Partition Aggregation for Budgeting

Laurent Bulteau, Pallavi Jain, Nimrod Talmon

Partition Aggregation for Budgeting

Laurent Bulteau [1] and **Pallavi Jain**[2] and **Nimrod Talmon** [3]

Abstract. Motivated by certain aggregation tasks related to participatory budgeting, such as clustering projects and modeling project interactions, we study several variants of the following aggregation problem: Given a set P of m projects, and n partitions of P, the task is to aggregate these n partitions into one aggregated partition. We consider several aggregation methods for this setting, including utility-based methods and Condorcet-based methods and evaluate these methods by analyzing their computational complexity and their behavior with respect to certain relevant axiomatic properties.

1 Introduction

In participatory budgeting (PB) [6] the task is to aggregate voter preferences over a set of projects, to decide upon a bundle of those projects to fund. It has received quite extensive recent attention from the research community, resulting in some aggregation methods to be used for such settings [11, 3, 12, 5, 2]. One aspect of PB which is neglected by existing methods is project interactions: E.g., consider a toy PB instance consisting of 3 projects – one school, s, and two parks, p_1 and p_2; in many cases, it is natural to assume that, while certain voters might wish to have one park built in their city, not many voters would feel that funding two parks (especially if these parks are geographically close to each other) is a good use of public funds. (The above example is of *substitutions*, as the two parks are substitutes for each other. Indeed, there could be other types of interactions, most notably complementarities, which correspond to positive interactions between projects; for presentation, we consider here only substitutions).

In this paper we are interested in figuring out the so-called *substitution structure* of the set of projects in a PB instance. E.g., in the toy example described above, it might indeed be the case that the two parks p_1 and p_2 are *substitutes* for most voters. So, the substitution structure of this example would be the partition $\{\{s\}, \{p_1, p_2\}\}$; each part (i.e., $\{s\}$ and $\{p_1, p_2\}$) of this partition is referred to a *substitution class*). In particular, taking a utilitarian approach, assume that each voter has a utility for each bundle of projects; so, the utility of a voter from a set of projects $P' \subseteq \{s, p_1, p_2\}$ is $u(P')$. Then, the substitution structure means that the utility function is submodular wrt. projects of the same part in the partition: That is, $u(\{p_1, p_2\}) \leq u(\{p_1\}) + u(\{p_2\})$.

There are several ways by which an organizer of a PB instance might tackle this aspect of substitutions:

Dictatorial decision: One possibility would be for the organizer to decide herself upon the substitution structure – then, she might, e.g., ask voters to approve projects (i.e., define the election to be an approval one), but do not allow voters to approve more than one project in each substitution class.

Explicit Exponential Elicitation: Another possibility would be for the organizer to allow each voter to explicitly specify her utility from each bundle of projects, however this would mean an exponential explosion and thus is probably not feasible.

Preliminary Election: A yet another possibility would be for the organizer to perform a preliminary election, in which she asks from the voters – perhaps only a subset of the voters – to provide their substitution structures. Then, the organizer can aggregate those partitions provided by the voters and use the aggregated partition as the global substitution structure.

Here we choose the last option, thus concentrate on the subproblem of aggregating partitions. Here, one would be wondering why do we want to conduct election in two phases, that is, first asking for substitution structures from some voters, and then conduct standard election. The one reason is that elicitation cost is high. Since this is the preliminary step of participatory budgeting, it is natural that we only ask some people to give substitution structure, say only secretaries of different societies. Of course, the question of how to use the aggregated partition in the subsequent participatory budgeting election is central; we do not tackle it in the current paper, however, to maintain focus, we leave it as the main future work. We acknowledge one natural criticism of our approach, namely that we assume that the substitution structure is somehow global, in the sense that it can be fixed to be the same for all voters. While, indeed, this might not always be the case, we believe that in most PB instances it is roughly global. Verifying this intuition and identifying cases in which this intuition is true and other cases in which it is violated is, again, an interesting avenue for future research.

Finally, while here we concentrate on identifying aggregated partitions to be used for deciding on substitution structures for PB instances, we briefly mention other motivations for this task:

Visualizing projects: A PB organizer shall distribute a pamphlet explaining the projects to the voters. On each page only a certain number of projects can be explained, thus in fact the pamphlet partitions the projects into pages. Using a partition aggregation method might help here, in particular, as the presentation affects the preferences.

Community structure: Consider the task of identifying communities in a set of agents. One way to achieve this is to request each voter to provide a partition of the set of agents to communities and then using an algorithm for aggregating partitions. However, in some cases, asking people to vote for communities might yield distorted results.

[1] LIGM, CNRS, Univ Gustave Eiffel, Marne-la-Vallée, France , email: laurent.bulteau@upem.fr

[2] Ben-Gurion University of the Negev, Beer-Sheva, Israel, email: pallavi@post.bgu.ac.il

[3] Ben-Gurion University of the Negev, Beer-Sheva, Israel, email: talmonn@bgu.ac.il

1.1 Related Work

A special case of the partition aggregation problem is *cluster ensembles* [15] which is also known as *cluster aggregation* [13] and *consensus clustering* [8]. In cluster aggregation, we are given a set of clusterings, and the goal is to find a clustering which agrees with the input clusterings as much as possible. Cluster aggregation is polynomial-time solvable when the input has two partitions, while it is APX hard when we have three partitions [4]. Strehl and Ghosh [15] proposed some techniques for cluster aggregation. In one of their approaches, given a set of clustering, they construct a hypergraph, and find a hyperedge separator that partitions the hypergraph into k unconnected components of approximately the same size.

Gionis et al. [13] gave some approximation algorithms. They considered cluster aggregation problem and correlation clustering. In cluster aggregation, they measure the dissimilarity between the clusterings. Let V be the given set of objects, and $C_1, \ldots, C_m$ be the set of clusterings. For two objects u and v in V, and two clusterings C_1, C_2, $d_{u,v}(C_1, C_2) = 1$, if u and v are in same part in C_1 and different parts in C_2, or vice-versa, otherwise 0. The dissimilarity between two clusterings C_1 and C_2 is defined as $d(C_1, C_2) = \sum_{u,v \in V} d_{u,v}(C_1, C_2)$. The goal is to find a clustering C such that the total dissimilarity, $\sum_{i=1}^{m} d(C, C_i)$, is minimized. This function is also known as total Mirkin distance. They also studied the maximization version of consensus clustering. Towards this, they defined similarity between two partitions C_1 and C_2, denoted by $s(C_1, C_2)$, as the number of objects which either belong to the same part in both the partitions or in different parts in both partitions. The goal is to find a partition C such that $\sum_{i=1}^{m} s(C_i, C)$ is maximised.

Dornfelder et al. [8] proved that cluster aggregation is NP-hard even when every partition contains at most two clusters. They proposed an FPT algorithm for cluster aggregation with respect to parameter average Mirkin distance. They also studied the local search variant of the problem, and showed that the problem is W[1]-hard when parameterized by radius of the Mirkin-distance neighborhood.

