Linear Combinations of Unordered Data Vectors

Linear Combinations of Unordered Data Vectors

Piotr Hofman
University of Warsaw, Poland

Jérôme Leroux
LaBRI, University of Bordeaux
Talence, France

Patrick Totzke
University of Edinburgh, UK

***Abstract*—Data vectors generalise finite multisets: they are finitely supported functions into a commutative monoid. We study the question whether a given data vector can be expressed as a finite sum of others, only assuming that 1) the domain is countable and 2) the given set of base vectors is finite up to permutations of the domain.**

Based on a succinct representation of the involved permutations as integer linear constraints, we derive that positive instances can be witnessed in a bounded subset of the domain.

For data vectors over a group we moreover study when a data vector is reversible, that is, if its inverse is expressible using only nonnegative coefficients. We show that if all base vectors are reversible then the expressibility problem reduces to checking membership in finitely generated subgroups. Moreover, checking reversibility also reduces to such membership tests.

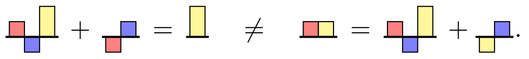
These questions naturally appear in the analysis of counter machines extended with unordered data: namely, for data vectors over $(\mathbb{Z}^d, +)$ expressibility directly corresponds to checking state equations for Coloured Petri nets where tokens can only be tested for equality. We derive that in this case, expressibility is in *NP*, and in *P* for reversible instances. These upper bounds are tight: they match the lower bounds for standard integer vectors (over singleton domains).

I. Introduction

Finite collections of named values are basic structures used in many areas of theoretical computer science. We can formalize these as functions $\mathbf{v} : \mathbb{D} \to X$ from some countable domain $\mathbb{D}$ of *names* or *data*, into some value space X, and call such functions (X-valued) *data vectors*. Often the actual names used are not relevant and instead one is interested in data vectors *up to renaming*, i.e., one wants to consider vectors $\mathbf{v}$ and $\mathbf{w}$ equivalent if $\mathbf{v} = \mathbf{w} \circ \theta$ for some permutation $\theta : \mathbb{D} \to \mathbb{D}$ of the domain.

We consider the case where the value space X has additional algebraic structure. Namely, we focus on data vectors where the values are from some commutative monoid $(M, +, 0)$ and where all but finitely many names are mapped to the neutral element. A natural question then asks if a given data vector is expressible as a sum of vectors from a given set, where the monoid operation is lifted to data vectors pointwise.

If the spanning set is finite only up to permutations, this problem does not immediately boil down to solving finite system of linear equations. Also, one cannot simply lift operations on data vectors to equivalence classes of data vectors because the result of pointwise applying the operation depends on the chosen representants. For example, if we have data vectors mapping colours to integers, then the vector $(red \mapsto -1, blue \mapsto 1)$ is equivalent to . Yet still,



We thus choose to keep permutations explicit and consider the *Expressibility* problem:

Input: A finite set V of data vectors and a target vector $\mathbf{x}$.
Question: does $\mathbf{x}$ equal $\sum_{i=1}^{n} \mathbf{v_i} \circ \theta_i$ for some $\mathbf{v_i} \in V$ and permutations $\theta_i : \mathbb{D} \to \mathbb{D}$?

If the domain $\mathbb{D}$ is finite then this just asks if some finite system of linear equations is satisfiable. Over infinite domains this corresponds to finding a solution of an infinite but regular set of linear equations. For brevity, we will call a vector $\mathbf{x}$ a *permutation sum* of V if it is expressible as a sum of permutations of vectors in V as above.

Contributions and Outline.: We provide two reductions from the Expressibility problem to problems of finding solutions for *finite* linear systems over $(M, +)$.

The more general approach, a small witness property, is presented in Sections III and IV and ultimately works by bounding the number of different data values necessary to express a permutation sum. This is based on an analysis of objects we call *histograms*, see Section III, which sufficiently characterize vectors expressible as permutation sums of single vectors. For any monoid $(M, +)$ the Expressibility problem then reduces to finding a non-negative integer solution of a finite system of linear inequations over $(M, +)$. In particular, for $(Z^d, +)$, the monoid of integer vectors of length d, this provides an *NP* algorithm, matching the lower bound from the feasibility of integer linear programs.

The second approach (Section V) reduces Expressibility to the problem of finding an (not necessarily non-negative) integer solution to a finite linear system. This assumes that $(M, +)$ is a group and that all data vectors in V are reversible, i.e., their inverses are expressible as permutation sums of V. We show that this reversibility condition can be verified by checking the existence of a solution of a simple system over $(M, +)$. For example, if we work over $(Z^d, +)$ then checking reversibility of V can be done in polynomial time. If V happens to be reversible then solving the Expressibility problem is also in polynomial time.

In Section VI we show how to use these results for the reachability analysis of counter programs extended with data.

Piotr Hofman acknowledges support from the NSC, grand number 2013/09/B/ST6/01575, and the NSC, grand number 2016/21/D/ST6/01368. Patrick Totzke acknowledges support from the EPSRC, grant EP/M027651/1.

The first application involves finding state invariants for unordered data Petri nets [1], [2], [3]; the second application is the reachability problem for blind counter automata [4] extended with data.

Related Research and Motivation.: Our main motivation for studying the Expressibility problem comes from the analysis of Petri nets extended with data [1], [2], [5], [6], [3]. For *unordered data Petri Nets* (UDPN), data vectors of the form $\mathbf{v} : \mathbb{D} \to \mathbb{Z}^d$ occur naturally. UDPNs are the only data extension of Petri nets, among those proposed in [1], for which reachability problem may be decidable (for all other models discussed there, undecidability follows by reduction from the boundedness problem for reset nets [7]). The first indicator that reachability may be decidable for UDPN is the decidability of place-boundedeness via a characterization of the coverability sets [3].

The Expressibility problem studied in this paper allows to generalize so-called *state equations* (also marking equations in the Petri net literature) [8] to UDPN. State equations are a fundamental invariant of the reachability relation for Petri nets and one of the important ingredients in the proof of decidability of the reachability problem [9], [10], [11]. They provide cheaply computable overapproximations for reachability that can be used to prune the seach space of the backwards coverability algorithm [12], [13], [14], [15]. This approach is especially efficient if combined with SMT-solver as done in [16], [17], [18].

UDPN can also be seen as a restriction of more general *Colored Petri nets* (CPN), see for instance [19]. There is a long history of research in the area of restricted CPNs. Here, we point to results about invariants and identification of certain syntactic substructures: in [20], [21] the authors investigate flows in subclasses of CPN. Another important branch of research concerns structural properties of Algebraic Nets [22] like detecting siphons [23] or other place invariants [24].

Finally, it is worth mentioning that UDPN can be interpreted as ordinary Petri nets for sets with *equality atoms*. We refer the reader to [25], [26] for work on sets with atoms/nominal sets. Very similar in spirit to our study of data vectors is the work in [27] that considers constraint satisfaction problems on infinite structures.

A broader overview of algebraic methods for Petri nets can be found in [28]. Algebraic methods are also used in restricted classes of Petri nets like conflict-free and free-choice Petri nets [29]. A beautiful application of algebraic techniques are results on reachability in continuous Petri nets [30], [31]. One can also find variants of state equations for Petri nets with resetting transitions [32] or with inhibitor arcs [33].

II. Data Vectors

In the sequel $\mathbb{D}$ is a countable set of elements called *data values* and $(M, +)$ is a commutative monoid with neutral element 0. A *data vector* (also *vector* for short) is a total function $\mathbf{v} : \mathbb{D} \to M$ such that the *support*, the set $\mathit{supp}(\mathbf{v}) \overset{\text{def}}{=} \{\alpha \in \mathbb{D} \mid \mathbf{v}(\alpha) \neq 0\}$ is finite. The monoid operation $+$ is lifted to vectors poinwise, so that $(\mathbf{v} + \mathbf{w})(\alpha) \overset{\text{def}}{=} \mathbf{v}(\alpha) + \mathbf{w}(\alpha)$.

Writing $\circ$ for function composition, we see that $\mathbf{v} \circ \theta$ is a data vector for any data vector $\mathbf{v}$ and permutation $\theta : \mathbb{D} \to \mathbb{D}$. A vector $\mathbf{x}$ is said to be a *permutation sum* of a set V of vectors if there are $\mathbf{v_1}, \dots, \mathbf{v_n}$ in V and permutations $\theta_1, \dots, \theta_n$ of $\mathbb{D}$ such that

$$\mathbf{x} = \sum_{i=1}^{n} \mathbf{v_i} \circ \theta_i.$$

Here, we have to emphasize that it is possible that $\mathbf{v_i} = \mathbf{v_j}$ for some i and j.

When working with vectors of the form $\mathbf{v} \circ \theta$ for permutations θ, it will be instrumental to specify θ indirectly using some injection of $\mathit{supp}(\mathbf{v})$ into $\mathbb{D}$. In the remainder of this section we show that one can always do this.

Take any finite subset $\mathbb{S}$ of $\mathbb{D}$ and $\pi : \mathbb{S} \to \mathbb{D}$ injective. Since π^{-1} is only a partial function, define the data vector $\mathbf{v} \circ \pi^{-1}$ so that any β not in the range of π maps to 0: More precisely, let $(\mathbf{v} \circ \pi^{-1}) : \mathbb{D} \to M$ be defined, for every $\beta \in \mathbb{D}$ as follows.

$$\mathbf{v} \circ \pi^{-1}(\beta) = \begin{cases} \mathbf{v}(\alpha) & \text{if } \pi(\alpha) = \beta \\ 0 & \text{if } \pi(\alpha) \neq \beta \text{ for all } \alpha \in \mathbb{S}. \end{cases}$$

Lemma 1. *Let $\mathbf{v}$ be a data vector. For any permutation $\theta : \mathbb{D} \to \mathbb{D}$ there exists an injection $\pi : \mathit{supp}(\mathbf{v}) \to \mathbb{D}$, such that $\mathbf{v} \circ \pi^{-1} = \mathbf{v} \circ \theta$, and vice versa.*

Proof: Let us introduce $\mathbb{S} = \mathit{supp}(\mathbf{v})$ and first consider a permutation θ. We show that the injection $\pi : \mathbb{S} \to \mathbb{D}$, defined by $\pi(\alpha) = \theta^{-1}(\alpha)$ for every $\alpha \in \mathbb{S}$, satisfies $\mathbf{v} \circ \theta = \mathbf{v} \circ \pi^{-1}$.

Pick any $\beta \in \mathbb{D}$ and let us write $\alpha \overset{\text{def}}{=} \theta(\beta)$. If $\alpha \in \mathbb{S}$ then $\pi(\alpha) = \beta$ so, we have that $(\mathbf{v} \circ \pi^{-1})(\beta) = \mathbf{v}(\alpha) = (\mathbf{v} \circ \theta)(\beta)$. If $\alpha \notin \mathbb{S}$ then $(\mathbf{v} \circ \pi^{-1})(\beta) = 0$ and $\mathbf{v} \circ \theta(\beta) = \mathbf{v}(\alpha) = 0$, by definition of $\mathbf{v} \circ \pi^{-1}$ and because $\mathbb{S}$ is the support of $\mathbf{v}$.

