Article:

Ganian, R, Ordyniak, S orcid.org/0000-0003-1935-651X and Rahul, CS (2023) Group Activity Selection with Few Agent Types. Algorithmica, 85. pp. 1111-1155. ISSN 0178-4617

https://doi.org/10.1007/s00453-022-01058-z

Noname manuscript No.
(will be inserted by the editor)

Group Activity Selection with Few Agent Types

Robert Ganian · Sebastian Ordyniak · *Rahul C. S.

Received: date / Accepted: date

Abstract In this paper we establish the complexity map for the GROUP ACTIVITY SELECTION PROBLEM (GASP), along with two of its prominent variants called sGASP and gGASP, focusing on the case when the number of types of agents is the parameter. In all these problems, one is given a set of agents (each with their own preferences) and a set of activities, and the aim is to assign agents to activities in a way which satisfies certain global as well as preference-based conditions.

Our positive results, consisting of one fixed-parameter algorithm and one XP algorithm, rely on a combination of novel Subset Sum machinery (which may be of general interest) and identifying certain *compression steps* that allow us to focus on solutions with a simpler, well-defined structure (in particular, they are "acyclic"). These algorithms are complemented by matching lower bounds, which among others close a gap to a recently obtained tractability result of Gupta, Roy, Saurabh and Zehavi (2017). In this direction, the techniques used to establish W[1]-hardness of sGASP are of particular interest: as an intermediate step, we use Sidon sequences to show the W[1]-hardness of a highly restricted variant of multi-dimensional Subset Sum, which may find applications in other settings as well.

A shortened and preliminary version of this paper has been published in the proceedings of the 27th Annual European Symposium on Algorithms (ESA 2019).

Robert Ganian
TU Wien
Vienna, Austria
E-mail: rganian@gmail.com

Sebastian Ordyniak
University of Leeds
Leeds, UK
E-mail: sordyniak@gmail.com

Rahul C.S.
IIT Goa
Goa, India
*E-mail: rahulcsbn@gmail.com

1 Introduction

In this paper we consider the GROUP ACTIVITY SELECTION PROBLEM (GASP) together with its two most prominent variants, the SIMPLE GROUP ACTIVITY SELECTION PROBLEM (sGASP), and the GROUP ACTIVITY SELECTION PROBLEM WITH GRAPH STRUCTURE (gGASP) [7,20]. Since their introduction, these problems have received considerable attention, notably in venues dedicated to multi-agent systems and game theory [4–6,21,16,17]. In GASP one is given a set of agents, a set of activities, and a set of preferences for each agent in the form of a complete transitive relation (also called the *preference list*[1]) over the set of pairs consisting of an activity a and a number s, expressing the willingness of the agent to participate in the activity a if it has s participants. The aim is to find a "good" assignment from agents to activities subject to certain *rationality* and *stability* conditions. Specifically, an assignment is individually rational if agents that are assigned to an activity prefer this outcome over not being assigned to any activity, and an assignment is (Nash) stable if every agent prefers its current assignment over moving to any other activity (given no one else changes their current assignment). In this way GASP—which asks whether an individually rational and stable assignment exists—captures a wide range of real-life situations such as event organization and work delegation.

sGASP is a simplified variant of GASP where the preferences of agents are expressed in terms of *approval sets* containing (activity, size) pairs instead of preference lists. In essence sGASP is GASP where each preference list has only two equivalence classes: the class of the approved (activity, size) pairs (which contains all pairs that are preferred over not being assigned to any activity), and the class of disapproved (activity, size) pairs (all possible remaining pairs). On the other hand, gGASP is a generalization of GASP where one is additionally given an undirected graph (network) on the set of all agents that can be employed to model for instance acquaintanceship or physical distance between agents. Crucially, in gGASP one only considers assignments for which the subnetwork induced by all agents assigned to some activity is connected. Note that if the network forms a complete graph, then gGASP is equivalent to the underlying GASP instance.

Related Work. sGASP, GASP, and gGASP, are known to be NP-complete even in very restricted settings [7,16,17,19,20]. It is therefore natural to study these problems through the lens of parameterized complexity [3,9]. Apart from parameterizing by the number of agents assigned to any activity in a solution [21], the perhaps most prominent parameterizations thus far have been the number of activities, the number of agents, and in the case of gGASP structural parameterizations tied to the structure of the network such as treewidth [7,10,16,19,20]. Consequently, the parameterized complexity of all three variants of GASP w.r.t. the number of activities and/or the number of agents is now almost completely understood.

Indeed, computing a stable assignment for a given instance of GASP is known to be W[1]-hard and contained in XP parameterized by either the number of activities [7, 17,19] or the number of agents [17,19] and known to be fixed-parameter tractable parameterized by both parameters [17,19]. Even though it has never been explicitly

[1] To avoid any confusion, we stress that in line with previous work our model allows for ties.

stated, the same results also hold for gGASP when parameterizing by the number of agents as well as when using both parameters. This is because both the XP algorithm for the number of agents as well as the fixed-parameter algorithm for both parameters essentially brute-force over every possible assignment and are hence also able to find a solution for gGASP. Moreover, the fact that gGASP generalizes GASP implies that the W[1]-hardness result for the number of agents also carries over to gGASP. On the other hand, if we consider the number of activities as a parameter then gGASP turns out to be harder than GASP: Gupta et al. ([16]) showed that gGASP is NP-complete even when restricted to instances with a single activity. The hardness of gGASP has inspired a series of tractability results [16,20] obtained by employing additional restrictions on the structure of the network. One prominent result in this direction has been recently obtained by Gupta et al. ([16]), showing that gGASP is fixed-parameter tractable parameterized by the number of activities if the network has constant treewidth. For sGASP, it was recently shown that the problem is also W[1]-hard when parameterized by the number of activities [10], and hence the only small gap left was the complexity of this problem parameterized by the number of agents.

Already with the introduction of GASP [7] the authors argued that instead of putting restrictions on the total number of agents, which can be very large in general, it might be much more useful to consider the number of distinct types of agents (where two agents have the same type if they have the same preferences). It is easy to imagine a setting with large groups of agents that share the same preferences (for instance due to inherent limitations of how preferences are collected). In contrast to the related parameter number of activity types, where it is known that sGASP remains NP-complete even for a constant number of activity types [7], the complexity of the problems parameterized by the number of agent types (with or without restricting the number of activities) has remained wide open thus far.

An overview of these results on group activity selection problems will later also be summarized (together with our results) in Table 1.

Our Results. In this paper we obtain a complete classification of the complexity of GASP and its variants sGASP and gGASP when parameterized by the number of agent types (t) alone, and also when parameterized by t plus the number of activities (a). In particular, for each of the considered problems and parameterizations, we determine whether the problem is in FPT, or W[1]-hard and in XP, or paraNP-hard. One distinguishing feature of our lower- and upper-bound results is that they make heavy use of novel Subset-Sum machinery. Below, we provide a high-level summary of the individual results presented in the paper.

Result 1. sGASP is fixed-parameter tractable when parameterized by $t + a$.

This is the only fixed-parameter tractability result presented in the paper, and is essentially tight: it was recently shown that sGASP is W[1]-hard when parameterized by a alone [10], and the W[1]-hardness of the problem when parameterized by t is obtained in this paper. Our first step towards obtaining the desired fixed-parameter algorithm for sGASP is to show that every YES-instance has a solution which is *acyclic*—in particular, a solution with no cycles formed by interactions between activities and agent types (captured in terms of the *incidence graph*[2] G of an assignment). This is proved via the identification of certain *compression steps* which can be applied on a solution in order to remove cycles.

Once we show that it suffices to focus on acyclic solutions, we branch over all acyclic incidence graphs (i.e., all acyclic edge sets of G); for each such edge set, we can reduce the problem of determining whether there exists an assignment realizing this edge set to a variant of SUBSET SUM embedded in a tree structure. The last missing piece is then to show that this problem, which we call TREE SUBSET SUM, is polynomial-time tractable; this is done via dynamic programming, where each step boils down to solving a simplified variant of SUBSET SUM.

Result 2. sGASP is W[1]-hard when parameterized by t.

Our second result complements Result 1. As a crucial intermediate step towards Result 2, we obtain the W[1]-hardness of a variant of SUBSET SUM with three distinct "ingredients":

1. **Partitioning**: items are partitioned into sets, and precisely one item must be selected from each set,
2. **Multidimensionality**: each item is a d-dimensional vector (d being the parameter) where the aim is to hit the target value for each component, and
3. **Simplicity**: each vector contains precisely one non-zero component.

We call this problem SIMPLE MULTIDIMENSIONAL PARTITIONED SUBSET SUM (SMPSS). Note that SMPSS is closely related to MULTIDIMENSIONAL SUBSET SUM (MSS), which (as one would expect) merely enhances SUBSET SUM via Ingredient 2. MSS has recently been used to establish W[1]-hardness for parameterizations of EDGE DISJOINT PATHS [14] and BOUNDED DEGREE VERTEX DELETION [13]. However, Ingredient 1 and especially Ingredient 3 are critical requirements for our reduction to work, and establishing the W[1]-hardness of SMPSS was the main challenge on the way towards the desired lower-bound result for sGASP. Since MSS has already been successfully used to obtain lower-bound results and SMPSS is a much more powerful tool in this regard, we believe that SMPSS will find applications in establishing lower bounds for other problems in the future.

Result 3. GASP is in XP when parameterized by t.

This is the only XP result required for our complexity map, as it implies XP algorithms for sGASP parameterized by t and for GASP parameterized by $t + a$ (this will become more evident in Table 1 later). We note that the techniques used to obtain Result 3 are disjoint from those used for Result 1; in particular, our first step is to reduce GASP parameterized by t to solving "XP-many" instances of sGASP parameterized by t. This is achieved by showing that once we know a "least preferred alternative" for every agent type that is active in an assignment, then the GASP instance becomes significantly easier—and, in particular, can be reduced to a (slightly modified version of) sGASP. It is interesting to note that the result provides a significant conceptual improvement over the known brute force algorithm for GASP parameterized by the number of agents which enumerates all possible assignments of agents to

[2] A formal definition is provided at the beginning of Section 4.

activities [18, Theorem 3] (see also [17]): instead of guessing an assignment for all agents, one merely needs to guess a least preferred alternative for every agent type.

The second part of our journey towards Result 3 focuses on obtaining an XP algorithm for sGASP parameterized by t. This algorithm has two components. Initially, we show that in this setting one can reduce sGASP to the problem of finding an assignment which is individually rational (i.e., without the stability condition) and satisfies some additional minor properties. To find such an assignment, we once again make use of SUBSET SUM: in particular, we develop an XP algorithm for the MPSS problem (i.e., SUBSET SUM enhanced by ingredients 1 and 2) and apply a final reduction from finding an individually rational assignment to MPSS.

Result 4. GASP **is W[1]-hard when parameterized by** $t + a$**.**

Result 5. gGASP **is W[1]-hard when parameterized by** $t + a$ **and the *vertex cover number* [12] of the network.**

The final two results are hardness reductions which represent the last pieces of the presented complexity map. Both are obtained via reductions from PARTITIONED CLIQUE (also called MULTICOLORED CLIQUE in the literature [3]), and both reductions essentially use $k + \binom{k}{2}$ activities whose sizes encode the vertices and edges forming a k-clique (i.e., a clique of size k). The main challenge lies in the design of (a bounded number of) agent types whose preference lists ensure that the chosen vertices are indeed endpoints of the chosen edges. The reduction for gGASP then becomes even more involved, as it can only employ a limited number of connections between the agents in order to ensure that vertex cover of the network is bounded (in this sense, it may be seen as an extension of the reductiion used for Result 4).

We note that Result 5 also closes an open gap left by Gupta, Roy, Saurabh and Zehavi [16], who showed that gGASP is fixed-parameter tractable parameterized by the number of activities if the network has constant treewidth and left it open whether their result can be improved to a more efficient fixed-parameter algorithm parameterized by the number of activities and treewidth. In this sense, our hardness result represents a substantial shift of the boundaries of (in)tractability: in addition to the setting of Gupta et al., it also rules out the use of agent types as a parameter and replaces treewidth by the more restrictive vertex cover number.

An overview of our five main results in the context of related work is provided in Table 1. Note that these results provide an almost complete picture of the complexity of Group Activity Selection problems w.r.t. any combination of the parameters number of agents, number of activities, and number of agent types. There is only one piece missing, namely, the parameterized complexity of sGASP parameterized by the number of agents, which we resolve via a fairly direct fixed-parameter algorithm that reduces the problem to an instance of matching.

Result 6. sGASP **is fixed-parameter tractable parameterized by the number of agents.**

Organization of the Paper. After introducing the required preliminaries in Section 2, we present all of our Subset Sum machinery in the dedicated Section 3. Each subsequent Section $i \leq 9$ then focuses on obtaining Result $i - 3$.

	Parameterization	Lower Bound	Upper Bound
sGASP	n	**FPT**[0]	
GASP		$\mathsf{W}[1]_a$	XP
gGASP		W[1]	XP_a
sGASP	a	$\mathsf{W}[1]_b$	XP
GASP		W[1]	XP_a
gGASP		paraNP_c	
sGASP	$n+a$	FPT_d	
GASP			
gGASP			
sGASP	t	**W[1]**[2]	**XP**
GASP		**W[1]**	**XP**[3]
gGASP		paraNP_c	
sGASP	$t+a$	**FPT**[1]	
GASP		**W[1]**[4]	**XP**
gGASP		paraNP_c	
gGASP	$t+a+\mathsf{vc}$ (or $t+a+\mathsf{tw}$)	**W[1]**[5]	XP_c

Table 1 Lower and upper bounds for sGASP, GASP, and gGASP parameterized by the number of agents (n), the number of agent types (t), and the number of activities (a). In the case of gGASP, also the parameters vertex cover number (vc) and treewidth (tw) of the network are considered. Entries in bold are shown in this paper, and the numbers 1 to 5 in the upper index are used to identify results 1 to 5. The result marked with [0] is provided in the concluding remarks.
References: a is [19], b is [10], c is [16], d is folklore.

2 Preliminaries

For an integer i, we let $[i] = \{1,2,\dots,i\}$ and $[i]_0 = [i] \cup \{0\}$. We denote by $\mathbb{N}$ the set of natural numbers, by $\mathbb{N}_0$ the set $\mathbb{N} \cup \{0\}$. For a set S and an integer k, we denote by S^k and 2^S the set of all k dimensional vectors over S and the set of all subsets of S, respectively. For a vector $\bar{p}$ of integers, we use $\bar{p}[i]$ to denote the value of the i-th coordinate.

We refer to the handbook by Diestel ([8]) for standard graph terminology. The *vertex cover number* of a graph G is the size of a minimum vertex cover of G, i.e., the minimum size of a vertex set X such that every edge has at least one endpoint in X.

2.1 Parameterized Complexity

In parameterized algorithmics [9,3,22] the run-time of an algorithm is studied with respect to a parameter $k \in \mathbb{N}_0$ and input size n. The basic idea is to find a parameter that describes the structure of the instance such that the combinatorial explosion can be confined to this parameter. In this respect, the most favourable complexity class is FPT (*fixed-parameter tractable*) which contains all problems that can be decided by an algorithm running in time $f(k) \cdot n^{\mathcal{O}(1)}$, where f is a computable function. Algorithms with this running time are called *fixed-parameter algorithms*. A less favourable outcome is an XP *algorithm*, which is an algorithm running in time $\mathcal{O}(n^{f(k)})$; problems admitting such algorithms belong to the class XP.

To obtain our lower bounds, we will need the notion of a parameterized reduction. Formally, a *parameterized problem* is a subset of $\Sigma^* \times \mathbb{N}_0$, where Σ is the input alphabet. Let $L_1 \subseteq \Sigma_1^* \times \mathbb{N}_0$ and $L_2 \subseteq \Sigma_2^* \times \mathbb{N}_0$ be parameterized problems. A *parameterized reduction* (or FPT-reduction) from L_1 to L_2 is a mapping $P : \Sigma_1^* \times \mathbb{N}_0 \to \Sigma_2^* \times \mathbb{N}_0$ such that

(i) $(x, k) \in L_1$ iff $P(x, k) \in L_2$,
(ii) the mapping can be computed by an FPT-algorithm w.r.t. parameter k, and
(iii) there is a computable function g such that $k' \leq g(k)$, where $(x', k') = P(x, k)$.

Finally, we introduce the complexity class used to describe our lower bounds. The class $\mathsf{W}[1]$ captures parameterized intractability and contains all problems that are FPT-reducible to INDEPENDENT SET (parameterized by solution size).

2.2 Group Activity Selection

The task in the GROUP ACTIVITY SELECTION PROBLEM (GASP) is to compute a *stable assignment* π from a given set N of *agents* to a set A of *activities*, where each agent participates in at most one activity in A. The assignment π is (Nash) stable if and only if it is *individually rational* and no agent has an *NS-deviation* to any other activity (both of these stability rules are defined in the next paragraph). We use a dummy activity $a_\emptyset$ to capture all those agents that do not participate in any activity in A and denote by A^* the set $A \cup \{a_\emptyset\}$. Thus, an assignment π is a mapping from N to A^*, and for an activity $a \in A$ we use $\pi^{-1}(a)$ to denote the set of agents assigned to a by π; we set $|\pi^{-1}(a_\emptyset)| = 1$ if there is at least one agent assigned to $a_\emptyset$ and 0 otherwise.

The set X of *alternatives* is defined as $X = (A \times [|N|]) \cup \{(a_\emptyset, 1)\}$. Each agent is associated with its own *preferences* defined on the set X. In the case of the standard GASP problem, an instance I is of the form $(N, A, (\succeq_n)_{n \in N})$ where each agent n is associated with a complete transitive preference relation (list) $\succeq_n$ over the set X. An assignment π is *individually rational* if for every agent $n \in N$ it holds that if $\pi(n) = a$ and $a \neq a_\emptyset$, then $(a, |\pi^{-1}(a)|) \succeq_n (a_\emptyset, 1)$ (i.e., n weakly prefers staying in a over moving to $a_\emptyset$). An agent n where $\pi(n) = a$ is defined to have an *NS-deviation* to a different activity a' in A if $(a', |\pi^{-1}(a')| + 1) \succ_n (a, |\pi^{-1}(a)|)$ (i.e., n prefers moving to an activity a' over staying in a). The task in GASP is to compute a stable assignment.

gGASP is defined analogously to GASP, however where one is additionally given a set L of *links* $L \subseteq \{\, \{n, n'\} \mid n, n' \in N \wedge n \neq n' \,\}$ between the agents on the input; specifically, L can be viewed as a set of undirected edges and (N, L) as a simple undirected graph. In gGASP, the task is to find an assignment π which is not only stable but also connected; formally, for every $a \in A$ the set of agents $\pi^{-1}(a)$

induces a connected subgraph of (N, L). Moreover, an agent $n \in N$ only has an NS-deviation to some activity $a \neq \pi(n)$ if (in addition to the conditions for NS-deviations defined above) n has an edge to at least one agent in $\pi^{-1}(a)$.

In sGASP, an instance I is of the form $(N, A, (P_n)_{n\in N})$, where each agent has an *approval set* $P_n \subseteq X \setminus \{(a_\emptyset, 1)\}$ of preferences (instead of an ordered preference list). We denote by $P_n(a)$ the set $\{\, i \mid (a, i) \in P_n \,\}$ for an activity $a \in A$, i.e., $P_n(a)$ is the set of approved sizes for activity a from the viewpoint of agent n. An assignment $\pi : N \to A^*$ is said to be *individually rational* if every agent $n \in N$ satisfied the following: if $\pi(n) = a$ and $a \neq a_\emptyset$, then $|\pi^{-1}(a)| \in P_n(a)$. Further, an agent $n \in N$ where $\pi(n) = a_\emptyset$, is said to have an *NS-deviation* to an activity a in A if $(|\pi^{-1}(a)| + 1) \in P_n(a)$.

We now introduce the notions and definitions required for our main parameter of interest, the "number of agent types". We say that two agents n and n' in N have the same *agent type* if they have the same preferences. To be specific, $P_n = P_{n'}$ for sGASP and $\succeq_n = \succeq_{n'}$ for GASP and gGASP. In the case of sGASP and GASP n and n' are indistinguishable, while in gGASP n and n' can still have different links to other agents. For a subset $N' \subseteq N$, we denote by $T(N')$ the set of agent types occurring in N'. Note that this notation requires that the instance is clear from the context. If this is not the case then we denote by $T(I)$ the set $T(N)$ if N is the set of agents for the instance I of sGASP, GASP, or gGASP.