We also mention work on aggregating graphs [9], as aggregating partitions is equivalent to aggregating cluster graphs (graphs that are a collection of disjoint cliques).

2 Formal Model

Formally, we have a set, $P = \{p_1, \ldots, p_m\}$, of projects, (In PB, there is a cost $c(p)$ for each $p \in P$; we do not include these in our formal model as they do not affect the substitution structure) and a set, $V = \{v_1, \ldots, v_n\}$, of voters, where voter v_i corresponds to a partition P_{v_i} of P. A *partition* P_v is a disjoint set of *parts* whose union is P; i.e., $P_v = \{p^1, \ldots, p^z\}$, where for every $i, j \in [z]$, $p^i \cap p^j = \emptyset$ and $\cup_{j \in [z]} p^j = P$. Each p^j is referred to as a *part* of the partition P_v. A *partition aggregation method* is a function taking n partitions of P and returning a partition S of P, referred to as the *aggregated partition*. (We ignore issues of tie-breaking as they clutter the presentation without adding significant insights.) We denote an instance of partition aggregation as $(P, \mathcal{C})$, where P is the set of projects and $\mathcal{C}$ is the collection of n partitions of P.

3 A Utilitiarian Approach

The main question we are studying is how to define a "good" partition. Here we take a utilitiarian approach: We assume that each voter v, based on her partition P_v, would derive a certain utility from each possible aggregated partition S. While these utilities are unknown, they might be estimated (similarly to set extensions, which are used to estimate utilities over committees based on utilities over single candidates in multiwinner elections). Given a specific way of estimating such utilities over the set of possible partitions, a natural partition aggregation method would return, as the aggregated partition, a partition which maximizes the sum – over the voters – of these utilities. Such a utilitarian approach has been applied successfully for many social choice settings, including multiwinner elections [10] and participatory budgeting [11].

Example 1 *Consider the set of projects $P = \{p_1, p_2, p_3, p_4\}$ and a voter v with the partition $P_v = \{\{p_1, p_2\}, \{p_3, p_4\}\}$. It is natural to assume that the utility of voter v from the partition $S_1 = \{\{p_1, p_2\}, \{p_3\}, \{p_4\}\}$ would be fairly high, as P_v and S_1 are quite similar. Furthermore, the utility of v from $S_2 = \{\{p_1\}, \{p_2\}, \{p_3\}, \{p_4\}\}$ might be less than her utility from S_1, as S_2 seems to be less similar to P_v than S_1 is.*

Our utilitarian approach is formally defined below.

Definition 1 (Utility function) *Let P be a set of m projects and let $\mathcal{P}$ be the set of all partitions of P. A* utility function *is a function $f : \mathcal{P} \times \mathcal{P} \to \mathbb{N}$. For a voter v and a possible aggregated partition p, $f(v, p)$ is understood as the utility that voter v gets from p; the higher the better.*

3.1 Two Utility Functions

We consider two utility functions. The PA utility function is perhaps the first utility function one might think of: The utility of a voter equals the number of pairs for which her vote agrees with the aggregated vote. The PAM utility function is slightly less natural as the utility is the number of pairs for which the voter agrees with the aggregated vote, minus the number of pairs not in the same partition in the voter's partition but in the same partition in the aggregated vote. Thus, in essence, we define a larger fine for merging a pair of projects unnecessarily (a fine of -1), than for splitting a pair of projects unnecessarily (a fine of 0). The reason for this asymmetry is our motivation from substitution structures: We believe that an aggregated partition that defines a set of projects as substitutions to each other, while most voters prefer not to have them defined as such is more harmful than an aggregated partition that simply fails to define a substitution in cases in which it should.

Number of pairwise agreements (PA): Let p_1, p_2 be two projects in P. For a voter v, and an aggregated partition S, let

$$\delta^{\mathtt{PA}}_{v,S}(p_1, p_2) = \begin{cases} 1 & \text{if } p_1, p_2 \text{ are in same (different) part(s)} \\ & \text{in both } P_v \text{ and } S\ ; \\ 0 & \text{otherwise}\ . \end{cases}$$

We define the *Number of pairwise agreements (PA)* utility function for voter v from S to be as follows:

$$f^{\mathtt{PA}}(v, S) = \sum_{p_1, p_2 \in P} \delta^{\mathtt{PA}}_{v,S}(p_1, p_2)\ .$$

Number of pairwise agreements minus number of mergings (PAM): Let p_1, p_2 be two projects in P. For a voter v, and aggre-

gated partition S, let

$$\delta^{\texttt{PAM}}_{v,S}(p_1,p_2) = \begin{cases} 1 & \text{if } p_1,p_2 \text{ are in same(different) part(s)} \\ & \text{in both } P_v \text{ and } S\ ; \\ 0 & \text{if } p_1 \text{ and } p_2 \text{ are in same part in } P_v \\ & \text{but in different parts in } S\ ; \\ -1 & \text{if } p_1 \text{ and } p_2 \text{ are in different parts in } P_v \\ & \text{but in same part in } S\ . \end{cases}$$

We define the *Number of pairwise agreements minus number of mergings (PAM)* utility function for voter v from S to be:

$$f^{\texttt{PAM}}(v,S) = \sum_{p_1,p_2 \in P} \delta^{\texttt{PAM}}_{v,S}(p_1,p_2)\ .$$

3.2 Aggregation Goals

Given a utility function f, we consider two partition aggregation goals: Maximizing the sum of utilities (Total) and maximizing the minimum utility (Egal).

Definition 2 (Total Utility (Total)) *Given a set of m projects, a collection of n partitions, $P_1,\ldots,P_n$, and a utility function f, the goal in* Total Utility *aggregation is to find a partition S such that $\sum_{i\in[n]} f(P_i,S)$ is maximized.*

That is, in Total Utility aggregation we look for a partition that maximizes the total utility (i.e., $\sum_{i\in[n]} f(P_i,S)$). In the decision version of the problem, given an integer k additionally, we look for a partition that has total utility at least k. We use the same problem name for optimisation as well as decision version of the problem as it will be clear from the context.

We also consider egalitarian aggregation methods, in which we care for the least satisfied voter (similarly in spirit to egalitarian committee scoring rules [1]).

Definition 3 (Egalitarian Utility (Egal)) *Given a set of m projects, a collection of n partitions, $P_1,\ldots,P_n$, and a utility function f, in* Egalitarian Utility *aggregation the goal is to find a partition S such that $\min_{i\in[n]} f(P_i,S)$ is maximized.*

In the decision version of the problem, given an integer k additionally, we look for a partition that has Egalitarian Utility at least k.

