other than i can be assigned to machine j, which, in turn, means that under π no agent other than i is assigned to activity a_j. Thus, in this case i is satisfied as well. Since this holds for each $i \in N$, it follows that π is a perfect assignment.

Conversely, let π be a perfect assignment for (N, A, P), and consider a mapping ρ given by $\rho(i) = j$ if and only if $\pi(i) = a_j$. Since under π each agent is assigned to a non-void activity, ρ assigns each job to some machine. Further, since π is individually rational, the completion time of each machine is finite, and each machine is assigned at most 2 jobs. Now, consider a machine j, $j \in \{1, \ldots, p\}$. If $p_{ij} = 1$ for each job i with $\rho(i) = j$, the completion time of machine j is at most 2. On the other hand, if $p_{ij} = 2$ for some job i with $\rho(i) = j$, then $S_i^{\downarrow a_j} = \{1\}$, which means that under π no agent other than i can be assigned to activity a_j. This, in turn, means that i is the only job assigned to machine j under ρ, and hence in this case the completion time of machine j is at most 2 as well. Since this holds for every machine, the makespan of ρ is at most 2. □

Theorem 3. *It is* NP*-complete to decide whether* a-GASP *admits a perfect assignment, even when all activities in A^* are simple, all agents have decreasing preferences, and, moreover, for every agent $i \in N$ it holds that* $|\{a \mid S_i^{\downarrow a} \neq \emptyset\}| \leq 3$.

Proof. We reduce MONOTONE 3-SAT to our problem. Recall that an instance of MONOTONE 3-SAT is given by a set of variables $X = \{x_1, \ldots, x_\ell\}$ and a set of three-literal clauses $\mathcal{C} = \{C_1, \ldots, C_q\}$ over X, where each clause in $\mathcal{C}$ contains positive literals only or negative literals only. It is a "yes"-instance if there exists a truth assignment to variables in X that satisfies all clauses and a "no"-instance otherwise. For a literal x, let $\mathrm{oc}(x)$ be the number of occurrences of x in $\mathcal{C}$. Without loss of generality, we can assume that the clauses $C_1, \ldots C_p$ contain positive literals only, while the clauses $C_{p+1}, \ldots, C_q$ contain negative literals only, and that $\mathrm{oc}(x_i) > 0$ and $\mathrm{oc}(\bar{x}_i) > 0$ for each $x_i \in X$.

We will now define an instance of restricted network flow problem $\mathcal{F}$ (for an illustration, see Fig. 1). Later, we will show that it corresponds to an instance of a-GASP with simple activities and decreasing preferences where each agent approves of at most three activities.

- We introduce a source s and a sink t.
- For each variable x_i
  - we introduce a node v_i and an arc of unlimited capacity from s to v_i;
  - we introduce two nodes x_i and $\bar{x}_i$, an arc labelled w_i from v_i to x_i, and an arc labelled $\bar{w}_i$ from v_i to $\bar{x}_i$.
- For each $i = 1, \ldots, \ell$, the capacity of the arc w_i is set to $\mathrm{oc}(x_i)$ and the capacity of the arc $\bar{w}_i$ is set to $\mathrm{oc}(\bar{x}_i) \cdot (q\ell + 1)$.

– For each "positive" clause C_j, $1 \le j \le p$, we introduce a node with the same label.
– For each "negative" clause C_j, $p+1 \le j \le q$, we introduce $q\ell+1$ nodes with the respective labels $C_j^{(1)}, \dots, C_j^{(q\ell+1)}$.
– For each $i = 1 \dots, \ell$ and each $j = 1, \dots, p$ we introduce unit capacity arcs from x_i to C_j if x_i appears in C_j.
– For each $i = 1, \dots, \ell$ and each $j = p+1, \dots, q$ we introduce unit capacity arcs from x_i and $\bar{x}_i$ to $C_j^{(1)}, \dots, C_j^{(q\ell+1)}$ if $\bar{x}_i$ appears in C_j.
– Finally, we connect each $C_1, \dots C_p, C_{p+1}^{(1)}, \dots, C_q^{(q\ell+1)}$ to t by a unit capacity arc.