For every agent type $t \in T(I)$, we denote by N_t the subset of N containing all agents of type t; observe that $\{\, N_t \mid t \in T(I) \,\}$ forms a partition of N. For an agent type $t \in T(I)$ we denote by P_t (sGASP) or $\succeq_t$ (GASP) the preference list assigned to all agents of type t and we use $P_t(a)$ (for an activity $a \in A$) to denote P_t restricted to activity a, i.e., $P_t(a)$ is equal to $P_n(a)$ for any agent n of type t. For an assignment $\pi : N \to A^*$, $t \in T(I)$, and $a \in A$ we denote by $\pi_{t,a}$ the set $\{\, n \mid n \in N_t \wedge \pi(n) = a \,\}$, i.e., $\pi_{t,a}$ is the set of agents of type t assigned to activity a by the assignment π. Moreover, we denote by π_t the set $\bigcup_{a\in A} \pi_{t,a}$, i.e., π_t is the set of all agents of type t that are assigned by π to some activity in A. Further, we denote by $\pi(t)$ the set of all activities that have at least one agent of type t participating in it. We say that π is a perfect assignment for some agent type $t \in T(I)$ if $\pi(n) \neq a_\emptyset$ for every $n \in N_t$. We denote by $\mathrm{PE}(I, \pi)$ the subset of $T(I)$ consisting of all agent types that are perfectly assigned by π, and say that π is a *perfect assignment* if $\mathrm{PE}(I, \pi) = T(I)$. To avoid any confusion, we remark that these definitions apply to all considered variants of group activity selection.

One notion that will appear throughout the paper is that of *compatibility*: for a subset $Q \subseteq T(I)$, we say that π is *compatible* with Q if $\mathrm{PE}(I, \pi) = Q$. We conclude this section with a technical lemma which provides a preprocessing procedure that will be used as a basic tool for obtaining our algorithmic results. In particular, Lemma 1 allows us to reduce the problem of computing a stable assignment for an sGASP instance compatible with Q to the problem of finding an individually rational assignment.

Lemma 1 *Let $I = (N, A, (P_n)_{n\in N})$ be an instance of* sGASP *and $Q \subseteq T(I)$. Then in time $\mathcal{O}(|N|^2|A|)$ one can compute an instance $\gamma(I, Q) = (N, A, (P'_n)_{n\in N})$ and $A_{\neq\emptyset}(I, Q) \subseteq A$ with the following property: for every assignment $\pi : N \to A^*$ that* *is compatible with Q, it holds that π is stable for I if and only if π is individually rational for $\gamma(I,Q)$ and $\pi^{-1}(a) \neq \emptyset$ for every $a \in A_{\neq\emptyset}(I,Q)$.*

Proof Let $\pi : N \to A^*$ be an assignment that is compatible with Q. Then there is an agent of type $t \in T(N)$ assigned to $a_\emptyset$ if and only if $t \notin Q$. Recall that an assignment is stable if and only if it is individually rational and does not have an NS-deviation, where for sGASP an NS-deviation occurs when an agent not currently assigned to any activity can join some other activity and the resulting size is in that agent's preference list. Hence π is stable for I if and only if (1) π is individually rational and furthermore (2) it holds that for every agent type $t \in T(N) \setminus Q$ and every activity $a \in A$, $|\pi^{-1}(a)| + 1 \notin P_t(a)$. This naturally leads us to a certain set of "forbidden" sizes for activities, and we will obtain the desired instance $\gamma(I,Q)$ by simply removing all tuples from all preference lists that would allow activities to reach a forbidden size. Formally, we obtain the desired instance $\gamma(I,Q)$ from I removing all tuples (a,i) from every preference list P_t, where $t \in T(N)$ such that there is an agent type $t' \in T(N) \setminus Q$ with $i+1 \in P_{t'}(a)$. This construction prevents the occurrence of all forbidden sizes of activities *except* for forbidding activities of size 0; that is where we use the set $A_{\neq\emptyset}(I,Q)$. Formally, the set $A_{\neq\emptyset}(I,Q)$ consists of all activities a such that there is an agent type $t \in T(N) \setminus Q$ with $1 \in P_t(a)$. It is now straightforward to verify that $\gamma(I,Q)$ and $A_{\neq\emptyset}(I,Q)$ satisfy the claim of the lemma.

Finally the running time of $\mathcal{O}(|N|^2|A|)$ for the algorithm can be achieved as follows. In a preprocessing step we first classify all agents into agent types and compute for every activity $a \in A$ the set of all forbidden numbers, i.e., the set of all numbers i such that there is an agent type $t \in T(N) \setminus Q$ with $(a, i+1) \in P_{t'}(a)$. For every activity a, we store the resulting set of numbers in such a way that determining whether a number i is contained in the set for activity a can be achieved in constant time; this can for instance be achieved by storing the set for each activity a as a Boolean array with $|N|$ entries, whose i-th entry is TRUE if and only if i is contained in the set of numbers for a. This preprocessing step takes time at most $\mathcal{O}(|N|^2|A|)$ and after it is completed we can use the computed sets to test for every agent type $t \in T(N)$, every activity $a \in A$, and every $i \in P_t(a)$, whether there is an agent type $t' \in T(N) \setminus Q$ such that $(a, i+1) \in P_{t'}(a)$ in constant time. If so we remove i from $P_t(a)$, otherwise we continue. This shows that $\gamma(I,Q)$ can be computed in $\mathcal{O}(|N|^2 \cdot |A|)$ time. The computation of $A_{\neq\emptyset}$ only requires to check for every activity $a \in A$ whether 1 is contained in the set of forbidden numbers for a; if so a is contained in $A_{\neq\emptyset}$ and otherwise it is not. After preprocessing, this can be achieved in time $\mathcal{O}(|A|)$. □

3 Subset Sum Machinery

In this section we introduce the subset sum machinery required for our algorithms and lower bound results. In particular, we introduce three variants of SUBSET SUM, obtain algorithms for two of them, and provide a W[1]-hardness result for the third. We note that it may be helpful to read the following three subsections in the context of the individual sections where they are used: in particular, Subsection 3.1 is used to obtain Result 1 (Section 4), Subsection 3.2 is used as a preprocedure for Result 3 (Section 6) and Subsection 3.3 (which is by far the most difficult of the three) is a crucial step in the reduction used for Result 2 (Section 5).

3.1 Tree Subset Sum

Here we introduce a useful generalization of SUBSET SUM, for which we obtain polynomial-time tractability under the assumption that the input is encoded in unary. Intuitively, our problem asks us to assign values to edges while meeting a simple criterion on the values of edges incident to each vertex.

TREE SUBSET SUM (TSS)	
Input:	A vertex-labeled undirected tree T with labeling function $\lambda : V(T) \to 2^{\mathbb{N}_0}$.
Question:	Is there an assignment $\alpha : E(T) \to \mathbb{N}_0$ such that for every $v \in V(T)$ it holds that $\sum_{e \in E(T) \wedge v \in e} \alpha(e) \in \lambda(v)$.

Let us briefly comment on the relationship of TSS with SUBSET SUM. Recall that given a set $S \subseteq \mathbb{N}_0$ and a number $t \in \mathbb{N}_0$, the SUBSET SUM problem asks whether there is a subset S' of S such that $\sum_{s \in S'} s = t$. One can easily construct a simple instance (G, λ) of TSS that is equivalent to a given instance (S, t) of SUBSET SUM as follows. G consists of a star having one leaf l_s for every $s \in S$ with $\lambda(l_s) = \{0, s\}$ and $\lambda(c) = \{t\}$ for the center vertex c of the star. Given this simple reduction from SUBSET SUM to TSS it becomes clear that TSS is much more general than SUBSET SUM. In particular, instead of a star TSS allows for the use of an arbitrary tree structure and moreover one can use arbitrary subsets of natural numbers to specify the constrains on the vertices. The above reduction in combination with the fact that SUBSET SUM is weakly NP-hard implies that TSS is also weakly NP-hard.

In the remainder of this section we will show that TSS (like SUBSET SUM) can be solved in polynomial-time if the input is given in unary.

Let $I = (T, \lambda)$ be an instance of TSS. We denote by $\max(I)$ the value of the maximum number occurring in any vertex label. The main idea behind our algorithm for TSS is to apply leaf-to-root dynamic programming. In order to execute our dynamic programming procedure, we will need to solve a special case of TSS which we call PARTITIONED SUBSET SUM; this is the problem that will later arise when computing the dynamic programming tables for TSS.

PARTITIONED SUBSET SUM	
Input:	A target set $R \subseteq \mathbb{N}_0$ and ℓ source sets $S_1, \dots, S_\ell \subseteq \mathbb{N}_0$.
Task:	Compute the set $S \subseteq \mathbb{N}_0$ such that for each $s \in S$, there are $s_1, \dots, s_\ell$, where $s_i \in S_i$ for every i with $1 \leq i \leq \ell$, satisfying $(\sum_{1 \leq i \leq \ell} s_i) + s \in R$.

For an instance $I = (R, S_1, \dots, S_\ell)$ of PARTITIONED SUBSET SUM, we denote by $\max(I)$ the value of the maximum number occurring in R.

Lemma 2 *An instance $I = (R, S_1, \dots, S_\ell)$ of* PARTITIONED SUBSET SUM *can be solved in time $\mathcal{O}(\ell \cdot \max(I)^2)$.*

Proof Here we also use a dynamic programming approach similar to the approach used for the well-known SUBSET SUM problem [15]. Let $I = (R, S_1, \dots, S_\ell)$ be an instance of PARTITIONED SUBSET SUM.

We first apply a minor modification to the instance which will allow us to provide a cleaner presentation of the algorithm. Namely, let $P_0, P_1, \dots, P_\ell$ be sets of integers defined as follows: $P_0 = R$, and for every $i \in [\ell]$, we set $P_i = \{ -s \mid s \in S_i \}$. Then the solution S for I is exactly the set of all numbers $n \in \mathbb{N}_0$ for which there are $p_0, \dots, p_\ell$ with $p_i \in P_i$ for every i with $0 \leq i \leq \ell$ such that $\sum_{0 \leq i \leq \ell} p_i = n$, and observe that we may assume w.l.o.g. that $n \leq (\max_{r \in R} r) \leq |I|$.

In order to compute the solution S for I (employing the above characterization for S), we compute a table D having one binary entry $D[i, n]$ for every i and n with $0 \leq i \leq \ell$ and $0 \leq n \leq \max(I)$ such that $D[i, n] = 1$ if and only if there are $p_0, \dots, p_i$ with $\sum_{0 \leq j \leq i} p_j = n$. Note that the solution S for I can be obtained from the table D as the set of all numbers n such that $D[\ell, n] = 1$. It hence remains to show how D can be computed.

We compute $D[i, n]$ via dynamic programming using the following recurrence relation. We start by setting $D[0, n] = 1$ for every n with $0 \leq n \leq \max(I)$ if and only if $n \in P_0$. Moreover, for every i with $1 \leq i \leq \ell$ and every n with $0 \leq n \leq \max(I)$, we set $D[i, n] = 1$ if and only if there is an n' with $n \leq n' \leq \max(I)$ and a $p \in P_i$ such that $n' + p = n$ and $D[i-1, n'] = 1$.

The running time of the algorithm is $\mathcal{O}(\ell \cdot \max(I)^2)$ since we require $\mathcal{O}(\ell \cdot \max(I))$ to initialize the table D and each of the ℓ recursive steps requires time $\mathcal{O}(\max(I)^2)$. The correctness of the algorithm follows from the correctness of each inductive step. To observe the correctness of each inductive step, assume that we have correctly computed all entries of $D[i-1, n]$ for all $n \leq |I|$, and notice that for each $n \leq |I|$ the following holds: $D[i, n] = 1$ if and only if there is an n' with $n \leq n' \leq \max(I)$ and a $p \in P_i$ such that $n' + p = n$ and $D[i-1, n'] = 1$. □

With Lemma 2 in hand, we can proceed to a pseudo-polynomial-time algorithm for TSS.

Lemma 3 *An instance $I = (T, \lambda)$ of* TSS *can be solved in time $\mathcal{O}(|V(T)|^2 \cdot \max(I)^2)$.*

Proof As mentioned earlier, the main idea behind our algorithm for TSS is to use a dynamic programming algorithm working from the leaves to an arbitrarily chosen root r of the tree T. Informally, the algorithm computes a set of numbers for each non-root vertex v of T representing the set of all assignments of the edge from v to its parent that can be extended to a valid assignment of all edges in the subtree of T rooted at v. Once this set has been computed for all children of the root we can construct a simple PARTITIONED SUBSET SUM instance (given below) to decide whether I has a solution.

More formally, for a vertex v of T we denote by T_v the subtree of T rooted at v and by T_v^* the subtree of T consisting of T_v plus the edge between v and its parent in

T; for the root r of T it holds that $T_v^* = T_v$. For every non-root vertex v with parent p we will compute a set $R(v)$ of numbers. Informally, $R(v)$ contains all numbers n such that the assignment setting $\{p, v\}$ to n can be extended to an assignment for all the edges in T_v^* satisfying all constrains given by the vertices in T_v. More formally, $n \in R(v)$ if and only if there is an assignment $\alpha : E(T_v^*) \to \mathbb{N}_0$ with $\alpha(\{p, v\}) = n$ such that for every $v \in V(T_v)$ it holds that $\sum_{e \in E(T_v^*) \wedge v \in e} \alpha(e) \in \lambda(v)$.

As stated above we will compute the sets $R(v)$ via a bottom-up dynamic programming algorithm starting at the leaves of T and computing $R(v)$ for every inner node v of T using solely the computed sets $R(c)$ of each child c of v in T. Note that having computed $R(c)$ for every child c of the root r of T we can decide whether I has a solution as follows. Let $c_1, \dots, c_\ell$ be the children of r in T; then I has a solution if and only if the solution set for the instance $(\lambda(r), R(c_1), \dots, R(c_\ell))$ of PARTITIONED SUBSET SUM contains 0.

The remaining step is to show how to compute $R(v)$ for the leaves and inner nodes of T. If v is a leaf then $R(v)$ is simply equal to $\lambda(v)$. Moreover, if v is an inner node with children $c_1, \dots, c_\ell$, then $R(v)$ is equal to the solution set for the instance $(\lambda(v), R(c_1), \dots, R(c_\ell))$ of PARTITIONED SUBSET SUM. This completes the description of the algorithm.

The running time of the algorithm is at most $\mathcal{O}(|V(T)|^2 \cdot \max(I)^2)$ since the time required at a leaf q of T is at most $\mathcal{O}(1)$ and the time required at any none-leaf node t of T with children $t_1, \dots, t_\ell$ is at most the time required to solve the instance $(\lambda(t), R(t_1), \dots, R(t_\ell))$ of PARTITIONED SUBSET SUM, which is at most $\mathcal{O}(|V(T)| \cdot \max(\lambda(v))^2)$ due to Lemma 2.

For correctness, it suffices to verify that our computation of the set $R(v)$ is correct for each non-root vertex v of T; indeed, once that is done, it is easily observed that I has a solution if and only if the solution set for the instance $(\lambda(r), R(c_1), \dots, R(c_\ell))$ of PARTITIONED SUBSET SUM contains 0. The correctness of the computation of $R(v)$ if v is a leaf is trivial, while for non-leaf vertices the correctness follows from the correctness of Lemma 2. □

3.2 Multidimensional Partitioned Subset Sum

Our second generalization of SUBSET SUM is a multi-dimensional variant of the problem that allows to separate the input set of numbers into several groups, and restricts the solution to take at most 1 vector from each group. For technical reasons, we will only search for solutions over non-negative integers up to a given bound r.

MULTIDIMENSIONAL PARTITIONED SUBSET SUM (MPSS)	
Input:	$d \in \mathbb{N}$, $r \in \mathbb{N}_0$, and a family $\mathcal{P} = \{P_1, \dots, P_l\}$ of sets of vectors over $\mathbb{N}_0^d$.
Task:	Compute the set of all vectors $\bar{t} \in \{0, \dots, r\}^d$ such that there are $\bar{p}_1, \dots, \bar{p}_l$ with $\bar{p}_i \in P_i$ for every i with $1 \leq i \leq l$ such that $\sum_{1 \leq i \leq l} \bar{p}_i = \bar{t}$.

It is easy to see that SUBSET SUM is a special case of MPSS: given an instance of SUBSET SUM, we can create an equivalent instance of MPSS by setting r to a sufficiently large number and simply making each group P_i contain two vectors: the all-zero vector and the vector that is equal to the i-th number of the SUBSET SUM instance in all entries.

Lemma 4 *An instance $I = (d, r, \mathcal{P})$ of* MPSS *can be solved in time $\mathcal{O}(|\mathcal{P}| \cdot r^d)$.*

Proof We use a dynamic programming procedure similar to the approach used for the well-known SUBSET SUM problem [15]. Let $I = (d, r, \mathcal{P})$ with $\mathcal{P} = \{P_1, \dots, P_l\}$ be an instance of MPSS.

We solve I by computing a table D having one binary entry $D[i, \bar{t}\,]$ for every i and $\bar{t}$ with $0 \leq i \leq l$ and $\bar{t} \in [r]_0^d$ such that $D[i, \bar{t}] = 1$ if and only if there are $\bar{p}_1, \dots, \bar{p}_i$ with $\bar{p}_j \in P_j$ for every j with $1 \leq j \leq i$ such that $\sum_{1 \leq j \leq i} \bar{p}_j = \bar{t}$. Note that the solution for I can be obtained from the table D as the set of all vectors $\bar{t} \in [r]_0^d$ such that $D[l, \bar{t}] = 1$. It hence remains to show how to compute the table D.

We compute $D[i, \bar{t}]$ via dynamic programming using the following recurrence relation. We start by setting $D[1, \bar{t}] = 1$ for every $\bar{t} \in [r]_0^d$ if and only if $\bar{t} \in P_1$. Moreover, for every i with $1 \leq i \leq l$ and every $\bar{t} \in [r]_0^d$, we set $D[i, \bar{t}] = 1$ if and only if there is a $\bar{p}_j \in P_j$ with $\bar{p}_j \leq \bar{t}$ such that $D[i-1, \bar{t} - \bar{p}_j] = 1$.

The running time of the algorithm is $\mathcal{O}(|\mathcal{P}| \cdot r^d)$ since we require $\mathcal{O}(|\mathcal{P}| \cdot r^d)$ to initialize the table D and each of the $|\mathcal{P}|$ recursive steps requires time $\mathcal{O}(r^d)$. Correctness then follows from the correctness of the recurrence relation provided in the previous paragraph. $\square$

3.3 Simple Multidimensional Partitioned Subset Sum

In this section, we are interested in a much more restrictive version of MPSS, where all vectors (apart from the target vector) are only allowed to have at most one non-zero component, and where the task is merely to determine whether the output of MPSS contains a specific vector. Surprisingly, we show that the W[1]-hardness of the previously studied MULTIDIMENSIONAL SUBSET SUM problem [13,14] carries over to this more restrictive variant using an intricate and involved reduction.

To formalize, we say that a set P of vectors in $\mathbb{N}_0^d$ is *simple* if each vector in P has exactly one non-zero component and the values of the non-zero components for any two distinct vectors in P are distinct.

SIMPLE MULTIDIMENSIONAL PARTITIONED SUBSET SUM (SMPSS)

Input:	$d \in \mathbb{N}$, $\bar{t} \in \mathbb{N}_0^d$, and a family $\mathcal{P} = \{P_1, \dots P_l\}$ of simple sets of vectors in $\mathbb{N}_0^d$.
Parameter:	d.
Question:	Are there vectors $\bar{p}_1, \dots, \bar{p}_l$ with $\bar{p}_i \in P_i$ for every i with $1 \leq i \leq l$ such that $\sum_{1 \leq i \leq l} \bar{p}_i = \bar{t}$.

Theorem 5 SMPSS *is strongly* W[1]*-hard.*

Proof We will employ a parameterized reduction from the PARTITIONED CLIQUE problem, which is well-known to be W[1]-complete [23].

PARTITIONED CLIQUE	
Input:	An integer k, a k-partite graph $G = (V, E)$ with partition $\{V_1, \dots, V_k\}$ of V into sets of equal size.
Parameter:	k
Question:	Does G have a k-clique, i.e., a set $C \subseteq V$ of k vertices such that $\forall u, v \in C$, with $u \neq v$ there is an edge $\{u, v\} \in E$?

We denote by $E_{i,j}$ the set of edges of G that have one endpoint in V_i and one endpoint in V_j and we assume w.l.o.g. that $|V_i = \{v_1^i, \dots, v_n^i\}| = n$ and $|E_{i,j}| = m$ for every i and j with $1 \leq i < j \leq k$ (see, e.g., [3, Theorem 13.7]) for a justification for these assumptions).

Given an instance (G, k) of PARTITIONED CLIQUE with partition $V_1, \dots, V_k$, we construct an equivalent instance $I = (d, \bar{t}, \mathcal{P})$ of SMPSS in polynomial time, where $d = k(k-1) + \binom{k}{2}$ and $|\mathcal{P}| = \binom{k}{2} + nk(2k-3)$. We will also make use of the following notation. For i and j with $1 \leq i \leq k$ and $1 \leq j < k$, we denote by $\mathsf{indJ}(i,j)$ the j-th smallest number in $[k] \setminus \{i\}$ and we denote by $\mathsf{indMin}(i)$ and $\mathsf{indMax}(i)$ the numbers $\mathsf{indJ}(i,1)$ and $\mathsf{indJ}(i,k-1)$, respectively.

We assign to every vertex v of G a unique number $\mathcal{S}(v)$ from a Sidon sequence $\mathcal{S}$ of length $|V(G)|$ [11]. A *Sidon sequence* is a sequence of natural numbers such that the sum of each pair of numbers is unique; it can be shown that it is possible to construct such sequences whose maximum value is bounded by a polynomial in its length [1, 11].