3.3 Illustrating Example

Our two utility functions and two aggregation types define four aggregation methods: Total-PA, Egal-PA, Total-PAM, and Egal-PAM. Consider the following set of partitions to illustrate the difference between these methods. Let $P_{v_1} = \{\{p_1,p_2,p_3\}\}$, $P_{v_2} = \{\{p_1\},\{p_2,p_3\}\}$, and $P_{v_3} = \{\{p_1\},\{p_2\},\{p_3\}\}$. Let $S = \{\{p_1,p_2\},\{p_3\}\}$ be an aggregated partition. We first describe Total-PA. For the pair of projects p_1,p_2, $\delta^{\texttt{PA}}_{v_1,S}(p_1,p_2) = 1$ as p_1 and p_2 are in same part in both the partitions; however $\delta^{\texttt{PA}}_{v_1,S}(p_1,p_3) = 0$ as p_1 and p_3 are in same part for v_1 and in different parts in S. Similarly, $\delta^{\texttt{PA}}_{v_1,S}(p_2,p_3) = 0$. Hence, the PA utility for voter v_1 from S, $f^{\texttt{PA}}(v_1,S)$, is 1. Similarly, the PA utility for voter v_2 and v_3 from S are 1 and 2, respectively. The Total-PA utility from S is 4. The Egal-PA utility is 1. Next, we describe Total-PAM utility. For the pair of projects a,b, $\delta^{\texttt{PAM}}_{S,v_1}(p_1,p_2) = 1$; however $\delta^{\texttt{PAM}}_{S,v_1}(p_1,p_3) = 0$ and $\delta^{\texttt{PAM}}_{S,v_1}(p_2,p_3) = 0$. Hence, the PAM utility for v_1 from S, $f^{\texttt{PAM}}(v_1,S)$, is 1. Furthermore, $\delta^{\texttt{PAM}}_{S,v_2}(p_1,p_2) = -1$ as p_1 and p_2 are in different parts for v_2, while they are in the same part in S; $\delta^{\texttt{PAM}}_{S,v_2}(p_1,p_3) = 1$ and $\delta^{\texttt{PAM}}_{S,v_2}(p_2,p_3) = 0$. Hence, the PAM utility for v_2 from S is 0. Similarly, the PAM utility for v_3 from S is 1. Therefore, the Total-PAM utility from S is 2, and the Egal-PAM utility is 0.

4 Computational Complexity

First, we study the computational complexity, and show intractability, of our partition aggregation methods.

4.1 Total-PA and Total-PAM

Recall that the problem of finding an aggregated partition with Maximum Total-PA is equivalent to the Consensus Clustering problem. Since Consensus Clustering is known to be NP-hard even for three partitions [8], we have the following result.

Proposition 1 *Total-PA is NP-hard even for three voters.*

Consensus Clustering is also known to be NP-hard when every input partition has at most two parts [8]. Hence, we have following result.

Proposition 2 *Total-PA is NP-hard even when every voter has at most two parts.*

We next present our intractability result for Total-PAM.

Theorem 1 *Total-PAM is NP-hard even when each partition contains at most two parts.*

Proof. We give a polynomial time reduction from the known NP-hard problem Cluster Deletion (Given a graph G, and an integer k; we shall decide the existence of at most k-sized set of edges whose deletion from G results into a cluster graph(i.e., a disjoint union of cliques)) [14]. Let (G,k) be an instance of the Cluster Deletion problem. Let $|V(G)| = n, |E(G)| = m$. Without loss of generality, assume that $n-2 = 2^\ell$, for some positive integer ℓ. We first construct the set of projects P. For each vertex $u \in V(G)$, we add a project u in the set P. Now, we construct a collection of partitions, $\mathcal{C}$, of projects. For every pair of vertices $u,v \in V(G)$, we create a collection of partitions $\mathcal{C}_{uv}$ as follows. If $uv \in E(G)$, then $|\mathcal{C}_{uv}| = 3\cdot 2^{2\ell-1}$, otherwise $|\mathcal{C}_{uv}| = 3\cdot 2^{4\ell-1}$. If $uv \in E(G)$, then in every partition in $\mathcal{C}_{uv}$, u and v are in same part; otherwise u and v are in different parts in every partition in $\mathcal{C}_{uv}$. For every pair of vertices $x,y \in V(G)\setminus\{u,v\}$, there are $|\mathcal{C}_{uv}|/3$ partitions in $\mathcal{C}_{uv}$ in which x and y are in different parts, and $2\cdot|\mathcal{C}_{uv}|/3$ partitions in which x and y are in same part. This collection of partitions $\mathcal{C}$ can be construed in polynomial time, however we skip the justification due to space constraint. The intuitive idea for such a collection of partitions is that for a pair of project x,y, the total PAM utility due to partitions in $\mathcal{C}_{uv}$, where u,v are distinct from x,y, is $2^{2\ell-1}$ for any aggregated partition, as if x,y are in different parts in the aggregated partition, then the total PAM utility due to these partitions is $2\cdot|\mathcal{C}_{uv}|/3 - |\mathcal{C}_{uv}|/3 = |\mathcal{C}_{uv}|/3$, otherwise $|\mathcal{C}_{uv}|/3$. In essence, the utility of pair of project for $\mathcal{C}_{u,v}$ does not depend on the parts to which x,y belongs in the aggregated partition. However, for project u,v, total PAM utility for $\mathcal{C}_{u,v}$ depends on their parts in the aggregated partition. The set of partitions in our instance is $\mathcal{C} = \cup_{u,v\in V(G)}\mathcal{C}_{uv}$. We set total PAM utility as

$$\begin{aligned} k' =& 2^{4\ell-1}\Big(\frac{n(n-1)}{2} - m\Big)\Big(\frac{n(n-1)}{2} + 2\Big) \\ &+ 2^{2\ell-1}\Big(m\Big(\frac{n(n-1)}{2} - m + n + 2\Big) - 3k\Big) \end{aligned}$$

Method	Complexity	Unanimity	Majority-based	IIP
Total-PA	NP-h even for $n = 3$ (Prop. 1) or $\ell \leq 2$ (Prop. 2)	Yes	Yes	No
Egal-PA	NP-h	No	No	No
Total-PAM	NP-h even for $\ell \leq 2$ (Theorem 1)	Yes	No	**Open**
Egal-PAM	NP-h	**Open**	No	**Open**

Table 1. Summary of Our Results. We denote the maximum number of parts in any partition by ℓ.

Next, we show the equivalence between the instance (G, k) of cluster deletion and the instance $(P, \mathcal{C})$ of total PAM.