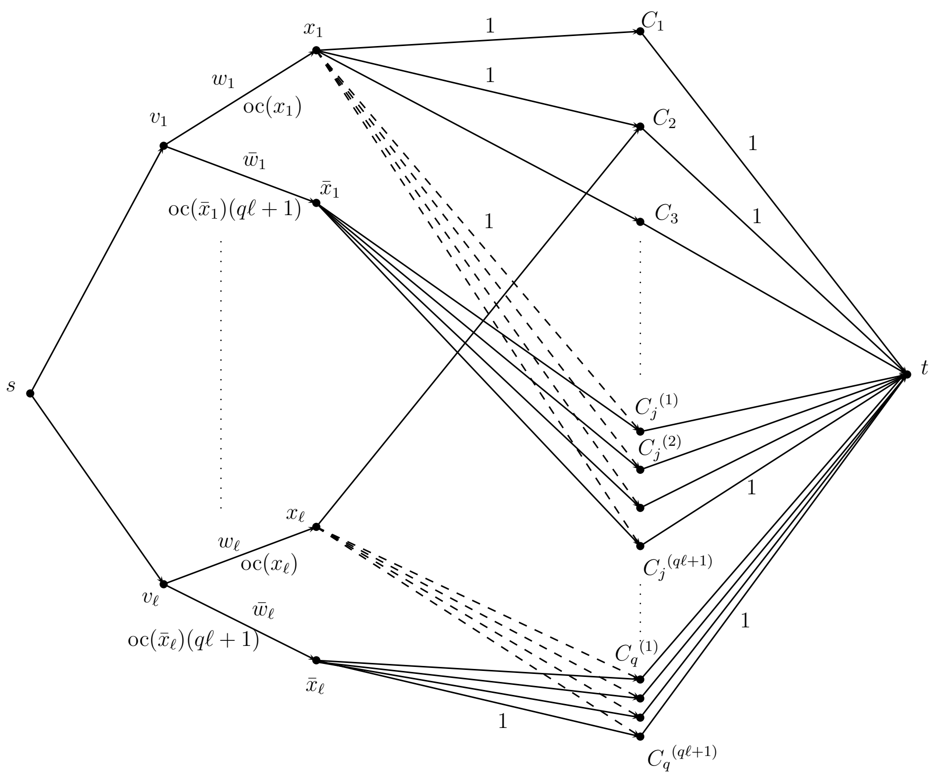

Figure1. Flow instance $\mathcal{F}$

We will now show that there is a satisfying truth assignment for X if and only if there is a flow of size $(q\ell+1)(q-p)+p$ in $\mathcal{F}$ that passes through at most one of $\{x_i, \bar{x}_i\}$ for each $i = 1, \dots, \ell$. The "if" direction is obvious, so let us consider the "only if" direction. Observe that a flow f of size $(q\ell+1)(q-p)+p$ that uses at most one of $\{x_i, \bar{x}_i\}$ for each $i = 1, \dots, \ell$ is a maximum flow in our network. Therefore it has to send a unit of flow through each of the nodes $C_1, \dots C_p, C_{p+1}^{(1)}, \dots, C_q^{(q\ell+1)}$. Now suppose that there is a $j \in \{p+1, \dots, q\}$ such that all of the nodes $C_j^{(1)}, \dots, C_j^{(q\ell+1)}$ are reached via positive literals only. Then we must have $\sum_{i=1}^{\ell} \mathrm{oc}(x_i) \geq q\ell + 1$, which is impossible. Thus, for each $j \in \{p+1, \dots, q\}$, f sends flow through at least one of $C_j^{(1)}, \dots, C_j^{(q\ell+1)}$ via at least one negative literal $\bar{x}_i$. By construction, the capacity of $\bar{w}_i$ is large enough that we can modify f so that through all of $C_j^{(1)}, \dots, C_j^{(q\ell+1)}$ the flow is sent via negative literals only. Recall that by construction, each "positive" clause $C_1, \dots, C_p$ is reachable only via nodes that are contained in the clause. Thus, there exists a flow that passes through at most one of $\{x_i, \bar{x}_i\}$ for each $i = 1, \dots, \ell$, sends flow from positive literals only to positive clauses and from negative literals only to (copies of) negative clauses, and passes through all clauses. Clearly, such a flow corresponds to a satisfying truth assignment.

Finally, we will now argue that the restricted flow problem described above can be interpreted as an instance of a-GASP. Indeed, we can view each node v_j as an activity, and each node $C_1, \dots C_p, C_{p+1}^{(1)}, \dots, C_q^{(q\ell+1)}$ as an agent. Each agent approves of the three activities they are reachable from. In more detail, for each agent $i \in \{C_1, \dots C_p\}$ we define his vote V_i so that the induced approval vote S_i is given by

$$S_i = \{(v_{i_j}, \mathrm{oc}(x_{i_j})) \mid i \text{ is reachable from } v_{i_j}\},$$

and for each agent $i \in \{C_{p+1}^{(1)}, \dots, C_q^{(q\ell+1)}\}$ we define his vote V_i so that the induced approval vote S_i is given by

$$S_i = \{(v_{i_j}, \mathrm{oc}(\bar{x}_{i_j})(q\ell+1)) \mid i \text{ is reachable from } v_{i_j}\}.$$