Conversely, let us consider a data injection π over $\mathbb{S}$ and let us prove that there exists a data permutation θ such that $\mathbf{v} \circ \pi^{-1} = \mathbf{v} \circ \theta$. We introduce $\mathbb{T} = \pi(\mathbb{S})$. Since the restriction of π on $\mathbb{S}$ is a bijection onto $\mathbb{T}$, there exists a bijection $\pi' : \mathbb{T} \to \mathbb{S}$ denoting its inverse. We introduce the sets $X = \mathbb{S} \backslash \mathbb{T}$, and $Y = \mathbb{T} \backslash \mathbb{S}$. Since $\mathbb{T}$ and $\mathbb{S}$ have the same cardinal, it follows that X and Y are two finite sets with the same cardinal. Hence, there exists a bijection $\pi_{X,Y} : X \to Y$ and its inverse $\pi_{Y,X} : Y \to X$. We introduce the function θ defined for every $\beta \in \mathbb{D}$ as follows:

$$\theta(\beta) = \begin{cases} \pi'(\beta) & \text{if } \beta \in \mathbb{T} \\ \pi_{X,Y}(\beta) & \text{if } \beta \in X \\ \beta & \text{otherwise.} \end{cases}$$

Observe that θ is a bijection since the function θ' defined for every $\alpha \in \mathbb{D}$ as follows is its inverse:

$$\theta'(\alpha) = \begin{cases} \pi(\alpha) & \text{if } \alpha \in \mathbb{S} \\ \pi_{Y,X}(\alpha) & \text{if } \alpha \in Y \\ \alpha & \text{otherwise.} \end{cases}$$

We show that $\mathbf{v} \circ \pi^{-1} = \mathbf{v} \circ \theta$. Fix some $\beta \in \mathbb{D}$ and assume first that $\beta \in \mathbb{T}$. There exists $\alpha \in \mathbb{S}$ such that $\pi(\alpha) = \beta$. It follows that $\pi'(\beta) = \alpha = \theta(\beta)$. Hence, $(\mathbf{v} \circ \theta)(\beta) = \mathbf{v}(\alpha) = (\mathbf{v} \circ \pi^{-1})(\beta)$.

Now, assume that $\beta \notin \mathbb{T}$. In that case, notice that $\theta(\beta)$ is not in $\mathbb{S}$ no matter if $\beta \in X$ or not. Thus $\mathbf{v} \circ \theta(\beta) = 0$. Observe that if $\pi^{-1}(\{\beta\})$ is empty, we deduce that $(\mathbf{v} \circ \pi^{-1})(\beta) = 0$. If $\pi^{-1}(\beta) = \{\alpha\}$ then $(\mathbf{v} \circ \pi^{-1})(\beta) = \mathbf{v}(\alpha)$. But since $\beta \notin \mathbb{T}$, we deduce that $\alpha \notin \mathbb{S}$. Therefore $\mathbf{v}(\alpha) = 0$ and we derive that $(\mathbf{v} \circ \pi^{-1})(\beta) = 0$. We have proved that $\mathbf{v} \circ \pi^{-1} = \mathbf{v} \circ \theta$. ■

III. Histograms

In this section we develop the notion of histograms. These are combinatorical objects that will be used in the next section to characterize permutation sums over singleton sets V. We elaborate more on the usefulness of histograms in the beginning of the next section.

Definition 2. A *histogram* over a finite set $\mathbb{S} \subseteq \mathbb{D}$ is a total function $H : \mathbb{S} \times \mathbb{D} \to \mathbb{N}$ such that for some $n \in \mathbb{N}$, called the *degree* of H, the following two conditions hold:

1) $\sum_{\beta \in \mathbb{D}} H(\alpha, \beta) = n$ for any $\alpha \in \mathbb{S}$,
2) $\sum_{\alpha \in \mathbb{S}} H(\alpha, \beta) \leq n$ for any $\beta \in \mathbb{D}$.

A histogram of degree $n = 1$ is called *simple*. Histograms with the same signature, i.e. the same sets $\mathbb{S}$ and $\mathbb{D}$, can be partially ordered and summed pointwise and the degree of the sum is the sum of degrees. The *support* of H is the set $\mathit{supp}(H) \overset{\text{def}}{=} \{\beta \mid \sum_{\alpha \in \mathbb{S}} H(\alpha, \beta) > 0\}$.

The following theorem states the main combinatorial property we are interested in, namely that simple histograms over $\mathbb{S}$ generate as finite sums the class of all histograms over $\mathbb{S}$. In particular, any histogram can be decomposed into finitely many simple histograms over the same signature (see Figure 1 for an illustration).

Theorem 3. *A function $H : \mathbb{S} \times \mathbb{D} \to \mathbb{N}$ is a histogram of degree $n \in \mathbb{N}$ if, and only if, H is the sum of n simple histograms over $\mathbb{S}$.*

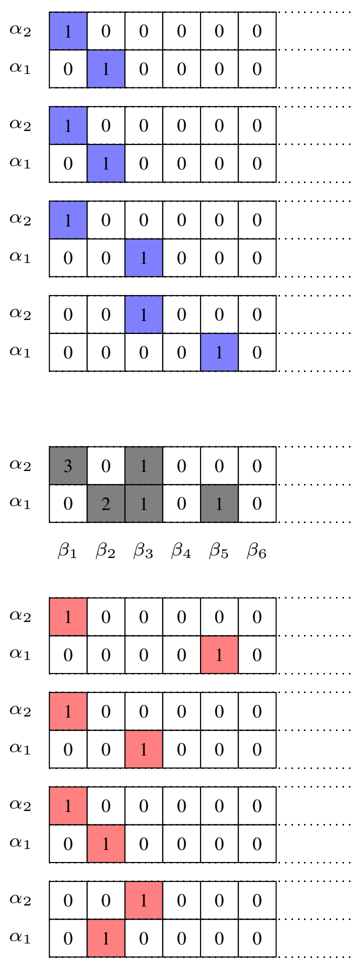
For the proof of Theorem 3 we need a lemma from graph theory. We refer the reader to [34] for relevant definitions and recall here only that in a graph (V, E), a *matching* of a set $S \subseteq V$ of nodes is a set $M \subseteq E$ of pairwise non-adjacent edges that covers all nodes in S.

Lemma 4. *Let $G = (L \cup R, E)$ be a bipartite graph. If there is a matching of $L' \subseteq L$ and a matching of $R' \subseteq R$ then there is a matching of $L' \cup R'$.*

Proof: Suppose, M_L and M_R are matchings that matches L' and R', respectively. Let $G' \overset{\text{def}}{=} (L \cup R, M_L \cup M_R)$ be a subgraph of G. We construct a matching M of $L' \cup R'$ as a matching in G'. Observe that G' is a union of single nodes, paths and cycles and M can be constructed in every strongly connected component independently.

We claim that for any strongly connected component C we can find a matching witch matches all elements in $C \cap L'$ and $C \cap R'$. This, if proved, ends the proof of Lemma 4.

First of all, every node in $L' \cup R'$ has a degree at least 1 so the claim holds for single nodes immediately.



	β_1	β_2	β_3	β_4	β_5	β_6
α_2	3	0	1	0	0	0
α_1	0	2	1	0	1	0

Fig. 1. The grey center depicts a histogram $H : \mathbb{S} \times \mathbb{D} \to \mathbb{N}$ of degree 4, where $\mathbb{S} = \{\alpha_1, \alpha_2\}$ and $\mathit{supp}(H) = \{\beta_1, \beta_2, \beta_3, \beta_5\}$. Below (in red) and to above (in blue) are decompositions into four simple histograms each.

Next we consider cycles, and paths of even and odd lengths, were the length denotes the number of vertices in the path.

If the strongly connected component is a cycle (in bipartite graph) or a path of an even length then there is a perfect matching in it so the claim holds as well.

The case of paths of odd length is the most complicated one. Let C be an odd length path. Without loss of generality, suppose that the first node x on the path C is in $L' \cup R'$. Indeed, if it is not, then we do not need to match x, so we match all nodes on the path except x; this case is done.

Further, without loss of generality, we can assume that $x \in L'$, as $x \in R'$ is symmetric. We prove that the path ends in a vertex from the set $L \setminus L'$, in other words, it is not possible that the both ends of the path C belong to $L' \cup R'$. Indeed the path has to end in L as it's length is odd. Furthermore, consider a walk along the path starting from x. Every vertex in L' on the path C we leave via an edge from M_L and to every vertex of C in $L' \setminus \{x\}$ we enter via an edge from M_R. From the above if the last vertex belonged to L' then there would be a contradiction because we would be able to leave it via an edge from M_L as from any vertex in L' there is an outgoing edge in M_L. Thus, the path cannot end in the element from L' and thus the last vertex has to belong to $L \setminus L'$. Now we match all vertices except of the last one. ■

Proof of Theorem 3: If $H = \sum_{j=1}^{n} H_j$, where the H_j are simple histograms over $\mathbb{S}$, then from $\sum_{\beta\in\mathbb{D}} H(\alpha,\beta) = \sum_{j=1}^{n}\sum_{\beta\in\mathbb{D}} H_j(\alpha,\beta)$ and because $\sum_{\beta\in\mathbb{D}} H_j(\alpha,\beta) = 1$ we derive that $\sum_{\beta\in\mathbb{D}} H(\alpha,\beta) = n$ for every $\alpha \in \mathbb{D}$. In the same way the second histogram condition for H follows from those of the H_j. So H is a histogram over $\mathbb{S}$.

For the converse direction, the proof proceeds by induction on n, the degree of the histogram H. If $n = 1$ then H is simple and the claim trivially holds. Suppose now that the claim holds for histograms of degree n and consider a histogram $H : \mathbb{S} \times \mathbb{D} \to \mathbb{N}$ of degree $n+1$. We show how that there exists a simple histogram $X : \mathbb{S} \times \mathbb{D} \to \mathbb{N}$ such that

1) $X(\alpha,\beta) \le H(\alpha,\beta)$ for every $\alpha \in \mathbb{S}$, and every $\beta \in \mathbb{D}$, and
2) for every $\beta \in \mathbb{D}$ if $\sum_{\alpha\in\mathbb{S}} H(\alpha,\beta) = n+1$ then $\sum_{\alpha\in\mathbb{S}} X(\alpha,\beta) = 1$.

The first condition guarantees that $Y = H - X$ is well defined and satisfies, for all $\alpha \in \mathbb{S}$, that

$$\begin{aligned}\sum_{\beta\in\mathbb{D}} Y(\alpha,\beta) &= \left(\sum_{\beta\in\mathbb{D}} H(\alpha,\beta)\right) - \left(\sum_{\beta\in\mathbb{D}} X(\alpha,\beta)\right)\\ &= (n+1) - 1 = n.\end{aligned}$$

The second condition implies that $\sum_{\alpha\in\mathbb{S}} Y(\alpha,\beta) \le n$ for all $\beta \in \mathbb{D}$. Hence, Y is a histogram of degree n and the claim follows by induction hypothesis.