To simplify the description of I, we will introduce names and notions to identify both components of vectors and sets in $\mathcal{P}$. Every vector in I has the following components:

- For every i and j with $1 \leq i, j \leq k$ and $i \neq j$, the *vertex component* $c_V^i(j)$. We set $\bar{t}[c_V^i(j)]$ to:
  - $n^6 + n^4$ if $j = \mathsf{indMin}(i)$,
  - $(n-1)n^8 + n^6 + n^4 + \sum_{\ell=1}^{n}(\ell + \ell n^2)$ if $j > \mathsf{indMin}(i)$ and $j < \mathsf{indMax}(i)$, and
  - $(n-1)n^8 + n^6 + \sum_{\ell=1}^{n} \ell$, otherwise.
- For every i and j with $1 \leq i < j \leq k$, the *edge component* $c_E(i,j)$ with $\bar{t}[c_E(i,j)] = \sum_{v \in V_i \cup V_j} \mathcal{S}(v)$.

Note that the total number of components d is equal to $k(k-1) + \binom{k}{2}$ and that for every i with $1 \leq i \leq k$, there are $k-1$ vertex components, i.e., the components $c_V^i(\mathsf{indJ}(i,1)), \dots, c_V^i(\mathsf{indJ}(i,k-1))$, which intuitively have the following tasks. The first component, i.e., the component $c_V^i(\mathsf{indJ}(i,1))$ identifies a vertex $v \in V_i$ that should be part of a k-clique in G. Moreover, every component $c_V^i(\mathsf{indJ}(i,j))$ (including the first component), is also responsible for: (1) Signalling the choice of the chosen vertex $v \in V_i$ to the next component, i.e., the component $c_V^i(\mathsf{indJ}(i,j+1))$ and (2) Signalling the choice of the vertex $v \in V_i$ to the component $c_E(i,j)$ that will then verify that there is an edge between the vertex chosen for V_i and the vertex chosen for V_j. This interplay between the components will be achieved through the sets of vectors in $\mathcal{P}$ that will be defined and explained next.

$\mathcal{P}$ consists of the following sets, which are illustrated in Table 2 and 3:

	$P^1_{EV}(2,\ell)$	$P^1_V(2,\ell)$	$P^1_{EV}(3,\ell)$	$P^1_V(3,\ell)$	$P^1_{EV}(4,\ell)$	$\bar{t}$
$c^1_V(2)$	$n^6+\ell$	$n^4-\ell$				n^6+n^4
$c^1_V(3)$		$n^8+\ell$ $+\ell n^2$	$n^6+\ell$	$n^4+\ell n^2$		$Z+n^4$ $+\sum_{\ell=1}^{n}(\ell n^2)$
$c^1_V(4)$				$n^8+\ell$	$n^6+\ell$	Z
$c_E(1,2)$	$\mathcal{S}(v^1_\ell)$					$\sum_{v\in V_1\cup V_2}\mathcal{S}(v)$
$c_E(1,3)$			$\mathcal{S}(v^1_\ell)$			$\sum_{v\in V_1\cup V_3}\mathcal{S}(v)$
$c_E(1,4)$					$\mathcal{S}(v^1_\ell)$	$\sum_{v\in V_1\cup V_4}\mathcal{S}(v)$

Table 2 An illustration of the vectors contained in the sets $P^1_{EV}(2,\ell),\dots,P^1_{EV}(4,\ell)$ and the sets $P^1_V(2,\ell), P^1_V(3,\ell)$. For example the column for the set $P^1_{EV}(2,\ell)$ shows that the set contains two vectors, one whose only non-zero component $c^1_V(2)$ has the value $n^6+\ell$ and a second one whose only non-zero component $c_E(1,2)$ and has the value $\mathcal{S}(v^1_\ell)$. The last column provides the value for the target vector for the component given by the rows. Finally, the value Z is equal to $(n-1)n^8+n^6+\sum_{l=1}^{n}(\ell)$.

	$P^i_{EV}(j,\ell)$	$P^j_{EV}(i,\ell)$	$P_E(i,j)$	$\bar{t}$
$c^i_V(j)$	$n^6+\ell$			
$c^j_V(i)$		$n^6+\ell$		
$c_E(i,j)$	$\mathcal{S}(v^i_\ell)$	$\mathcal{S}(v^j_\ell)$	$\{\mathcal{S}(v)+\mathcal{S}(u) \mid \{v,u\}\in E_{i,j}\}$	$\sum_{v\in V_i\cup V_j}\mathcal{S}(v)$

Table 3 An illustration of the vectors contained in the sets $P^i_{EV}(j,\ell)$, $P^j_{EV}(i,\ell)$, and $P_E(i,j)$ and their interplay with the components $c^i_V(j)$, $c^j_V(i)$, and $c_V(i,j)$. For the conventions used in the table please refer to Table 2. Additionally, note that the column for $P_E(i,j)$ indicates that the set contains one vector for every edge $\{v,u\}\in E_{i,j}$, whose only non-zero component $c_E(i,j)$ has the value $\mathcal{S}(v)+\mathcal{S}(u)$.

- For every i, j', and ℓ with $1\le i\le k$, $1\le j'\le k-2$, and $1\le \ell\le n$, the *vertex set* $P^i_V(j,\ell)$, where $j=\mathsf{indJ}(i,j')$, containing two vectors $\bar{v}^+_{i,j,\ell}$ and $\bar{v}^-_{i,j,\ell}$ defined as follows:
  - if $j'=1$, then $\bar{v}^+_{i,j,\ell}[c^i_V(j)]=n^4-\ell$ and $\bar{v}^-_{i,j,\ell}[c^i_V(\mathsf{indJ}(i,j'+1))]=n^8+\ell+\ell n^2$ or
  - if $1<j'<k-2$, then $\bar{v}^+_{i,j,\ell}[c^i_V(j)]=n^4+\ell n^2$ and $\bar{v}^-_{i,j,\ell}[c^i_V(\mathsf{indJ}(i,j'+1))]=n^8+\ell+\ell n^2$ or
  - if $j'=k-2$, then $\bar{v}^+_{i,j,\ell}[c^i_V(j)]=n^4+\ell n^2$ and $\bar{v}^-_{i,j,\ell}[c^i_V(\mathsf{indJ}(i,j'+1))]=n^8+\ell$.

  We denote by $P^i_V(j)$, $P^i_{V+}(j)$, and $P^i_{V-}(j)$ the sets $\bigcup_{\ell=1}^{n}(P^i_V(j,\ell))$, $P^i_V(j)\cap\{\,\bar{v}^+_{i,j,\ell}\mid 1\le\ell\le n\,\}$, and $P^i_V(j)\setminus P^i_{V+}(j)$, respectively.
- For every i, j, and ℓ with $1\le i,j\le k$, $i\ne j$, and $1\le\ell\le n$, the *vertex incidence set* $P^i_{EV}(j,\ell)$, which contains the two vectors $\bar{a}^+_{i,j,\ell}$ and $\bar{a}^-_{i,j,\ell}$ such that $\bar{a}^+_{i,j,\ell}[c^i_V(j)]=n^6+\ell$ and $\bar{a}^-_{i,j,\ell}[c_E(i,j)]=\mathcal{S}(v^i_\ell)$.
  We denote by $P^i_{EV}(j)$, $P^i_{EV+}(j)$, and $P^i_{EV-}(j)$ the sets $\bigcup_{\ell=1}^{n}(P^i_{EV}(j,\ell))$, $P^i_V(j)\cap\{\,\bar{a}^+_{i,j,\ell}\mid 1\le\ell\le n\,\}$, and $P^i_{EV}(j)\setminus P^i_{EV+}(j)$, respectively.
- For every i, j with $1\le i<j\le k$, the *edge set* $P_E(i,j)$, which for every $e=\{v,u\}\in E_{i,j}$ contains the vector $\bar{e}$ such that $\bar{e}[c_E(i,j)]=\mathcal{S}(v)+\mathcal{S}(u)$; note that $P_E(i,j)$ is indeed a simple set, because $\mathcal{S}$ is a Sidon sequence.

Note that altogether there are $nk(k-2)+\binom{k}{2}+nk(k-1)=\binom{k}{2}+nk(2k-3)$ sets in $\mathcal{P}$.

Informally, the two vectors $\bar{v}^+_{i,j,\ell}$ and $\bar{v}^-_{i,j,\ell}$ in $P^i_V(j,\ell)$ represent the choice of whether or not the vertex v^i_ℓ should be included in a k-clique for G, i.e., if a solution for I chooses $v^+_{i,j,\ell}$ then v^i_ℓ should be part of a k-clique and otherwise not. The component $c^i_V(j)$, more specifically the value for $\bar{t}[c^i_V(j)]$, now ensures that a solution can choose at most one such vector in $P^i_{V+}(j)$. For instance, if $j = \mathsf{indMin}(i)$, then $\bar{t}[c^i_V(j)] = n^6 + n^4$, $\bar{v}^+_{i,j,\ell}[c^i_V(j)] = n^4 - \ell$, and $\bar{a}^+_{i,j,\ell}[c^i_V(j)] = n^6 + \ell$ for every ℓ with $1 \leq \ell \leq n$ and therefore at most one vector $\bar{v}^+_{i,j,\ell}$ in $P^i_{V+}(j)$ and at most one vector $\bar{a}^+_{i,j,\ell}$ in $P^i_{EV+}(j)$ can be choosen and those vectors must agree on ℓ. This also means that all but one of the vectors $\bar{v}^-_{i,j,1}, \dots, \bar{v}^-_{i,j,n}$ need to be chosen by a solution for I and this in turn signals the choice of the vertex for V_i to the next component, i.e., either the component $c^i_V(j+1)$ if $j+1 \neq i$ or the component $c^i_V(j+2)$ if $j+1 = i$. This is ensure by the value of the target vector for the next component and the carefully chosen values for the vectors $\bar{v}^-_{i,j,1}, \dots, \bar{v}^-_{i,j,n}$ which are non-zero at the next component. Note that we only need $k-2$ sets $P^i_V(j)$ for every i, because we need to copy the vertex choice for V_i to only $k-1$ components. A similar idea underlies the two vectors $\bar{a}^+_{i,j,\ell}$ and $\bar{a}^-_{i,j,\ell}$ in $P^i_{EV}(j,\ell)$, i.e., again the component $c^i_V(j)$ ensures that $\bar{a}^+_{i,j,\ell}$ can be chosen for only one of the sets $P^i_{EV}(j,1), \dots, P^i_{EV}(j,n)$ and $\bar{a}^-_{i,j,\ell}$ must be chosen for all the remaining ones. Note that the component $c^i_V(j)$ now also ensures that the choice made for the sets in $P^i_V(j)$ is the same as the choice made for the sets in $P^i_{EV}(j)$. Moreover, the choice made for the sets in $P^i_{EV}(j)$ is now propagated to the component $c_E(i,j)$ (instead of the next vertex component). Finally, the vectors in the set $P_E(i,j)$ represent the choice of the edge used in a k-clique between V_i and V_j and the component $c_E(i,j)$ ensures that only an edge, whose endpoints are the two vertices signalled by the sets $P^i_{EV}(j)$ and $P^j_{EV}(i)$ can be chosen. We are now ready to give a formal proof for the equivalence between the two instances.

This completes the construction of I. It is straightforward to verify that all sets in $\mathcal{P}$ are simple, I can be constructed in polynomial-time, and all the component values of all vectors are bounded by a polynomial in n. It hence only remains to show that (G,k) has a solution if and only if so does I.

Towards showing the forward direction, let $v^1_{\ell_1}, \dots, v^k_{\ell_k}$ with be the vertices of a k-clique of G and for every i and j with $1 \leq i < j \leq k$, let $e_{i,j}$ be the edge between $v^i_{\ell_i}$ and $v^j_{\ell_j}$ in G. We obtain a solution $S \subseteq \bigcup_{P \in \mathcal{P}} P$ for I with $\sum_{\bar{s} \in S} \bar{s} = \bar{t}$ and $|S \cap P| = 1$ for every $P \in \mathcal{P}$ by choosing the following vectors:

- For every i and j with $1 \leq i,j \leq k$ and $j \neq i$, we choose the vector $\bar{a}^+_{i,j,\ell_i}$ from the set $P^i_{EV}(j,\ell)$ as well as the vector $\bar{a}^-_{i,j,\ell}$ for every $\ell \neq \ell_1$.
- For every i and j with $1 \leq i,j \leq k$ and $j \notin \{i, \mathsf{indMax}(i)\}$, we choose the vector $\bar{v}^+_{i,j,\ell_i}$ from the set $P^i_V(j,\ell)$ as well as the vector $\bar{v}^-_{i,j,\ell}$ for every $\ell \neq \ell_i$.
- For every i and j with $1 \leq i < j \leq k$, we choose the vector $\bar{e}_{i,j}$ from the set $P_E(i,j)$.

Here and in the following we denote by $v^+_{i,j\ell}$, $v^-_{i,j\ell}$, $a^+_{i,j\ell}$, $a^-_{i,j\ell}$, e the value of the unique non-zero component of the vector $\bar{v}^+_{i,j\ell}$, $\bar{v}^-_{i,j\ell}$, $\bar{a}^+_{i,j\ell}$, $\bar{a}^-_{i,j\ell}$, and $\bar{e}$ respectively. Towards showing that S is indeed a solution for I, let $\bar{x} = \sum_{\bar{s} \in S} \bar{s}$ and note that:

- For every i and j with $1 \leq i,j \leq k$ and $i \neq j$, it holds that:

– if $j = \mathsf{indMin}(i)$, then:

$$\bar{x}[c_V^i(j)] = v_{i,j,\ell_i}^+ + a_{i,j,\ell_i}^+ = n^6 + \ell_i + n^4 - \ell_i = \bar{t}[c_V^i(j)],$$

– if $\mathsf{indMin}(i) < j < \mathsf{indMin}(i)$, let $j' = j-1$ if $j-1 \neq i$ and let $j' = j-2$ otherwise, then:

$$\begin{aligned}\bar{x}[c_V^i(j)] &= (\sum_{\ell\neq\ell_i} v_{i,j',\ell}^-) + (v_{i,j,\ell_i}^+) + (a_{i,j,\ell_i}^+)\\ &= ((n-1)n^8 + \sum_{\ell\neq\ell_i} \ell + \ell n^2) + (n^4 + \ell_i n^2) + (n^6 + \ell_i)\\ &= \bar{t}[c_V^i(j)],\end{aligned}$$

– otherwise (with j' as above):

$$\begin{aligned}\bar{x}[c_V^i(j)] &= (\sum_{\ell\neq\ell_i} v_{i,j',\ell}^-) + (a_{i,j,\ell_i}^+)\\ &= ((n-1)n^8 + \sum_{\ell\neq\ell_i} \ell) + (n^6 + \ell_i)\\ &= \bar{t}[c_V^i(j)].\end{aligned}$$

– For every i and j with $1 \leq i < j \leq k$, we obtain:

$$\begin{aligned}\bar{x}[c_E(i,j)] &= (\sum_{\ell\neq\ell_i} a_{i,j,\ell}^-) + (\sum_{\ell\neq\ell_j} a_{j,i,\ell}^-) + (e_{i,j})\\ &= (\sum_{\ell\neq\ell_i} \mathcal{S}(v_\ell^i)) + (\sum_{\ell\neq\ell_j} \mathcal{S}(v_\ell^j)) + (\mathcal{S}(v_{\ell_i}^i) + \mathcal{S}(v_{\ell_j}^j))\\ &= \bar{t}[c_E(i,j)].\end{aligned}$$

Therefore, S constitutes a solution for I.

Towards showing the reverse direction, let $S \subseteq \bigcup_{P\in\mathcal{P}} P$ be a solution for I, i.e., $\sum_{\bar{s}\in S} \bar{s} = \bar{t}$ and $|S \cap P| = 1$ for every $P \in \mathcal{P}$. We show the reverse direction using the following series of claims.

(C1) For every i and j with $1 \leq i,j \leq k$ and $i \neq j$, it holds that $|S \cap P_{EV+}^i(j)| = 1$ and $|S \cap P_{EV-}^i(j)| = n-1$. In the following let $\bar{v}_{EV}^i(j)$ be the unique vector in $S \cap P_{EV+}^i(j)$.

(C2) For every i and j' with $1 \leq i \leq k$ and $1 \leq j' \leq k-2$, it holds that $|S \cap P_{V+}^i(j)| = 1$ and $|S \cap P_{V-}^i(j)| = n-1$, where $j = \mathsf{indJ}(i,j')$. In the following let $\bar{v}_V^i(j)$ be the unique vector in $S \cap P_{V+}^i(j)$.

(C3) For every i, j and j' with $1 \leq i, j, j' \leq k$, $j \neq i$, $j' \neq i$, and $j' \neq j$, it holds that $\bar{v}^i_{EV}(j)[c^i_V(j)] = \bar{v}^i_{EV}(j')[c^i_V(j')]$; see (C1) for the definition of the vector $\bar{v}^i_{EV}(j)$. In particular, for i as above, there exists a unique value ℓ_i such that $\bar{v}^i_{EV}(j)[c^i_V(j)] = n^6 + \ell_i$ for every j as above.

(C4) For every i and j with $1 \leq i < j \leq k$, G contains an edge between $v^i_{\ell_i}$ and $v^j_{\ell_j}$.

(C5) The vertices $v^1_{\ell_1}, \dots, v^k_{\ell_k}$ induce a clique in G.

Towards showing (C1) consider the component $c^i_V(j)$. Note that $\bar{t}[c^i_V(j)]$ contains the term n^6 and moreover the only vectors of I having a non-zero component at $c^i_V(j)$ (apart from the vectors in $P^i_{EV+}(j)$) are the vectors in $P^i_{V-}(j')$ (only if $j >$ $\mathsf{indMin}(i)$), where $j' = j - 1$ if $j - 1 \neq i$ and $j' = j - 2$ otherwise, and the vectors in $P^i_{V+}(j')$ (only if $j < \mathsf{indMax}(i)$). The former all have values larger than $n^8 \geq n^6$ and the sum of all values of the latter is at most $\sum_{\ell=1}^{n} n^4 + \ell n^2 \leq 2n^5 < n^6$. Hence the only vectors that can contribute the term n^6 are the vectors in $P^i_{EV+}(j)$ and since n^6 appears exactly once in $\bar{t}[c^i_V(j)]$, (C1) follows. Recall that we denote by $\bar{v}^i_{EV}(j)$ the unique vector in $S \cap P^i_{EV+}(j)$.

Towards showing (C2) consider the component $c^i_V(j)$. Note that $\bar{t}[c^i_V(j)]$ contains the term n^4 and moreover the only vectors of I having a non-zero component at $c^i_V(j)$ (apart from the vectors in $P^i_{E+}(j)$) are the vectors in $P^i_{V-}(j')$ (only if $j >$ $\mathsf{indMin}(i)$), where $j' = j-1$ if $j-1 \neq i$ and $j' = j-2$ otherwise, and the vectors in $P^i_{EV-}(j)$. The former and the latter have values larger than n^8 and n^6, respectively. Hence the only vectors that can contribute the term n^4 are the vectors in $P^i_{V+}(j)$ and since n^4 appears exactly once in $\bar{t}[c^i_V(j)]$, (C2) follows. Recall that we denote by $\bar{v}^i_V(j)$ the unique vector in $S \cap P^i_{V+}(j)$.

Towards showing (C3), we show that $\bar{v}^i_{EV}(j)[c^i_V(j)] = \bar{v}^i_{EV}(j')[c^i_V(j')]$, where $j = \mathsf{indJ}(i, r)$ and $j' = \mathsf{indJ}(i, r+1)$ for every r with $1 \leq r < k$. Since we can assume that w.l.o.g. $k > 3$, we only need to distinguish the following three cases:

(A) $r = 1$ and $r + 1 < k - 1$,
(B) $r > 1$ and $r + 1 < k - 1$,
(C) $r > 1$ and $r + 1 = k - 1$.

For the case (A), consider the component $c^i_V(j)$. Note that due to (C1) and (C2), the vectors $\bar{v}^i_{EV}(j)$ and $\bar{v}^i_V(j)$ are the only vectors in S, for which the component $c^i_V(j)$ is non-zero; refer to (C1) and (C2) for a definition of these vectors. Hence, $\bar{v}^i_{EV}(j)[c^i_V(j)] + \bar{v}^i_V(j)[c^i_V(j)] = \bar{t}[c^i_V(j)] = n^6 + n^4$, which is only possible if $\bar{v}^i_{EV}(j) = \bar{a}^+_{i,j,\ell_1}$ and $\bar{v}^i_V(j) = \bar{v}^+_{i,j,\ell_1}$ for some ℓ_1 with $1 \leq \ell_1 \leq n$. Now consider the component $c^i_V(j')$. Because of (C2), we obtain that $\sum_{s \in S \cap P^i_{V-}(j)} s[c^i_V(j)] = (\sum_{\ell=1}^{n} n^8 + \ell + \ell n^2) - (n^8 + \ell_1 + \ell_1 n^2)$. Moreover, because of (C1) and (C2), we obtain that $(\sum_{\ell=1}^{n} n^8 + \ell + \ell n^2) - (n^8 + \ell_1 + \ell_1 n^2) + \bar{v}^i_{EV}(j') + \bar{v}^i_V(j') = \bar{t}[c^i_V(j')]$, which is only possible if $\bar{v}^i_{EV}(j') = \bar{a}^+_{i,j',\ell_1}$ and $\bar{v}^i_V(j') = \bar{v}^+_{i,j',\ell_1}$. Hence $\bar{v}^i_{EV}(j)[c^i_V(j)] = \bar{v}^i_{EV}(j')[c^i_V(j')]$, as required. The proof for the cases (B) and (C) is analogous.

