Theory Comput Syst (2012) 50:675–693
DOI 10.1007/s00224-011-9367-y

Parameterizing by the Number of Numbers

Michael R. Fellows · Serge Gaspers · Frances A. Rosamond

Published online: 29 October 2011

Abstract The usefulness of parameterized algorithmics has often depended on what Niedermeier has called "the art of problem parameterization". In this paper we introduce and explore a novel but general form of parameterization: *the number of numbers*. Several classic numerical problems, such as SUBSET SUM, PARTITION, 3-PARTITION, NUMERICAL 3-DIMENSIONAL MATCHING, and NUMERICAL MATCHING WITH TARGET SUMS, have multisets of integers as input. We initiate the study of parameterizing these problems by the number of distinct integers in the input. We rely on an FPT result for INTEGER LINEAR PROGRAMMING FEASIBILITY to show that all the above-mentioned problems are fixed-parameter tractable when parameterized in this way. In various applied settings, problem inputs often consist in part of multisets of integers or multisets of weighted objects (such as edges in a graph, or jobs to be scheduled). Such number-of-numbers parameterized problems often reduce to subproblems about transition systems of various kinds, parameterized by the size of the system description. We consider several core problems of this kind relevant to number-of-numbers parameterization. Our main hardness result considers the problem: given a non-deterministic Mealy machine M (a finite state automaton

A preliminary version of this paper appeared in the proceedings of IPEC 2010 [7].
M.R.F. and F.A.R. acknowledge support from the Australian Research Council. S.G. acknowledges partial support from the European Research Council (COMPLEX REASON, 239962), from Conicyt Chile (Basal-CMM), and from the Australian Research Council.

M.R. Fellows · F.A. Rosamond
School of Engineering and IT, Charles Darwin University, Darwin, NT 0909, Australia

M.R. Fellows
e-mail: michael.fellows@cdu.edu.au

F.A. Rosamond
e-mail: frances.rosamond@cdu.edu.au

S. Gaspers (✉)
Institute of Information Systems, Vienna University of Technology, Vienna, Austria
e-mail: gaspers@kr.tuwien.ac.at

outputting a letter on each transition), an input word x, and a census requirement c for the output word specifying how many times each letter of the output alphabet should be written, decide whether there exists a computation of M reading x that outputs a word y that meets the requirement c. We show that this problem is hard for $W[1]$. If the question is whether there exists an input word x such that a computation of M on x outputs a word that meets c, the problem becomes fixed-parameter tractable.

Keywords Parameterized complexity · Problem parameterization · Variety of a multiset · Numerical problems

1 Introduction

Parameterized complexity and algorithmics has been developing for more than twenty years. Some important progress of the field has depended on what Niedermeier has called "the art of problem parameterization" (see Chap. 5 of his monograph [23]). For example, it was Cristina Bazgan who first suggested that the parameter might be $k = 1/\epsilon$ in the study of the complexity of approximation, leading eventually to the study of EPTASs [3].

Here we explore, for the first time (to our knowledge), a parameterization that seems widely relevant: *the number of numbers*. Many problems take as input information that consists (in part) of multisets of integers or multisets of weighted objects, such as weighted edges in a weighted graph, the time-requirements of jobs to be scheduled, or the sequence of molecular weights of a spectrographic dataset. Our investigations are of importance for problem input distributions where the number of distinct numerical values is small compared to the overall input size, and when the modeling of the problem allows rounding as a way to get to fewer distinct values.

In classical complexity, this "parameterization" has been explored in distribution-sensitive algorithmics [29]. For example, while $\Omega(n \log n)$ is a lower bound on sorting n values in the comparison model [18], a multiset of cardinality n and h distinct values can be sorted using $O(n \log h)$ comparisons [22].

It is perhaps surprising that this parameterization in the sense of Niedermeier's "art of problem parameterization" [23, 24] has not been considered before in parameterized complexity, as it seems entirely well-motivated. While weighted combinatorial optimization problems have generally strong claims to model realism, it is often the case that, e.g., the jobs to be scheduled may be of certain standard sizes arising in a limited number of ways, or that the costs of the nodes in a network problem may depend on the model and vendor of the device, of which there are a limited number of possibilities. Many similar scenarios easily come to mind. A bounded number of numbers may also arise naturally and implicitly in parameterized problems when numbers are associated to other parameterized aspects of a problem, such as alphabet size.

As an initial foray, we first show that a number of classic NP-hard problems about multisets of integers, when parameterized in this way, become fixed-parameter tractable. The proofs are easy, and the knowledgeable reader might anticipate them almost

as exercises today—they use the relatively deep result that INTEGER LINEAR PROGRAMMING, parameterized by the number of variables, is FPT. Until recently, as noted in the 2006 monograph by Niedermeier [23], there were not so many interesting applications of this fundamental result (see [1, 11, 12, 16] for some exceptions).