To show the existence of a suitable simple histogram X, we consider now the bipartite graph G where the sets of nodes are $\mathbb{S}$ and $\mathbb{B} \overset{\text{def}}{=} \mathit{supp}(H)$ and where there is an edge between α and β whenever $H(\alpha,\beta) > 0$ (We assume here w.l.o.g. that $\mathbb{S}$ and $\mathit{supp}(H)$ are disjoint; otherwise take $\mathbb{B}$ as some suitable duplication). Moreover, let $\mathbb{T}$ denotes the set of those "maximal" data values β where $\sum_{\alpha\in\mathbb{D}} H(\alpha,\beta) = n+1$. Note that $\mathbb{T} \subseteq \mathbb{B}$.

We claim that the required simple histogram X exists iff there is a matching in the graph G that matches both $\mathbb{S}$ and $\mathbb{T}$. Indeed, any such histogram X provides a matching $M \overset{\text{def}}{=} \{(\alpha,\beta) \mid X(\alpha,\beta) = 1\}$. By the first histogram condition, M matches all nodes in $\mathbb{S}$; and all nodes $\beta \in \mathbb{T}$ are matched since X must satisfy $\sum_{\alpha\in\mathbb{S}} X(\alpha,\beta) = 1$. Conversely, for a given matching M we define $X(\alpha,\beta) = 1$ if $(\alpha,\beta) \in M$ and 0 otherwise. It is easy to check that X is a simple histogram that satisfies required properties.

To finish the proof we show that a matching M of $\mathbb{S} \cup \mathbb{T}$ exists. By Lemma 4, it suffices to find two matchings, one of $\mathbb{S}$ and one of $\mathbb{T}$. In both cases, we will make use of Hall's marriage Theorem [34].

We write $\mathit{Nb}(S)$ for the *neighbourhood* (formally open neighbourhood) of a set S of nodes (the set of nodes $v \notin S$ adjacent to some node in S).

Hall's marriage Theorem states that in a finite bipartite graph there is a matching of a set S of nodes if, and only if every subset $S' \subseteq S$ has at least as many neighbours as elements:

$$|S'| \le |\mathit{Nb}(S')|. \tag{1}$$

We start by proving the existence of a matching of $\mathbb{S}$. If we label the edges (α,β) in our graph by the respective values $H(\alpha,\beta)$, then we observe that the total weight of edges connecting any subset $\mathbb{S}' \subseteq \mathbb{S}$ is at most the total weight of edges connecting its neighbours: $\sum_{\alpha\in\mathbb{S}'}\sum_{\beta\in\mathbb{D}} H(\alpha,\beta) \le \sum_{\alpha\in\mathbb{S}}\sum_{\beta\in\mathit{Nb}(\mathbb{S}')} H(\alpha,\beta)$. Consequently,

$$\begin{aligned}|\mathbb{S}'| \cdot (n+1) = \sum_{\substack{\alpha\in\mathbb{S}'\\ \beta\in\mathbb{D}}} H(\alpha,\beta) &\le \sum_{\substack{\alpha\in\mathbb{S}\\ \beta\in\mathit{Nb}(\mathbb{S}')}} H(\alpha,\beta)\\ &\le \sum_{\beta\in\mathit{Nb}(\mathbb{S}')} (n+1) = |\mathit{Nb}(\mathbb{S}')| \cdot (n+1).\end{aligned}$$

The first equality is due to the first histogram condition and the second inequality is by the second histogram condition. So all subsets $\mathbb{S}' \subseteq \mathbb{S}$ satisfy Equation (1), so Hall's theorem applies and there exists a matching of $\mathbb{S}$.

The proof that a matching of $\mathbb{T}$ exists follows the same pattern. For any subset $\mathbb{T}' \subseteq \mathbb{T}$ we get

$$\begin{aligned}|\mathbb{T}'| \cdot (n+1) = \sum_{\beta\in\mathbb{T}'}\sum_{\alpha\in\mathbb{S}} H(\alpha,\beta) &\le \sum_{\alpha\in\mathit{Nb}(\mathbb{T}')}\sum_{\beta\in\mathbb{D}} H(\alpha,\beta)\\ &= |\mathit{Nb}(\mathbb{T}')| \cdot (n+1),\end{aligned}$$

where the first equality holds by the definition of $\mathbb{T}$. So $\mathbb{T}$ satisfies the assumption of Hall's theorem and we conclude that some matching of $\mathbb{T}$ exists, as required. ■

IV. Expressibility

In this section, we show that histograms provide a natural tool for deciding if a data vector is a permutation sum.

Histograms should be seen as a compressed representation of a permutation sum. In Lemmas 5 and 6, we show how to produce a histogram from a sum of permuted data vectors, and that this encoding preserves the outcome of the sum. The idea is that even if we could bound the number of data appearing in the sum by some polynomial then still we would need to deal with the exponential number of possible permutations of data vectors i.e. possibly exponential length of the sum; this would need another trick to obtain NP upper-bound. Histograms are exactly the objects that allow forgetting about actual permutations and focusing on the effect of some existing sum, that we do not know explicitly.

We first establish the connection between histograms and permutation sums, and then (in Theorem 7) that permutation sums can be represented by histograms with bounded support sets. Finally, we derive an *NP* complexity upper bound for the Expressibility problem for vectors over monoids $(\mathbb{Z}^d, +)$.

Given a data vector $\mathbf{v}$ and a histogram H over $\mathbb{S} \overset{\text{def}}{=} \mathit{supp}(\mathbf{v})$ we define the vector $\mathit{eval}(\mathbf{v}, H)$ by

$$\mathit{eval}(\mathbf{v}, H)(\beta) \overset{\text{def}}{=} \sum_{\alpha\in\mathbb{S}} \mathbf{v}(\alpha) \cdot H(\alpha,\beta).$$

Observe that eval is a homomorphism in the sense that for any vector $\mathbf{v}$ and histograms H_1, H_2 over $\mathbb{S}$ it holds that

$$\mathit{eval}(\mathbf{v}, H_1 + H_2) = \mathit{eval}(\mathbf{v}, H_1) + \mathit{eval}(\mathbf{v}, H_2). \quad (2)$$

Recall that by Lemma 1, permutation sums are of the form $\mathbf{x} = \sum_{i=1}^{n} \mathbf{v}_i \circ \pi_i^{-1}$ where $\pi_i : \mathit{supp}(\mathbf{v}_i) \to \mathbb{D}$ are injections. We now associate each injective function $\pi : \mathbb{S} \to \mathbb{D}$ with the simple histogram H_π over $\mathbb{S}$ defined as

$$H_\pi(\alpha, \beta) \overset{\text{def}}{=} \begin{cases} 1 & \text{if } \beta = \pi(\alpha) \\ 0 & \text{otherwise.} \end{cases}$$

Notice that conversely, each simple histogram $H : \mathbb{S} \times \mathbb{D} \to \mathbb{D}$ provides a unique injection $\pi_H : \mathbb{S} \to \mathbb{D}$ satisfying $H(\alpha, \pi_H(\alpha)) = 1$ for every $\alpha \in \mathbb{S}$. So, $H_{\pi_H} = H$. The next lemma makes the connection between histograms and permutation sums.

Lemma 5. *Let $\mathbf{v}$ be a vector and $\pi : \mathit{supp}(\mathbf{v}) \to \mathbb{D}$ injective. Then $\mathbf{v} \circ \pi^{-1} = \mathit{eval}(\mathbf{v}, H_\pi)$.*

Proof: If $\pi(\alpha) = \beta$ then $\mathbf{v} \circ \pi^{-1}(\beta) = \mathbf{v}(\alpha) = \sum_{\alpha' \in \mathbb{S}} \mathbf{v}(\alpha') H_\pi(\alpha', \beta) = \mathit{eval}(\mathbf{v}, H_\pi)(\beta)$ where the second equation holds because H_π is simple. If $\pi(\alpha) \neq \beta$ for all α then $\mathbf{v} \circ \pi^{-1}(\beta) = 0 = \sum_{\alpha \in \mathbb{S}} \mathbf{v}(\alpha) \cdot 0 = \mathit{eval}(\mathbf{v}, H_\pi)(\beta)$. ■

Lemma 6. *Let $\mathbf{v}$ be a vector. A vector $\mathbf{x}$ is a permutation sum of $\{\mathbf{v}\}$ if, and only if, there exists a histogram H over $\mathit{supp}(\mathbf{v})$ such that $\mathbf{x} = \mathit{eval}(\mathbf{v}, H)$.*

Proof: Assume first that $\mathbf{x}$ is a permutation sum of $\{\mathbf{v}\}$. Then there are permutations $\theta_1, \ldots, \theta_n$ such that $\mathbf{x} = \sum_{j=1}^{n} \mathbf{v} \circ \theta_j$. By Lemma 1, $\mathbf{x} = \sum_{j=1}^{n} \mathbf{v} \circ \pi_j^{-1}$ for injections $\pi_j : \mathit{supp}(\mathbf{v}) \to \mathbb{D}$. From Lemma 5 we derive $\mathbf{x} = \sum_{j=1}^{n} \mathit{eval}(\mathbf{v}, H_{\pi_j})$ and by Equation (2) the histogram $H \overset{\text{def}}{=} \sum_{j=1}^{n} H_{\pi_j}$ satisfies $\mathbf{x} = \mathit{eval}(\mathbf{v}, H)$.

Conversely, let us assume that there exists a histogram H over $\mathit{supp}(v)$ such that $\mathbf{x} = \mathit{eval}(\mathbf{v}, H)$. Theorem 3 shows that H can be decomposed as $H = \sum_{j=1}^{n} H_j$ where H_j are simple histograms over $\mathit{supp}(\mathbf{v})$. From Equation (2) it follows that $\mathbf{x} = \sum_{j=1}^{n} \mathit{eval}(\mathbf{v}, H_j)$, and since all H_j are simple, there are injections $\pi_1, \ldots, \pi_n$ with $H_j = H_{\pi_j}$. The claim thus follows by Lemma 5 and Lemma 1. ■

We now show how to bound the supports of histograms $H_\mathbf{v}$ such that $\mathbf{x} = \sum_{\mathbf{v} \in V} \mathit{eval}(\mathbf{v}, H_\mathbf{v})$ with respect to $\mathbf{x}$ and V.