In the forward direction, let X be a subset of edges of size k such that $G' = G - X$ is a cluster graph. We construct an aggregated partition Y for the instance $(P, \mathcal{C})$ as follows. Two projects $u, v \in P$ are in same part in Y if and only if their corresponding vertices in G' are in same clique. Note that the number of parts in Y is same as the number of cliques in G'. Now, we show that the total PAM utility for Y is at least k'. Recall that for any pair of projects $u, v \in P$, for all collections $\mathcal{C}_{xy}$, where $x, y \in V(G)\backslash\{u, v\}$, $\sum_{C\in\mathcal{C}_{xy}} \delta_{C,Y}(u, v) = |\mathcal{C}_{xy}|/3$. Note that if $uv \notin E(G)$, then $uv \notin E(G')$, as we only delete edges from G. Therefore, for $u, v \in P$, if $(u, v) \notin E(G)$, then $\sum_{C\in\mathcal{C}\setminus\mathcal{C}_{uv}} \delta_{C,Y}(u, v) = 2^{4\ell-1}\binom{n(n-1)}{2} - m - 1) + 2^{2\ell-1}m$. Furthermore, $\sum_{C\in\mathcal{C}_{uv}} \delta_{C,Y}(u, v) = 3 \cdot 2^{4\ell-1}$. Therefore, if $uv \notin E(G)$, then $\sum_{C\in\mathcal{C}} \delta_{C,Y}(u, v) = 2^{4\ell-1}\binom{n(n-1)}{2} - m + 2) + 2^{2\ell-1}m$. We further note that if $uv \in E(G')$, then $uv \in E(G)$, as we do not add any edge to G. Therefore, for $u, v \in P$, if $(u, v) \in E(G)$, then $\sum_{C\in\mathcal{C}\setminus\mathcal{C}_{uv}} \delta_{C,Y}(u, v) = 2^{4\ell-1}\binom{n(n-1)}{2} - m) + 2^{2\ell-1}(m - 1)$. Furthermore, if $uv \in E(G')$, then $\sum_{C\in\mathcal{C}_{uv}} \delta_{C,Y}(u, v) = 3 \cdot 2^{2\ell-1}$. Therefore, if $uv \in E(G')$, then $\sum_{C\in\mathcal{C}} \delta_{C,Y}(u, v) = 2^{4\ell-1}\binom{n(n-1)}{2} - m) + 2^{2\ell-1}(m + 2)$; and if $uv \in E(G)$ but $uv \notin E(G')$, then $\sum_{C\in\mathcal{C}} \delta_{C,Y}(u, v) = 2^{4\ell-1}\binom{n(n-1)}{2} - m) + 2^{2\ell-1}(m - 1)$. Since there can be at most k edges in G which are not in G', total PAM utility for P is

$$\Big(2^{4\ell-1}\Big(\frac{n(n-1)}{2} - m + 2\Big) + 2^{2\ell-1}m\Big)\Big(\frac{n(n-1)}{2} - m\Big)$$
$$+\Big(2^{4\ell-1}\Big(\frac{n(n-1)}{2} - m\Big) + 2^{2\ell-1}(m-1)\Big)m$$
$$+3 \cdot 2^{2\ell-1}(m - k) = k'$$

This completes the proof in the forward direction.

In the reverse direction, let Y be an aggregated partition to $(P, \mathcal{C})$ such that total PAM utility for Y is at least k'. Let $X = \{uv \in E(G) \mid u \text{ and } v \text{ are in different parts in } Y\}$.

Claim 1 *If u and v belong to same part in Y, then $uv \in E(G)$.*

Proof. Towards the contradiction, suppose that there exists projects u and v in a part in Y such that $uv \notin E(G)$. Then, $\sum_{C\in\mathcal{C}_{uv}} \delta_{C,Y}(u, v) = -3 \cdot 2^{4\ell-1}$ (as u and v are in different parts in each partition in $\mathcal{C}_{uv}$). Therefore,

$$\sum_{C\in\mathcal{C}} \delta_{C,Y}(u, v) = 2^{4\ell-1}\Big(\frac{n(n-1)}{2} - m - 1\Big) + 2^{2\ell-1}m - 3 \cdot 2^{4\ell-1}$$

Hence, total PAM utility for Y is

$$\Big(2^{4\ell-1}\Big(\frac{n(n-1)}{2} - m + 2\Big) + 2^{2\ell-1}m\Big)\Big(\frac{n(n-1)}{2} - m - 1\Big)$$
$$+ 2^{4\ell-1}\Big(\frac{n(n-1)}{2} - m - 1\Big) + 2^{2\ell-1}m - 3 \cdot 2^{4\ell-1}$$
$$+ \Big(2^{4\ell-1}\Big(\frac{n(n-1)}{2} - m\Big) + 2^{2\ell-1}(m-1)\Big)m$$
$$+ 3 \cdot 2^{2\ell-1}(m - k_1)$$
$$= 2^{4\ell-1}\Big(\frac{n(n-1)}{2} - m\Big)\Big(\frac{n(n-1)}{2} + 2\Big) - 6 \cdot 2^{4\ell-1}$$
$$+ 2^{2\ell-1}\Big(m\Big(\frac{n(n-1)}{2} - m + n + 2\Big) - 3k_1\Big) < k'$$

as $k < m \leq 2^{2\ell+1}$, a contradiction to the assumption that Y is a solution to $(P, \mathcal{C})$. □

Hence, $G - X$ is a cluster graph. Next, we show the size bound on X.

Claim 2 $|X| \leq k$.

Proof. Towards the contradiction, suppose that $|X| > k$. Since $G - X$ is a cluster graph, if $uv \notin E(G)$, then $uv \notin E(G - X)$. Since $|X| > k$, there are at least $k + 1$ pairs $u, v \in V(G)$ such that u and v are in same part in each partition in $\mathcal{C}_{uv}$ but in different parts in Y. We can show that in this case the total PAM utility for Y is less than k', a contradiction to that Y is a solution to $(P, \mathcal{C})$. □

The reverse direction follows from Claim 1 and 2.

□

4.2 Egalitarian-PA and Egalitarian-PAM

Proposition 3 *Egalitarian-PA and Egalitarian-PAM are NP-hard.*

Proof. Due to lack of space, we only give a sketch of the reduction. We reduce from Unary Bin Packing, seen as a partition problem (one seeks a partition of $[kB]$ into k parts of size B where items, represented as disjoint subsets, must be included in single parts). We use $[kB]$ as the set of projects, and first build two voters: $R := \{[kB]\}$ and $S := \{\{1\}, \ldots, \{kB\}\}$. We can enforce that the solution must have utility at least $t_R := k\binom{B}{2}$ with R and $t_S := \binom{kB}{2} - t_R$ with S ($t_S := \binom{kB}{2} - 2t_R$ in the PAM model). These two thresholds yield a partition with k blocks of size B. Then, for each pair of elements $e = (x, y)$ from the same item, we enforce that both elements are in the same part using a partition $Q_e = \{\{x\}, \{y\}, [kB] \setminus \{x, y\}\}$ with utility threshold $t_Q := t_R + 2(m - 2B) + 3$ (or $t_Q := t_R + 4(m - 2B) + 6$ in the PAM model). Together, these constraints ensure that the target solution is a valid bin packing of the original instance (we achieve distinct utility thresholds with additional gadgets appended to each partition). □

5 Axiomatic Properties

Here we consider various axiomatic properties that are relevant for partition aggregation methods, and test our partition aggregation methods against them. While it is possible to define many axiomatic properties, we chose axioms that seem especially relevant, with the application of aggregating substitution structures in mind. Our results, regarding both the computational complexity and the axiomatic properties of our four partition aggregation methods – Total-PA, Egal-PA, Total-PAM, and Egal-PAM – are summarized in Table 1.