At a deeper level of engagement with this parameterization, we describe some examples of how number-of-numbers parameterized problems reduce to numerical problems about Mealy machines, parameterized by the size of the description of the machine. We show that one basic problem about Mealy machines, parameterized in this way, is FPT, and that another is $W[1]$-hard.

2 Preliminaries

Integer Linear Programming In the INTEGER LINEAR PROGRAMMING FEASIBILITY problem (ILPF), the input is an $m \times n$ matrix $\mathbf{A}$ of integers and an m-vector $\mathbf{b}$ of integers, the parameter is n, and the question is whether there exists an n-vector $\mathbf{x}$ of integers satisfying the m inequalities $\mathbf{Ax} \leq \mathbf{b}$. ILPF, parameterized by the number of variables, was shown to be fixed-parameter tractable by Lenstra [20] and the running time has been improved by Kannan [17] and by Frank and Tardos [14].

Multisets Let A be a multiset. The *cardinality* of A, denoted $|A|$, is the total number of elements in A, including repeated memberships. The *variety* of A, denoted $\|A\|$, is the number of distinct elements in A. Element a has *multiplicity* m in A if it occurs m times in A. We denote the set of integers from 1 to n by $[n] = \{1, \ldots, n\}$.

Graphs Let $G = (V, E)$ be a graph, $v \in V$ be a vertex of G, and $S \subseteq V$ be a subset of vertices of G. The subgraph of G induced on S is the graph $G[S] = (S, E \cap \{uv : u, v \in S\})$. The set S is a *clique* of G if $G[S]$ is *complete*, i.e. there is an edge between every two distinct vertices of $G[S]$. The set S is an *independent set* of G if $G[S]$ is *empty*, i.e. $G[S]$ has no edge. The *neighborhood* of v is the set of vertices incident to v and denoted $N(v)$. The *degree* of v is $d(v) = |N(v)|$. We also define $N_S(v) = N(v) \cap S$ and $d_S(v) = |N_S(v)|$.

Words Let Σ be an *alphabet*. The elements of Σ are called *letters*, and a *word* x of length $n = |x|$ is a sequence of n letters. The symbol ϵ denotes the empty letter. We denote the concatenation of two words $x_1, x_2 \in \Sigma^*$ by $x_1 x_2$. The ith *power* of a word x is denoted x^i or $(x)^i$ and represents the word $\underbrace{xx\ldots x}_{i \text{ times}}$.

Parameterized Complexity We define the basic notions of Parameterized Complexity and refer to other sources [6, 13, 23] for an in-depth treatment. A *parameterized problem* is a set of pairs (I, k), the instances, where I is the main part and k is the parameter. A parameterized problem is *fixed-parameter tractable* if there exist a computable function f and an algorithm that solves any instance (I, k) of size n in time $f(k)n^{O(1)}$. FPT denotes the class of all fixed-parameter tractable parameterized decision problems.

Parameterized complexity offers a completeness theory that allows the accumulation of strong theoretical evidence that some parameterized problems are not fixed-parameter tractable. This theory is based on a hierarchy of complexity classes

$$\text{FPT} \subseteq \text{W}[1] \subseteq \text{W}[2] \subseteq \text{W}[3] \subseteq \cdots \subseteq \text{XP},$$

where all inclusions are believed to be strict. Each class W[i] contains all parameterized decision problems that can be reduced to a canonical parameterized satisfiability problem P_i under *parameterized reductions*. These are many-to-one reductions where the parameter for one problem maps into the parameter for the other. More specifically, a parameterized problem L reduces to a parameterized problem L' if there is a mapping R from instances of L to instances of L' such that

1. (I,k) is a YES-instance of L if and only if $(I',k') = R(I,k)$ is a YES-instance of L',
2. there is a computable function g such that $k' \leq g(k)$, and
3. there is a computable function f such that R can be computed in time $f(k) \cdot n^{O(1)}$, where n denotes the size of (I,k).

A parameterized problem L is then in W[i], for $i \in \mathbb{N}$, if it has a parameterized reduction to the problem of deciding whether a Boolean decision circuit (a decision circuit is a circuit with exactly one output), with AND, OR, and NOT gates, of constant depth such that on each path from an input to the output, all but i gates have a constant number of inputs, parameterized by the number of ones in a satisfying assignment to the inputs of the circuit [6].

A parameterized problem is in XP if there exist computable functions f and g and an algorithm that solves any instance (I,k) of size n in time $f(k)n^{g(k)}$.

3 Subset Sum and Partition

We start with two classic problems on multisets and show that they are fixed-parameter tractable, parameterized by the number of numbers.

variety-SUBSET SUM (*var*-SUBSUM)

Input:	A multiset A of integers and an integer s.
Parameter:	$k = \|A\|$, the number of distinct integers in A.
Question:	Is there a multiset $X \subseteq A$ such that $\sum_{a \in X} a = s$?

variety-PARTITION (*var*-PART)

Input:	A multiset A of integers.
Parameter:	$k = \|A\|$.
Question:	Is there a multiset $X \subseteq A$ such that $\sum_{a \in X} a = \sum_{b \in A \setminus X} b$?