Towards showing (C4) consider the component $c_E(i, j)$. Note that the set $P^i_{EV}(j)$, $P^j_{EV}(i)$, and $P_E(i, j)$ are the only sets in $\mathcal{P}$ containing vectors that are non-zero at $c_E(i, j)$. Moreover, because of (C1) it holds that $\sum_{\bar{s} \in S \cap P^i_{EV}(j)} \bar{s}[c_E(i, j)] =$

$(\sum_{\ell=1}^{n} \mathcal{S}(v_\ell^i)) - \mathcal{S}(v_{\ell_i}^i)$ and similarly $\sum_{\bar{s} \in S \cap P_{EV}^j(i)} \bar{s}[c_E(i,j)] = (\sum_{\ell=1}^{n} \mathcal{S}(v_\ell^j)) - \mathcal{S}(v_{\ell_j}^j)$. Since $\bar{t}[c_E(i,j)] = \sum_{v \in V_i \cup V_j} \mathcal{S}(v)$, we obtain that the unique vector $\bar{e} \in S \cap P_E(i,j)$ must satisfy $\bar{e}[c_E(i,j)] = \mathcal{S}(v_{\ell_i}^i) + \mathcal{S}(v_{\ell_j}^j)$, which due the properties of Sidon sequences is only possible if e is an edge between $v_{\ell_i}^i$ and $v_{\ell_j}^j$ in G. Finally, (C5) follows immediately from (C3) and (C4). $\square$

4 Result 1: Fixed-Parameter Tractability of sGASP

In this section we will establish that sGASP is FPT when parameterized by the number of agent types and the number of activities by proving Theorem 6.

Theorem 6 sGASP *can be solved in time* $\mathcal{O}(2^{|T(N)|\cdot(1+|A|)} \cdot ((|N|+|A|)|N|)^2)$.

Let $I = (N, A, (P_n)_{n \in N})$ be a sGASP instance and let $\pi : N \to A^*$ be an assignment of agents to activities. We denote by $G_I(\pi)$ the incidence graph between $T(N)$ and A, which is defined as follows. $G_I(\pi)$ has vertices $T(N) \cup A$ and contains an edge between an agent type $t \in T(N)$ and an activity $a \in A$ if $\pi_{t,a} \neq \emptyset$. We say that π is *acyclic* if $G_I(\pi)$ is acyclic.

Our first aim towards the proof of Theorem 6 is to show that if I has a stable assignment, then it also has an acyclic stable assignment (Lemma 9). We will then show in Lemma 11 that finding a stable assignment whose incidence graph is equal to some given acyclic pattern graph can be achieved in polynomial-time via a reduction to the TSS problem defined and solved in Subsection 3.1. Since the number of (acyclic) pattern graphs is bounded in our parameters, we can subsequently solve sGASP by enumerating all acyclic pattern graphs and checking for each of them whether there is an acyclic solution matching the selected pattern.

A crucial notion towards showing that it is sufficient to consider only acyclic solutions is the notion of (strict) compression. We say that an assignment τ is a *compression* of π if it satisfies the following conditions:

(C1) for every $t \in T(N)$ it holds that $|\pi_t| = |\tau_t|$,
(C2) for every $a \in A$ it holds that $|\pi^{-1}(a)| = |\tau^{-1}(a)|$, and
(C3) for every $a \in A$ it holds that the set of agent types which τ assigns to a is a subset of the agent types π assigns to a.

Moreover, if τ additionally satisfies:

(C4) $|E(G_I(\tau))| < |E(G_I(\pi))|$,

we say that τ is a *strict compression* of π.

Intuitively, an assignment τ is a compression of π if it maintains all the properties required to preserve stability and compatibility with a given subset $Q \subseteq T(N)$. We note that condition (C3) can be formalized as $T(\tau^{-1}(a)) \subseteq T(\pi^{-1}(a))$. Observe that if τ is a strict compression then there is at least one activity $a \in A$ such that $T(\tau^{-1}(a)) \subset T(\pi^{-1}(a))$. The following lemma shows that every assignment that is not acyclic admits a strict compression (and how it may be computed).

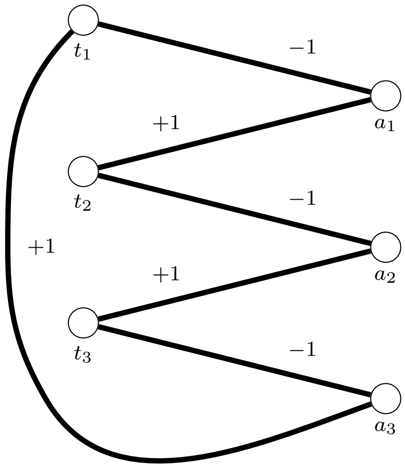

Fig. 1 Illustration of the modification (M) in the proof of Lemma 7 for a cycle of length six. A label of $+1$ on an edge $\{t, a\}$ of $G_I(\pi)$ means that $|\kappa_{t,a}|$ is by one larger than $|\pi_{t,a}|$. Similarly, a label of -1 on an edge $\{t, a\}$ of $G_I(\pi)$ means that $|\kappa_{t,a}|$ is by one smaller than $|\pi_{t,a}|$.

Lemma 7 *If π is not acyclic, then there exists an assignment τ that strictly compresses π.*

Proof Let $C = (t_1, a_1, \ldots, t_l, a_l, t_1)$ be a cycle of $G_I(\pi)$. Consider the following modification (M) of our instance: reassign one (arbitrary) agent in π_{t_1,a_1} to a_l, and for every i with $1 < i \leq l$ reassign one (arbitrary) agent in π_{t_i,a_i} to a_{i-1}. An illustration of (M) is also provided in Figure 1.

First, we show that the assignment κ obtained from π after applying modification (M) satisfies (C1)–(C3). Towards showing (C1), observe that $|\pi_t| = |\kappa_t|$ for any $t \in T(N) \setminus \{t_1, \ldots, t_l\}$. Moreover, for every i with $1 < i \leq l$, we have:

$$\begin{aligned}
|\kappa_{t_i}| &= \sum_{a \in A} |\kappa_{t_i,a}| \\
&= \big(\sum_{a \in (A \setminus \{a_i, a_{i-1}\})} |\kappa_{t_i,a}| \big) + |\kappa_{t_i,a_i}| + |\kappa_{t_i,a_{i-1}}| \big) \\
&= \big(\sum_{a \in (A \setminus \{a_i, a_{i-1}\})} |\pi_{t_i,a}| \big) + \big(|\pi_{t_i,a_i}| - 1\big) + \big(|\pi_{t_i,a_{i-1}}| + 1\big) \\
&= \sum_{a \in A} |\pi_{t_i,a}| = |\pi_{t_i}|
\end{aligned}$$

and for $i = 1$ we have:

$$
\begin{aligned}
|\kappa_{t_1}| &= \sum_{a \in A} |\kappa_{t_1,a}| \\
&= \big(\sum_{a \in (A \setminus \{a_1, a_l\})} |\kappa_{t_i,a})| + |\kappa_{t_1,a_1}| + |\kappa_{t_1,a_l}|\big) \\
&= \big(\sum_{a \in (A \setminus \{a_1, a_l\})} |\pi_{t_1,a}|\big) + (|\pi_{t_1,a_1}| - 1) + (|\pi_{t_1,a_l}| + 1) \\
&= \sum_{a \in A} |\pi_{t_1,a}| = |\pi_{t_1}|
\end{aligned}
$$

Towards showing (C2) note that $|\pi^{-1}(a)| = |\kappa^{-1}(a)|$ for every $a \in A \setminus \{a_1, \ldots, a_l\}$ and moreover for every i with $1 \leq i < l$ we obtain:

$$
\begin{aligned}
|\kappa^{-1}(a_i)| &= \sum_{t \in T(N)} |\kappa_{t,a_i}| \\
&= (\sum_{t \in (T(N) \setminus \{t_i, t_{i+1}\})} |\kappa_{t,a_i})| + |\kappa_{t_i,a_i}| + |\kappa_{t_{i+1},a_i}| \\
&= (\sum_{t \in (T(N) \setminus \{t_i, t_{i+1}\})} |\pi_{t,a_i})| + (|\pi_{t_i,a_i}| - 1) + (|\pi_{t_{i+1},a_i}| + 1) \\
&= \sum_{t \in T(N)} |\pi_{t,a_i}| = |\pi^{-1}(a_i)|
\end{aligned}
$$

and for $i = l$ we have:

$$
\begin{aligned}
|\kappa^{-1}(a_l)| &= \sum_{t \in T(N)} |\kappa_{t,a_l}| \\
&= (\sum_{t \in (T(N) \setminus \{t_l, t_1\})} |\kappa_{t,a_l})| + |\kappa_{t_l,a_l}| + |\kappa_{t_1,a_l}| \\
&= (\sum_{t \in (T(N) \setminus \{t_l, t_1\})} |\pi_{t,a_l})| + (|\pi_{t_l,a_l}| - 1) + (|\pi_{t_1,a_l}| + 1) \\
&= \sum_{t \in T(N)} |\pi_{t,a_l}| = |\pi^{-1}(a_l)|
\end{aligned}
$$

Because $|\kappa_{t,a}|$ and $|\pi_{t,a}|$ can only differ if $|\pi_{t,a}| \neq 0$, i.e., $|\kappa_{t,a}|$ will never be non-equal to zero if $|\pi_{t,a}| = 0$, we obtain that $|\kappa|$ satisfies also condition (C3).

Having settled that (M) does not violate (C1)–(C3), we observe that if $G_I(\kappa)$ still contains the cycle C, we can apply modification (M) again to κ and the obtained assignment still satisfies conditions (C1)–(C3). Hence we can repeatedly apply modification (M) as long as the cycle C is not destroyed in the resulting assignment. Namely let $m = \min\{\pi_{t_1,a_1}, \ldots, \pi_{t_l,a_l}\}$, let i be an index with $m = \pi_{t_i,a_i}$, and let τ be the assignment obtained after m applications of modification (M) to π. Note that m applications of modification (M) are possible since the cycle C remains preserved up to the $m - 1$-th modification of (M). Furthermore, since $|\tau_{t_i,a_i}| = 0$, we obtain that τ satisfies (C4). $\square$

The following lemma shows that any assignment can be compressed into an acyclic assignment.

Lemma 8 *Let $\pi : N \to A^*$ be an assignment for I. Then there exists an acyclic assignment π' that compresses π.*

Proof The lemma follows via an exhaustive application of Lemma 7. Namely, we start by checking whether π is acyclic. If yes, then π itself is the acyclic assignment that compresses π. If not, then we apply Lemma 7 to π and obtain the assignment π' that strictly compresses π. If π' is acyclic, we are done; otherwise, we repeat the above procedure with π' instead of π. Because at every step $|E(G_I(\pi))| < |E(G_I(\pi'))|$ and $G_I(\pi)$ has at most $|T(N)| \cdot |A|$ edges, this process concludes after at most $|T(N)| \cdot |A|$ steps and results in an acyclic assignment that compresses π. $\square$

The following lemma provides the first cornerstone for our algorithm by showing that it is sufficient to consider only acyclic solutions (and, in particular, shows how an acyclic solution can be constructed from an arbitrary solution). Intuitively, it is a consequence of Lemma 8 along with the observation that compression preserves stability and individual rationality.

Lemma 9 *If I has a stable assignment, then I has an acyclic stable assignment.*

Proof Let π be a stable assignment for I. Because of Lemma 8, there is an acyclic assignment π' that compresses π. We claim that π' is also a stable assignment. Conditions (C2) and (C3) together with the fact that π is individually rational imply that also π' is individually rational. Moreover, it follows from Condition (C1) that $\text{PE}(I, \pi) = \text{PE}(I, \pi')$, which together with Condition (C2) and the stability of π implies the stability of π'. $\square$

Our next step is the introduction of terminology related to the pattern graphs mentioned at the beginning of this section. Let G be a bipartite graph with bi-partition $\{T(N), A\}$. We say that G *models* an assignment $\pi : N \to A^*$ if $G_I(\pi) = G$; in this sense every such bipartite graph can be seen as a pattern (or model) for assignments. For a subset $Q \subseteq T(N)$ we say that G is *compatible* with Q if every vertex in Q and every vertex in $A_{\neq\emptyset}(I, Q)$ (recall the definition of $A_{\neq\emptyset}(I, Q)$ given in Lemma 1) has at least one neighbour in G; note that if G is compatible with Q then any assignment π modelled by G satisfies $\pi^{-1}(a) \neq \emptyset$ for every $a \in A_{\neq\emptyset}(I, Q)$. Intuitively, the graph G captures information about which types of agents are mapped to which activities (without specifying numbers), while Q captures information about which agent types are perfectly (i.e., "completely") assigned.

Let $Q \subseteq T(N)$ and let G be a bipartite graph with bi-partition $\{T(N), A\}$ that is compatible with Q. The following simple lemma shows that, modulo compatibility requirements, finding a stable assignment for I can be reduced to finding an individually rational assignment for $\gamma(I, Q)$ (recall the definition of $\gamma(I, Q)$ given in Lemma 1).

Lemma 10 *Let $Q \subseteq T(N)$ and let G be a bipartite graph with bi-partition $\{T(N), A\}$ that is compatible with Q. Then for every assignment $\pi : N \to A^*$ modelled by G and compatible with Q, it holds that π is stable for I if and only if π is individually rational for $\gamma(I, Q)$.*

Proof Since π is compatible with Q, it follows from Lemma 1 that π is stable for I if and only if π is individually rational for $\gamma(I,Q)$ and $\pi^{-1}(a) \neq \emptyset$ for every $a \in A_{\neq\emptyset}(I,Q)$. Consider an arbitrary $a \in A_{\neq\emptyset}(I,Q)$. Because G is compatible with Q it holds that every a has at least one neighbour in G and since π is modelled by G, we obtain that $\pi^{-1}(a) \neq \emptyset$, as required. □

The next, final lemma forms (together with Lemma 9) the core component of our proof of Theorem 6.

Lemma 11 *Let $Q \subseteq T(N)$ and let G be an acyclic bipartite graph with bipartition $\{T(N), A\}$ that is compatible with Q. Then one can decide in time $\mathcal{O}((|N|+|A|)^2|N|^2)$ whether I has a stable assignment which is modelled by G and compatible with Q.*

Proof By Lemma 10, I has a stable assignment that is modelled by G and compatible with Q if and only if $\gamma(I,Q)$ has an individually rational assignment that is modelled by G and compatible with Q. To determine whether $\gamma(I,Q)$ has an individually rational assignment that is compatible with Q and modelled by G, we will employ a reduction to the TSS problem, which can be solved in polynomial-time by Lemma 3.

The reduction proceeds as follows. We will construct an instance $I' = (T,\lambda)$ of TSS that is a YES-instance if and only if $\gamma(I,Q) = (N, A, (P'_n)_{n\in N})$ has an individually rational assignment that is compatible with Q and modelled by G. We set T to be the graph G and let λ be defined for every $v \in V(G)$ as follows:

- if $v \in Q$, then $\lambda(v) = \{|N_v|\}$,
- if $v \in T(N) \setminus Q$, then $\lambda(v) = \{0, \ldots, |N_v| - 1\}$,
- otherwise, i.e., if $v \in A$, then $\lambda(v) = \bigcap_{t\in T(N)\wedge\{v,t\}\in E(G)} P'_t(v)$.

Note that the reduction can be achieved in time $\mathcal{O}(|E(G)||N|)$, assuming that the sets $P'_t(a)$ for every $t \in T(N)$ and $a \in A$ are given in terms of a data structure that allows to test containment in constant time; such a data structure could for instance be a Boolean array with $|N|$ entries, whose i-th entry is TRUE if and only if i is contained in $P'_t(a)$. Because the time to construct $\gamma(I,Q)$ and $A_{\neq\emptyset}(I,Q)$ is at most $\mathcal{O}((|N|\cdot|A|)|N|)$ (Lemma 1) and the time to solve I' is at most $\mathcal{O}(|V(T)|^2\cdot|N|^2) = \mathcal{O}(|V(G)|^2\cdot|N|^2) = \mathcal{O}\big((|N|+|A|)^2|N|^2\big)$ (Lemma 3), we obtain $\mathcal{O}\big((|N|+|A|)^2|N|^2\big)$ as the total running time of the algorithm.

It remains to show that I' is a YES-instance if and only if I has an individually rational assignment that is compatible with Q and modelled by G. Towards showing the forward direction let $\alpha : E(T) \to \mathbb{N}_0$ be a solution for I'. We claim that the assignment $\pi : N \to A^*$ that for every edge $e = \{t,a\}$ of G (with $t \in T$ and $a \in A$) assigns exactly $\alpha(\{t,a\})$ agents of type t to activity a and assigns all remaining agents (if any) to $a_\emptyset$ is an individually rational assignment for $\gamma(I,Q)$ that is compatible with Q and modelled by G. First observe that for every $t \in T(N)$ and every $a \in A$ it holds that $\pi_{t,a} = \alpha(\{t,a\})$ if $\{t,a\} \in E(T)$ and $\pi_{t,a} = 0$ otherwise. Hence $G_I(\pi) = G = T$ which implies that π is modelled by G. We show next that π is also compatible with Q, i.e., π satisfies:

(P1) for every $t \in Q$, it holds that $\sum_{a\in A} |\pi_{t,a}| = |N_t|$, (P2) for every $t \in T(N) \setminus Q$, it holds that $\sum_{a\in A} |\pi_{t,a}| < |N_t|$,

Towards showing (P1) first note that because G is compatible with Q, it holds that every $t \in Q$ is adjacent to at least one edge in G and thus also in T. Moreover, because α is a solution for I', we obtain that $\sum_{e=\{t,a\}\in E(T)} \alpha(e) \in \lambda(t) = \{|N_t|\}$. Since $\sum_{e=\{t,a\}\in E(T)} \alpha(e) = \sum_{a\in A} |\pi_{t,a}|$, we obtain (P1).

Towards showing (P2) let $t \in T(N)\setminus Q$. If t is isolated in T then $\sum_{a\in A} |\pi_{t,a}| = 0$ but $N_t \neq \emptyset$ and hence $\sum_{a\in A} |\pi_{t,a}| < |N_t|$, as required. If on the other hand t is not isolated in T then because α is a solution for I', we obtain that $\sum_{e=\{t,a\}\in E(T)} \alpha(e) \in \lambda(t) = \{0,\dots,|N_t|-1\}$. Since $\sum_{e=\{t,a\}\in E(T)} \alpha(e) = \sum_{a\in A} |\pi_{t,a}|$, we obtain (P2).

Finally it remains to show that π is individually rational for $\gamma(I,Q)$, i.e., for every $a \in A$ and $t \in T(N)$, it holds that if $\pi_{t,a} \neq \emptyset$ then $|\pi^{-1}(a)| \in P'_t(a)$. Let $a \in A$. If a has no neighbour in T then $\pi_{t,a} = 0$ for every $t \in T(N)$ and the claim holds. Hence let $t_1,\dots,t_l$ be the neighbours of a in T. Then because α is a solution for I', we obtain:

$$\begin{aligned}
|\pi^{-1}(a)| &= \sum_{1\le i\le l} |\pi_{t_i,a}| \\
&= \sum_{1\le i\le l} \alpha(\{t_i,a\}) \\
&\in \lambda(a) = \bigcap_{1\le i\le l} P'_{t_i}(a)
\end{aligned}$$

Hence for every $1 \le i \le l$ it holds that $|\pi^{-1}(a)| \in P'_{t_i}(a)$, as required.

Towards showing the reverse direction let π be an individually rational assignment for $\gamma(I,Q)$ that is compatible with Q and modelled by G. We claim that the assignment $\alpha : E(G) \to \mathbb{N}_0$ with $\alpha(\{t,a\}) = \pi_{t,a}$ for every $\{t,a\} \in E(T)$ is a solution for I'. Observe that because π is modelled by G, we have that $|\pi_{t,a}| \neq 0$ if and only if $\{t,a\} \in E(G)$. Hence $|\pi_t| = \sum_{a\in A} |\pi_{t,a}| = \sum_{\{t,a\}\in E(T)} \alpha(t,a)$ for every $t \in T(i)$ and $|\pi^{-1}(a)| = \sum_{t\in T(N)} |\pi_{t,a}| = \sum_{\{t,a\}\in E(T)} \alpha(\{t,a\})$. Since π is compatible with Q we obtain that $\pi_t = |N_t|$ for every $t \in Q$ and hence $\alpha(t) = |N_t| \in \lambda(t)$. Moreover for every $t \in T(N) \setminus Q$ it holds that $\pi_t < |N_t|$ and hence $\alpha(t) \in \lambda(t)$. Because π is an individually rational assignment, it holds that $|\pi^{-1}(a)| \in \bigcap_{t\in T(\pi^{-1}(a))} P'_t(a)$ and thus $\sum_{\{t,a\}\in E(T)} \alpha(\{t,a\}) \in \lambda(a)$, as required. $\square$

We are now ready to establish Theorem 6.