Note that we always have $\mathrm{oc}(x_{i_j}) < \mathrm{oc}(\bar{x}_{i_j})(q\ell+1)$. By construction, an agent is connected to x_{i_j} if she accepts up to $\mathrm{oc}(x_{i_j})$ participants in activity v_{i_j}; if she accepts up to $\mathrm{oc}(\bar{x}_{i_j})(q\ell+1)$ participants, she is connected to $\bar{x}_{i_j}$ as well. Thus, a flow of size $(q\ell+1)(q-p)+p$ corresponds to a perfect assignment. □

Theorem 4. *It is* NP*-complete to decide whether* a-GASP *admits a perfect assignment, even when all activities in A^* are equivalent (i.e., A^* consists of a single ∞-copyable activity a) and for every $i \in N$ we have $|S_i^{\downarrow a}| \leq 2$.*

Proof. Note that this variant of a-GASP can be viewed as an anonymous hedonic game, and a perfect assignment corresponds to a partition of players into coalitions so that each agent approves of the size of his coalition. Therefore, when constructing an instance of a-GASP in our NP-hardness reduction, we will only describe the coalition sizes that each agent approves of, without mentioning the activity explicitly.

We reduce from a restricted version of X3C (see the proof of Theorem 1), where it is assumed that every element of the ground set X appears in exactly three sets in $\mathcal{Y}$; X3C is known to remain NP-hard under this restriction [6]. Given an $(X, \mathcal{Y})$ instance of X3C, for each $x \in X$ let $t(x)$ denote the number of subsets in $\mathcal{Y}$ that contain x; note that $0 \leq t(x) \leq 3$. We pick $|X| + |\mathcal{Y}|$ pairwise distinct positive integers that are multiples of 4: for every subset $Y \in \mathcal{Y}$ we pick an integer $z(Y) \geq 4$, and for every element $x \in X$ we pick an integer $z(x) \geq 4$. We will now construct an instance of a-GASP as follows:

- for every element $x \in X$, we introduce a group G_x of $z(x)-t(x)+1$ corresponding element-players and set $S_i^{\downarrow a} = \{z(x)\}$ for all $i \in G_x$;
- for every subset $Y \in \mathcal{Y}$, we introduce a group G_Y of $z(Y) - 3$ corresponding subset-players and set $S_i^{\downarrow a} = \{z(Y) - 3, z(Y)\}$ for all $i \in G_Y$;
- for every element $x \in X$ and every subset $Y \in \mathcal{Y}$ with $x \in Y$, we introduce a corresponding choice-player $P(x, Y)$ and set set $S_{P(x,Y)}^{\downarrow a} = \{z(x), z(Y)\}$.

We claim that $(X, \mathcal{Y})$ is a "yes"-instance of X3C if and only if the constructed instance of a-GASP admits a perfect assignment.

First, suppose that $(X, \mathcal{Y})$ is a "yes"-instance of X3C, which is certified by a collection of subsets $\mathcal{Y}' \subseteq \mathcal{Y}$.

- For every subset $Y \in \mathcal{Y}'$ with $Y = \{x, y, z\}$, we form a coalition consisting of the $z(Y) - 3$ subset-players in G_Y and of the three choice-players $P(x, Y)$, $P(y, Y)$, $P(z, Y)$. The coalition size is $z(Y)$, and all these players are happy.
- For every subset $Y \in \mathcal{Y} \setminus \mathcal{Y}'$, the players in G_Y form their own coalition of size $z(Y) - 3$; all these players are happy.
- For every element $x \in X$, the $z(x)-t(x)+1$ element-players in G_x form a coalition together with the remaining $t(x)-1$ choice-players $P(x, Y)$ with $Y \in \mathcal{Y} \setminus \mathcal{Y}'$. The coalition size is $z(x)$, and all these players are happy.