The parameterizon of SUBSET SUM by $|X|$ is $W[1]$-hard [9]. This hardness also holds for the parameterization of PARTITION by $|X|$ as an easy reduction from SUBSET SUM adds the integer $(\sum_{a \in A} a) - 2s$ to A if $s \leq (\sum_{a \in A} a)/2$, and if

$s > (\sum_{a\in A} a)/2$, the reduction looks instead for the complement set $A \setminus X$ that sums to $(\sum_{a\in A} a) - s$ and uses the previous construction.

Our FPT results use a deep result of Lenstra, stating that INTEGER LINEAR PROGRAMMING FEASIBILITY (ILPF), parameterized by the number of variables, is FPT. They are obtained by very natural formulations of the respective problems as integer programs.

Theorem 1 *var*-SUBSUM *is fixed-parameter tractable.*

Proof Given an instance (A, s) for *var*-SUBSUM, with $\|A\| = k$, we create an equivalent instance of ILPF whose number of variables is upper bounded by a function of k. Let $a_1, \ldots, a_k$ denote the distinct elements of A and let $m_1, \ldots, m_k$ denote their respective multiplicities in A. The ILPF instance has the integer variables $x_1, \ldots, x_k$ and the following inequalities and equalities.

$$\begin{aligned} &x_i \leq m_i, \quad \forall i \in [k], \\ &x_i \geq 0, \quad \forall i \in [k], \\ &\sum_{i=1}^{k} x_i \cdot a_i = s. \end{aligned}$$

For each $i \in [k]$, the variable x_i represents the number of times a_i occurs in X, the set summing to s in a valid solution. Using standard techniques in mathematical programming, these constraints can be transformed into the form $\mathbf{Ax} \leq \mathbf{b}$. □

A very similar proof shows that *var*-PART is fixed-parameter tractable.

Theorem 2 *var*-PART *is fixed-parameter tractable.*

Proof Given an instance A for *var*-PART, with $\|A\| = k$, we create an equivalent instance of ILPF whose number n of variables is upper bounded by a function of k.

Let $a_1, \ldots, a_k$ denote the distinct elements of A and let $m_1, \ldots, m_k$ denote their respective multiplicities in A. The ILPF instance has the integer variables $x_1, \ldots, x_k$ and the following inequalities and equalities.

$$\begin{aligned} &x_i \leq m_i, \quad \forall i \in [k], \\ &x_i \geq 0, \quad \forall i \in [k], \\ &\sum_{i=1}^{k} x_i \cdot a_i = \sum_{a\in A} a/2. \end{aligned}$$

For each $i \in [k]$, the variable x_i represents the number of times a_i occurs in X, such that $\sum_{a\in X} a = \sum_{b\in A\setminus X} b = \sum_{a\in A} a/2$ in a valid solution.

Using standard techniques in mathematical programming, these constraints can be transformed such that they respect the form $\mathbf{Ax} \leq \mathbf{b}$. □

4 Other Classic Numerical Problems

Using the ILPF machinery, we show in this section that several other problems, which are often used in NP-hardness proofs, become fixed-parameter tractable when parameterized by the number of numbers.

variety-NUMERICAL 3-DIMENSIONAL MATCHING (*var*-NUM3-DM)

Input:	Three multisets A, B, C of n integers each and an integer s.
Parameter:	$k = \|A \cup B \cup C\|$.
Question:	Are there n triples $S_1, \ldots, S_n$, each containing one element from each of A, B, and C such that for every $i \in [n]$, $\sum_{a \in S_i} a = s$?

Theorem 3 *var*-NUM3-DM *is fixed-parameter tractable.*

Proof Let (A, B, C, s) be an instance for *var*-NUM3-DM, with $k_1 = \|A\|$, $k_2 = \|B\|$, $k_3 = \|C\|$, and $k = \|A \cup B \cup C\|$. Let $a_1, \ldots, a_{k_1}$ denote the distinct elements of A, $b_1, \ldots, b_{k_2}$ denote the distinct elements of B, and $c_1, \ldots, c_{k_3}$ denote the distinct elements of C. Also, let $m_{1,a}, \ldots, m_{k_1,a}, m_{1,b}, \ldots, m_{k_2,b}, m_{1,c}, \ldots, m_{k_3,c}$ denote their respective multiplicities in A, B, and C. We create an instance of ILPF with at most k^3 integer variables $x_{i,j,\ell}$, for $i \in [k_1]$, $j \in [k_2]$, $\ell \in [k_3]$:

$$x_{i,j,\ell} = 0 \quad \text{for each } (i, j, \ell) \in [k_1] \times [k_2] \times [k_3] \text{ such that } a_i + b_j + c_\ell \neq s,$$

$$\sum_{(j,\ell) \in ([k_2],[k_3])} x_{i,j,\ell} = m_{i,a}, \quad \forall i \in [k_1],$$

$$\sum_{(i,\ell) \in ([k_1],[k_3])} x_{i,j,\ell} = m_{j,b}, \quad \forall j \in [k_2],$$

$$\sum_{(i,j) \in ([k_1],[k_2])} x_{i,j,\ell} = m_{\ell,c}, \quad \forall \ell \in [k_3].$$