Theorem 7. *A data vector $\mathbf{x}$ is a permutation sum of V if and only if $\mathbf{x} = \sum_{\mathbf{v} \in V} \mathit{eval}(\mathbf{v}, H_\mathbf{v})$ where for each vector $\mathbf{v} \in V$, $H_\mathbf{v}$ is a histogram over $\mathit{supp}(\mathbf{v})$, and $|\bigcup_{\mathbf{v} \in V} \mathit{supp}(H_\mathbf{v})|$ is bounded by $|\mathit{supp}(\mathbf{x})| + 1 + \sum_{\mathbf{v} \in V}(2|\mathit{supp}(\mathbf{v})| - 1)$.*

Proof: Any permutation sum of V is a sum $\mathbf{x} = \sum_{\mathbf{v} \in V} \mathbf{x}_\mathbf{v}$, where each $\mathbf{x}_\mathbf{v}$ is a permutation sum of $\{\mathbf{v}\}$. So, from Lemma 6 we conclude that $\mathbf{v}$ is a permutation sum of V iff there is a set of histograms $H_\mathbf{v}$ with $\mathbf{x}_\mathbf{v} = \mathit{eval}(\mathbf{v}, H_\mathbf{v})$. Now, we only need to prove that existence of any solution implies existence of a solution with a 'small' support.

For each histogram $H_\mathbf{v}$ we write $n_\mathbf{v}$ for its degree and define the set of *big* data values as

$$\mathbb{B}_\mathbf{v} \overset{\text{def}}{=} \left\{ \beta \in \mathbb{D} \mid \sum_{\alpha \in \mathit{supp}(\mathbf{v})} H_\mathbf{v}(\alpha, \beta) > \frac{n_\mathbf{v}}{2} \right\}.$$

The cardinality of $\mathbb{B}_\mathbf{v}$ is bounded by $2|\mathit{supp}(\mathbf{v})| - 1$. Indeed, if $\mathbb{B}_\mathbf{v}$ is empty, the property is immediate. Otherwise, we have $\sum_{\beta \in \mathbb{B}_\mathbf{v}} \sum_{\alpha \in \mathit{supp}(\mathbf{v})} H_\mathbf{v}(\alpha, \beta) > |\mathbb{B}_\mathbf{v}| \frac{n_\mathbf{v}}{2}$. It moreover holds that $\sum_{\beta \in \mathbb{B}_\mathbf{v}} \sum_{\alpha \in \mathit{supp}(\mathbf{v})} H_\mathbf{v}(\alpha, \beta) \leq \sum_{\alpha \in \mathit{supp}(\mathbf{v})} \sum_{\beta \in \mathbb{D}} H_\mathbf{v}(\alpha, \beta) = |\mathit{supp}(\mathbf{v})| \cdot n_\mathbf{v}$. Therefore, $|\mathbb{B}_\mathbf{v}| \leq 2|\mathit{supp}(\mathbf{v})| - 1$ holds.

To provide the bound claimed in the theorem, suppose that the histograms $H_\mathbf{v}$ are chosen such that their combined support $\mathbb{T} \overset{\text{def}}{=} \bigcup_{\mathbf{v} \in V} \mathit{supp}(H_\mathbf{v})$ is minimal. We show that this set cannot have more than $|\mathit{supp}(\mathbf{x})| + 1 + \sum_{\mathbf{v} \in V}(2|\mathit{supp}(\mathbf{v})| - 1)$ elements, which implies the claim.

Suppose towards a contradiction that $|\mathbb{T}|$ exceeds this bound. Then it must contain two distinct elements β_1 and β_2 that are both neither in $\bigcup_{\mathbf{v} \in V} \mathbb{B}_\mathbf{v}$ nor in $\mathit{supp}(\mathbf{x})$. Notice that the first condition, that β_1, β_2 are not big in any histogram $H_\mathbf{v}$ guarantees that for all $\mathbf{v} \in V$,

$$\sum_{\alpha \in \mathit{supp}(\mathbf{v})} H_\mathbf{v}(\alpha, \beta_1) + \sum_{\alpha \in \mathit{supp}(\mathbf{v})} H_\mathbf{v}(\alpha, \beta_2) \leq n_\mathbf{v}. \quad (3)$$

Based on β_1 and β_2 we introduce, for each $\mathbf{v} \in V$, the function $F_\mathbf{v} : \mathit{supp}(\mathbf{v}) \times \mathbb{D} \to \mathbb{N}$ as

$$F_\mathbf{v}(\alpha, \beta) = \begin{cases} H_\mathbf{v}(\alpha, \beta_1) + H_\mathbf{v}(\alpha, \beta_2) & \text{if } \beta = \beta_1 \\ 0 & \text{if } \beta = \beta_2 \\ H_\mathbf{v}(\alpha, \beta) & \text{otherwise.} \end{cases}$$

Then for any $\alpha \in \mathit{supp}(v)$ we have $\sum_{\beta \in \mathbb{D}} F_\mathbf{v}(\alpha, \beta) = n_\mathbf{v}$ and by Equation (3) we have $\sum_{\alpha \in \mathit{supp}(\mathbf{v})} F_\mathbf{v}(\alpha, \beta_1) \leq n_\mathbf{v}$. So $F_\mathbf{v}$ is a histogram over $\mathit{supp}(\mathbf{v})$ of degree $n_\mathbf{v}$. We claim that

$$\sum_{\mathbf{v} \in V} \mathit{eval}(\mathbf{v}, F_\mathbf{v}) = \sum_{\mathbf{v} \in V} \mathit{eval}(\mathbf{v}, H_\mathbf{v}).$$

Indeed, $F_\mathbf{v}$ trivially satisfies $\mathit{eval}(\mathbf{v}, H_\mathbf{v}) = \mathit{eval}(\mathbf{v}, F_\mathbf{v})$ for all data except of β_1 and β_2. Thus

$$\mathbf{x}(\beta) = \left(\sum_{\mathbf{v} \in V} \mathbf{x}_\mathbf{v} \right)(\beta) = \left(\sum_{\mathbf{v} \in V} \mathit{eval}(\mathbf{v}, F_\mathbf{v}) \right)(\beta)$$

for all $\beta \notin \{\beta_1, \beta_2\}$. Moreover, $\beta_2 \notin \mathit{supp}(\mathbf{x})$ so $\mathbf{x}(\beta_2) = 0$. On the other hand $\left(\sum_{\mathbf{v} \in V} \mathit{eval}(\mathbf{v}, F_\mathbf{v})\right)(\beta_2) = 0$ as for every $\mathbf{v}$ it holds that $\mathit{eval}(\mathbf{v}, F_\mathbf{v})(\beta_2) = 0$. Finally, $\beta_1 \notin \mathit{supp}(\mathbf{x})$ so $\mathbf{x}(\beta_1) = 0$. On the other hand

$$\begin{aligned} &\left(\sum_{\mathbf{v} \in V} \mathit{eval}(\mathbf{v}, H_\mathbf{v}) \right)(\beta_1) + \left(\sum_{\mathbf{v} \in V} \mathit{eval}(\mathbf{v}, H_\mathbf{v}) \right)(\beta_2) \\ &= \left(\sum_{\mathbf{v} \in V} \mathbf{x}_\mathbf{v} \right)(\beta_1) + \left(\sum_{\mathbf{v} \in V} \mathbf{x}_\mathbf{v} \right)(\beta_2) \\ &= \mathbf{x}(\beta_1) + \mathbf{x}(\beta_2) = 0 + 0 = 0. \end{aligned}$$

We conclude that $\mathbf{x} = \sum_{\mathbf{v}\in V} \mathit{eval}(\mathbf{v}, F_{\mathbf{v}})$. But this contradicts the minimality of $|\mathbb{T}|$ as $\bigcup_{\mathbf{v}\in V} \mathit{supp}(F_{\mathbf{v}}) = \mathbb{T}\setminus\{\beta_2\}$ is a strict subset of $\mathbb{T}$. ■

Corollary 8. *The Expressibility problem for data vectors with values in $(\mathbb{Z}^d, +)$ is NP-complete.*

Proof: We show the upper bound only, as a matching lower bound holds already for singleton domains $\mathbb{D}$, where the problem is equivalent to the feasibility of integer linear programs.

Suppose, we ask if $\mathbf{x}$ is a permutation sum of a set V. By Theorem 7, positive instances imply the existence of a set of histograms $H_{\mathbf{v}\in V}$. Each $H_{\mathbf{v}\in V}$ is over $\mathit{supp}(\mathbf{v})$ and the size of the sum of their supports is bounded by $|\mathit{supp}(\mathbf{x})| + 1 + \sum_{\mathbf{v}\in V}(2|\mathit{supp}(\mathbf{v})| - 1)$. More precise, the sum of supports has to contain $\mathit{supp}(\mathbf{x})$ and at most $1 + \sum_{\mathbf{v}\in V}(2|\mathit{supp}(\mathbf{v})| - 1)$ of other data values (as they are auxiliary data, we don't need to specify them precisely). Further, $\mathbf{x} = \sum_{\mathbf{v}\in V} \mathit{eval}(\mathbf{v}, H_{\mathbf{v}})$. By Theorem 7, the existence of such histograms is also a sufficient condition for $\mathbf{x}$ to be a permutation sum of V.

Now we explain that existence of such set of histograms $H_{\mathbf{v}\in V}$ can be reduced to finding a non-negative integer solution of a system of linear equations of polynomial size. So the claim follows from standard results for integer linear programming, see e.g. [35].

From above, each histogram $H_{\mathbf{v}}$ can be specified by $|\mathit{supp}(\mathbf{v})| \cdot (|\mathit{supp}(\mathbf{x})| + 1 + \sum_{\mathbf{v}\in V}(2|\mathit{supp}(\mathbf{v})| - 1))$ non-negative integer numbers. Thus for each $H_{\mathbf{v}}$ we create a unique set of variables $Var_{\mathbf{v}}$, one variable for every possible non-negative integer number. So in total the number of variables is polynomial. Next we to write equations which enforce that any correct valuation of variables will form histograms $H_{\mathbf{v}}$. Precisely, as liner equations we encode, the histogram conditions (Definition 2) and the equation $\mathbf{x} = \sum_{\mathbf{v}\in V} \mathit{eval}(\mathbf{v}, H_{\mathbf{v}})$. Histogram conditions are simple as we only need to check *(i)* if sums of variables forming rows of a histogram are equal each other and *(ii)* that they are not smaller than sums of variables that form columns. The $\mathit{eval}()$ condition is also simple to encode, as for a fixed size histogram the $\mathit{eval}()$ function is linear.

It is worth emphasizing that the system of equations can be written immediately from the problem definition so it can be easily encoded in any SMT solver supporting integers. ■

V. Reversibility

In this section we consider data vectors $\mathbf{v} : \mathbb{D} \to G$ where $(G, +)$ is a commutative group. In this case the set of all vectors is itself a commutative group with identity $\mathbf{0}$. We write $-\mathbf{v}$ for the inverse of vector $\mathbf{v}$ and $\mathbf{v} - \mathbf{w} \overset{\text{def}}{=} \mathbf{v} + (-\mathbf{w})$.

Definition 9. A vector $\mathbf{x} : \mathbb{D} \to G$ is *reversible in* V if both $\mathbf{x}$ and $-\mathbf{x}$ are permutation sums of V. A set of vectors V is reversible if every vector $\mathbf{v} \in V$ is reversible in V.

We will provide in this section a way to reduce the Expressibility problem to the membership problems in finitely generated subgroups of $(G, +)$, assuming the given set of data vectors is reversible. This result is stated as Theorem 15. We also show (as Theorem 11), that checking the reversibility condition boils down to solving a finite linear system over $(G, +)$.