Proof (Proof of Theorem 6) Let $I = (N, A, (P_n)_{n\in N})$ be the given instance of sGASP. It follows from Lemma 9 that it suffices to decide whether I has an acyclic stable assignment. Observe that every acyclic stable assignment π is compatible with $\mathrm{PE}(I,\pi)$ and the acyclic bipartite graph $G_I(\pi)$. Hence there is an acyclic stable assignment π if and only if there is a set $Q \subseteq T(N)$ and an acyclic bipartite graph G with bi-partition $\{T(N), A\}$ compatible with Q such that there is a stable assignment for I modelled by G and compatible with Q.

Consequently, we can determine the existence of a stable assignment for I by first branching over every $Q \subseteq T(N)$, then over every acyclic bipartite graph G

with bi-partition $\{T(N), A\}$ compatible with Q, and checking whether I has a stable assignment that is compatible with Q and modelled by G. Since there are $2^{|T(N)|}$ many subsets Q of $T(N)$ and at most $2^{|T(N)|\cdot|A|}$ (acyclic) bipartite graphs G, and we can determine whether I has a stable assignment compatible with Q and modelled by G in time $\mathcal{O}\big((|N|+|A|)^2|N|^2\big)$ (see Lemma 11), it follows that the total running time of the algorithm is at most $\mathcal{O}\big(2^{|T(N)|(1+|A|)}(|N|+|A|)^2|N|^2\big)$. □

5 Result 2: Lower Bound for sGASP

In this subsection we complement Theorem 6 by showing that if we drop the number of activities in the parameterization, then sGASP becomes W[1]-hard. We achieve this via a parameterized reduction from SMPSS that we have shown to be strongly W[1]-hard in Theorem 5.

Theorem 12 sGASP *is* W[1]*-hard parameterized by the number of agent types.*

Proof Let $(d, \bar{t}, \mathcal{S})$ with $\mathcal{S} = (S_1, \dots, S_m)$ be an instance of SMPSS. Because SMPSS is strongly W[1]-hard, we can assume that all numbers of the instance $(d, \bar{t}, \mathcal{S})$, i.e., the values of all components of the vectors in $\{\bar{t}\} \cup \bigcup_{R\in\mathcal{S}} R$, are encoded in unary. We will also assume that all non-zero components of the vectors $\{\bar{t}\} \cup \bigcup_{R\in\mathcal{S}} R$ are at least 3 (this can for instance be achieved by multiplying every vector in $\{\bar{t}\} \cup \bigcup_{R\in\mathcal{S}} R$ with the number 3). We will now construct the instance $I = (N, A, (P_n)_{n\in N})$ of sGASP in polynomial time with $|T(I)| = d + 3$ such that $(d, \bar{t}, \mathcal{S})$ has a solution if and only if so does I. The instance I has one agent type t_i for every $1 \leq i \leq d$ that comes with $\bar{t}[i]$ agents as well as three additional agent types t_P, $t^1_{\neq\emptyset}$, and $t^2_{\neq\emptyset}$ having one agent each. The last three agent types will be employed to ensure that every agent type t_i must be perfectly assigned and every activity apart from $a_\emptyset$ is assigned at least one agent. Moreover, I has one activity a_ℓ for every ℓ with $1 \leq \ell \leq m$ as well as one activity a_P, which will be used in conjunction with the agent type t_P to ensure that all agent types t_i must be perfectly assigned. The approval set for the agent type t_i w.r.t. an activity a_ℓ is given by $P_{t_i}(a_\ell) = \{\, \bar{v}[i] \mid \bar{v} \in S_\ell \wedge \bar{v}[i] \neq 0 \,\}$. Note that because all sets in $\mathcal{S}$ are simple, it holds that $P_{t_i}(a_\ell) \cap P_{t_j}(a_\ell) = \emptyset$ for every i, j, and ℓ with $1 \leq i, j \leq d$, $j \neq i$, and $1 \leq \ell \leq m$ and hence every such activity a_ℓ is populated by agents of at most one type in any individually rational assignment for I. Finally, we set $P_{t_i}(a_P) = \{2\}$ for every i with $1 \leq i \leq d$, $P_{t_P}(a_P) = \{1, 3\}$, $P_{t^1_{\neq\emptyset}}(a) = \{1\}$ and $P_{t^2_{\neq\emptyset}}(a) = \{2\}$ for every activity $a \in A\setminus\{a_P\}$. Note that indeed $|T(I)| = d+3$ and I can be constructed in polynomial-time (recall that we assumed that all numbers of $(d, \bar{t}, \mathcal{S})$ are encoded in unary). It remains to show that $(d, \bar{t}, \mathcal{S})$ has a solution if and only if so does I.

Towards showing the forward direction, let $\bar{p}_1, \dots, \bar{p}_m$ with $\bar{p}_\ell \in S_\ell$ and $\sum_{\ell=1}^m \bar{p}_\ell = \bar{t}$ be a solution for $(d, \bar{t}, \mathcal{S})$. Let $\pi : N \to A^*$ be the assignment defined as follows:

- For every agent type t_i and every vector $\bar{p}_\ell$ such that $\bar{p}_\ell[i] \neq 0$, π assigns exactly $\bar{p}_\ell[i]$ agents of type t_i to activity a_j.
- $\pi(n_P) = \{a_P\}$, where n_P is the unique agent of type t_P,

– $\pi(n^1_{\neq\emptyset}) = \pi(n^2_{\neq\emptyset}) = a_\emptyset$, where $n^1_{\neq\emptyset}$ and $n^2_{\neq\emptyset}$ are the unique agents having type $t^1_{\neq\emptyset}$ and $t^2_{\neq\emptyset}$, respectively.

We claim that π is a stable assignment for I, which we will show using the following sequence of observations:

(O1) Due to the construction of π, we obtain that $|\pi_{t_i,a_\ell}| = \bar{p}_\ell[i]$ for every i and j with $1 \leq i \leq d$ and $1 \leq \ell \leq m$.
(O2) Because of (O1) and the fact that $\sum_{\ell=1}^{m} \bar{p}_\ell = \bar{t}$, we obtain that $|\pi_{t_i}| = \bar{t}[i]$ for every i with $1 \leq i \leq d$. Since furthermore I has exactly $\bar{t}[i]$ agents of type t_i, we obtain that the agents of type t_i are perfectly assigned by π.
(O3) Because every set in $\mathcal{S}$ is simple, it also holds that $\pi^{-1}(a_\ell)$ consists of exactly $\bar{p}_\ell[i]$ agents of type i, where i is the unique non-zero component of $\bar{p}_\ell$.
(O4) Since $\pi^{-1}(a_P) = \{n_P\}$ the assignment π is stable for the agent n_P.
(O5) Since $\pi(n^1_{\neq\emptyset}) = \pi(n^2_{\neq\emptyset}) = a_\emptyset$ and $|\pi^{-1}(a)| \geq 2$ for every activity $a \in A \setminus \{a_P\}$ (because of (O3)), it holds that the assignment π is stable for the agents $n^1_{\neq\emptyset}$ and $n^2_{\neq\emptyset}$.
(O6) Consider an agent n of type t_i. Because of (O2), we have that $\pi(n) \neq a_\emptyset$. Hence $\pi(n) = a_\ell$ for some ℓ with $1 \leq \ell \leq m$ and since $|\pi^{-1}(a_\ell)| = \bar{p}_\ell[i]$ (because of (O3)), we have that $|\pi^{-1}(a_\ell)| \in P_n(a_\ell)$, which implies that π is a stable assignment for n.

Consequently, π is a stable assignment for I.

Towards showing the reverse direction, let $\pi : N \to A^*$ be a stable assignment for I. We start by showing that π satisfies the following two properties:

(P1) for every i with $1 \leq i \leq d$, all agents of type t_i are assigned to some activity in $A \setminus \{a_P\}$,
(P2) for every activity $a \in A \setminus \{a_P\}$, it holds that $\pi^{-1}(a) \neq \emptyset$ and moreover $T(\pi^{-1}(a)) \subseteq \{t_1, \dots, t_d\}$ and $|T(\pi^{-1}(a))| = 1$.

We will show (P1) and (P2) using the following series of claims that hold for any stable assignment $\pi : N \to A$ for I:

(C1) $\pi(n^2_{\neq\emptyset}) = a_\emptyset$ for the unique agent $n^2_{\neq\emptyset}$ of type $t^2_{\neq\emptyset}$,
(C2) $\pi(n^1_{\neq\emptyset}) = a_\emptyset$ for the unique agent $n^1_{\neq\emptyset}$ of type $t^1_{\neq\emptyset}$,
(C3) $\pi(n_P) = a_P$ for the unique agent n_P of type t_P,
(C4) $\pi^{-1}(a_P) = \{n_P\}$,

Towards showing (C1), assume for the contrary that $\pi(n^2_{\neq\emptyset}) = a$ for some $a \in A$. Then $a \in A \setminus \{a_P\}$ and $|\pi^{-1}(a)| = 2$. However, this is not possible since $n^2_{\neq\emptyset}$ is the only agent in I that approves size 2 for any activity in $A \setminus \{a_P\}$.

Towards showing (C2), assume for the contrary that $\pi(n^1_{\neq\emptyset}) = a$ for some $a \in A$. Then $a \in A \setminus \{a_P\}$ and $|\pi^{-1}(a)| = 1$. Moreover, because of (C1), we have that $\pi(n^2_{\neq\emptyset}) = a_\emptyset$ and hence $n^2_{\neq\emptyset}$ would prefer a over his current assignment, contradicting the stability of π.

Towards showing (C3), assume for the contrary that $\pi(n_P) \neq a_P$. Then $\pi(n_P) = a_\emptyset$. Moreover, due to the approval set of n_P, it must hold that $\pi^{-1}(a_P) \neq \emptyset$. Since $P_t(a_P) \in \{\emptyset, \{2\}\}$ for every agent type in $T(I) \setminus \{t_P\}$, it follows that $|\pi^{-1}(a_P)| =$

2. However, this contradicts the stability of π, since n_P would prefer a_P over his current assignment. (C4) is a direct consequence of (C3) since $P_{n_P}(a_P) = \{1, 3\}$ and moreover n_P is the only agent with $3 \in P_{n_P}(a_P)$.

We are now ready to show (P1) and (P2). Towards showing (P1) assume for a contradiction that there is an agent n whose type is in $\{t_1, \ldots, t_d\}$ such that $\pi(n) \in \{a_\emptyset, a_P\}$. Because of (C4), we obtain that $\pi(n) = a_\emptyset$. Moreover, since $2 \in P_n(a_P)$ and $|\pi^{-1}(a_P)| = 1$ (because of (C4)), it follows that n would prefer a_P over his current assignment, which contradicts the stability of π.

Towards showing (P2) assume for a contradiction that there is an activity $a \in A \setminus \{a_P\}$ with $\pi^{-1}(a) = \emptyset$. Consider the agent $n^1_{\neq\emptyset}$, i.e., the only agent of type $t^1_{\neq\emptyset}$, then because of (C2), we have $\pi(n^1_{\neq\emptyset}) = a_\emptyset$. Moreover, since $1 \in P_{n^1_{\neq\emptyset}}(a)$ for every $a \in A \setminus \{a_P\}$, the agent $n^1_{\neq\emptyset}$ would prefer a over his current assignment, which contradicts the stability of π. Hence $\pi^{-1}(a) \neq \emptyset$ for every $a \in A \setminus \{a_P\}$. Furthermore, since the agent types in $\{t_1, \ldots, t_d, t^1_{\neq\emptyset}, t^2_{\neq\emptyset}\}$ are the only types that approve of an activity in $A \setminus \{a_P\}$ and it follows from (C1) and (C2) that neither $n^1_{\neq\emptyset}$ nor $n^2_{\neq\emptyset}$ are assigned to an activity in $A \setminus \{a_P\}$, we obtain that $T(\pi^{-1}(a)) \subseteq \{t_1, \ldots, t_d\}$. It remains to show that $|T(\pi^{-1}(a_\ell))| = 1$ for every $1 \leq \ell \leq m$. Assume for a contradiction that there are two distinct agent types t and t' in $\{t_1, \ldots, t_d\}$ with $t, t' \in T(\pi^{-1}(a_\ell))$. Since π is individually rational, we obtain that $|\pi^{-1}(a_\ell)| \in P_t(a_\ell)$ and $|\pi^{-1}(a_\ell)| \in P_{t'}(a_\ell)$. Hence the set S_ℓ in $\mathcal{S}$ contains two vectors that share the same value at their non-zero component, which contradicts our assumption that S_ℓ is a simple set. This concludes the proof for (P1) and (P2) and we are now ready to complete the proof of the reverse direction.

Consider an activity a_ℓ for some $1 \leq \ell \leq m$. Because of (P2), we obtain that all agents in $\pi^{-1}(a_\ell)$ have the same type say t_i. Because π is stable it holds that $|\pi^{-1}(a_\ell)| \in P_{t_i}(a_\ell)$ and hence there is a vector, say $\bar{p}_\ell$, in S_ℓ with $\bar{p}_\ell[i] = |\pi^{-1}(a_\ell)|$. We claim that the vectors $\bar{p}_1, \ldots, \bar{p}_m$ chosen in this way form a solution for $(d, \bar{t}, \mathcal{S})$. Consider a component i with $1 \leq i \leq d$, then because of (P1), we obtain that $\sum_{j=1}^{m} \bar{p}_j[i]$ is equal to the number of agents of type t_i, which in turn is equal to $\bar{t}[i]$ by the construction of I. Hence $\sum_{j=1}^{m} \bar{p}_j = \bar{t}$ and $\bar{p}_1, \ldots, \bar{p}_m$ is a solution for $(d, \bar{t}, \mathcal{S})$. $\square$

6 Result 3: XP Algorithms for sGASP and GASP

In this section, we present our XP algorithm for GASP parameterized by the number of agent types. In order to obtain this result, we observe that the stability of an assignment for GASP can be decided by only considering the stability of agents that are assigned to a "minimal alternative" w.r.t. their type. We then show that once one guesses (i.e., branches over) a minimal alternative for every agent type, the problem of finding a stable assignment for GASP that is compatible with this guess can be reduced to the problem of finding a perfect and individually rational assignment for a certain instance of sGASP, where one additionally requires that certain activities are assigned to at least one agent. Our first task will hence be to obtain an XP algorithm which can find such a perfect and individually rational assignment for sGASP.

6.1 An XP Algorithm for sGASP

The aim of this section is twofold. First of all, we obtain Lemma 13, which allows us to find certain individually rational assignments in sGASP instances and forms a core part of our XP algorithm for GASP.

Lemma 13 *Let $I = (N, A, (P_n)_{n\in N})$ be an instance of* sGASP*, $Q \subseteq T(N)$, and $A_{\neq\emptyset} \subseteq A$. Then one can decide in time $\mathcal{O}(|A| \cdot (|N|)^{|T(N)|})$ whether I has an individually rational assignment π that is compatible with Q such that $\pi^{-1}(a) \neq \emptyset$ for every $a \in A_{\neq\emptyset}$.*

Proof Let $k = |T(N)|$ and $r = \max_{t\in T(N)} |N_t|$. We construct an instance $I' = (k, \mathcal{S}, \bar{t})$ of MPSS such that the solution to I' contains a specific vector if and only if $I = (N, A, (P_n)_{n\in N})$ has an individually rational assignment $\pi : N \to A^*$ that is compatible with Q and $\pi^{-1}(a) \neq \emptyset$ for every $a \in A_{\neq\emptyset}$. Let $T(N) = \{t_1, \ldots, t_k\}$. The set $\mathcal{S}$ contains one set S_a for every activity $a \in A$, defined as follows. For every number $p \in \bigcup_{t\in T(N)} P_t(a)$ the set S_a contains the set of all vectors $\bar{p} \in [r]_0^k$ such that $(\sum_{i=1}^{k} \bar{p}[i]) = p$ and $\bar{p}[i] = 0$ for every i with $1 \leq i \leq k$ such that $p \notin P_{t_i}(a)$. Moreover, if $a \notin A_{\neq\emptyset}$, then the set S_a additionally contains the all-zero vector $\bar{0}$. This completes the construction of I'.

We claim that I has an individually rational assignment $\pi : N \to A^*$ that is compatible with Q and $\pi^{-1}(a) \neq \emptyset$ for every $a \in A_{\neq\emptyset}$ if and only if the solution T for I' contains a vector $\bar{t}$ with $\bar{t}[i] = |N_{t_i}|$ for every $t_i \in Q$ and $\bar{t}[i] < |N_{t_i}|$ otherwise (i.e. for every $t_i \in T(N) \setminus Q$). Note that establishing this claim completes the proof of the lemma since I' can be constructed in time $\mathcal{O}(|A|r^k)$ and solved in time $\mathcal{O}(|A| \cdot r^k)$ by Lemma 4.

Towards showing the forward direction, let $\pi : N \to A^*$ be an individually rational assignment for I that is compatible with Q and $\pi^{-1}(a) \neq \emptyset$ for every $a \in A_{\neq\emptyset}$. For every $a \in A$ let $\bar{p}_a$ be the vector with $\bar{p}_a[i] = |\pi_{t_i,a}|$ for every i with $1 \leq i \leq k$. Note that if $\bar{p}_a \neq \bar{0}$ then $\bar{p}_a \in S_a$ for every $a \in A$ because π is individually rational. On the other hand, if $\bar{p}_a = \bar{0}$ then $a \notin A_{\neq\emptyset}$ and so we also obtain $\bar{p}_a \in S_a$. Hence the vector $\bar{t} = \sum_{a\in A} \bar{p}_a$ is in the solution for I' and moreover $\bar{t}[i] = \sum_{a\in A} \bar{p}_a[i] = \sum_{a\in A} |\pi_{t_i,a}| = |\pi_{t_i}|$ for every i with $1 \leq i \leq k$. Finally, because π is compatible with Q, we obtain that $\bar{t}[i] = |\pi_{t_i}| = |N_{t_i}|$ for every i with $t_i \in Q$ and also $\bar{t}[i] = |\pi_{t_i}| < |N_{t_i}|$ for every i with $t_i \in T(N) \setminus Q$, as required.

Towards showing the reverse direction, assume that the solution T for I' contains a vector $\bar{t}$ with $\bar{t}[i] = |N_{t_i}|$ for every $t_i \in Q$ and $\bar{t}[i] < |N_{t_i}|$ otherwise (i.e., for every $t_i \in T(N) \setminus Q$) and for every $a \in A$ let $\bar{p}_a$ be the vector in S_a such that $\sum_{a\in A} \bar{p}_a = \bar{t}$. We claim that the assignment $\pi : N \to A^*$ that for every $1 \leq i \leq k$ and every $a \in A$ assigns exactly $\bar{p}_a[i]$ agents of type t_i to activity a and all remaining agents to activity $a_\emptyset$ is an individually rational assignment for I that is compatible with Q and $\pi^{-1}(a) \neq \emptyset$ for every $a \in A_{\neq\emptyset}$. First observe that for every i with $1 \leq i \leq k$, it holds that $|\pi_{t_i}| = \sum_{a\in A} |\pi_{t_i,a}| = \sum_{a\in A} \bar{p}_a[i] = \bar{t}[i]$. Note also that because $\bar{t}[i] \leq |N_{t_i}|$, i.e., there is a sufficient number of agents for every agent type, we know that it is possible to assign the agents according to π. Since in addition it holds that $\bar{t}[i] = |N_{t_i}|$ if $t_i \in Q$ and $\bar{t}[i] < |N_{t_i}|$ if $t_i \notin Q$, it follows that π is compatible with Q. Moreover, because $\bar{0} \notin S_a$ for every $a \in A_{\neq\emptyset}$, it also holds that $|\pi^{-1}(a)| = \sum_{1 \le i \le k} |\pi_{t_i,a}| = \sum_{1 \le i \le k} \bar{p}_a[i] \neq 0$ for every such a. It remains to show that π is individually rational for I. By the definition of π it holds that whenever π assigns an agent of type t_i to some activity $a \in A$, then $\bar{p}_a[i] \neq 0$. Moreover, by the construction of I' it holds that if $\bar{p}_a[i] \neq 0$ then $(\sum_{i=1}^{k} \bar{p}_a[i]) \in P_{t_i}(a)$. Hence because $|\pi^{-1}(a)| = |p_a|$, we obtain that $|\pi^{-1}(a)| \in P_{t_i}(a)$. □

As a secondary result, we can already obtain an XP algorithm for sGASP parameterized by the number of agent types. This may also be of interest, as the obtained running time is strictly better than that of the algorithm obtained for the more general GASP. The last thing we need for this result is the following corollary, obtained as a direct consequence of Lemma 1 and Lemma 13.