A variable $x_{i,j,\ell}$ represents the number of times the elements $a_i \in A$, $b_j \in B$ and $c_\ell \in C$ are used together to form a triple summing to s. The first constraint makes sure that such a triple is formed only if it sums to s. The remaining equalities make sure that each element of $A \cup B \cup C$ appears in a triple. Thus n such triples are formed, all summing to s if the integer program is feasible. □

Note that the problem is also fixed-parameter tractable if parameterized by $\|A \cup B\|$ only: we face a NO-instance if $\|C\| > \|\{a + b : a \in A, b \in B\}\|$. A closely related, well known numerical problem, is the following.

variety-NUMERICAL MATCHING WITH TARGET SUMS (*var*-NMTS)

Input:	Three multisets A, B, S of n integers each.
Parameter:	$k = \|\|A \cup B \cup S\|\|$.
Question:	Are there n triples $C_1, \ldots, C_n \in A \times B \times S$, such that the A-element and the B-element from each C_i sum to its S-element?

Corollary 1 *var*-NMTS *is fixed-parameter tractable.*

By the previous discussion, the natural parameterization by $\|A \cup B\|$ is also fixed-parameter tractable. A straightforward adaptation of the proof of Theorem 3 shows that *variety*-3-PARTITION is fixed-parameter tractable.

variety-3-PARTITION (*var*-3-PART)

Input:	A multiset A of $3n$ integers.
Parameter:	$k = \|\|A\|\|$.
Question:	Are there n triples $S_1, \ldots, S_n \subseteq A$, all summing to the same number?

Theorem 4 *var*-3-PART *is fixed-parameter tractable.*

Proof Let A be an instance for *var*-3-PART, with $\|A\| = k$ and $|A| = 3n$. Let $s = \sum_{a \in A} a/n$. Let $a_1, \ldots, a_k$ denote the distinct elements of A and let $m_1, \ldots, m_k$ denote their multiplicities in A. We create an instance of ILPF with at most k^3 integer variables $x_{i,j,\ell}$, for $i, j, \ell \in [k]$:

$$x_{i,j,\ell} = 0 \quad \text{for each } i, j, \ell \in [k] \text{ such that } a_i + a_j + a_\ell \neq s,$$

$$\sum_{\substack{j,\ell \in [k] \\ j,\ell \neq i}} (x_{i,j,\ell} + x_{j,i,\ell} + x_{j,\ell,i}) + 2 \cdot \sum_{\substack{j \in [k] \\ j \neq i}} (x_{i,i,j} + x_{i,j,i} + x_{j,i,i}) + 3 \cdot x_{i,i,i} = m_i, \quad \forall i \in [k].$$

A variable $x_{i,j,\ell}$ represents the number of times the elements a_i, a_j and a_ℓ are used together to form a triple summing to s. The first constraint makes sure that such a triple is formed only if it sums to s. The second set of equalities make sure that each element of A appears in a triple. Thus n such triples are formed, all summing to s if the integer program is feasible. □

5 Mealy Machines

In this section, we explore how far we can generalize the rather simple FPT results of the previous two sections. To this end, we investigate the parameterized complexity of two problems about Mealy Machines. Both problems can be viewed as parameterized problems implicitly parameterized by the number of numbers, because in each case the size of the alphabet is part of the parameterization, and each letter of the alphabet is associated with a census requirement. The richer structure of these problems means that a simple appeal to integer linear programming no longer suffices: one turns out to be FPT, and the other W[1]-hard. In Sect. 6, we show that other problems parameterized by the number of numbers reduce to these two seemingly general problems of this kind.

Mealy machines [21] are finite-state transducers, generating an output based on their current state and input. They have important applications in cryptanalysis [2], computational linguistics [27], and control and system theory [30]. A *deterministic Mealy machine* is a dual-alphabet state transition system given by a 5-tuple $M = (S, s_0, \Gamma, \Sigma, T)$:

- a finite set of states S,
- a start state $s_0 \in S$,
- a finite set Γ, called the input alphabet,
- a finite set Σ, called the output alphabet, and
- a transition function $T : S \times \Gamma \to S \times \Sigma$ mapping pairs of a state and an input letter to the corresponding next state and output letter.

The alphabets Γ and Σ may contain the empty letter ϵ, as in [28]. This eases some of the description, but all our results also hold if we restrict $\epsilon \notin \Gamma \cup \Sigma$.

In a *non-deterministic Mealy machine*, the only difference is that the transition function is defined $T : S \times \Gamma \to \mathcal{P}(S \times \Sigma)$ as for a given state and input letter, there may be more than one possibility for the next state and output letter. (Here $\mathcal{P}(X)$ denotes the powerset of a set X.)