Our constructions are based on the homomorphism *weight*, which projects data vectors into the underlying group: the weight of a data vector $\mathbf{v} : \mathbb{D} \to G$ is the element of G defined as

$$\mathit{weight}(\mathbf{v}) \overset{\text{def}}{=} \sum_{\alpha\in\mathbb{D}} \mathbf{v}(\alpha).$$

For a set V of data vectors define $\mathit{weight}(V) \overset{\text{def}}{=} \{\mathit{weight}(\mathbf{v}) \mid \mathbf{v} \in V\}$.

In addition we introduce some useful notation. Fix any total order on $\mathbb{D}$. The *rotation* of a finite set $\mathbb{S} \subseteq \mathbb{D}$ is the permutation $\mathit{rotate}_{\mathbb{S}} : \mathbb{D} \to \mathbb{D}$ defined as

$$\mathit{rotate}_{\mathbb{S}}(\alpha) \overset{\text{def}}{=} \begin{cases} \min\{\mathbb{S}\} & \text{if } \alpha = \max\{\mathbb{S}\} \\ \min\{\beta \in \mathbb{S} \mid \beta > \alpha\} & \text{if } \max\{\mathbb{S}\} \neq \alpha \in \mathbb{S} \\ \alpha & \text{if } \alpha \notin \mathbb{S}. \end{cases}$$

This allows us to express for instance the vector $\mathbf{v} \circ \mathit{rotate}_{\{\alpha,\beta\}}$, which results from the vector $\mathbf{v}$ by exchanging the values of α and β. Clearly, for any finite set $\mathbb{S} \subseteq \mathbb{D}$, the $|\mathbb{S}|$-fold composition of $\mathit{rotate}_{\mathbb{S}}$ with itself is the identity on $\mathbb{D}$.

Finally, we introduce vectors $\mathbf{rot}_{\mathbb{S}}(\mathbf{v}) : \mathbb{D} \to \mathbb{Z}^d$ as

$$\mathbf{rot}_{\mathbb{S}}(\mathbf{v}) \overset{\text{def}}{=} \sum_{i=0}^{|\mathbb{S}|-1} \mathbf{v} \circ \mathit{rotate}_{\mathbb{S}}^{i}$$

where $\mathbf{v}$ is a data vector, $\mathit{supp}(\mathbf{v}) \subseteq \mathbb{S} \subseteq \mathbb{D}$ is finite, and the superscripts denote i-fold iteration. It is the result of summing up all different $\mathbb{S}$-rotations of $\mathbf{v}$. This vector is useful because it is a permutation sum of $\mathbf{v}$ that "equalizes" all values for $\alpha \in \mathbb{S}$ to $\mathit{weight}(\mathbf{v})$, as stated in the proposition below.

Proposition 10. *Let $\mathbf{v} : \mathbb{D} \to G$ be a data vector and $\mathbb{S} \subseteq \mathbb{D}$ finite such that $\mathit{supp}(\mathbf{v}) \subseteq \mathbb{S}$.*

1) $\mathbf{rot}_{\mathbb{S}}(\mathbf{v})$ *is a permutation sum of* $\{\mathbf{v}\}$.
2) $\mathbf{rot}_{\mathbb{S}}(\mathbf{v})(\alpha) = \mathit{weight}(\mathbf{v})$ *if* $\alpha \in \mathbb{S}$ *and* $\mathbf{rot}_{\mathbb{S}}(\mathbf{v})(\alpha) = 0$ *if* $\alpha \notin \mathbb{S}$.

Proof: The first property is immediate by definition of $\mathbf{rot}_{\mathbb{S}}(\mathbf{v})$. Concerning the second one, observe that if $\alpha \notin \mathbb{S}$ then $\mathbf{v} \circ \mathit{rotate}_{\mathbb{S}}^{i}(\alpha) = \mathbf{v}(\alpha) = 0$ for every i. Hence $\mathbf{rot}_{\mathbb{S}}(\mathbf{v})(\alpha) = 0$. If $\alpha \in \mathbb{S}$. If $\alpha \in \mathbb{S}$, notice that $\{\mathit{rotate}_{\mathbb{S}}^{i}(\alpha) \mid 0 \leq i < |\mathbb{S}|\} = \mathbb{S}$. It follows that $\mathbf{rot}_{\mathbb{S}}(\mathbf{v})(\alpha) = \sum_{\beta\in\mathbb{S}} v(\beta)$. From $\mathit{supp}(\mathbf{v}) \subseteq \mathbb{S}$ we derive $\sum_{\beta\in\mathbb{S}} v(\beta) = \mathit{weight}(\mathbf{v})$. We have proved the second property. ■

Identifying Reversible Sets of Vectors:

Theorem 11. *Let V be a set of data vectors and $\mathbf{x} \in V$. Then $\mathbf{x}$ is reversible in V if, and only if, $\mathit{weight}(\mathbf{x})$ is reversible in $\mathit{weight}(V)$, i.e., there exist $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n \in V$ such that $-\mathit{weight}(\mathbf{x}) = \sum_{i=1}^{n} \mathit{weight}(\mathbf{v}_i)$.*

Proof: For the only if direction we need to show that if $-\mathbf{x} = \sum_{i=1}^{n} \mathbf{v_i} \circ \theta_i$ for vectors $\mathbf{v_i} \in V$ and permutations $\theta_i : \mathbb{D} \to \mathbb{D}$, then $-\mathit{weight}(\mathbf{x})$ may be expressed as a sum of vectors $\mathit{weight}(\mathbf{v}_1), \mathit{weight}(\mathbf{v}_2), \ldots, \mathit{weight}(\mathbf{v}_n)$. Since *weight* is a homomorphism we observe that $-\mathit{weight}(\mathbf{x})$ is expressible as the sum $\sum_{i=1}^{n} \mathit{weight}(\mathbf{v_i} \circ \theta_i)$. The claim follows from the fact that $\mathit{weight}(\mathbf{v_i} \circ \theta_i) = \mathit{weight}(\mathbf{v_i})$ for all $\mathbf{v_i} : \mathbb{D} \to G$.

For the opposite direction assume vectors $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_n} \in V$ such that $-\mathit{weight}(\mathbf{x}) = \sum_{j=1}^{n} \mathit{weight}(\mathbf{v_j})$ and let $\mathbb{S} \overset{\text{def}}{=} \bigcup_{i=1}^{n} \mathit{supp}(\mathbf{v_i})$. First, we aim to show that

$$-\mathbf{rot}_{\mathbb{S}}(\mathbf{x}) = \sum_{j=1}^{n} \mathbf{rot}_{\mathbb{S}}(\mathbf{v_j}), \tag{4}$$

that is, $-\mathbf{rot}_{\mathbb{S}}(\mathbf{x})(\alpha) = \sum_{j=1}^{n} \mathbf{rot}_{\mathbb{S}}(\mathbf{v_j})(\alpha)$ for all $\alpha \in \mathbb{D}$. As $\mathbf{x} \in V$, by definition of $\mathbf{rot}_{\mathbb{S}}(\mathbf{x})$, this trivially holds for $\alpha \notin \mathbb{S}$. For the remaining $\alpha \in \mathbb{S}$, note that by point 2 of Proposition 10 we have $\mathbf{rot}_{\mathbb{S}}(\mathbf{v})(\alpha) = \mathit{weight}(\mathbf{v})$ for any $\mathbf{v} \in V$. In particular this holds for $\mathbf{x}$ and all $\mathbf{v_j}$. So,

$$-\mathbf{rot}_{\mathbb{S}}(\mathbf{x})(\alpha) = -\mathit{weight}(\mathbf{x}) = \sum_{j=1}^{n} \mathbf{rot}_{\mathbb{S}}(\mathbf{v_j})(\alpha) \tag{5}$$

which proves Equation (4). Unfolding the definition of $\mathbf{rot}_{\mathbb{S}}(\mathbf{x})$ we therefore see that

$$-\mathbf{rot}_{\mathbb{S}}(\mathbf{x}) = -\left(\mathbf{x} + \sum_{i=1}^{|\mathbb{S}|-1} \mathbf{x} \circ \mathit{rotate}_{\mathbb{S}}^{i}\right) = \sum_{j=1}^{n} \mathbf{rot}_{\mathbb{S}}(\mathbf{v_j})$$

and consequently that $-\mathbf{x} = (\sum_{i=1}^{|\mathbb{S}|-1} \mathbf{x} \circ \mathit{rotate}_{\mathbb{S}}^{i}) + \sum_{j=1}^{n} \mathbf{rot}_{\mathbb{S}}(\mathbf{v_j})$. Now, $(\sum_{i=1}^{|\mathbb{S}|-1} \mathbf{x} \circ \mathit{rotate}_{\mathbb{S}}^{i})$ is clearly a permutation sum of $\{\mathbf{x}\}$ and thus also of V. By point 1 of Proposition 10, $\sum_{j=1}^{n} \mathbf{rot}_{\mathbb{S}}(\mathbf{v_j})$ is also a permutation sum of V. We conclude that $-\mathbf{x}$ is a permutation sum of V and therefore that $\mathbf{x}$ is reversible in V. ■

Corollary 12. *Let $d \in \mathbb{N}$ and V a finite set of data vectors $\mathbf{v} : \mathbb{D} \to \mathbb{Z}^d$. There is a polynomial time procedure that checks if V is reversible.*

Proof: By Theorem 11, it suffices to verify for all $\mathbf{v} \in V$ that $-\mathit{weight}(\mathbf{v})$ is the sum of elements in $\mathit{weight}(V)$. In other words, we need to check for an element h of a finite set $H = \{h_1, \ldots, h_k\} \subseteq \mathbb{Z}^d$, that there exist $n_1, \ldots, n_k \in \mathbb{N}$ with $-h = n_1 h_1 + \cdots + n_k h_k$.

We show that the condition above is satisfied if, and only if, there exists $\lambda_1, \ldots, \lambda_k \in \mathbb{Q}_{\geq 0}$ such that $-h = \lambda_1 h_1 + \cdots + \lambda_k h_k$. The "only if" direction is immediate. For the converse, assume factors $\lambda_1, \ldots, \lambda_k \in \mathbb{Q}_{\geq 0}$ such that $-h = \lambda_1 h_1 + \cdots + \lambda_k h_k$. There exists a positive integer p such that $p\lambda_j \in \mathbb{N}$ for every j and thus $-h = (p\lambda_1)h_1 + \cdots + (p\lambda_k)h_k + (p-1)h$. We have proved the claim.

Now, checking if $-h = \lambda_1 h_1 + \cdots + \lambda_k h_k$ has a solution in the non-negative rationals is doable in polynomial time using linear programming [36]. ■

Solving Expressibility in Reversible Sets of Vectors: In the remainder of this subsection we prove Theorem 15. We start with a lemma. All constructions in this section assume $\max_{\mathbf{v} \in V} |\mathit{supp}(\mathbf{v})| < |\mathbb{D}|$, that there exists at least one fresh datum in the domain.