The axiom of *Unanimity*, defined next, says that if all voters agree on whether some two projects shall be in the same part or in different parts (i.e., all voters are unanimous wrt. to this pair of projects), then the aggregated partition shall also agree with the voters regarding these two projects.

Definition 4 (Unanimity) *A partition aggregation method $\mathcal{R}$ satisfies* Unanimity *if the following hold: Let $P_1, \ldots, P_n$ be the set of partitions of projects $P = \{p_1, \ldots, p_m\}$. If two projects p_i and p_j, $i, j \in [m]$, belong to the same part in P_i, for all $i \in [n]$, then p_i, p_j belong to the same part in the aggregated partition. Similarly, if p_i, p_j belong to different parts in P_i, for all $i \in [n]$, then p_i, p_j belong to different parts in the aggregated partition.*

While Unanimity requires that the aggregated partition agrees with the voters on those pairs of projects for which all voters are in complete agreement, majority-based aggregation considers pairs of projects with majority agreement among the voters. As we show next, such aggregated partitions need not exist; thus, we require an aggregation method to output such partitions only when they exist.

Definition 5 (Majority based aggregation) *An aggregated partition $\tilde{P}$ is majority-based if any two projects p_1, p_2 are in the same part in $\tilde{P}$ if and only if p_1 and p_2 are placed in the same part for more than half of the voters. A partition aggregation method $\mathcal{R}$ satisfies* Majority-based aggregation *if it always outputs majority-based aggregated partitions, whenever such an aggregated partition exists.*

Remark 1 *One might consider a continuum of axioms between Majority-based aggregation and Unanimity, by employing supermajorities: An aggregated partition p would place a pair of projects in the same partition iff at least a δ-Supermajority among the voters does so.*

We also consider an adaptation of the fundamental axiom of Independent of Irrelevant Alternatives to our setting of aggregating partitions: We refer to our adaptation as *Independent of Irrelevant Projects*. In essence, it means that if the restriction of some two profiles to a pair of projects is the same, then the restriction to this pair of projects of both aggregated partition shall be the same.

Definition 6 (Independent of Irrelevant Projects) *A partition aggregation method $\mathcal{R}$ satisfies* Independent of Irrelevant Projects *if the following holds: Let $\mathtt{P}$ and $\mathtt{P}'$ be two partition profiles and let s and s' be their aggregated partitions according to $\mathcal{R}$. Let p_1 and p_2 be two projects such that the number of voters that places them in the same part in $\mathtt{P}$ and in $\mathtt{P}'$ are the same. Then, s and s' shall either both place p_1 and p_2 in the same part or in different parts.*

5.1 Unanimity

Here we compare our four aggregation methods wrt. to the axiom of Unanimity: We wish to identify which of our methods are Unanimous and which are not. First, we note that a unanimous partition always exists, and can be found in polynomial time:

Observation 1 *A unanimous partition always exists, and can be found in polynomial time.*

The proof follows from the fact that we can output any partition that is provided by some voter. Note that it is unanimous.

Next we show that Total-PA is also Unanimous. This is intuitively appealing, as, to maximize the total utility, it seems natural for the aggregated partition to agree with the voters at least on those pairs of projects for which the voters are in total agreement among themselves.

Theorem 2 *Total-PA is unanimous.*

Proof. Let S be an optimal aggregated partition for a set of voters, $\{v_1, \ldots, v_n\}$, that maximizes total PA. Let p_1 and p_2 be two projects that are in same part in all the partitions while in different parts in S. We claim that there exists an aggregated partition S' in which p_1 and p_2 are in same part and Total-PA utility for S' is same as that of S.

We first create S' as follows. Initially, S' is same as S. Let S_1 and S_2 be two parts containing p_1 and p_2, respectively. We delete p_2 from S_2, and add it to S_1. Next, we prove that Total-PA utility from S' is larger than that from S. We denote new S_1 and S_2 as S_1' and S_2', respectively. Consider parts S_1 and S_2 in the partition S. Let $\sum_{i=1}^{n} \sum_{p \in S_1 \cup S_2} \delta_{S,v_i}(p, p_1) = n_1$ (number of pairwise agreements between project p_1 and the projects in $S_1 \cup S_2$), $\sum_{i=1}^{n} \sum_{p \in S_1 \cup S_2} \delta_{S,v_i}(p, p_2) = n_2$ (number of pairwise agreements between project p_2 and the projects in $S_1 \cup S_2$). Now, we first claim that $n_1 = n_2$. Towards the contradiction, suppose that $n_1 > n_2$. Then, when we move p_2 from S_2 to S_1, the utility for p_2 and the vertices in $S_1 \cup S_2$ is n_1 as p_1 and p_2 are in same part in all the partitions. Note that the utility for other pairs of projects do not change. Therefore, the Total PA for this partition is greater than for S, a contradiction. Similarly, if $n_2 > n_1$, then by moving p_1 from S_1 to S_2, we obtain a partition whose Total-PA is more than that from S, a contradiction. Therefore, $n_1 = n_2$, and hence, total PA utility for S' is same as that of S.

Next, we consider the case when p_1 and p_2 are in different parts in all the partitions while in the same part, say S_1, in S. Let $\sum_{i=1}^{n} \sum_{p \in S_1} \delta_{S,v_i}(p, p_1) = n_1$ (number of pairwise agreements between project p_1 and the projects in S_1), and $\sum_{i=1}^{n} \sum_{p \in S_1} \delta_{S,v_i}(p, p_2) = n_2$ (number of pairwise agreements between project p_2 and the projects in S_1). Since p_1 and p_2 are in different parts for all the voters, if $p_2 \notin S_1$, $\sum_{i=1}^{n} \sum_{p \in S_1} \delta_{S,v_i}(p, p_2) = n_1$. Similarly, if $p_1 \notin S_1$, $\sum_{i=1}^{n} \sum_{p \in S_1} \delta_{S,v_i}(p, p_1) = n_2$. Therefore, if $n_1 > n_2$, then deleting p_2 from S_1, and adding a part $\{p_2\}$ to S increase the total PA utility, a contradiction to that S maximises total utility. Similarly, if $n_2 > n_1$, then deleting p_1 from S_1, and adding a part $\{p_1\}$ to S increase the total PA utility, a contradiction. Hence, $n_1 = n_2$. Thus, if we create a partition by deleting p_2 from S_1, and adding a part $\{p_2\}$ to S, total PA utility remains same. □

In contrast, Egal-PA is not Unanimous. This is also intuitively appealing, as egalitarian methods care for the least satisfied voter.