A *census requirement* $c : \Sigma \setminus \{\epsilon\} \to \mathbb{N}$ is a function assigning a non-negative integer to each letter of the output alphabet (except ϵ). It is used to constrain how many times each letter should appear in the output of a Mealy machine. A word $x \in \Sigma^*$ *meets* the census requirement if every letter $b \in \Sigma \setminus \{\epsilon\}$ appears exactly $c(b)$ times in x.

The notion of census requirement is related to Parikh images [25]. Let $\Sigma \setminus \{\epsilon\} = \{b_1, \ldots, b_\sigma\}$. For $x \in \Sigma^*$, the *Parikh image* is $\Psi(x) = (c(b_1), \ldots, c(b_\sigma))$, where c is the census requirement such that x meets c. The *Parikh image of a language* L is $\Psi(L) = \{\Psi(x) : x \in L\}$. Parikh's theorem [25] states that the Parikh image of a context-free language is semilinear, i.e., that for every context-free language there is a regular language with the same Parikh image.

Our first problem about Mealy machines asks whether there exists an input word and a computation of the Mealy machine such that the output word meets the census requirement.

variety-EXISTS WORD MEALY MACHINE (*var*-EWMM)

Input:	A non-deterministic Mealy machine $M = (S, s_0, \Gamma, \Sigma, T)$, and a census requirement $c : \Sigma \setminus \{\epsilon\} \to \mathbb{N}$.
Parameter:	$\lvert S\rvert + \lvert\Gamma\rvert + \lvert\Sigma\rvert$.
Question:	Does there exist a word $x \in \Gamma^*$ for which a computation of M on input x generates an output y that meets c?

Our proof that *var*-EWMM is fixed-parameter tractable is inspired by the proof from [10] showing that BANDWIDTH is fixed-parameter tractable when parameterized by the maximum number of leaves in a spanning tree of the input graph. We need the following definition and lemma from [10].

In a digraph D, two directed walks Δ and Δ' from a vertex s to a vertex t are *arc-equivalent*, if for every arc a of D, Δ and Δ' pass through a the same number of times.

Lemma 1 [10] *Any directed walk Δ through a finite digraph D on n vertices from a vertex s to a vertex t of D is arc-equivalent to a directed walk Δ' from s to t, where Δ' has the form*:

(1) *Δ' consists of an underlying directed walk ρ from s to t of length at most n^2,*
(2) *together with some number of* short loops, *where each such short loop l begins and ends at a vertex of ρ, and has length at most n.*

The algorithm will first subdivide state transitions in order to make the underlying directed graph simple. As suggested by Lemma 1, the algorithm goes over all possible choices for selecting an underlying directed walk ρ starting from s_0. For every short loop starting and ending at a vertex from ρ, the algorithm associates an integer variable representing the number of times this short loop is executed while moving along ρ. Again by INTEGER LINEAR PROGRAMMING FEASIBILITY, it can be checked whether there is a set of integers, representing the number of executions of the short loops, such that the number of times each output letter is written is compatible with the census requirement.

Theorem 5 *var*-EWMM *is fixed-parameter tractable.*

Proof Let $(M' = (S', s'_0, \Gamma', \Sigma', T'), c)$ be an instance for *var*-EWMM with $k = \lvert S'\rvert + \lvert\Gamma'\rvert + \lvert\Sigma'\rvert$. As M' might have multiple transitions from one state to another, we first subdivide each transition in order to obtain a simple digraph underlying the Mealy machine (so we can use Lemma 1): create a new non-deterministic Mealy machine $M = (S, s_0, \Gamma, \Sigma, T)$ such that, initially, $S = S'$, $s_0 = s'_0$, $\Gamma = \Gamma' \cup \{\epsilon\}$, and $\Sigma = \Sigma' \cup \{\epsilon\}$; for each transition t of T' from a couple $(s_i, \langle i\rangle)$ to a couple $(s_o, \langle o\rangle)$, add a new state s_t to S and add the transition from $(s_i, \langle i\rangle)$ to $(s_t, \langle o\rangle)$ and the transition from (s_t, ϵ) to (s_o, ϵ) to T. Clearly, there is at most one transition between every two states in M.

Our algorithm goes over all transition walks in M of length at most $\lvert S\rvert^2$ that start from s_0. There are at most $\lvert S\rvert^{(\lvert S\rvert^2)}$ such transition walks and each such transition

walk has at most $|S|^{|S|}$ short loops, as they have length at most $|S|$ by Lemma 1. Let $P = (s_0, s_1, \ldots, s_{|P|})$ be such a transition walk and $L = (\ell_1, \ell_2, \ldots, \ell_{|L|})$ be its short loops. It remains to check whether there exists a set of integers $X = \{x_1, x_2, \ldots, x_{|L|}\}$ such that a word output by a computation of M moving from s_0 to $s_{|P|}$ along the walk P, and executing x_i times each short loop ℓ_i, $1 \leq i \leq |L|$, meets the census requirement. Note that if one such word meets the census requirement, then all such words meet the census requirement, as it does not matter in which order the short loops are executed. We verify whether such a set X exists by ILPF.