We first prove that some special data vectors are permutation sums of V. Those vectors are defined by introducing for every element $g \in G$ and every data value $\alpha \in \mathbb{D}$, the data vector $[\![\alpha \mapsto g]\!]$ defined for every $\beta \in \mathbb{D}$ by:

$$[\![\alpha \mapsto g]\!](\beta) \overset{\text{def}}{=} \begin{cases} g & \text{if } \beta = \alpha \\ 0 & \text{otherwise.} \end{cases}$$

Lemma 13. *Let V be a finite, reversible set of data vectors such that $\bigcup_{\mathbf{v} \in V} \mathit{supp}(\mathbf{v}) \subsetneq \mathbb{D}$. Then, $[\![\beta \mapsto g]\!]$ is a permutation sum of V for every g in the subgroup of $(G, +)$ generated by $\mathit{weight}(V)$, and for every $\beta \in \mathbb{D}$.*

Proof: Since g is in the subgroup generated by $\mathit{weight}(V)$, there exist a sequence $\mathbf{v_1}, \ldots, \mathbf{v_n}$ of elements in V such that:

$$g = \sum_{j=1}^{n} \mathit{weight}(\mathbf{v_j}).$$

Consider now the vector $\mathbf{x} \overset{\text{def}}{=} \sum_{j=1}^{n} \mathbf{v_j}$. It has three relevant properties:

1) it is a permutation sum of V,
2) its support is contained in $\bigcup_{\mathbf{v} \in V} \mathit{supp}(\mathbf{v})$, and
3) it satisfies $g = \mathit{weight}(\mathbf{x})$.

By point 2 and the assumption that the combined support $\bigcup_{\mathbf{v} \in V} \mathit{supp}(\mathbf{v})$ is strictly included in $\mathbb{D}$, we can pick some $\alpha \notin \mathit{supp}(\mathbf{x})$. Let $\mathbb{S}, \mathbb{T} \subseteq \mathbb{D}$ be defined as

$$\mathbb{S} \overset{\text{def}}{=} \mathit{supp}(\mathbf{x}) \qquad \text{and} \qquad \mathbb{T} \overset{\text{def}}{=} \mathit{supp}(\mathbf{x}) \cup \{\alpha\}.$$

Then by Proposition 10 point 1, both $\mathbf{rot}_{\mathbb{S}}(\mathbf{x})$ and $\mathbf{rot}_{\mathbb{T}}(\mathbf{x})$ are permutation sums of V. Since V is reversible, so is the inverse $-\mathbf{rot}_{\mathbb{S}}(\mathbf{x})$. It remains to observe that

$$[\![\alpha \mapsto g]\!] = \mathbf{rot}_{\mathbb{T}}(\mathbf{x}) - \mathbf{rot}_{\mathbb{S}}(\mathbf{x}).$$

Indeed, for all $\delta \notin \mathbb{T}$ we have $[\![\alpha \mapsto g]\!](\delta) = \mathbf{rot}_{\mathbb{S}}(\mathbf{x})(\delta) = \mathbf{rot}_{\mathbb{T}}(\mathbf{x})(\delta) = 0$. For all $\delta \in \mathbb{S}$, by Proposition 10 point 2, it holds that $\mathbf{rot}_{\mathbb{S}}(\mathbf{x})(\delta) = \mathbf{rot}_{\mathbb{T}}(\mathbf{x})(\delta) = \mathit{weight}(\mathbf{x}) = g$ and therefore that $[\![\alpha \mapsto g]\!](\delta) = \mathbf{rot}_{\mathbb{S}}(\mathbf{x})(\delta) - \mathbf{rot}_{\mathbb{T}}(\mathbf{x})(\delta) = g - g = 0$. For the last case that $\delta = \alpha \in \mathbb{T} \setminus \mathbb{S}$, again by Proposition 10 point 2, we have $[\![\alpha \mapsto g]\!](\delta) = \mathbf{rot}_{\mathbb{S}}(\mathbf{x})(\delta) - \mathbf{rot}_{\mathbb{T}}(\mathbf{x})(\delta) = g - 0 = g$. Now, the data vector $[\![\beta \mapsto g]\!] = [\![\alpha \mapsto g]\!] \circ \mathit{rotate}_{\{\alpha,\beta\}}$ completes the proof. ■

Lemma 14. *Let V be a finite, reversible set of data vectors such that $\bigcup_{\mathbf{v} \in V} \mathit{supp}(\mathbf{v}) \subsetneq \mathbb{D}$. The data vector $[\![\alpha \mapsto g]\!] - [\![\beta \mapsto g]\!]$ is a permutation sum of V for every g in the subgroup of $(G, +)$ generated by $\{\mathbf{v}(\delta) \mid \delta \in \mathbb{D},\ \mathbf{v} \in V\}$, and for every $\alpha, \beta \in \mathbb{D}$.*

Proof: Let $\mathbb{T} \overset{\text{def}}{=} \bigcup_{\mathbf{v} \in V} \mathit{supp}(\mathbf{v})$ and assume w.l.o.g. that $\alpha \neq \beta$ since otherwise the claim is trivial. It suffices to show the claim for $\alpha \in \mathbb{T}$ and $\beta \notin \mathbb{T}$.

We first show that there exists a data vector $\mathbf{x}$ that is a permutation sum of V such that $\mathbf{x}(\alpha) = g$ and such that $supp(\mathbf{x}) \subseteq \mathbb{T}$. Since g is in the subgroup generated by $\{\mathbf{v}(\delta) \mid \delta \in \mathbb{D},\ \mathbf{v} \in V\}$, there exist vectors $\mathbf{v_1}, \ldots, \mathbf{v_n} \in V$ and data values $\delta_1, \ldots, \delta_n \in \mathbb{D}$ such that:

$$g = \sum_{j=1}^{n} \mathbf{v_j}(\delta_j).$$

We can assume without loss of generality that $\mathbf{v_j}(\delta_j)$ are not equal to zero and hence $\delta_j \in \mathbb{T}$.

Let $\theta_j \overset{\text{def}}{=} rotate_{\{\alpha,\delta_j\}} : \mathbb{D} \to \mathbb{D}$ be the permutation that exchanges α and δ_j and consider the vector $\mathbf{x}$, defined as follows.

$$\mathbf{x} = \sum_{j=1}^{n} \mathbf{v_j} \circ \theta_j.$$

Observe that $\mathbf{x}$ is a permutation sum of V and $\mathbf{x}(\alpha) = g$ as it was required. Moreover, since $\delta_j, \alpha \in \mathbb{T}$, we deduce that $supp(\mathbf{v_j} \circ \theta_j)$ is included in $\mathbb{T}$ and therefore that $supp(\mathbf{x}) \subseteq \mathbb{T}$.

To show the claim, let $\theta \overset{\text{def}}{=} rotate_{\{\alpha,\beta\}}$ be the permutation that swaps α and β and consider the vector

$$\mathbf{y} \overset{\text{def}}{=} \mathbf{x} - \mathbf{x} \circ \theta.$$

For all $\delta \in \mathbb{D} \setminus \{\alpha, \beta\}$ we get $\mathbf{y}(\delta) = \mathbf{x}(\delta) - \mathbf{x}(\theta(\delta)) = \mathbf{x}(\delta) - \mathbf{x}(\delta) = 0$. Moreover, $\mathbf{y}(\alpha) = \mathbf{x}(\alpha) - \mathbf{x}(\theta(\alpha)) = g - \mathbf{x}(\beta) = g$, similarly $\mathbf{y}(\beta) = -g$. We conclude that $\mathbf{y}$ is a permutation sum of V and $\mathbf{y} = [\![\alpha \mapsto g]\!] - [\![\beta \mapsto g]\!]$. ■

We can now prove our main theorem. It allows reducing question about the reversibility of data vector to questions about expressibility in the given group, for example, in Z^d with addition. The conditions (below) may look technical, but they are natural constraints; for example the first one says that the weight of the sum has to be expressible as a sum of weights (which is obvious considering that weight is a homomorphism). The not trivial part of the theorem is that the conditions imply expressibility, as stated below.

Theorem 15. *Let V be a finite, reversible set of data vectors with $\bigcup_{\mathbf{v} \in V} supp(\mathbf{v}) \subsetneq \mathbb{D}$. A data vector $\mathbf{x}$ is a permutation sum of V if, and only if, the following two conditions hold.*

- *$weight(\mathbf{x})$ is in the subgroup of $(G, +)$ generated by $\{weight(\mathbf{v}) \mid \mathbf{v} \in V\}$, and*
- *$\mathbf{x}(\alpha)$ is in the subgroup of $(G, +)$ generated by $\{\mathbf{v}(\delta) \mid \delta \in \mathbb{D},\ \mathbf{v} \in V\}$ for every $\alpha \in \mathbb{D}$.*

Proof: If $\mathbf{x}$ is a permutation sum of V there exists a sequence $\mathbf{v_1}, \ldots, \mathbf{v_n}$ of data vectors in V and a sequence $\theta_1, \ldots, \theta_n$ of data permutations such that $\mathbf{x} = \sum_{j=1}^{n} \mathbf{v_j} \circ \theta_j$. We derive that $weight(\mathbf{x}) = \sum_{j=1}^{n} weight(\mathbf{v_j} \circ \theta_j)$. Since $weight(\mathbf{v_j} \circ \theta_j) = weight(\mathbf{v_j})$, it follows that $weight(\mathbf{x})$ is in the group generated by $weight(V)$. Moreover, for every $\alpha \in \mathbb{D}$, we have $\mathbf{x}(\alpha) = \sum_{j=1}^{n} \mathbf{v_j}(\theta_j(\alpha))$. Thus $\mathbf{x}(\alpha)$ is in the group generated by $\{\mathbf{v}(\delta) \mid \delta \in \mathbb{D},\ \mathbf{v} \in V\}$.

For the converse direction, assume that $\mathbf{x}$ is a data vector satisfying the two conditions. We pick $\delta \in \mathbb{D}$. From condition 2 and Lemma 14 we derive that for every $\alpha \in supp(\mathbf{x})$, the data vector $[\![\alpha \mapsto \mathbf{x}(\alpha)]\!] - [\![\delta \mapsto \mathbf{x}(\alpha)]\!]$ is a permutation sum of V. It follows that the $\mathbf{y} \overset{\text{def}}{=} \sum_{\alpha \in supp(\mathbf{x})} ([\![\alpha \mapsto \mathbf{x}(\alpha)]\!] - [\![\delta \mapsto \mathbf{x}(\alpha)]\!])$ is a permutation sum of V. Notice that this vector is equal to $\mathbf{x} - [\![\delta \mapsto weight(\mathbf{x})]\!]$. By condition 1, Lemma 13 applies and implies that $[\![\delta \mapsto weight(\mathbf{x})]\!]$ is a permutation sum of V. So, $\mathbf{x}$ must be a permutation sum of V. ■

Corollary 16. *Let $\mathbb{D}$ be infinite, $d \in \mathbb{N}$, and V a finite, reversible set of data vectors $\mathbf{v} : \mathbb{D} \to \mathbb{Z}^d$. There is a polynomial time procedure that determines if a given target data vector $\mathbf{x} : \mathbb{D} \to \mathbb{Z}^d$ is a permutation sum of V.*

Remark 17. To motivate the freshness assumption, $\bigcup_{\mathbf{v} \in V} supp(\mathbf{v}) \subsetneq \mathbb{D}$ consider the following example, which shows that the two conditions in the claim of Theorem 15 are not necessarily sufficient on its own.