Proposition 4 *Egalitarian-PA is not unanimous.*

Proof. We show this by an example. Suppose that we have 2 voters and 5 projects, p_1, p_2, p_3, p_4, p_5. Let the partition for first voter be $\{\{p_1, p_2, p_3, p_4, p_5\}\}$, and for second voter it is $\{\{p_1, p_2\}, \{p_3, p_4, p_5\}\}$. Let us consider some possible aggregated partition. In particular, we consider all partitions in which p_1 and

p_2 are in the same part, and one partition in which p_1 and p_2 are in different parts to show that any partition containing p_1 and p_2 in same part can not be optimal solution of Egal-PA. Note that partitions $\{\{p_1, p_2, p_3, p_4\}, \{p_5\}\}$, $\{\{p_1, p_2, p_3, p_4\}, \{p_4\}\}$, and $\{\{p_1, p_2, p_4, p_5\}, \{p_3\}\}$ have same PA utility for both the voters; therefore in Table 2, we only mention one of these. Similarly, we omit some partitions in the table that contains p_1, p_2 in same part but have same utilities as some partition in the table. □

Aggregated Partition	PA utility	
	1st voter	2nd voter
$S = \{\{p_1, p_2, p_3, p_4, p_5\}\}$	10	4
$S = \{\{p_1, p_2, p_3, p_4\}, \{p_5\}\}$	6	4
$S = \{\{p_1, p_2, p_3\}, \{p_4\}, \{p_5\}\}$	3	5
$S = \{\{p_1, p_2, p_3\}, \{p_4, p_5\}\}$	4	6
$S = \{\{p_1, p_2\}, \{p_3\}, \{p_4, p_5\}\}$	2	7
$S = \{\{p_1, p_2\}, \{p_3\}, \{p_4\}, \{p_5\}\}$	1	7
$S = \{\{p_1, p_2\}, \{p_3, p_4, p_5\}\}$	4	10
$S = \{\{p_1, p_3, p_4, p_5\}, \{p_2\}\}$	6	6

Table 2. Example for non-unanimity of Egalitarian PA (Lemma 4)

Next we consider Total-PAM. While we show that, similarly to Total-PA, Total-PAM also satisfies Unanimity, we conjecture that Egal-PAM, similarly to Egal-PA, does not satisfy Unanimity.

Theorem 3 *Total-PAM is unanimous.*

Proof. Let S be an optimal aggregated partition for a set of voters, $\{v_1, \ldots, v_n\}$. Let p_1 and p_2 be two projects which are in same part for all the voters, while in different parts, say S_1, S_2, respectively, in S. Let $\sum_{i=1}^{n} \sum_{p \in S_1 \cup S_2} \delta_{S,v_i}(p, p_1) = n_1 - n_2$ (number of pairwise agreements between project p_1 and the projects in $S_1 \cup S_2$ minus number of mergings), and $\sum_{i=1}^{n} \sum_{p \in S_1 \cup S_2} \delta_{S,v_i}(p, p_2) = n_3 - n_4$. Note that if $n_1 - n_2 > n_3 - n_4$, then moving p_2 from S_2 to S_1 increases the total PAM utility, a contradiction to optimality of S. Similarly, if $n_3 - n_4 > n_1 - n_2$, then moving p_1 from S_1 to S_2 increases the total PAM utility, a contradiction. Therefore, $n_1 - n_2 = n_3 - n_4$. Thus, moving p_1 from S_1 to S_2 does not change the total PAM utility. Hence, there exists an optimal aggregated partition in which p_1 and p_2 are in same part.

Next, we show that if p_1 and p_2 are in different parts for all the voters, then p_1 and p_2 are not in the same part in any optimal aggregated partition. Towards the contradiction, suppose that there exists an optimal aggregated partition, S, in which p_1 and p_2 are in same part, say S_1. Note that for a project p, if $\sum_{i=1}^{n} \delta_{S,v_i}(p, p_1) = n_1$, then $\sum_{i=1}^{n} \delta_{S,v_i}(p, p_2) = -n_1$, and vice-versa. Let $\sum_{i=1}^{n} \sum_{p \in S_1} \delta_{S,v_i}(p, p_1) = n_1 - n_2$, and $\sum_{i=1}^{n} \sum_{p \in S_1} \delta_{S,v_i}(p, p_2) = n_3 - n_4$. Using above observation, $n_4 = n_4' + n_1$ and $n_2 = n_2' + n_3$. Note that $n_2' = n_4'$. Therefore, $n_1 - n_2 + n_3 - n_4 < 0$. Let us consider two cases: either $n_1 - n_2 < 0$ or $n_1 - n_2 \geq 0$. If $n_1 - n_2 < 0$, then we delete p_1 from S_1 and add it as a singleton in S. Note that now $\sum_{i=1}^{n} \delta_{S,v_i}(p, p_1) \geq n$ (as only merging two projects gives negative terms, not splitting in two parts). Hence, $\sum_{i=1}^{n} \delta_{S,v_i}(p, p_1) + \sum_{i=1}^{n} \delta_{S,v_i}(p, p_2) \geq n + n_3 - n_4$. Since $n_1 - n_2 < 0$, the total PAM utility from new partition is larger than from the former one. Consider the other case. Let $n_1 - n_2 \geq 0$. In this case, we delete p_2 from S_1 and add it as a singleton in S. As argued above, here also total utility increases. □

5.2 Majority-Based Aggregation

Next we consider Majority-based partitions. First, we show that, contrary to Unanimous partitions, Majority-based partitions do not always exist.

Proposition 5 *A majority based aggregated partition need not exist.*

Proof. Consider three voters, v_1, v_2, v_3, and three projects p_1, p_2, p_3. Let the partitions corresponding to v_1, v_2, and v_3 be $P_{v_1} = \{\{p_1, p_2\}, \{p_3\}$, $P_{v_2} = \{\{p_1, p_2, p_3\}\}$, and $P_{v_3} = \{\{p_1, p_3\}, \{p_2\}\}$. According to the property of Majority-based aggregation, we shall have the following: (1) p_1, p_3 shall be in the same part; (2) p_1, p_2 shall be in the same part; and (3) p_2, p_3 shall be in different parts. As the three constraints above cannot be simultaneously satisfied, we conclude that there is no Majority-based partition for this profile. □

In a sense, the counterexample in the above proof is similar to the canonical counterexample showing the existence of Condorcet-cycles in single-winner elections (i.e., voters $a > b > c$, $b > c > a$, and $c > a > b$, requiring the contradicting requirements of $a > b$, $b > c$, and $c > a$ as the canonical example of Condorcet paradox.

Nevertheless, Majority-based aggregation exists for some profiles; for these profiles, it can be found in polynomial time.

Theorem 4 *A majority based aggregated partition can be found in polynomial time, if it exists.*

Proof. Let P be the set of projects. Let V be the set of voters. We construct an aggregated partition S iteratively as follows. Initially, let $S = \emptyset$. In each iteration i, we add a part S_i in S as follows: consider a project $x \in P \setminus (\cup_{j=1}^{i-1} S_j)$ (that is a project that has not been added in any part in previous iterations); add it in the part S_i, and add all the projects $x' \in P \setminus (\cup_{j=1}^{i-1} S_j \cup \{x\})$, such that x and x' are in same part for more than half of the voters, in S_i. Now, we check whether every two projects in S_i are in same part for more than half of the voters. If not, then the algorithm returns No. We also check whether there exists projects $y \in S_i$ and $y' \in S_j$, where $i \neq j$, such that y and y' are in different parts for at least half of the voters. If not, then the algorithm returns No. We repeat until there is a project which is not placed in any part in S; and return set S. Clearly, the algorithm runs in polynomial time.