Corollary 14 *Let $I = (N, A, (P_n)_{n\in N})$ be an instance of* sGASP *and $Q \subseteq T(N)$. Then one can decide in time $\mathcal{O}(|N|^2 \cdot |A| + |A| \cdot (|N|)^{|T(N)|})$ whether I has a stable assignment compatible with Q.*

We can now prove the following.

Theorem 15 *An instance $I = (N, A, (P_n)_{n\in N})$ of* sGASP *can be solved in time $|A| \cdot |N|^{\mathcal{O}(|T(N)|)}$.*

Proof Let $I = (N, A, (P_n)_{n\in N})$ be the given instance of sGASP. The algorithm loops through every $Q \subseteq T(N)$ and in each branch checks whether I has a stable assignment that is compatible with Q. Note that due to Corollary 14 this can be achieved in time $\mathcal{O}(|N|^2 \cdot |A| + |A| \cdot (|N|)^{|T(N)|})$ for every $Q \subseteq T(N)$. Since there are $2^{|T(N)|}$ subsets of $T(N)$, we obtain $\mathcal{O}(2^{|T(N)|}(|N|^2 \cdot |A| + |A| \cdot (|N|)^{|T(N)|})) = |A| \cdot |N|^{\mathcal{O}(|T(N)|)}$ as the total running time of the algorithm. □

6.2 An XP algorithm for GASP

Our aim here is to use Lemma 13 to obtain an XP algorithm for GASP. To simplify the presentation, we start by showing how a simple modification of GASP instances allows us to only consider perfect assignments and to express individual rationality in terms of NS-deviations. Namely, we say that a GASP instance $I = (N, A, (\succeq_n)_{n\in N})$ is *nice* if A contains a special activity a_ϕ such that $(a_\phi, i) \succeq_n (a_\emptyset, 1) \succeq_n (a_\phi, i)$ for every $n \in N$ and every integer i with $1 \le i \le |N|$. Clearly, one can transform an arbitrary GASP instance into a nice GASP instance in linear-time by adding the special activity a_ϕ to A and every preference list. Moreover, it is easy to see that the nice instance obtained in this way is equivalent to the originial instance with respect to stability, since any agent assigned to a_ϕ could instead be assigned to $a_\emptyset$ and because a_ϕ does not induce any new NS-deviations. More importantly, it also holds that a nice instance has a perfect stable assignment if and only if it has a stable assignment and that an assignment for a nice instance is stable if and only if no agent has an NS-deviation. The following observation and lemma summarizes the above-mentioned properties of nice instances.

Observation 16 *Let $I = (N, A, (\succeq_n)_{n\in N})$ be a* GASP *instance. Then, in time $\mathcal{O}(|N|)$, we can construct a nice* GASP *instance $I_\star$ such that I has a stable assignment if and only if $I_\star$ has a perfect stable assignment.*

Lemma 17 *Let $I = (N, A, (\succeq_n)_{n\in N})$ be a nice* GASP *instance. Then, an assignment $\pi : N \to A^*$ is stable if and only if no agent has an NS-deviation.*

Proof The forward direction is trivial because no agent has an NS-deviation in a stable assignment. Towards showing the backward direction, let $\phi : N \to A^*$ be an assignment for I such that no agent has an NS-deviation. Then, π is individually rational because if not there would be an agent $n \in N$ assigned to an activity $a \in A$ such that $(a_\emptyset, 1) \succ_n (a, |\pi^{-1}(a)|)$. However, this would imply that n has an NS-deviation to the activity a_ϕ, contradicting our assumption that no agent in I has an NS-deviation. □

We are now ready to introduce the notion of minimal alternative, which plays a crucial role in our algorithm. Let $I = (N, A, (\succeq_n)_{n\in N})$ be a nice GASP instance, let $\pi : N \to A^*$ be a perfect assignment and let $t \in T(I)$ be an agent type. We say that an alternative $(a, |\pi^{-1}(a)|)$ is *active* for t if $a \in \pi(t)$ (recall that $\pi(t)$ is the set of all activities being assigned at least one agent of type t). Moreover, we say that an active alternative $(a, |\pi^{-1}(a)|)$ is a *minimal alternative* for t if $(a', |\pi^{-1}(a')|) \succeq_t (a, |\pi^{-1}(a)|)$ for all active alternatives $(a', |\pi^{-1}(a')|)$ for t. We say that a is a *minimal activity* for t if $(a, |\pi^{-1}(a)|)$ is a minimal alternative. The following lemma now provides our first key insight, by showing that a non-stable perfect assignment for I is always accompanied by an NS-deviation of an agent in a minimal activity or of an agent to a minimal activity.

Lemma 18 *Let $I = (N, A, (\succeq_n)_{n\in N})$ be a nice* GASP *instance, let $\pi : N \to A$ be a perfect assignment and for every agent type $t \in T(I)$ let a_t be a minimal activity for t. Then, π is stable if and only if for each $t \in T(I)$:*

(P1) no agent of type t assigned to a_t has an NS-deviation, and
(P2) no agent of type t has an NS-deviation to a_t.

Proof It follows from Lemma 17 that π is stable if and only if no agent has an NS-deviation. Therefore, it remains to show that no agent has an NS-deviation if and only if π satisfies (P1) and (P2). The forward direction is trivial. Towards showing the reverse direction, suppose for a contradiction that there is an agent $n \in N$ of type t that has an NS-deviation to some activity $a \in A$. Because of (P2), it holds that $a \neq a_t$. But then every agent n' of type t participating in a_t also has an NS-deviation to a, and since a_t must be active by definition, this contradicts assumption (P1). □

Recall that $X = (A \times [|N|])$ is the set of all alternatives of a GASP instance I. The following theorem now employs the above lemma to construct an instance I' of sGASP together with a subset $A_{\neq\emptyset}$ of activities such that for every function $f_{\min} : T(I) \to X$ (representing every guess of minimal alternatives in an assignment), it holds that a nice GASP instance I has a stable assignment such that $f_{\min}(t)$ is a minimal alternative w.r.t. t for every $t \in T(I)$ if and only if I' has a perfect and individually rational assignment π such that $\pi^{-1}(a) \neq \emptyset$ for every $a \in A_{\neq\emptyset}$. For brevity, we will say that an assignment π is *compatible with* $f_{\min}$ if and only if $f_{\min}(t)$ is a minimal alternative w.r.t. t for every $t \in T(I)$.

Theorem 19 *Let $I = (N, A, (\succeq_n)_{n\in N})$ be a nice instance of* GASP *and let $f_{\min} : T(N) \to X \setminus \{(a_\emptyset, 1)\}$, which informally represents a guess of a minimal alternative for every agent type. Then one can in time $\mathcal{O}(|N|^2|A|)$ construct an instance $I' = (N, A, (P_n)_{n\in N})$ of* sGASP *together with a subset $A_{\neq\emptyset}$ of activities such that $|T(I')| \leq 2|T(I)|$ and I has a perfect stable assignment compatible with $f_{\min}(t)$ if and only if I' has a perfect individually rational assignment π with $\pi^{-1}(a) \neq \emptyset$ for every $a \in A_{\neq\emptyset}$.*

Proof Let $I = (N, A, (\succeq_n)_{n\in N})$ be a nice instance of GASP and let $f_{\min} : T(N) \to X \setminus \{(a_\emptyset, 1)\}$. We define the preferences $(P_n)_{n\in N}$ for I' as follows. Let $t \in T(I)$, let $(a_t, i_t) = f_{\min}(t)$ and let $n_t \in N_t$ be an arbitrary agent of type t. Then, we set $P_{n_t} = \{(a_t, i_t)\}$. This will ensure that every perfect assignment π for I' must have the property $\pi(n_t) = a_t$ and $(a_t, |\pi^{-1}(a_t)|) = (a_t, i_t)$.

Moreover, for every other agent n of type t, i.e., $n \in N_t \setminus \{n_t\}$, we define the preferences P_n as follows. Let X_t be the set of all alternatives that agent type t prefers at least as much as the minimal alternative $f_{\min}(t)$, i.e., $X_t = \{\, (a, i) \in X \mid (a, i) \succeq_t f_{\min}(t) \,\}$. Informally, starting from X_t we now remove all alternatives that are not allowed to be active by Lemma 18. Namely, let R be the set of all alternatives that would lead to an agent assigned to a minimal activity having an NS-deviation, i.e., $R = \bigcup_{t\in T(N)}(\{\, (a, i) \in X \mid (a, i+1) \succ_t f_{\min}(t) \,\})$. Note that because of Lemma 18 (P1), we know that no alternative in R can be activated by any perfect stable assignment for I. Let R_t be the set of alternatives in X_t such that if an agent of type t would be assigned to the corresponding activity he would have an NS-deviation to a_t (in any perfect assignment for I compatible with $f_{\min}$), i.e., $R_t = \{\, (a, i) \in X_t \mid (a_t, i_t + 1) \succ_t (a, i) \,\}$. Note that because of Lemma 18 (P2), it follows that no agent of type t can be assigned to an activity corresponding to such an alternative in any perfect stable assignment for I compatible with $f_{\min}$. Finally, we set $P_n = X_t \setminus (R \cup R_t)$. Informally, because of Lemma 18, P_n is the set of alternatives that an agent of type t is allowed to participate in with respect to any perfect stable assignment for I compatible with $f_{\min}$.

The set $A_{\neq\emptyset}$ contains all activities $a \in A$ for which there is an agent type t such that $(a, 1) \succ_t f_{\min}(t)$, i.e., all activities that must be non-empty for any stable assignment π for I compatible with $f_{\min}$.

This completes the construction of I' and $A_{\neq\emptyset}$. Clearly, the construction can be achieved in time $\mathcal{O}(|N|^2|A|)$ and moreover $|T(I')| \leq 2|T(I)|$. It remains to show that for every choice of $f_{\min}$, I has a perfect stable assignment π compatible with $f_{\min}(t)$ if and only if I' has a perfect individually rational assignment π with $\pi^{-1}(a) \neq \emptyset$ for every $a \in A_{\neq\emptyset}$.

Towards showing the forward direction, let π be a perfect stable assignment for I compatible with $f_{\min}(t)$. Note that, without loss of generality, we can assume that $\pi(n_t) = a_t$, where $(a_t, i_t) = f_{\min}(t)$ for every $t \in T(I)$, since otherwise we can switch the assignment for n_t with some agent that is assigned to a_t, which exists because π is compatible with $f_{\min}$. We claim that $\pi_1 : N \to A$ such that for every

$n \in N$, $\pi_1(n) = \pi(n)$, is a perfect individually rational assignment for I' such that $\pi_1^{-1}(a) \neq \emptyset$ for every $a \in A_{\neq\emptyset}$. The assignment π_1 is obviously perfect for I', because π is perfect for I.

Towards showing that π_1 is individually rational, let $n \in N$ be any agent of type t. If $n = n_t$, then $\pi_1(n) = \pi(n) = a_t$ (using our assumption that $\pi(n_t) = a_t$). Moreover, because π is compatible with $f_{\min}$, we obtain that $|\pi^{-1}(a_t)| = |\pi_1^{-1}(a_t)| = i_t$ and therefore $(\pi_1(n), \pi_1^{-1}(\pi_1(n))) = (a_t, i_t) \in P_n = \{(a_t, i_t)\}$, as required. If on the other hand $n \neq n_t$, then because π is a perfect assignment compatible with $f_{\min}$, we obtain that $(\pi(n), |\pi^{-1}(\pi(n))|) \in X_t$. Moreover, using Lemma 18 and the fact that π is stable, we obtain that $(\pi(n), |\pi^{-1}(\pi(n))|) \notin R \cup R_t$. Therefore, $(\pi_1(n), |\pi_1^{-1}(\pi_1(n))|) = (\pi(n), |\pi^{-1}(\pi(n))|) \in P_n = X_t \setminus (R \cup R_t)$, as required.

To complete the argument for the forward direction, it remains to show that $\pi_1^{-1}(a) \neq \emptyset$ for every $a \in A_{\neq\emptyset}$. Assume for a contradiction that there is an activity $a \in A_{\neq\emptyset}$ such that $\pi_1^{-1}(a) = \emptyset$. Note that then also $\pi^{-1}(a) = \emptyset$ and because of the definition of $A_{\neq\emptyset}$ it follows that there is an agent type t such that $(a, 1) \succ_t f_{\min}(t)$. But this contradicts the stability of π since now each agent assigned to the minimal activity a_t (which must have received at least one agent) has an NS-deviation to a in I.

Towards showing the reverse direction, let π_1 be a perfect individually rational assignment for I' such that $\pi_1^{-1}(a) \neq \emptyset$ for every $a \in A_{\neq\emptyset}$. We claim that the assignment π with $\pi(n) = \pi_1(n)$ is a perfect stable assignment for I that is compatible with $f_{\min}$. Clearly, π is perfect because so is π_1. We show next that π is compatible with $f_{\min}$. Let t be an arbitrary agent type and consider the special agent n_t with $P_{n_t} = \{(a_t, i_t)\}$. Then, because π_1 is perfect and individually rational, we obtain that $\pi(n_t) = \pi_1(n_t) = a_t$ and $|\pi^{-1}(a_t)| = |\pi_1^{-1}(a_t)| = i_t$. Therefore, the alternative (a_t, i_t) is active for t in π. Moreover, because $P_n \subseteq X_t = \{\, (a, i) \in X \mid (a, i) \succeq_t f_{\min}(t) \,\}$ for every agent n of type t, we obtain from the individual rationality of π_1 that π assigns all agents of type t to alternatives (a, i) with $(a, i) \succeq_t f_{\min}(t)$. Therefore, π is compatible with $f_{\min}$.

Towards showing that π is also stable, assume for a contradiction that this is not the case. Then, because of Lemma 18 either (P1) or (P2) does not hold. In the former case, there is an agent n of type t assigned to a_t having an NS-deviation to some activity $a \in A \setminus \{a_t\}$. But then, $(a, |\pi^{-1}(a)|) \in R$, which implies that $(a, |\pi^{-1}(a)|) \notin P_{n'}$ for any agent $n' \in N$. If additionally $|\pi^{-1}(a)| \neq 0$, this contradicts the individual rationality of π_1 for any agent in $\pi^{-1}(a)$. Moreover, if $|\pi^{-1}(a)| = 0$, then $a \in A_{\neq\emptyset}$ contradicting our assumption that $\pi_1^{-1}(a') \neq \emptyset$ for every $a' \in A_{\neq\emptyset}$.

In the latter case (corresponding to (P2)), there is an agent n of type t that has an NS-deviation to a_t. However, then $(a_t, i_t + 1) \in R_t$ and therefore $(a_t, i_t + 1) \notin P_t$, which contradicts the indivual rationality of π_1. $\square$

We can now proceed to the main result of this section.

Theorem 20 *An instance* $I = (N, A, (\succeq_n)_{n \in N})$ *of* GASP *can be solved in time* $(|A| \cdot |N|)^{\mathcal{O}(|T(I)|)}$.

Proof Let $I_\star$ be the nice GASP instance obtained from I in time $\mathcal{O}(|N|)$ using Observation 16, which has a perfect stable assignment if and only if I has a stable assignment. The algorithm now enumerates all of the at most $((|A| + 1) \cdot |N|)^{|T(I)|}$

possible functions $f_{\min}$ and for each such function $f_{\min}$ the algorithm uses Theorem 19 to construct the instance $I' = (N, A, (P_n)_{n\in N})$ of SGASP from $I_\star$ with $|T(I')| \leq 2|T(I)|$ together with the set $A_{\neq\emptyset}$ of activities in time $\mathcal{O}(|N|^2(|A|+1))$. It then uses Lemma 13 to decide whether I' has a perfect individually rational assignment π_1 such that $\pi_1^{-1}(a) \neq \emptyset$ for every $a \in A_{\neq\emptyset}$ in time $\mathcal{O}((|A|+1)(|N|)^{|T(I')|}) = \mathcal{O}((|A|+1)(|N|^{2|T(I)|})$. If this is true for at least one of the functions $f_{\min}$, the algorithm returns that I has a stable assignment, and correctly returns no otherwise. The total running time of the algorithm is hence $\mathcal{O}((|A|+1)^{|T(I)|} \cdot |N|^{2|T(I)|}) = (|A| \cdot |N|)^{\mathcal{O}(|T(I)|)}$. □

7 Result 4: Lower Bound for GASP

This section presents our hardness result for GASP. In particular, we show that GASP is unlikely to be fixed-parameter tractable parameterized by both the number of activities and the number of agent types.

Theorem 21 GASP *is* W[1]*-hard parameterized by the number of activities and the number of agent types.*

Proof We will employ a parameterized reduction from the PARTITIONED CLIQUE problem, which is well-known to be W[1]-complete [23].

PARTITIONED CLIQUE	
Input:	An integer k, a k-partite graph $G = (V, E)$ with partition $\{V_1, \dots, V_k\}$ of V into sets of equal size.
Parameter:	k
Question:	Does G have a k-clique, i.e., a set $C \subseteq V$ of k vertices such that $\forall u, v \in C$, with $u \neq v$ there is an edge $\{u, v\} \in E$?

We denote by $E_{i,j}$ the set of edges of G that have one endpoint in V_i and one endpoint in V_j and we assume w.l.o.g. that $|V_i| = n$ and $|E_{i,j}| = m$ for every i and j with $1 \leq i < j \leq k$ (see, e.g., [3, Theorem 13.7] for a justification of these assumptions).

Given an instance (G, k) of PARTITIONED CLIQUE with partition $V_1, \dots, V_k$, we construct an equivalent instance $I = (N, A, (\succeq_n)_{n\in N})$ of GASP in polynomial time with $\binom{k}{2} + k$ activities and $2k + 1$ agent types.

The instance I has the following activities:

- For every i with $1 \leq i \leq k$ the activity a_i, whose size in a stable assignment for I will be used to identify the vertex in V_i chosen to be part of a k-clique in G.
- For every i and j with $1 \leq i < j \leq k$ the activity $a_{i,j}$, whose size in a stable assignment for I will be used to identify the edge in $E_{i,j}$ chosen to be part of a k-clique in G.

For every i and j with $1 \leq i < j \leq k$ let α_i be a bijection from V_i to the set $\{3, 5, \dots, 2(n-1)+1, 2n+1\}$ and similarly let $\alpha_{i,j}$ be a bijection from $E_{i,j}$ to the set $\{1, 3, \dots, 2m-1\}$. The main ideas behind the reduction are as follows. First the reduction ensures that for every stable assignment $\pi : N \to A^*$ for I the size $s = |\pi^{-1}(a_i)|$ of activity a_i uniquely identifies a vertex in V_i, i.e., the vertex $\alpha_i^{-1}(s)$, and the size $s = |\pi^{-1}(a_{i,j})|$ of activity $a_{i,j}$ uniquely identifies an edge in $E_{i,j}$, i.e., the edge $\alpha_{i,j}^{-1}(s)$. Employing a set of "special agents" and their associated preference lists, the reduction will then ensure that the vertices identified by the sizes of the activities $a_1, \dots, a_k$ are endpoints of all the edges identified by the sizes of the activities $a_{1,2}, a_{1,3}, \dots, a_{k-1,k}$, which implies that these vertices form a k-clique in G.

For a pair i, j of numbers, we denote by $o(i,j)$, the (ordered) pair i, j if $i \leq j$ and the (ordered) pair j, i otherwise. We will now partition the set of alternatives into certain equivalence classes, which will help us present the preference lists constructed in the reduction. Namely, we define the following equivalence classes:

- For $x \in \{1, 2\}$ we define the set $C_A^x = \{\, (a_i, x) \mid 1 \leq i \leq k \,\}$,
- For every i with $1 \leq i \leq k$, we define the following sets:
  - $C_V(i) = \{\, (a_i, \alpha_i(v)) \mid v \in V_i \,\}$,
  - $C_V^{+1}(i) = \{\, (a_i, \alpha_i(v) + 1) \mid v \in V_i \,\}$,
- For every i and j with $1 \leq i < j \leq k$, we define the set $C_E(i,j) = \{\, (a_{i,j}, \alpha_{i,j}(e)) \mid e \in E_{i,j} \,\}$,
- For every i and $v \in V_i$, we define the set $C_I(i,v) = \{\, (a_i, \alpha_i(v)) \,\} \cup \{\, (a_{o(i,j)}, \alpha_{o(i,j)}(e) + 1) \mid 1 \leq j \leq k \wedge j \neq i \wedge e \in E_{i,j} \wedge v \in e \,\}$, i.e., $C_I(i,v)$ contains the tuple $(a_i, \alpha(v))$ and all tuples $(a_{o(i,j)}, \alpha_{o(i,j)}(e) + 1)$ such that $j \neq i$ and the edge $e \in E_{o(i,j)}$ is incident to v.

We let $C_V = \bigcup_{i=1}^k C_V(i)$ and $C_E = \bigcup_{1 \leq i < j \leq k} C_E(i,j)$.

We are now ready to define the required preference lists. When defining a preference list we will only list the equivalence classes that are more or equally preferred to the alternative $(a_\emptyset, 1)$ and assume that all remaining alternatives, i.e., all alternatives that are not listed, are less preferred than $(a_\emptyset, 1)$[3].