$\mathbb{D}$ is the finite set $\{\alpha_1, \ldots, \alpha_k\}$ where $\alpha_1, \ldots, \alpha_k$ are distinct and $k \geq 2$. Assume that there exists $m \in G$ such that $k \cdot m \neq 0$. We introduce $V = \{\mathbf{v}, -\mathbf{v}\}$ where $\mathbf{v} = [\![\alpha_1 \mapsto m]\!] + \cdots + [\![\alpha_k \mapsto m]\!]$ and $\mathbf{x} = [\![\alpha_1 \mapsto (k \cdot m)]\!]$. Observe that $\mathbf{x}$ satisfies the two conditions of Theorem 15. Assume by contradiction that $\mathbf{x}$ is a permutation sum of V. Since $\mathbf{v} \circ \theta = \mathbf{v}$ for every data permutation θ it follows that $\mathbf{x} = z \cdot \mathbf{v}$ for some $z \in \mathbb{Z}$. Thus $0 = \mathbf{x}(\alpha_2) = z \cdot \mathbf{v}(\alpha_2) = z \cdot m$, and $k \cdot m = \mathbf{x}(\alpha_1) = z \cdot \mathbf{v}(\alpha_1) = z \cdot m$. Hence $k \cdot m = 0$ and we get a contradiction. Hence $\mathbf{x}$ is not a permutation sum of V.

VI. Applications

A. Unordered data Petri nets

Unordered data nets [2], [5], [6] extend the classical model of Petri nets [37], [38], [39] such that each token carries a datum from a countable set $\mathbb{D}$. In this basic algebraic extension, transition firing can only depend on equality constraints over the involved tokens. We recall an equivalent definition from [40] which has more of a *vector addition system* [41] flavour.

Definition 18 (UDPN)**.** Fix an integer $d \in \mathbb{N}$ and a countable domain $\mathbb{D}$. A d-dimensional *unordered data Petri net* (UDPN) is a finite set $\mathcal{T}$ of data vectors over $\mathbb{Z}^d$.

A *transition* is a data vector $\mathbf{t} \overset{\text{def}}{=} \mathbf{f} \circ \sigma$, where $\mathbf{f} \in \mathcal{T}$ and σ is a permutation of $\mathbb{D}$. A *configuration* is a data vector over $\mathbb{N}^d$, i.e., a finitely supported function $\mathbf{conf} : \mathbb{D} \to \mathbb{N}^d$.

We say that there is a *step* $\mathbf{conf_0} \xrightarrow{t} \mathbf{conf_1}$ between configurations $\mathbf{conf_0}, \mathbf{conf_1}$ if $\mathbf{conf_1} = \mathbf{conf_0} + \mathbf{t}$ for some transition $\mathbf{t}$. Note that this enforces that $\mathbf{conf_0}(\alpha) + \mathbf{t}(\alpha) \geq 0$ for all α in $\mathbb{D}$ since $\mathbf{conf_1}$ is non-negative. We simply write $\mathbf{conf_0} \to \mathbf{conf_1}$ if $\mathbf{conf_0} \xrightarrow{\mathbf{t}} \mathbf{conf_1}$ for some transition $\mathbf{t}$ and let $\xrightarrow{*}$ denote the transitive and reflexive closure of $\to$.

As usual, the *reachability problem* asks, if $\mathbf{conf_0} \xrightarrow{*} \mathbf{conf_1}$ holds for given configurations $\mathbf{conf_0}$ and $\mathbf{conf_1}$.

Observe that UDPNs over any domain with cardinality $|\mathbb{D}| = 1$ are classical vector addition systems [41]. Notice also that, the set of transitions in an UDPN is finite up to permutations of $\mathbb{D}$ and that the step relation is closed under permutations: $\mathbf{conf_0} \to \mathbf{conf_1}$ implies that $\mathbf{conf_0} \circ \sigma \to \mathbf{conf_1} \circ \sigma$.

The decidability status of the reachability problem for UDPN is currently open. We will discuss here a necessary condition for positive instances, an invariant sometimes called *state equations* in the Petri net literature [33]. This can be formulated as follows.

Proposition 19 (State-Equation). *If* $\mathbf{conf_0} \xrightarrow{*} \mathbf{conf_1}$*, then* $(\mathbf{conf_1} - \mathbf{conf_0})$ *is a permutation sum of* $\mathcal{T}$.

We say configurations $\mathbf{conf_0}$ and $\mathbf{conf_1}$ satisfy the state-equation, if $(\mathbf{conf_1} - \mathbf{conf_0})$ is indeed a permutation sum of $\mathcal{T}$. That is, if there exists data vectors $\mathbf{t_1}, \mathbf{t_2}, \ldots, \mathbf{t_k} \in \mathcal{T}$ and permutations $\sigma_1, \sigma_2, \ldots, \sigma_k$ such that $(\mathbf{conf_1} - \mathbf{conf_0}) = \sum_{i=1}^{k} \mathbf{t_i} \circ \sigma_i$. Observe that $(\mathbf{conf_1} - \mathbf{conf_0}) \in \mathbb{D} \to \mathbb{Z}^d$ so the above is exactly the expressibility problem for data vectors with values in $(\mathbb{Z}^d, +)$.

A direct consequence of Corollaries 8 and 16 is that one can check this condition in (nondeterministic) polynomial time.

Theorem 20. *There is an NP algorithm that checks if two configurations of a UPDN satisfy the state equation.*

For reversible UDPN, checking state equations is in P.

The complexity of checking state equations for UDPN thus matches that of the same problem for ordinary Petri nets (via linear programming).

Finally, we remark that the decision problem whether a given UDPN is reversible is in P, by Corollary 12. Notice that reversibility (cf. Definition 9) is a fairly natural condition and compatible with the usual definition of reversibility in Petri nets: that the reachability relation $\xrightarrow{*}$ is symmetric. Indeed, if $\xrightarrow{*}$ is symmetric then $\mathcal{T}$ is reversible as otherwise there must be some $\mathbf{t} \in \mathcal{T}$ where $-\mathbf{t}$ is not a permutation sum of $\mathcal{T}$. So there are configurations $\mathbf{conf_1}, \mathbf{conf_0}$ and permutation σ with $\mathbf{conf_1} = \mathbf{conf_0} + (\mathbf{t} \circ \sigma)$ and $\mathbf{conf_0} \not\xrightarrow{*} \mathbf{conf_1}$.

B. Blind Counter Automata

Blind counter automata [4] are finite automata equipped with a number of registers that store integer values and which can be independently incremented or decremented in each step. These systems correspond to vector addition systems with states (VASS) over the integers [32], where transitions are always enabled. The model can be equipped with data in a natural way.

Definition 21. A (d-dimensional) *unordered data blind counter automaton* is given by a finite labelled transition system $\mathcal{A} \stackrel{\text{def}}{=} (Q, E, L)$ where edges in E are labelled by the function L with data vectors in $\mathbb{D} \to \mathbb{Z}^d$.

A *configuration* of the automaton is a pair $(q, \mathbf{v})$ where $q \in Q$ is a state and $\mathbf{v} \in \mathbb{D} \to \mathbb{Z}^d$. There is a step $(q, \mathbf{c_0}) \xrightarrow{e} (q', \mathbf{c_1})$ between two configurations $(q, \mathbf{c_0}), (q', \mathbf{c_1})$ if $e = (q, q') \in E$ and $\mathbf{c_1} = \mathbf{c_0} + L(e) \circ \theta$ for a permutation $\theta : \mathbb{D} \to \mathbb{D}$. The reachability relation is a transitive closure of the step relation and we denote it by $\xrightarrow{*}$. Finally, a sequence of configurations and transitions of a form $(q_0, \mathbf{x_0}) \xrightarrow{e_1} (q_1, \mathbf{x_1}) \xrightarrow{e_2} \ldots \xrightarrow{e_n} (q_n, \mathbf{x_n})$ we call a *path*.

Theorem 22. *The reachability problem for unordered data blind counter automata is in NP.*

Proof: We can provide a proof by decomposing paths into skeleton paths and multisets of simple cycles connected to it (like in [42], [43]). We prefer to provide a direct proof based on extract dimensions that count the number of times transitions are used by a path as follows.

Suppose $Q = \{q_1, \ldots, q_{|Q|}\}$, $E = \{e_1, e_2, \ldots, e_{|E|}\}$, and we ask if from $(q_1, \mathbf{c_0})$ it is possible to reach $(q_{|Q|}, \mathbf{c_1})$. We show how to encode the problem as the expressibility problem. Let $\alpha \in \mathbb{D}$ be a fresh data value i.e. $\alpha \notin supp(\mathbf{c_0}) \cup supp(\mathbf{c_1}) \cup supp(L(e))$ for any $e \in E$. First we introduce a new labelling function L' with an image in $\mathbb{D} \to \mathbb{Z}^{d+|Q|+|E|}$ defined as follows for any $\beta \in \mathbb{D} \setminus \{\alpha\}$ and $e_k = (q_i, q_j) \in E$ $L'(e_k)(\beta) = L(e_k)(\beta)$; $L'(\alpha)(d+i) = -1$, $L'(\alpha)(d+j) = 1$, $L'(\alpha)(d + |Q| + k) = 1$, and $L'(\alpha)(d + h) = 0$ for $h \notin \{i, j, |Q| + k\}$.

Our encoding involves a guess, namely we guess a set of edges explored by the path, let us denote this set by E'. We assume that the guessed E' induces a connected graph, that contains q_0 and $q_{|Q|}$.

We claim that there is a path from $(q_1, \mathbf{c_0})$ to $(q_{|Q|}, \mathbf{c_1})$ if and only if there is a permutation sum of elements of $L'(E')$ such that its effect $\mathbf{x}$ satisfies the following conditions:

1) projection of $\mathbf{x}$ on to the d first coordinates is equal $\mathbf{c_1} - \mathbf{c_0}$,
2) $weight(\mathbf{x})$ projected to the coordinates from $d+1$ to $d+|Q|$ equals $(1, 0, 0, \ldots, 0, 1)$,
3) $weight(\mathbf{x})$ is positive on all coordinates $d+|Q|+i$ for $e_i \in E'$ and 0 on all coordinates $d+|Q|+j$ for $e_j \notin E'$.

The first condition corresponds to the effect of the path, third condition guaranties that every edge in E' is visited along the path, and second is the encoding of Kirchhoff's condition guaranteeing existence of an Eulerian path (when combined with the connectivity due to the third condition).