Next we prove the correctness of the algorithm. Indeed, if the algorithm returns a partition S, then it is a majority based aggregated partition, as at every step the algorithm checks this property. Now, we show that if the algorithm returns No, then there is no majority based aggregated partition for V. Towards the contradiction, let S' be a majority based aggregation for V. Note that the algorithm returns No in two cases: (1) there are two projects, say y, z, in a part, say S_i, in S that are not in same part for more than half of the voters, or (2) there are two projects, say y, z, in different parts, say S_i, S_j, respectively, of S that are in same part for more than half of the voters. Consider the case (1). Clearly, y and z are in different parts in S'. Let the algorithm first added project x to S_i. Thus, x, y and x, z are in same part for more than half of the voters, as the algorithm added y, z in S_i, a contradiction that S' is majority based aggregation. Next, consider case (2). Clearly, y, z are in same part in S'. Let the algorithm first added project x to S_i. Then, algorithm added y to S_i but not z. This implies that x, y are in same part for more thant half of the voters but not x, z. Since the part containing y, z in S' does not contain x as x, z are in different parts for at

least half of the voters, a contradiction that S' is a majority based aggregation, as x, y are in different parts in S'. □

We have following corollary from the algorithm described above.

Corollary 1 *The existence of a Majority-based partition for a given profile can be decided in polynomial time.*

Moreover, note that, if it exists, then a Majority-based aggregated partition is unique. Next, we check whether our partition aggregation methods output the Majority-based partition whenever it exists.

Theorem 5 *If a majority based aggregated partition exists, then it maximizes Total-PA.*

Proof. Let V be a set of partitions corresponding to all the voters. Let S be the majority based aggregated partition for V. Let $S' \neq S$ be an optimal solution of Total-PA for V. Since majority based aggregated partition is unique, S' is not a majority based aggregated partition. Thus, either (1) there exists a part S_i in S' containing two projects that are not in same part for more than half of the voters, or (2) there exists two parts S_i, S_j in S' such that there is a project in S_i and a project in S_j that are in same part for more than half of the voters.

Consider case (1). Let x and y be two projects in S_i that are in different parts for at least half of the voters. Let M_x be the set of projects that are in same part with x for more than half of the voters, and N_x be the set of projects that are not in same part with x for more than half of the voters. Note that a pair of projects a, b, where $a \in M_x$ and $b \in N_x$ are not in same part for more than half of the voters, otherwise majority based aggregated partition does not exist. Also, note that every two projects in M_x are in same part for more than half of the voters, otherwise majority based aggregated partition does not exist. Now, create a new partition by deleting N_x from S_i, and add a new set of projects N_x to S. Since every pair of project a, b, where $a \in \{x\} \cup M_x$ and $b \in N_x$, are in different parts for at least half of the voter; either the total PA utility of new partition is same as that of S' or larger than that of S. It can not be larger as S' is an optimal solution of Total-PA. We apply this step for all the parts in S_i that contains at least two projects that are not in same part for more than half of the voters. Note that the total PA utility for new partition is same as that of S'. Let us denote this new partition by S''.

Next, we consider case (2). Note that if there exists two parts S_i, S_j in S' such that there is a project in S_i and a project in S_j that are in same part for more than half of the voters, then such parts also exist in S'' (because we have not merged any two parts). Let S_p and S_q be two parts in S'' that contains x and y, respectively, such that x and y are in same part for more than half of the voters. We merge parts S_p and S_q in the partition S''. Since S'' does not contain any part that violates majority based aggregation property due to case (1), all the pair of projects in S_p(or S_q) are in same part for more than half of the voters. Note that every pair of projects a, b such that $a \in S_p$ and $b \in S_q$ are in same part for more than half of the voters, otherwise majority based aggregated partition does not exist. Therefore, merging S_p and S_q increases the total PA utility, a contradiction to the optimality of S. Hence, S' does not contain parts that satisfies condition in case (2). □

Using the same example as in Lemma 4, we have following result.

Proposition 6 *Egalitarian-PA does not satisfy Majority-based aggregation.*

We go on to consider Total-PAM and Egal-PAM.

Proposition 7 *Total-PAM does not satisfy Majority-based aggregation.*

Proof. We show this by a counterexample. Consider a set of 2 projects, $\{p_1, p_2\}$; and a set of 5 voters. Let the partition for 3 voters be $\{\{p_1, p_2\}\}$, and for 2 voters, the partition is $\{\{p_1\}, \{p_2\}\}$. Note that $\{\{p_1, p_2\}\}$ is a majority based aggregated partition; its total PAM utility is 0. However, the total PAM utility for $\{\{p_1\}, \{p_2\}\}$ is 1. □

Using the same example as in Lemma 7, we have following result.

Proposition 8 *Egalitarian-PAM does not satisfy the property of majority based aggregation.*

5.3 Independent of Irrelevant Projects

Next we consider our adaptation of the axiom of Independence of Irrelevant Alternatives to our setting.

Proposition 9 *Total-PA does not satisfy Independence of Irrelevant Projects.*

Proof. We show this by a counterexample. Let $\mathtt{P}$ and $\mathtt{P}'$ be two profiles, and we have 3 voters. Suppose that the partitions for $\mathtt{P}$ are $P_{v_1} = \{\{p_1, p_2, p_3\}, \{p_4\}\}$, $P_{v_2} = \{\{p_1, p_2, p_3\}, \{p_4\}\}$, and $P_{v_3} = \{\{p_1\}, \{p_2, p_3\}, \{p_4\}\}$; and for $\mathtt{P}'$, partitions are $P'_{v_1} = \{\{p_1, p_2, p_3\}, \{p_4\}\}$, $P'_{v_2} = \{\{p_1, p_2, p_4\}, \{p_3\}\}$, and $P'_{v_3} = \{\{p_1, p_3\}, \{p_2, p_4\}\}$. We consider here pair of project p_1, p_2. In both the profiles p_1 and p_2 are in same part for 2 voters. Let S and S' be aggregated partitions for $\mathtt{P}$ and $\mathtt{P}'$, respectively. Since Total-PA is unanimous (Lemma 2), projects p_3 and p_4 are in different parts in both the partitions S and S'. Moreover, in S, $\{p_4\}$ is a singleton, and p_2, p_3 are in same part. Now, if p_1 is in the same part as in p_2, p_3; total PA utility for S' is 16, otherwise 14. Now, let us consider partition S'. Suppose that p_1 and p_2 are in same part. Then, there are three cases: either $S' = \{\{p_1, p_2\}, \{p_3\}, \{p_4\}\}$ or $S' = \{\{p_1, p_2, p_3\}, \{p_4\}\}$ or $S' = \{\{p_3\}, \{p_1, p_2, p_4\}\}$. In all the case total PA utility is 11. However, if $S' = \{\{p_1, p_3\}, \{p_2, p_4\}\}$, then total PA utility is 12. □

Observe that, if an aggregation rule is not unanimous, then it can not satisfy independent of irrelevant projects properties as we can have one profile in which all the projects are in the same part (or, equivalently, in different parts) for all voters, and another profile which is an instance that violates the Unanimity axiom. Hence, due to Lemma 4, we have following result.