Let $\Sigma \setminus \{\epsilon\} = \{\langle \ell, 1\rangle, \langle \ell, 2\rangle, \ldots, \langle \ell, \sigma\rangle\}$. Define $m(i, j)$, for $1 \leq i \leq |L|$, $1 \leq j \leq \sigma$, to denote the number of times that M writes the letter $\langle \ell, j\rangle$ when it executes the loop ℓ_i once. Define $m(j)$, for $1 \leq j \leq \sigma$, to be the number of times that M writes the letter $\langle \ell, j\rangle$ when it transitions from s_0 to $s_{|P|}$ along the walk P. Then, we only need to verify that there exist integers $x_1, x_2, \ldots, x_{|L|}$ such that

$$m(j) + \sum_{i=0}^{|L|} x_i \cdot m(i, j) = c(\langle \ell, j\rangle), \quad \forall j \in [\sigma].$$

By construction, $|S| \leq |S'| + |T'| \leq |S'| + |S'|^2 \cdot |\Gamma'| \cdot |\Sigma'| \leq k + k^4$. As the number of integer variables of this program is at most $|L| \leq |S|^{|S|} \leq (k + k^4)^{k+k^4}$, and the number of transition walks that the algorithm considers is at most $|S|^{(|S|^2)} \leq (k + k^4)^{k^2+2k^5+k^8}$, *var*-EWMM is fixed-parameter tractable. □

We note that the proof in [10] concerned a special case of a deterministic Mealy machine where the input and output alphabet are the same, and all transitions that read a letter $\langle \ell \rangle$ also write $\langle \ell \rangle$.

In our second Mealy machine problem, the question is whether, for a given input word, there is a computation of the Mealy machine which outputs a word that meets the census requirement.

variety-GIVEN WORD MEALY MACHINE (*var*-GWMM)

Input:	A non-deterministic Mealy machine $M = (S, s_0, \Gamma, \Sigma, T)$, a word $x \in \Gamma^*$, and a census requirement $c : \Sigma \setminus \{\epsilon\} \to \mathbb{N}$.
Parameter:	$\lvert S\rvert + \lvert\Gamma\rvert + \lvert\Sigma\rvert$.
Question:	Is there a computation of M on input x generating an output y that meets c?

By dynamic programming we show that two restrictions of this problem are in XP. In the first one, the census requirement is encoded in unary. This restriction of the problem seems lenient, especially when one is actually interested in finding the output word, as the census function acts then as a placeholder for the produced word.

Theorem 6 *var*-GWMM *is in XP if c is encoded in unary.*

Proof Let $|\Sigma \setminus \{\epsilon\}| = \sigma$ and $\Sigma \setminus \{\epsilon\} = \{b_1, \ldots, b_\sigma\}$. Our dynamic programming algorithm computes the entries of a boolean table A. The table A has an entry $A[s, c_1, \ldots, c_\sigma, i, p]$ for each state $s \in S$, each $c_j \in \{0, \ldots, c(b_j)\}$, $j \in [\sigma]$,

each index $i \in \{0, \ldots, |x|\}$, and each integer $p \in P = \{0, \ldots, |S| - 1\}$. The entry $A[s, c_1, \ldots, c_\sigma, i, p]$ is set to `true` if there exists a computation of M reading the first i letters of x, outputting a word y in which the letter b_j occurs c_j times, for each $j \in [\sigma]$, followed by p transitions that read ϵ and write ϵ, and ending up in state s, and to `false` otherwise.

Set $A[s, c_1, \ldots, c_\sigma, 0, 0]$ to `true` if $s = s_0$ and $c_1 = \cdots = c_\sigma = 0$, and to `false` otherwise. Compute the values of the table by increasing values of $\sum_{i=1}^{\sigma} c_i$, index i, propagation integer p, and state number s:

$$
\begin{aligned}
&A[s, c_1, \ldots, c_\sigma, i, p] \\
&\quad = \bigvee_{\substack{s' \in S, b_j \in \Sigma \setminus \{\epsilon\}, p' \in P: \\ (s, b_j) \in T(s', x[i])}} A[s', c_1, \ldots, c_{j-1}, c_j - 1, c_{j+1}, \ldots, c_\sigma, i - 1, p'] \\
&\qquad \vee \bigvee_{\substack{s' \in S, p' \in P: \\ (s, \epsilon) \in T(s', x[i])}} A[s', c_1, \ldots, c_\sigma, i - 1, p'] \\
&\qquad \vee \bigvee_{\substack{s' \in S, b_j \in \Sigma \setminus \{\epsilon\}, p' \in P: \\ (s, b_j) \in T(s', \epsilon)}} A[s', c_1, \ldots, c_{j-1}, c_j - 1, c_{j+1}, \ldots, c_\sigma, i, p'] \\
&\qquad \vee \bigvee_{s' \in S: (s, \epsilon) \in T(s', \epsilon)} A[s', c_1, \ldots, c_\sigma, i, p - 1].
\end{aligned}
$$