To encode existence of a permutation sum satisfying the three conditions as the expressibility problem, we extend the set of data vectors $\{L'(e) : e \in E\}$ by data vectors which allow us to modify second and third condition.

First, we define $\mathbf{v}_{\beta,\mathbf{i}}$ as a data vector equal 0 for all data and coordinates except of a pair (β, i) for which $\mathbf{v}_{\beta,\mathbf{i}}(\beta)(i) = 1$.

Now we define V as $L'(E)$ extended in a following way; for every $0 < i \leq |Q|+|E|$ we add a data vector $\mathbf{v}_{\alpha,\mathbf{d+i}} - \mathbf{v}_{\beta,\mathbf{d+i}}$ for some $\beta \in \mathbb{D}\setminus\{\alpha\}$; and for every $0 < j \leq |E|$ we add a data vector $-\mathbf{v}_{\alpha,\mathbf{d+|Q|+j}}$. The first type of data vector allows us to freely manipulate with coordinates from $d+1$ to $d+|Q|+|E|$ without changing weight of a data vector. The second type of data vectors allows us to decrement weight on coordinates from $d+|Q|+1$ to $d+|Q|+|E|$.

Thus we may reformulate the claim with the three conditions as follows.

There is a path from $(q_1, \mathbf{c_0})$ to $(q_{|Q|}, \mathbf{c_1})$ if and only if there is a permutation sum of elements of V such that its effect $\mathbf{x}$ satisfies the following conditions

1) projection of $\mathbf{x}$ on to d first coordinates is equal $\mathbf{c_1} - \mathbf{c_0}$,

2) $\mathbf{x}(\alpha)$ projected to the coordinates from $d+1$ to $d+|Q|$ equals $(1,0,0,\ldots,0,1)$, and is equal to 0 for all data not equal α,
3) $\mathbf{x}(\alpha)$ is equal 1 on all coordinates $d+|Q|+i$ for $e_i \in E'$ and 0 on all coordinates $d+|Q|+j$ for $e_j \notin E'$ and is equal to 0 for all data not equal to α.

Now the three conditions defines precisely a data vector $\mathbf{x}$, so the existence of the path is equivalent to the expressibility of the data vector $\mathbf{x}$ in V.

Concluding the NP time algorithm is as follows:

- guess the set of edges E',
- check if the graph induced by E' is connected, and if it contains q_1 and $q_{|Q|}$,
- check in NP if for a chosen E' there is a solution of the expressibility problem for a set of data vectors V and data vector $\mathbf{x}$ ($\mathbf{x}$ depends on the choice of E'). ∎

VII. Conclusions

The main contribution of this paper is a technique to use *histograms* to finitely summarize permutations used in permutation sums. This allows solving the expressibility problem by deriving bounds on the number of different data values necessary to express a given vector.

For data vectors over $(\mathbb{Z}^d, +)$, this shows that the Expressibility is in *NP* (see Corollary 8). It follows that the unordered data extension does not make linear integer programming problems more difficult.

We think that our result will be central for analysing data extensions of many classical computational models like the UDPN or the blind counter automata presented respectively in Sections VI-A and VI-B.

Our results can be extended in several directions. For instance, one can ask similar questions for ordered data domains or for data matrices, i.e. functions from $\mathbb{D} \times \mathbb{D}$ to M. In the case of ordered data domain, it is not hard to prove that the complexity of the expressibility problem is at least *EXPSPACE*. As future work, the decidability of expressibility problems for ordered data vectors, and ordered/unordered data matrices seem to be difficult but interesting problems.

References

[1] R. Lazić, T. Newcomb, J. Ouaknine, A. Roscoe, and J. Worrell, "Nets with tokens which carry data," *Fund. Inform.*, vol. 88, no. 3, pp. 251–274, 2008.

[2] F. Rosa-Velardo, M. Martos-Salgado, and D. de Frutos-Escrig, "Accelerations for the coverability set of Petri nets with names," *Fund. Inform.*, vol. 113, no. 3–4, pp. 313–341, 2011.

[3] P. Hofman, S. Lasota, R. Lazić, J. Leroux, S. Schmitz, and P. Totzke, "Coverability Trees for Petri Nets with Unordered Data," in *FoSSaCS*, 2016.

[4] S. Greibach, "Remarks on blind and partially blind one-way multicounter machines," *Theoretical Computer Science*, vol. 7, no. 3, pp. 311 – 324, 1978.

[5] F. Rosa-Velardo and D. de Frutos-Escrig, "Decidability and complexity of Petri nets with unordered data," *Theor. Comput. Sci.*, vol. 412, no. 34, pp. 4439–4451, 2011.

[6] F. Rosa-Velardo, "Ordinal recursive complexity of unordered data nets," *Information and Computation*, pp. –, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0890540117300160

[7] C. Dufourd, A. Finkel, and P. Schnoebelen, "Reset nets between decidability and undecidability," in *ICALP*, 1998, pp. 103–115.

[8] T. Murata, "State equation, controllability, and maximal matchings of Petri nets," *IEEE Transactions on Automatic Control*, vol. 22, no. 3, pp. 412–416, Jun 1977.

[9] E. W. Mayr, "An algorithm for the general Petri net reachability problem," in *STOC*, 1981, pp. 238–246.

[10] S. R. Kosaraju, "Decidability of reachability in vector addition systems," in *STOC*, 1982, pp. 267–281.

[11] J. Leroux and S. Schmitz, "Demystifying reachability in vector addition systems," in *Annual IEEE Symposium on Logic in Computer Science*, 2015, pp. 56–67.

[12] A. Finkel and Ph. Schnoebelen, "Well-structured transition systems everywhere!" *Theor. Comput. Sci.*, vol. 256, no. 1–2, pp. 63–92, 2001.

[13] P. A. Abdulla, K. Čerāns, B. Jonsson, and Y.-K. Tsay, "Algorithmic analysis of programs with well quasi-ordered domains," *Inform. and Comput.*, vol. 160, no. 1/2, pp. 109–127, 2000.

[14] S. Schmitz and P. Schnoebelen, "The power of well-structured systems," in *CONCUR 2013*, 2013, pp. 5–24.

[15] P. A. Abdulla, G. Delzanno, and L. Van Begin, "A classification of the expressive power of well-structured transition systems," *Information and computation*, vol. 209, no. 3, pp. 248–279, 2011.

[16] J. Esparza, R. Ledesma-Garza, R. Majumdar, P. Meyer, and F. Niksic, "An smt-based approach to coverability analysis," in *International Conference on Computer Aided Verification*, 2014, pp. 603–619.

[17] M. Blondin, A. Finkel, C. Haase, and S. Haddad, "Approaching the coverability problem continuously," in *TACAS*, 2016, pp. 480–496.

[18] K. Athanasiou, P. Liu, and T. Wahl, *Unbounded-Thread Program Verification using Thread-State Equations*, 2016, pp. 516–531.

[19] K. Jensen, *Coloured Petri Nets - Basic Concepts, Analysis Methods and Practical Use - Volume 1, Second Edition*, 1996.

[20] S. Haddad and C. Girault, "Algebraic structure of flows of a regular coloured net," in *Advances in Petri Nets*, 1987, pp. 73–88.

[21] S. Evangelista, C. Pajault, and J.-F. Pradat-Peyre, "A simple positive flows computation algorithm for a large subclass of colored nets," in *FORTE*, 2007, pp. 177–195.

[22] W. Reisig, "Petri nets and algebraic specifications." *Theor. Comput. Sci.*, vol. 80, no. 1, pp. 1–34, 1991.

[23] K. Schmidt, "Verification of siphons and traps for algebraic Petri nets," in *Proceedings of the 18th International Conference on Application and Theory of Petri Nets*, 1997, pp. 427–446.

[24] M. Triebel and J. Sürmeli, "Homogeneous equations of algebraic Petri nets," in *CONCUR*, 2016, pp. 14:1–14:14.

[25] M. Bojanczyk, B. Klin, S. Lasota, and S. Torunczyk, "Turing machines with atoms," in *LICS*, 2013, pp. 183–192.

[26] M. Bojanczyk, B. Klin, and S. Lasota, "Automata theory in nominal sets," *Logical Methods in Computer Science*, vol. 10, no. 3, 2014.

[27] B. Klin, E. Kopczynski, J. Ochremiak, and S. Toruńczyk, "Locally finite constraint satisfaction problems," in *LICS*, 2015, pp. 475–486.

[28] M. Silva, E. Terue, and J. M. Colom, "Linear algebraic and linear programming techniques for the analysis of place/transition net systems," in *Lectures on Petri Nets I: Basic Models: Advances in Petri Nets*, 1998, pp. 309–373.

[29] J. Desel and J. Esparza, *Free Choice Petri Nets*, 1995.

[30] L. Recalde, S. Haddad, and M. Silva, "Continuous Petri nets: Expressive power and decidability issues," in *ATVA*, 2007, pp. 362–377.

[31] E. Fraca and S. Haddad, "Complexity analysis of continuous petri nets," *Fundam. Inform.*, vol. 137, no. 1, pp. 1–28, 2015. [Online]. Available: http://dx.doi.org/10.3233/FI-2015-1168

[32] C. Haase and S. Halfon, "Integer vector addition systems with states," in *RP 2014*, 2014, pp. 112–124.

[33] C. Başkocagİl and S. Kurtulan, "Generalized state equation for Petri nets," *WTOS*, vol. 10, no. 9, pp. 295–305, Sep. 2011.

[34] R. Diestel, *Graph Theory*, 2nd ed., 2000.

[35] A. Schrijver, *Theory of Linear and Integer Programming*, 1986.

[36] B. Aspvall and R. E. Stone, "Khachiyan's linear programming algorithm," *Journal of Algorithms*, vol. 1, no. 1, pp. 1 – 13, 1980.

[37] E. W. Mayr, "An algorithm for the general Petri net reachability problem," in *Symposium on Theory of Computing*, 1981, pp. 238–246.

[38] W. Reisig, *Petri Nets: An Introduction*, 1985.

[39] R. Durrett, H. Kesten, and G. Lawler, "Making money from fair games," in *Random walks, Brownian motion, and interacting particle systems*, 1991, pp. 255–267.

[40] P. Hofman, S. Lasota, R. Lazić, J. Leroux, S. Schmitz, and P. Totzke, "Coverability trees for Petri nets with unordered data," in *International*

Conference on Foundations of Software Science and Computational Structures, vol. 9634, 2016, pp. 445–461.

[41] R. M. Karp and R. E. Miller, “Parallel program schemata,” vol. 3, no. 2, pp. 147–195, 1969.

[42] M. Blondin, A. Finkel, S. Göller, C. Haase, and P. McKenzie, “Reachability in Two-Dimensional Vector Addition Systems with States Is PSPACE-Complete,” in *LICS*, 2015, pp. 32–43.

[43] A. Lechner, R. Mayr, J. Ouaknine, A. Pouly, and J. Worrell, “Model checking flat freeze LTL on one-counter automata,” in *27th International Conference on Concurrency Theory, CONCUR 2016, August 23-26, 2016, Québec City, Canada*, 2016, pp. 1–14.