Corollary 2 *Egalitarian-PA does not satisfy Independence of Irrelevant Projects.*

6 Outlook

We have studied several partition aggregation methods for the setting in which each voter provides an arbitrary partition over the same set of projects, and the aggregation goal is to output a single aggregated partition. Here we discuss few other variants of this setting and describe some use-cases for each of them.

6.1 Partitions with k Parts

In certain cases it might be desired to require that each input partition, as well as the output partition, shall consist of at most a given number k of parts. As a natural use-case, consider a set of projects for participatory budgeting where the election organizer wishes to partition the projects into thematic subjects, to ease the elicitation process. It is natural to require not-too-many parts, does requiring some upper bound of k parts. Such scenarios can be modeled as follows: Given n input partitions, each consisting of at most k parts, the aggregation method shall output a single aggregated partition, also consisting of at most k parts.

As a different example, of a different flavor, consider a system employing liquid democracy, in which the system wishes to cluster the proposals on the table into predefined subjects (say, education, health, etc.). Then, one approach would be to ask a subset of the voters to assign the proposals to the different, given subjects. Such a scenario can be modeled as follows: We have a set of k *types* and each voter shall partition the projects into these types. The aggregation method shall then output a partition of the projects into these types.

Observe that, in the first example, the k parts are indistinguishable, while, in the second example, these are distinguishable, as they have different IDs. We thus refer to the problem corresponding to the first example as *partition into k anonymous parts*, and to the problem corresponding to the second example as *partition into k distinguishable parts*.

k Anonymous Parts: The problem of partitioning a set of projects into k anonymous parts is formally equivalent to the problem we mainly study in this paper, with the only difference that we restrict the input partitions, as well as the output partition, to consist of at most k parts. Due to NP-hardness of Mirkin Distance Minimization given by Chen et al. [7], Total PA is NP-hard even for $k = 2$.

An interesting approach to the problem of aggregating partitions into k anonymous parts is via clustering: For example, embed the set of projects into a metric space where each project p_i is an element and the distance between a pair of projects p_i and p_j equals the number of voters placing p_i and p_j in different parts. Then, one can use, e.g., k-means algorithm to find k projects acting as centers, and, viewing each center as a representative of a part, assign to each center all projects that are the closest to this center among all k centers. Other option would be to define those k centers as the set of k projects that, if chosen as centers, would minimize the maximum distance between any pair of projects assigned to the same cluster; we mention that this aggregation method does not seem to be definable via voter-utilities of the sort we consider in this paper. Studying such clustering algorithm is left for future work.

k Distinguishable Parts: This problem variant is significantly different from the problem we mainly study in this paper, as we are not interested in standard partitions, but in partitions into distinguishable parts. In fact, we can view the input votes, as well as the aggregated output as strings: Each vote is a string of length m (recall that m is the number of projects) and each character i in such a string is a number between 1 to k, representing the type of the project p_i. Thus, we have in fact that, for this variant, Egal-PA is equivalent to the NP-hard problem Closest String.

6.2 Aggregating Signed Partitions

Recall that the main motivation for our study of partition aggregation methods is of deciding on a substitution structure for participatory budgeting instances. A related task would be to decide upon a complementarities structure, in which projects inside a complementarity part in the output partition would enjoy supermodular relations (in contrast to projects inside a substitution structure, which should suffer from submodular relations). Indeed one can use our partition aggregation methods for deciding upon a complementarities structure as well as for deciding upon a substitution structure.

A more involved task would be to simultaneously aggregate substitution structures and complementarity structures: That is, given a set of projects, we would like to partition them into parts, and decide, for each part, whether it is a substitution part or a complementarity part. Modeling this task would be natural via studying signed-partitions aggregation methods: Each voter provides a signed-partition, which is a partition of the set of projects such that each partition is labeled with, say, either $+$ (for complementarity part) or $-$ (for substitution part); the task of a signed-partitions aggregation method would be to output a signed-partition over the same set of projects.

The problem of aggregating signed-partitions is fundamentally different than the problem we mainly study here (i.e., aggregation non-signed partitions), and we leave it as a natural future research direction.

REFERENCES

[1] Haris Aziz, Piotr Faliszewski, Bernard Grofman, Arkadii Slinko, and Nimrod Talmon. Egalitarian committee scoring rules. In *Proceedings of IJCAI '18*, pages 56–62, 2018.

[2] Haris Aziz, Barton E Lee, and Nimrod Talmon. Proportionally representative participatory budgeting: Axioms and algorithms. In *Proceedings of AAMAS '18*, pages 23–31, 2018.

[3] Gerdus Benade, Swaprava Nath, Ariel D Procaccia, and Nisarg Shah. Preference elicitation for participatory budgeting. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[4] Paola Bonizzoni, Gianluca Della Vedova, Riccardo Dondi, and Tao Jiang. On the approximation of correlation clustering and consensus clustering. *Journal of Computer and System Sciences*, 74(5):671–696, 2008.

[5] Florian Brandl, Felix Brandt, Dominik Peters, Christian Stricker, and Warut Suksompong. Donor coordination: Collective distribution of individual contributions, 2019.

[6] Yves Cabannes. Participatory budgeting: a significant contribution to participatory democracy. *Environment and Urbanization*, 16(1):27–46, 2004.

[7] Jiehua Chen, Danny Hermelin, and Manuel Sorge. A note on clustering aggregation. *CoRR*, abs/1807.08949, 2018.

[8] Martin Dörnfelder, Jiong Guo, Christian Komusiewicz, and Mathias Weller. On the parameterized complexity of consensus clustering. *Theoretical Computer Science*, 542:71–82, 2014.

[9] Ulle Endriss and Umberto Grandi. Graph aggregation. *Artificial Intelligence*, 245:86–114, 2017.

[10] Piotr Faliszewski, Piotr Skowron, Arkadii Slinko, and Nimrod Talmon. Committee scoring rules: Axiomatic classification and hierarchy. pages 250–256, 2016.

[11] Piotr Faliszewski and Nimrod Talmon. A framework for approval-based budgeting methods. In *AAAI '19*, 2019.

[12] Rupert Freeman, David M Pennock, Dominik Peters, and Jennifer Wortman Vaughan. Truthful aggregation of budget proposals, 2019.

[13] Aristides Gionis, Heikki Mannila, and Panayiotis Tsaparas. Clustering aggregation. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1:4, 2007.

[14] Ron Shamir, Roded Sharan, and Dekel Tsur. Cluster graph modification problems. *Discrete Applied Mathematics*, 144(1-2):173–182, 2004.

[15] Alexander Strehl and Joydeep Ghosh. Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *Journal of machine learning research*, 3:583–617, 2002.