Finally, there exists an x-computation of M generating a word y that meets the census requirement if and only if $\bigvee_{s \in S, p \in P} A[s, c(b_1), \ldots, c(b_\sigma), |x|, p]$ is `true`. Denote by n is the length of the description of an input instance. The table has $|S| \cdot |x| \cdot |P| \cdot \Pi_{j=1}^{\sigma} c(b_j) \leq |S|^2 \cdot n^{\sigma+1}$ entries, and each entry can be computed in time $O(|S|^2 \cdot \sigma)$. The running time of the algorithm is thus upper bounded by $O(n^{\sigma+1} \cdot k^5)$, where k is the parameter. □

For the version where c is encoded in binary, a restriction on the input alphabet gives an XP algorithm as well.

Corollary 2 *var*-GWMM *is in XP if* $\epsilon \notin \Gamma$.

Proof If $\sum_{b \in \Sigma \setminus \{\epsilon\}} c(b) > |x|$, then return `false`, as M cannot output more than $|x|$ letters. Otherwise, run the algorithm described in the proof of Theorem 6. Its running time is $O(k^5 \cdot |x| \cdot \Pi_{j=1}^{\sigma} c(b_j)) = O(n^{\sigma+1} \cdot k^5)$. □

Note that the XP-results also hold if the parameter is only $|\Sigma|$.

To show that *var*-GWMM is $W[1]$-hard, we reduce from the MULTICOLORED CLIQUE problem, which is $W[1]$-hard [8, 26].

MULTICOLORED CLIQUE (MCC)	
Input:	An integer k and a connected undirected graph $G = (V(1) \cup V(2) \cup \cdots \cup V(k), E)$ such that for every $i \in [k]$, the vertices of $V(i)$ induce an independent set in G.
Parameter:	k.
Question:	Is there a clique of size k in G?

Clearly, a solution to this problem has one vertex from each color.

Our parameterized reduction encodes G in the input word x of the Mealy machine M, and the description of M depends only on k. The Mealy machine is divided into k parts, one for each color class $V(i)$, with $1 \leq i \leq k$. Its ith part is responsible for selecting a vertex v_i from $V(i)$ and edges $v_i v_j$ for every $v_j \in V(j)$, with $1 \leq j \neq i \leq k$. All consistency issues and communication is done via the census requirement. Within part i, we need to make sure that the selected edges are all incident to the selected vertex v_i. This is achieved by making M output p times each letter $\langle \mathbf{l}, i, j \rangle$, with $1 \leq j \neq i \leq k$, if it selects the pth vertex in $V(i)$. The census requirement for $\langle \mathbf{l}, i, j \rangle$ is $|V(i)| + 1$, meaning that $\langle \mathbf{l}, i, j \rangle$ needs to be output $|V(i)| + 1 - p$ times later. To select an edge $v_i v_j$, the machine M will be constrained to select this edge among the edges incident to v_i. To achieve this, edges from $V(i)$ to $V(j)$ appear in x grouped by the vertex from $V(i)$ on which they are incident. After each group of edges incident on one vertex from $V(i)$, there is a special state where $\langle \mathbf{l}, i, j \rangle$ is output if and only if the edge towards $V(j)$ has already been selected. As $\langle \mathbf{l}, i, j \rangle$ needs to be output exactly $|V(i)| + 1 - p$ times, we force in this way that an edge is selected which is incident on v_i. This enforces that all edges selected in the ith part are incident on the same vertex. It remains to make sure that distinct parts i and j select the same edge between $V(i)$ and $V(j)$. This is again achieved by a census requirement where a part of the census of letter $\langle \mathbf{l}, \bar{\mathbf{e}}, i, j \rangle$ is output in the ith part and the remaining part in the jth part of M.

Theorem 7 *var*-GWMM *is* $W[1]$*-hard.*

Proof Let $(k, G = (V(1) \cup V(2) \cup \cdots \cup V(k), E))$ be an instance of MCC. Suppose $V(i) = \{v_{i,1}, v_{i,2}, \ldots, v_{i,|V(i)|}\}$ is the vertex set of color i, for each color class $i \in [k]$, $E = \{e_1, e_2, \ldots, e_{|E|}\}$, and $E(i, j) = \{e(i, j, 1), e(i, j, 2), \ldots, e(i, j, |E(i, j)|)\}$ is the subset of edges with one vertex in color class i and the other in color class j, for $i, j \in [k]$. Moreover, suppose $E(i, j)$ follows the same order as E, that is if $e_p = e(i, j, p')$, $e_q = e(i, j, q')$, and $p \leq q$, then $p' \leq q'$. For a vertex $v_{i,p}$ and two integers $j \in [k] \setminus \{i\}$ and $q \in [d_{V(j)}(v_{i,p}) + 1]$, we define $\mathsf{gap}(v_{i,p}, j, q) = t - s$, where $e(i, j, t)$ is the qth edge in $E(i, j)$ incident to $v_{i,p}$ (respectively, $t = |E(i, j)|$ if $q = d_{V(j)}(v_{i,p}) + 1$) and $e(i, j, s)$ is the $(q - 1)$th edge in $E(i, j)$ incident to $v_{i,p}$ (respectively, $s = 0$ if $q = 1$).