Conversely, suppose that our instance of a-GASP admits a perfect assignment π. Consider an element $x \in X$, and consider the corresponding $z(x) - t(x) + 1$ element-players in G_x. All these players must be in a coalition of size $z(x)$, and for this they need exactly $t(x) - 1$ choice-players $P(x, Y)$ with $x \in Y$ to join. Hence exactly one of the choice-players $P(x, Y)$ with $x \in Y$ remains unassigned, and must look for another coalition. The only option for him is a coalition of size $z(Y)$; hence he must be together with the $z(Y) - 3$ subset-players in G_Y and with the two other choice-players $P(y, Y)$ and $P(z, Y)$ for this subset Y. Then we say that the corresponding subset Y is *selected* by its three choice-players. It is easy to verify that there are exactly q subsets selected, and that they form an exact cover. □

B Proofs for Section 4.3

Theorem 8. *It is* NP-*complete to decide whether* a-GASP *admits a Nash stable assignment.*

Proof. It is easy to see that this problem is in NP. To prove NP-hardness, we reduce from the problem of deciding whether a-GASP admits a perfect assignment.

Let (N, A, P) be an instance of a-GASP. Let $N' = N \cup \{n + 1\}$, $A' = A \cup \{b\}$. For each agent $i \in N$ define his vote V_i' so that the induced approval vote S_i' is given by $S_i' = S_i \cup \{(b, 1)\}$; also, define V_{n+1}' so that the induced approval vote S_{n+1}' is given by $S_{n+1}' = \{(b, 2)\}$. Set $P' = (V_1', \dots, V_{n+1}')$. Clearly, (N', A', P') is an instance of a-GASP; we will argue that it admits a Nash stable assignment if and only if (N, A, P) admits a perfect assignment.

Suppose first that (N, A, P) admits a perfect assignment π. Then we define an assignment π' for (N', A', P') by setting $\pi'(i) = \pi(i)$ for all $i \in N$ and $\pi'(n+1) = a_\emptyset$. Clearly, π' is Nash stable.

Conversely, suppose that (N', A', P') admits a Nash stable assignment π'. If π' assigns two or more agents to b, then it is not individually rational (and *a fortiori* not Nash stable), because for every agent $i \neq n+1$ we have $2 \notin S_i^{\downarrow b}$. If π' assigns exactly one agent to b, it has to be an agent $i \in N$, in which case π' is not Nash stable, because $n+1$ would like to join b. Hence, no agent is assigned to b. Further, if $\pi'(i) = a_\emptyset$ for some $i \in N$ then π' is not Nash stable, because i would like to join b. Thus, each agent $i \in N$ is assigned to a non-void activity in A. Since π' is individually rational, it follows that the restriction of π' to N is a perfect assignment for (N, A, P). □

Theorem 9. *If (N, A, P) is an instance of* a-GASP *that is increasing, decreasing, or mixed increasing-decreasing, a Nash stable assignment always exists and can be found in polynomial time.*

Proof. For increasing preferences, we can start by choosing an arbitrary individually rational assignment π (e.g., $\pi(i) = a_\emptyset$ for all $i \in N$). If π is not Nash stable, there exists an agent $i \in N$ with $\pi(i) = a_\emptyset$ and an activity $a_j \in A^*$ such that $(a, |\pi^j|+1) \in S_i$. We can then modify π by setting $\pi(i) = a_j$; clearly, this assignment remains individually rational. If the resulting assignment is still not Nash stable, we can repeat this step. Since at each step the number of agents assigned to the void activity goes down by 1, this process stops after at most n steps.

For decreasing preferences, we proceed as follows. We consider the activities one by one; at step j, we consider activity a_j. Let $N_j \subseteq N$ be the set of agents that remain unassigned at the beginning of step j. Let $N_{j,\ell} = |\{i \in N_j \mid (a_j, \ell) \in S_i\}|$, and set $k = \max\{\ell \mid N_{j,\ell} \geq \ell\}$. Thus, k is the size of the largest group of currently unassigned agents that can be assigned to a_j. By our choice of k, the set N_j contains at most k agents that are willing to share a_j with $k+1$ or more other agents. We assign all these agents to a_j; if the resulting coalition contains $\ell < k$ agents, we assign $k - \ell$ additional agents that approve of (a_j, k) to a_j (the existence of these $k - \ell$ agents is guaranteed by our choice of k). This completes the description of the j-th step. Note that no agent that remains unassigned after this step want to be assigned to a_j: indeed, this activity is currently shared among k agents, so if he were to join, the size of the group that is assigned to a_j would increase to $k+1$, and none of the unassigned agents is willing to share a_j with $k+1$ other agents. If some agents remain unassigned after n steps, we assign them to the void activity. To see that this assignment is Nash stable, consider an agent i assigned to the void activity. For each activity a_j he did not want to join the coalition of agents assigned to a_j during step j. Since the set of agents assigned to a_j did not change after step j, this is still the case.

For mixed decreasing-increasing instances, we first remove all activities in A^+ and apply our second algorithm to the remaining instance; we then consider the unassigned agents and assign them to activities in A^+ using the first algorithm. □