- The *validity* preference list, denoted by P_{VAL}, defined as $C_V \cup C_E > C_A^2 > C_A^1 > (a_\emptyset, 1)$. Informally, P_{VAL} is crucial in ensuring that $|\pi^{-1}(a_i)| \in \{\, \alpha_i(v) \mid v \in V_i \,\}$ and $|\pi^{-1}(a_{i,j})| \in \{\, \alpha_{i,j}(e) \mid e \in E_{i,j} \,\}$ for every stable assignment π for I and every i and j with $1 \leq i < j \leq k$.
- For every i with $1 \leq i \leq k$, let $v_1, \dots, v_u$ be the unique ordering of the vertices in V_i in ascending order w.r.t. α_i. We define the following two preference lists for every i with $1 \leq i \leq k$:
  - The *forward-vertex* preference list, denoted by $P_V^{\rightarrow}(i)$, defined as $C_V^{+1}(i) > C_I(i, v_u) > C_I(i, v_{u-1}) > \cdots > C_I(i, v_1) > (a_\emptyset, 1)$. Informally, $P_V^{\rightarrow}(i)$ is crucial to ensure that for every j with $1 \leq j \leq k$ and $j \neq i$ the edge e with $\alpha_{o(i,j)}(e) = |\pi^{-1}(a_{o(i,j)})|$ is not adjacent with any vertex $v' \in V_i$ such that $\alpha_i(v') > \alpha_i(v)$ for the vertex v ($v \in e$) with $\alpha_i(v) = |\pi^{-1}(a_i)|$. This intuition will be made precise in Claim 1.
  - The *backward-vertex* preference list, denoted by $P_V^{\leftarrow}(i)$, is defined as $C_V^{+1}(i) > C_I(i, v_1) > C_I(i, v_2) > \cdots > C_I(i, v_u) > (a_\emptyset, 1)$. Informally, $P_V^{\leftarrow}(i)$ is crucial to ensure that for every j with $1 \leq j \leq k$ and $j \neq i$ the edge e with $\alpha_{o(i,j)}(e) = |\pi^{-1}(a_{o(i,j)})|$ is not adjacent with any vertex $v' \in V_i$ such

[3] Preference lists of this form are sometimes called Ballester encodings [2]

that $\alpha_i(v') < \alpha_i(v)$ for the vertex v ($v \in e$) with $\alpha_i(v) = |\pi^{-1}(a_i)|$. This intuition will be made precise in Claim 1.

Informally, $P_V^{\rightarrow}(i)$ and $P_V^{\leftarrow}(i)$ together ensure that for every j with $1 \leq j \leq k$ and $j \neq i$, the edge e with $\alpha_{o(i,j)}(e) = |\pi^{-1}(a_{o(i,j)})|$ is adjacent with the vertex v with $\alpha_i(v) = |\pi^{-1}(a_i)|$. This intuition will be made precise in Claim 1

We are now ready to define the set N of agents:

- for every i with $1 \leq i \leq k$:
  - one agent $n_i^{\rightarrow}$ with preference list $P_V^{\rightarrow}(i)$ and
  - one agent $n_i^{\leftarrow}$ with preference list $P_V^{\leftarrow}(i)$.
- a set N_V of $\binom{k}{2}(2m-1) + k(2n+1) + 1$ agents with preference list P_{VAL}.

This completes the construction of the instance I. Clearly the given reduction can be achieved in polynomial-time. Moreover, since I has exactly $\binom{k}{2} + k$ activities and exactly $2k + 1$ distinct types of preference lists, both parameters are bounded by a function of k, as required. It remains to show that G has a k-clique if and only if I has a stable assignment.

Towards showing the forward direction let $C = \{v_1, \ldots, v_k\}$ be a k-clique of G such that $v_i \in V_i$ for every i with $1 \leq i \leq k$ and for every i and j with $1 \leq i < j \leq k$ let $e_{i,j}$ be the edge between v_i and v_j in G. We claim that the assignment $\pi : N \rightarrow A^*$ defined in the following is a stable assignment for I. We set:

- $\pi(n_i^{\rightarrow}) = \pi(n_i^{\leftarrow}) = a_i$ and
- for every i and j with $1 \leq i < j \leq k$, π assigns exactly $\alpha_{i,j}(e_{i,j})$ agents from N_V to activity $a_{i,j}$,
- for every i with $1 \leq i \leq k$, π assigns exactly $\alpha_i(v_i) - 2$ agents from N_V to activity a_i,
- all remaining agents (which are only in N_V) are assigned to $a_\emptyset$.

Note that $|\pi^{-1}(a_{i,j})| = \alpha_{i,j}(e_{i,j})$ i and j with $1 \leq i < j \leq k$ and $|\pi^{-1}(a_i)| = \alpha_i(v_i) - 2 + 2 = \alpha_i(v_i)$ for every i with $1 \leq i \leq k$.

Towards showing that the assignment π is stable, we consider any agent n and distinguish the following cases:

- if n is one of $n_i^{\rightarrow}$ or $n_i^{\leftarrow}$ for some i with $1 \leq i \leq k$, then π is stable w.r.t. to n because for every j with $1 \leq j \leq k$ and $j \neq i$, the edge $e_{o(i,j)}$ is incident with v_i.
- if $n \in N_V$, we consider the following cases:
  - ($\pi(n) = a_i$) In this case the assignment is stable w.r.t. n because the tuple $(a_i, |\pi^{-1}(a_i)|) = (a_i, \alpha_i(v_i))$ is in the most preferred equivalence class of P_{VAL}.
  - ($\pi(n) = a_{i,j}$) In this case the assignment is stable w.r.t. n because the tuple $(a_{i,j}, |\pi^{-1}(a_{i,j})|) = (a_{i,j}, \alpha_{i,j}(e_{i,j}))$ is in the most preferred equivalence class of P_{VAL}.
  - ($\pi(n) = a_\emptyset$) In this case the assignment is stable w.r.t. n because the tuples $(a_i, |\pi^{-1}(a_i)| + 1) = (a_i, \alpha_i(v_i) + 1)$ and $(a_{i,j}, |\pi^{-1}(a_{i,j})| + 1) = (a_{i,j}, \alpha_{i,j}(e_{i,j}) + 1)$ are less preferred than the tuple $(a_\emptyset, 1)$ in the preference list P_{VAL} for n (for every i and j with $1 \leq i < j \leq k$).

Towards showing the backward direction, we start by formalizing the intuition given above about the preference lists $P_V^{\leftarrow}(i)$ and $P_V^{\rightarrow}(i)$.

Claim 1 *Let i be an integer with $1 \leq i \leq k$ and let π be a stable assignment for I satisfying:*

(A1) $|\pi^{-1}(a_i)| \in \{\, \alpha_i(v) \mid v \in V_i \,\}$ *and*
(A2) for every j with $1 \leq j \leq k$ and $j \neq i$ it holds that $|\pi^{-1}(a_{o(i,j)})| \in \{\, \alpha_{o(i,j)}(e) \mid e \in E_{o(i,j)} \,\}$.

Then the following holds for π:

(C1) $\pi(n_i^{\rightarrow}) = \pi(n_i^{\leftarrow}) = a_i$.
(C2) For every j with $1 \leq j \leq k$ and $j \neq i$, the unique edge $e_{o(i,j)} \in E_{o(i,j)}$ with $\alpha_{o(i,j)}(e) = |\pi^{-1}(a_{o(i,j)})|$ is incident with the unique vertex $v \in V_i$ such that $\alpha(v) = |\pi^{-1}(a_{o(i,j)})|$.

In order to establish the claim, assume that both assumption (A1) and assumption (A2) holds, and recall that i and v are defined in the claim statement. Towards showing (C1) assume for a contradiction that this is not the case, i.e., one of the agents $n_i^{\rightarrow}$ or $n_i^{\leftarrow}$, in the following denoted by n is not assigned to a_i. Because $(a_i, \alpha(v) + 1) \in C_V^{+1}(i)$ and $C_V^{+1}(i)$ only contains alternatives for activity a_i, the agent n would prefer to change from his current activity to activity a_i contradicting our assumption that π is stable.

Towards showing (C2) assume for a contradiction that this is not the case and let j be an index witnessing this, i.e., $v \notin e_{o(i,j)}$. Let v' be the vertex in V_i that is incident with $e_{o(i,j)}$. We distinguish two analogous cases (note that $v' \neq v$): (1) $\alpha(v') > \alpha(v)$ and (2) $\alpha(v') < \alpha(v)$. In the former case π would not be stable because the agent $n_i^{\rightarrow}$ would prefer to join activity $a_{i,j}$ over his current activity a_i; this is because $\alpha(v') > \alpha(v)$ and hence the equivalence class $C_I(i, v')$, which contains the tuple $(a_{o(i,j)}, \alpha_{o(i,j)}(e) + 1)$, is more preferred in $P_V^{\rightarrow}(i)$ than the equivalence class $C_I(i, v)$, which contains the tuple $(a_i, \alpha_i(v))$. The proof for the latter case is analogous, using the agent $n_i^{\leftarrow}$ instead of the agent $n_i^{\rightarrow}$.

Note that once we show that the assumptions (A1) and (A2) hold for every i with $1 \leq i \leq k$, property (C2) will ensure that the vertices $\alpha^{-1}(|\pi^{-1}(a_1)|), \ldots, \alpha^{-1}(|\pi^{-1}(a_k)|)$ form a k-clique in G. Indeed:

Claim 2 *For every stable assignment $\pi : N \to A^*$ for I, it holds that:*

(A0) $\pi(u) = a_\emptyset$ *for at least one agent $u \in N_V$.*
(A1) $|\pi^{-1}(a_i)| \in \{\, \alpha_i(v) \mid v \in V_i \,\}$ *for every i with $1 \leq i \leq k$.*
(A2) $|\pi^{-1}(a_{i,j})| \in \{\, \alpha_{i,j}(e) \mid e \in E_{i,j} \,\}$ *for every i and j with $1 \leq i < j \leq k$.*

Towards showing (A0), assume for a contradiction that this is not the case, i.e., all $\binom{k}{2}(2m-1)+k(2n+1)+1$ agents in N_V are assigned to one of the $\binom{k}{2}+k$ activities in A. Then there either exists an activity $a_{i,j}$ such that more than $2m-1$ agents in N_V are assigned to $a_{i,j}$ by π or there exists an activity a_i such that more than $2n+1$ agents in N_V are assigned to a_i by π. In the former case let $u \in N_V$ be an agent with $\pi(u) = a_{i,j}$. Then the assignment is not stable because n would prefer being assigned to $a_\emptyset$ over its current assignment to $a_{i,j}$. The latter case is analogous. This completes the proof for (A0).

Because of (A0) there is at least one agent $u \in N_V$ such that $\pi(u) = a_\emptyset$. Hence because of the preference list P_{VAL} for n, we obtain that $|\pi^{-1}(a_i)| \notin \{0,1,2\} \cup \{\, \alpha_i(v) - 1 \mid v \in V_i \,\}$, since otherwise n would prefer activity a_i over $a_\emptyset$. It follows that either $|\pi^{-1}(a_i)| \in \{\, \alpha_i(v) \mid v \in V_i \,\}$ or $|\pi^{-1}(a_i)| > \max\{\, \alpha_i(v) \mid v \in V_i \,\} = 2n+1$. In the former case (A1) holds, so assume that the latter case applies. Since we can assume w.l.o.g. that $2n + 1 > 2k$ (and there are only $2k$ agents in $N \setminus N_V$), we obtain that there is at least one agent $u \in N_V$ such that $\pi(u) = a_i$. But then because of the preference list P_{VAL} of n, n would prefer activity $a_\emptyset$ over a_i, contradicting our assumption that π is a stable assignment. The completes the proof of (A1). The proof of (A2) is analogous to the proof of (A1). □

8 Result 5: Lower Bound for gGASP

From our previous result in conjunction with the equivalence between GASP and gGASP on complete networks, we can immediately conclude that gGASP is also W[1]-hard parameterized by the number of agent types and the number of activities. However, here we strengthen this result by providing a modified reduction which establishes the W[1]-hardness of the problem even when one additionally parameterizes by the vertex cover number of the network. As noted in the introduction, this also implies the W[1]-hardness of the problem when parameterized by the treewidth of the network, a question raised in previous work [16]; in fact, the presented lower-bound result not only shows the (conditional) fixed-parameter intractability of the problem with a more restrictive graph parameter, but also when additionally parameterizing by the number of agent types.

Theorem 22 *gGASP is W[1]-hard parameterized by the number of activities, the number of agent types, and the vertex cover number of the network.*

Proof The proof is via a parameterized reduction from PARTITIONED CLIQUE, i.e., given an instance (G,k) of PARTITIONED CLIQUE with partition $V_1, \dots, V_k$, we construct an equivalent instance $I = (N, A, (\succeq_n)_{n \in N}, L)$ of gGASP in polynomial time with $\binom{k}{2} + k$ activities, $\binom{k}{2} + 3k$ agent types, and whose network (N,L) has vertex cover number at most $\binom{k}{2} + 2k$.

The main ideas behind the reduction are quite similar to the reduction used in the proof of Theorem 21. The main differences is that to ensure that the network (N,L) has a small vertex cover number, it is necessary to split the set N_V of agents used in the previous reduction, into sets N_i and $N_{i,j}$ for every i and j with $1 \leq i < j \leq k$ such that the agents in a set N_i can only be assigned to activity a_i (or $a_\emptyset$) and the agents in a set $N_{i,j}$ can only be assigned to activity $a_{i,j}$ (or $a_\emptyset$). This way the agents $n_i^{\rightarrow}$ and $n_i^{\leftarrow}$ only need to be connected to agents in N_i and $N_{i,j}$ (for any $j \neq i$) but not with all agents in N_V.

Let I' be the instance of GASP as defined in the proof of Theorem 21. Since the instance I is defined quite similar to the instance I', we will refer to I' for the construction of I. In particular, I has the same set of activities as I', i.e., I has one activity a_i for every i with $1 \leq i \leq k$ and one activity $a_{i,j}$ for every i and j with $1 \leq i < j \leq k$. For every i and j with $1 \leq i < j \leq k$ let α_i be a bijection from V_i to the set $\{3, 5, \dots, 2(n-1)+1, 2n+1\}$ and similarly let $\alpha_{i,j}$ be a bijection from $E_{i,j}$ to the set $\{3, 5, \dots, 2m+1\}$. Note that α_i and $\alpha_{i,j}$ are defined almost the same as in the proof of Theorem 21, the only difference being the definition of the image $\alpha_{i,j}$, which is now $\{3, 5, \dots, 2m+1\}$ instead of $\{1, 3, \dots, 2m-1\}$.

For the definition of the preference lists, we will mainly use the equivalence classes defined in the proof of Theorem 21, i.e., the classes $C_V(i)$, C_V^{+1}, $C_E(i,j)$, and $C_I(i,v)$. Apart from those we will also need a slightly modified version of the equivalence class $C_V^{+1}(i)$, which we denote by $C_V^{+1,2}(i)$ and set to $\{(a_i, 2)\} \cup C_V^{+1}(i)$. We are now ready to define the preference lists required by the reduction.

- For every i with $1 \leq i \leq k$, let $v_1, \dots, v_u$ be the unique ordering of the vertices in V_i in ascending order w.r.t. α_i. We define the following two preference lists for every i with $1 \leq i \leq k$:
  - The *vertex-validity* preference list, denoted by $P^V_{\text{VAL}}(i)$, defined as $C_V(i) > (a_i, 2) > (a_i, 1) > (a_\emptyset, 1)$.
  - The *forward-vertex* preference list, denoted by $P_V^{\rightarrow}(i)$, defined as $C_V^{+1,2}(i) > C_I(i, v_u) > C_I(i, v_{u-1}) > \cdots > C_I(i, v_1) > (a_\emptyset, 1)$.
  - The *backward-vertex* preference list, denoted by $P_V^{\leftarrow}(i)$, defined as $C_V^{+1,2}(i) > C_I(i, v_1) > C_I(i, v_2) > \cdots > C_I(i, v_u) > (a_\emptyset, 1)$.
- For every i and j with $1 \leq i < j \leq k$ the *edge-validity* preference list, denoted by $P^E_{\text{VAL}}(i,j)$, and defined as $C_E(i,j) > (a_{i,j}, 2) > (a_{i,j}, 1) > (a_\emptyset, 1)$.

We are now ready to define the set N of agents:

- for every i with $1 \leq i \leq k$:
  - one agent $n_i^{\rightarrow}$ with preference list $P_V^{\rightarrow}(i)$,
  - one agent $n_i^{\leftarrow}$ with preference list $P_V^{\leftarrow}(i)$, and
  - a set N_i of $2n+3$ agents with preference list $P^V_{\text{VAL}}(i)$.
- for every i and j with $1 \leq i < j \leq k$, a set $N_{i,j}$ of $2m+3$ agents with preference list $P^E_{\text{VAL}}(i,j)$. In the following let $n_{i,j}$ be one of the agents in $N_{i,j}$.

Finally, the links L between the agents are given by:

- for every i with $1 \leq i \leq k$:
  - a link between $n_i^{\rightarrow}$ and $n_i^{\leftarrow}$,
  - for every $n \in N_i$ a link between $n_i^{\rightarrow}$ and n,
  - for every j with $1 \leq j \leq k$ and $j \neq i$ a link between $n_i^{\rightarrow}$ and $n_{o(i,j)}$,
- for every i, j with $1 \leq i < j \leq k$ and every $n \in N_{i,j} \setminus \{n_{i,j}\}$ a link between $n_{i,j}$ and n.

An illustration of the network (N, L) as defined above is given in Figure 2.

This completes the construction of the instance I. Observe that the set $\{\, n_i^{\rightarrow} \mid 1 \leq i \leq k \,\} \cup \{\, n_{i,j} \mid 1 \leq i < j \leq k \,\}$ is a vertex cover of the network (N, L) of size at most $\binom{k}{2} + k$, and hence the network has vertex cover number at most $\binom{k}{2} + k$. Clearly the given reduction can be achieved in polynomial-time. Moreover, since I has exactly $\binom{k}{2} + k$ activities, exactly $\binom{k}{2} + 3k$ distinct types of preference lists, and the vertex cover number of the network (N, L) is at most $\binom{k}{2} + 2k$, all parameters are

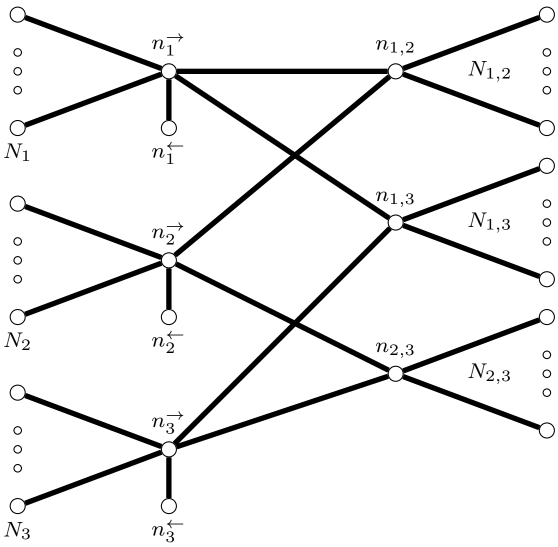

Fig. 2 An illustration of the network (N, L) obtained in the reduction of Theorem 22 for the case that $k = 3$.

bounded by a function of k, as required. It remains to show that G has a k-clique if and only if I has a stable assignment.

Towards showing the forward direction let $C = \{v_1, \ldots, v_k\}$ be a k-clique of G such that $v_i \in V_i$ for every i with $1 \leq i \leq k$ and for every i and j with $1 \leq i < j \leq k$ let $e_{i,j}$ be the edge between v_i and v_j in G. We claim that the assignment $\pi : N \rightarrow A^*$ defined in the following is a stable assignment for I. We set:

- $\pi(n_i^{\rightarrow}) = \pi(n_i^{\leftarrow}) = a_i$,
- for every i with $1 \leq i \leq k$, π assigns exactly $\alpha_i(v_i) - 2$ agents from N_i to activity a_i,
- for every i and j with $1 \leq i < j \leq k$, $\pi(n_{i,j}) = a_{i,j}$, and
- for every i and j with $1 \leq i < j \leq k$, π assigns exactly $\alpha_{i,j}(e_{i,j}) - 1$ agents from $N_{i,j}$ to activity $a_{i,j}$,
- all remaining agents are assigned to $a_\emptyset$.

Note that for every i and j with $1 \leq i < j \leq k$ the agents assigned to activities a_i and $a_{i,j}$ are connected (see Figure 2 for an illustration of why this is the case) and moreover $|\pi^{-1}(a_{i,j})| = \alpha_{i,j}(e_{i,j})$ and $|\pi^{-1}(a_i)| = \alpha_i(v_i)$. Let n be an agent, we consider the following cases:

- if n is one of $n_i^{\rightarrow}$ or $n_i^{\leftarrow}$ for some i with $1 \leq i \leq k$, then the assignment π is stable w.r.t. to n because for every j with $1 \leq j \leq k$ and $j \neq i$, the edge $e_{o(i,j)}$ is incident to v_i in G and hence the tuples $(a_i, \alpha_i(v_i))$ and $(a_{o(i,j)}, \alpha_{o(i,j)}(e_{o(i,j)}) + 1)$ are in the same equivalence class of $P_V^{\rightarrow}$ and $P_V^{\leftarrow}$.
- if $n \in N_i$ and $\pi(u) = a_i$, then the assignment π is stable w.r.t. n because the tuple $(a_i, \alpha_i(v_i))$ is in the most preferred equivalence class of $P_{\text{VAL}}^V(i)$.

- if $n \in N_i$ and $\pi(u) = a_\emptyset$, then the assignment π is stable w.r.t. n because $(a_\emptyset, 1)$ is preferred to $(a_i, \alpha_i(v_i) + 1)$ as well as to any tuple with any other activity in $P^V_{\text{VAL}}(i)$.
- if $n \in N_{i,j}$ and $\pi(n) = a_{i,j}$, then the assignment π is stable w.r.t. n because the the tuple $(a_{i,j}, \alpha_{i,j}(e_{i,j}))$ is in the most preferred equivalence class of $P^E_{\text{VAL}}(i,j)$.
- if $n \in N_{i,j}$ and $\pi(n) = a_\emptyset$, then the assignment π is stable w.r.t. n because $(a_\emptyset, 1)$ is preferred to $(a_{i,j}, \alpha_{i,j}(e_{i,j}) + 1)$ as well as to any tuple with any other activity in $P^E_{\text{VAL}}(i,j)$.

The reverse direction follows immediately from the following claim.

Claim 3 *For every stable assignment $\pi : N \to A^*$ for I, it holds that:*

(C1) for every i with $1 \leq i \leq k$, there is an agent $n \in N_i$ such that $\pi(n) = (a_\emptyset, 1)$,
(C2) for every i and j with $1 \leq i < j \leq k$, there is an agent $n \in N_{i,j}$ such that $\pi(n) = (a_\emptyset, 1)$,
(C3) $|\pi^{-1}(a_i)| \in \{\, \alpha_i(v) \mid v \in V_i \,\}$ for every i with $1 \leq i \leq k$,
(C4) $\pi(n_i^{\rightarrow}) = \pi(n_i^{\leftarrow}) = a_i$ for every i with $1 \leq i \leq k$,
(C5) $|\pi^{-1}(a_{i,j})| \in \{\, \alpha_{i,j}(e) \mid e \in E_{i,j} \,\}$ for every i and j with $1 \leq i < j \leq k$,
(C6) $\pi(n_{i,j}) = a_{i,j}$ for every i and j with $1 \leq i < j \leq k$,
(C7) $\alpha_i^{-1}(|\pi^{-1}(a_i)|) \in \alpha^{-1}_{o(i,j)}(|\pi^{-1}(a_{o(i,j)})|)$ for every i and j with $1 \leq i, j \leq k$ and $i \neq j$.

Towards showing (C1) suppose for a contradiction that this is not the case. Then because all agents in N_i must either be assigned to a_i or $a_\emptyset$ by π, we obtain that all $2n+3$ agents in N_i must be assigned to a_i. However such an assignment would not be stable because any tuple (a_i, x) with $x \geq 2n+3$ is less preferred than $(a_\emptyset, 1)$ in every preference list.

The proof of (C2) is very similar to the proof of (C1). Namely, suppose for a contradiction that this is not the case. Then all $2m+3$ agents in $N_{i,j}$ must be assigned to $a_{i,j}$, however such an assignment would not be stable because any tuple $(a_{i,j}, x)$ with $x \geq 2m+3$ is less preferred than $(a_\emptyset, 1)$ in every preference list.

Towards showing (C3) we first show that $|\pi^{-1}(a_i)| > 1$. Because of (C1) there is an agent $n \in N_i$ with $\pi(n) = a_\emptyset$. Hence $|\pi^{-1}(a_i)| \neq 0$ since otherwise the agent n would prefer to switch to a_i. Now suppose for a contradiction that $|\pi^{-1}(a_i)| = 1$. Because the agents in N_i are the only agents that prefer the tuple $(a_i, 1)$ over $(a_\emptyset, 1)$, it holds that $\pi^{-1}(a_i) \subseteq N_i$. But since the tuple $(a_i, 2)$ is a tuple that is in the highest equivalence class $C_V^{+1,2}(i)$ of the preference list $P_i^{\rightarrow}$ for $n_i^{\rightarrow}$ and $n_i^{\rightarrow}$ is linked with every vertex in N_i, the agent $n_i^{\rightarrow}$ would prefer to switch to a_i, contradicting our assumption that π is stable. Consequently $|\pi^{-1}(a_i)| > 1$ and we show next that $\pi^{-1}(a_i)$ contains $n_i^{\rightarrow}$. Observe that the agents in $\{n_i^{\leftarrow}, n_i^{\rightarrow}\} \cup N_i$ are the only agents in N that can be assigned to a_i; all other agents prefer the tuple $(a_\emptyset, 1)$ over any tuple involving the activity a_i. Since $|\pi^{-1}(a_i)| > 1$ the set $\pi^{-1}(a_i)$ can only be connected if it contains $n_i^{\rightarrow}$. Hence we have $|\pi^{-1}(a_i)| > 1$ and $n_i^{\rightarrow} \in \pi^{-1}(a_i)$. Because of (C1) there is an agent $n \in N_i$ with $\pi(n) = a_\emptyset$. Since n is linked with $n_i^{\rightarrow}$ and prefers a_i over $a_\emptyset$, whenever $|\pi^{-1}(a_i)| \in \{0,1\} \cup \{\, \alpha_i(v) - 1 \mid v \in V_i \,\} = \{0, 1, 2, 4, 6, \ldots, 2n\}$, we obtain that either $|\pi^{-1}(a_i)| \in \{\, \alpha_i(v) \mid v \in V_i \,\} = \{3, 5, \ldots, 2n+1\}$ or $|\pi^{-1}(a_i)| > 2n+1$. In the former case (C3) holds, so assume that the latter case applies. Since the agents in $\{n_i^{\leftarrow}, n_i^{\rightarrow}\} \cup N_i$ are the only agents in N that can be assigned to a_i and we can assume w.l.o.g. that $2n+1 > 2$, we obtain that there is at least one agent $u \in N_i$ such that $\pi(u) = a_i$. But then because of the preference list $P_V(i)$ of n, n would prefer activity $a_\emptyset$ over a_i, contradicting our assumption that π is a stable assignment. The completes the proof of (C3).

Towards showing (C5) first observe that the agents in $\{\, n_i^{\leftarrow}, n_i^{\rightarrow} \mid 1 \leq i \leq k\} \cup N_{i,j}$ are the only agents in N that can be assigned to $a_{i,j}$; all other agents prefer the tuple $(a_\emptyset, 1)$ over any tuple involving the activity $a_{i,j}$. Moreover because of (C4), we obtain that only the agents in $N_{i,j}$ can actually be assigned to $a_{i,j}$. Moreover because all agents in $N_{i,j}$ must either be assigned to $a_{i,j}$ or to $a_\emptyset$ and due to (C2) there is always an agent $n' \in N_{i,j}$ with $\pi(n') = a_\emptyset$, and since the set $N_{i,j}$ is connected by L, we obtain that there is always an agent $n \in N_{i,j}$ with $\pi(n) = a_\emptyset$ that is linked to an agent in $\pi^{-1}(a_{i,j})$. Hence it follows from the preference list $P_E(i,j)$ of the agents in $N_{i,j}$ that $|\pi^{-1}(a_{i,j})| \notin \{0,1\} \cup \{\, \alpha_{i,j}(e) - 1 \mid e \in E_{i,j}\,\}$, since otherwise the agent n would prefer to switch to $a_{i,j}$. Hence either $|\pi^{-1}(a_{i,j})| \in \{\, \alpha_{i,j}(e) \mid e \in E_{i,j}\,\} = \{3, 5, \dots, 2m+1\}$ or $|\pi^{-1}(a_{i,j})| > 2m+1$. In the former case (C5) holds, so assume that the latter case applies. Then there is an agent $n \in N_{i,j}$ with $\pi(n) = a_{i,j}$, but since $|\pi^{-1}(a_{i,j})| > 2m+1$ the agent would prefer to be assigned to $a_\emptyset$, contradicting our assumption that the assignment π is stable. This completes the proof of (C5).

(C6) can be obtained as follows. Because of (C4), we have that $|\pi^{-1}(a_{i,j})| \geq 3$. Since (as observed already in the proof of (C5)) only the agents in $N_{i,j}$ can be assigned to activity $a_{i,j}$, we obtain that $\pi(n_{i,j}) = a_{i,j}$ since otherwise $\pi^{-1}(a_{i,j})$ would not be connected.

Towards showing (C7) assume for a contradiction that there are i and j with $1 \leq i,j \leq k$ and $i \neq j$ such that $v \notin e$, where $v = \alpha_i^{-1}(|\pi^{-1}(a_i)|)$ and $e = \alpha_{o(i,j)}(|\pi^{-1}(a_{o(i,j)})|)$. Observe first that because of (C3) and (C4) v and e are properly defined and moreover $v \in V_i$ and $e \in E_{i,j}$. Let v' be the endpoint of e in V_i, which because $v \notin e$ is not equal to v. We distinguish two analogous cases: (1) $\alpha_i(v') < \alpha_i(v)$ and (2) $\alpha_i(v') > \alpha_i(v)$. In the former case consider the agent $n_i^{\leftarrow}$. Because of (C4) and (C6), it holds that $\pi(n_i^{\leftarrow}) = a_i$ and $\pi(n_{i,j}) = a_{o(i,j)}$, which implies that $n_i^{\leftarrow}$ is linked with an agent, namely the agent $n_{i,j}$ assigned to $a_{o(i,j)}$ by π. Together with the facts that the equivalence class $C_I(i,v')$ contains the tuple $(a_{o(i,j)}, \alpha(e))$, the equivalence class $C_I(i,v)$ contains the tuple $(a_i, \alpha(v))$, and $C_E(i,v')$ is preferred over $C_E(i,v)$ in the preference list for $n_i^{\leftarrow}$, we obtain that the agent $n_i^{\leftarrow}$ prefers the activity $a_{o(i,j)}$ over its current activity a_i, contradicting the stability of π. The proof for the latter case is analogous using the agent $n_i^{\rightarrow}$ instead of the agent $n_i^{\leftarrow}$. This completes the proof for (C7). $\square$

9 Result 6: sGASP Parameterized by the Number of Agents

As mentioned at the end of the Introduction, the parameterized complexity of sGASP when the number of agents is taken as the parameter is the last question that remains open for the considered problems and parameterizations. We settle this question by providing a fixed-parameter algorithm.

Theorem 23 sGASP *can be solved in time* $\mathcal{O}(|N|^{|N|} \cdot \sqrt{|N \cup A|} \cdot |N||A|)$*, and hence is fixed-parameter tractable parameterized by the number of agents.*

Proof Let $I = (N, A, (P_i)_{i \in N})$ be a sGASP instance with $n = |N|$. The main idea behind the algorithm is to guess (i.e., branch over) the set $M_\emptyset$ of agents that are assigned to $a_\emptyset$ as well as a partition $\mathcal{M}$ of the remaining agents, i.e., the agents in $N \setminus M_\emptyset$, and then check whether there is a stable assignment π for I such that:

(P1) $\pi^{-1}(a_\emptyset) = M_\emptyset$ and
(P2) $\{\, \pi^{-1}(a) \mid a \in A \,\} \setminus \{\emptyset\} = \mathcal{M}$, i.e., $\mathcal{M}$ corresponds to the grouping of agents into activities by π.

Since there are at most n^n possibilities for $M_\emptyset$ and $\mathcal{M}$ and those can be enumerated in time $\mathcal{O}(n^n)$, it remains to show how to decide whether there is a stable assignent for I satisfying (P1) and (P2) for any given $M_\emptyset$ and $\mathcal{M}$. Towards showing this, we first consider the implications for a stable assignment resulting from assigning the agents in $M_\emptyset$ to $a_\emptyset$. Namely, let P'_n for every $n \in N$ be the approval set obtained from P_n after removing every alternative (a, i) (where a is an activity and i and integer) such that $i \neq 0$ and there is an agent $n_\emptyset \in M_\emptyset$ with $(a, i+1) \in P_{n_\emptyset}$. Moreover, let $A_{\neq\emptyset}$ be the set of all activities that cannot be left empty if the agents in $M_\emptyset$ are assigned to $a_\emptyset$, i.e., the set of all activities such that there is an agent $n_\emptyset \in M_\emptyset$ with $(a, 1) \in P_{n_\emptyset}$. Now consider a set $M \in \mathcal{M}$, and observe that the set A_M of activities that the agents in M can be assigned to in any stable assignment satisfying (P1) and (P2) is given by: $A_M = \{\, a \mid |M| \in \bigcap_{n \in M} P_n(a)' \,\}$. Let B be the bipartite graph having $\mathcal{M}$ on one side and A on the other side and having an edge between a vertex $M \in \mathcal{M}$ and a vertex $a \in A$ if $a \in A_M$. We claim that I has a stable assignment satisfying (P1) and (P2) if and only if B has a matching that saturates $\mathcal{M} \cup A_{\neq\emptyset}$. Since deciding the existence of such a matching can be achieved in time $\mathcal{O}(\sqrt{|V(B)|}|E(B)|) = \mathcal{O}(\sqrt{|N \cup A|}n|A|)$ (see e.g. [14, Lemma 4]), establishing this claim is the last component required for the proof of the theorem.

Towards showing the forward direction, let π be a stable assignment for I satisfying (P1) and (P2). Then $O = \{\, \{a, \pi^{-1}(a)\} \mid a \in A \,\}$ is a matching in B that saturates $\mathcal{M} \cup A_{\neq\emptyset}$. Note that O saturates $\mathcal{M}$ due to (P2), moreover, O saturates $A_{\neq\emptyset}$ since otherwise there would be an activity $a \in A_{\neq\emptyset}$ with $\pi^{-1}(a) = \emptyset$, which due to the definition of $A_{\neq\emptyset}$ and (P1) implies there is an agent n with $\pi(n) = a_\emptyset$ such that $1 \in P_n(a)$, contradicting our assumption that π is stable.

Towards showing the reverse direction, let O be a matching in B that saturates $\mathcal{M} \cup A_{\neq\emptyset}$. Then the assignment π mapping all agents in M (for every $M \in \mathcal{M}$) to its partner in O and all other agents to $a_\emptyset$ clearly already satisfies (P1) and (P2). It remains to show that it is also stable. Note that π is individually rational because of the construction of B. Moreover, assume for a contradiction that there is an agent $n \in N_\emptyset$ with $\pi(n) = a_\emptyset$ and an activity $a \in A$ such that $(a, |\pi^{-1}(a)| + 1) \in P_n$. If $|\pi^{-1}(a)| = 0$, then $a \in A_{\neq\emptyset}$ and hence $|\pi^{-1}(a)| > 0$ (because O saturates $A_{\neq\emptyset}$), a contradiction. If on the other hand $|\pi^{-1}(a)| \neq 0$, then $\{M, a\} \in O$ (for some $M \in \mathcal{M}$), but $(a, |\pi^{-1}(a)|) \notin P'_n$ and hence $\{M, a\} \notin E(B)$, also a contradiction. □

10 Conclusion

We obtained a comprehensive picture of the parameterized complexity of Group Activity Selection problems parameterized by the number of agent types, both with and without the number of activities as an additional parameter. Our positive results suggest that using the number of agent types is a highly appealing parameter for GASP and its variants; indeed, this parameter will often be much smaller than the number of agents due to the way preference lists are collected or estimated (as also argued in initial work on GASP [7]). For instance, in the large-scale event management setting of GASP (or sGASP), one would expect that preference lists for event participants are collected via simple questionnaires—and so the number of agent types would remain fairly small regardless of the size of the event.

We believe that the techniques used to obtain the presented results, and especially the Subset Sum tools developed to this end, are of broad interest to the algorithms community. For instance, MULTIDIMENSIONAL SUBSET SUM (MSS) has been used as a starting point for W[1]-hardness reductions in at least two different settings over the past year [13, 14], but the simple and partitioned variant of the problem (i.e., SMPSS) is much more restrictive and hence forms a strictly better starting point for any such reductions in the future. This is also reflected in our proof of the W[1]-hardness of SMPSS, which is *significantly* more involved than the analogous result for MSS. Likewise, we expect that the developed algorithms for TREE SUBSET SUM and MULTIDIMENSIONAL PARTITIONED SUBSET SUM may find applications as subroutines for (parameterized and/or classical) algorithms in various settings.

For future work, we believe that it would be interesting to see how the complexity map changes if one were to consider the number of activity types instead of the number of activities in our parameterizations.

Acknowledgements Rahul C. S. acknowledges support by project TOTAL funded by the European Research Council (ERC) under the European Unions Horizon 2020 research and innovation programme (grant agreement No 677651). Robert Ganian acknowledges support from the Austrian Science Fund (FWF, projects Y1329 and P31336). Sebastian Ordyniak acknowledges support from the Engineering and Physical Sciences Research Council (EPSRC, project EP/V00252X/1).

Conflict of interest

The authors declare that they have no conflict of interest.

References

1. Martin Aigner and Günter M. Ziegler. *Proofs from the Book (3. ed.)*. Springer, 2004.
2. Coralio Ballester. Np-completeness in hedonic games. *Games and Economic Behavior*, 49(1):1–30, 2004.
3. Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015.
4. Andreas Darmann. Group activity selection from ordinal preferences. In *Algorithmic Decision Theory - 4th International Conference, ADT 2015*, volume 9346 of *Lecture Notes in Computer Science*, pages 35–51. Springer, 2015.

5. Andreas Darmann, Janosch Döcker, Britta Dorn, Jérôme Lang, and Sebastian Schneckenburger. On simplified group activity selection. In *Algorithmic Decision Theory - 5th International Conference, ADT 2017*, volume 10576 of *Lecture Notes in Computer Science*, pages 255–269. Springer, 2017.
6. Andreas Darmann, Edith Elkind, Sascha Kurz, Jérôme Lang, Joachim Schauer, and Gerhard Woeginger. Group activity selection problem with approval preferences. *International J. of Game Theory*, pages 1–30, 2017.
7. Andreas Darmann, Edith Elkind, Sascha Kurz, Jérôme Lang, Joachim Schauer, and Gerhard J. Woeginger. Group activity selection problem. In *Internet and Network Economics - 8th International Workshop, WINE 2012*, volume 7695 of *Lecture Notes in Computer Science*, pages 156–169. Springer, 2012.
8. Reinhard Diestel. *Graph Theory, 4th Edition*, volume 173 of *Graduate texts in mathematics*. Springer, 2012.
9. Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer Verlag, 2013.
10. Eduard Eiben, Robert Ganian, and Sebastian Ordyniak. A structural approach to activity selection. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018*, pages 203–209. ijcai.org, 2018.
11. Paul Erdős and Paul Turán. On a problem of Sidon in additive number theory, and on some related problems. *Journal of the London Mathematical Society*, 1(4):212–215, 1941.
12. Michael R. Fellows, Daniel Lokshtanov, Neeldhara Misra, Frances A. Rosamond, and Saket Saurabh. Graph layout problems parameterized by vertex cover. In *Algorithms and Computation, 19th International Symposium, ISAAC 2008*, pages 294–305, 2008.
13. Robert Ganian, Fabian Klute, and Sebastian Ordyniak. On structural parameterizations of the bounded-degree vertex deletion problem. *Algorithmica*, 83(1):297–336, 2021.
14. Robert Ganian, Sebastian Ordyniak, and M. S. Ramanujan. On structural parameterizations of the edge disjoint paths problem. *Algorithmica*, 83(6):1605–1637, 2021.
15. Michael R. Garey and David R. Johnson. *Computers and Intractability*. W. H. Freeman and Company, New York, San Francisco, 1979.
16. Sushmita Gupta, Sanjukta Roy, Saket Saurabh, and Meirav Zehavi. Group activity selection on graphs: Parameterized analysis. In *Algorithmic Game Theory - 10th International Symposium, SAGT 2017*, volume 10504 of *Lecture Notes in Computer Science*, pages 106–118. Springer, 2017.
17. Ayumi Igarashi, Robert Bredereck, and Edith Elkind. On parameterized complexity of group activity selection problems on social networks. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, AAMAS 2017, pages 1575–1577. International Foundation for Autonomous Agents and Multiagent Systems, 2017.
18. Ayumi Igarashi, Robert Bredereck, and Edith Elkind. On parameterized complexity of group activity selection problems on social networks. *CoRR*, abs/1703.01121, 2017.
19. Ayumi Igarashi, Robert Bredereck, Dominik Peters, and Edith Elkind. Group activity selection on social networks. *CoRR*, abs/1712.02712, 2017.
20. Ayumi Igarashi, Dominik Peters, and Edith Elkind. Group activity selection on social networks. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pages 565–571. AAAI Press, 2017.
21. Hooyeon Lee and Virginia Vassilevska Williams. Parameterized complexity of group activity selection. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2017*, pages 353–361. ACM, 2017.
22. Rolf Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Oxford Lecture Series in Mathematics and its Applications. Oxford University Press, Oxford, 2006.
23. Krzysztof Pietrzak. On the parameterized complexity of the fixed alphabet shortest common supersequence and longest common subsequence problems. *J. of Computer and System Sciences*, 67(4):757–771, 2003.