We construct an instance $(M = (S, s_0, \Gamma, \Sigma, T), x, c)$ for *var*-GWMM as follows. M's input alphabet, Γ, is $\{\langle i \rangle, \langle i, j \rangle, \langle \bar{\mathbf{e}}, i, j \rangle, \langle \mathbf{e}, i, j \rangle : i, j \in [k], i \neq j\}$. M's output alphabet, Σ, is $\{\epsilon\} \cup \{\langle \mathbf{l}, i, j \rangle, \langle \mathbf{l}, \bar{\mathbf{e}}, i, j \rangle : i, j \in [k], i \neq j\}$. The word x is defined

$$x := x_1 x_2 \ldots x_k,$$

$$
\begin{aligned}
x_i &:= x_{i,0}x_{i,1}\dots x_{i,i-1}x_{i,i+1}x_{i,i+2}\dots x_{i,k}\langle i\rangle, \quad \forall i \in [k],\\
x_{i,0} &:= (\langle i,1\rangle\langle i,2\rangle\dots\langle i,i-1\rangle\langle i,i+1\rangle\langle i,i+2\rangle\dots\langle i,k\rangle)^{|V(i)|}, \quad \forall i \in [k],\\
x_{i,j} &:= \langle i,j\rangle x_{i,j,1}\langle i,j\rangle x_{i,j,2}\dots\langle i,j\rangle x_{i,j,|V(i)|}\langle i,j\rangle, \quad \forall i,j \in [k], i \neq j,\\
x_{i,j,p} &:= \langle \bar{\mathbf{e}},i,j\rangle^{\mathsf{gap}(v_{i,p},j,1)}\langle \mathbf{e},i,j\rangle\langle \bar{\mathbf{e}},i,j\rangle^{\mathsf{gap}(v_{i,p},j,2)}\langle \mathbf{e},i,j\rangle,\\
&\quad \dots\langle \bar{\mathbf{e}},i,j\rangle^{\mathsf{gap}(v_{i,p},j,d_{V(j)}(v_{i,p}))}\langle \mathbf{e},i,j\rangle\langle \bar{\mathbf{e}},i,j\rangle^{\mathsf{gap}(v_{i,p},j,d_{V(j)}(v_{i,p})+1)}.
\end{aligned}
$$

The census requirement c is, for every $i,j \in [k], i \neq j$,

$$c(\langle \mathbf{l},i,j\rangle) := |V(i)| + 1,$$

$$c(\langle \mathbf{l},\bar{\mathbf{e}},i,j\rangle) := |E(i,j)|.$$

On reading a subword x_i, the Mealy machine will select a vertex $v_{i,p}$ in $V(i)$ and one edge incident to $v_{i,p}$ for each color class $j \in [k] \setminus \{i\}$. The vertex $v_{i,p}$ is selected in the subword $x_{i,0}$ of x_i. Next, for each $j \in [k] \setminus \{i\}$, a vertex in $V(i)$ and a vertex in $V(j)$ are selected in the subword $x_{i,j}$. The census requirement for $\langle \mathbf{l},i,j\rangle$ makes sure that the vertex from $V(i)$ is $v_{i,p}$. The subword $x_{i,j,p}$ ensures that $v_{i,p}$ and the vertex that is selected from $V(j)$ are joined by an edge. Finally, the census requirement for $\langle \mathbf{l},\bar{\mathbf{e}},i,j\rangle$ is responsible for the inter-partition communication and makes sure that the edge selected in $x_{i,j}$ is equal to the edge selected in $x_{j,i}$.

The Mealy machine M consists of k parts. The ith part of M is depicted in Fig. 1. Its initial state is $s_{v,1}$. There is a transition from the last state of each part, $s^{(4)}_{e,i,k}$, to the first state of the following part, $s_{v,i+1}$ (from the kth part, there is a transition to a final state): it reads the letter $\langle i\rangle$ and writes the letter ϵ. We set $\langle \mathbf{l}',\bar{\mathbf{e}},i,j\rangle = \langle \mathbf{l},\bar{\mathbf{e}},j,i\rangle$ for all $i \neq j \in [k]$. Note that, in the description of M, the letter $\langle \mathbf{l},i,j\rangle$ can only be output on reading $\langle i,j\rangle$, and $\langle \mathbf{l},\bar{\mathbf{e}},i,j\rangle$ can only be output on reading $\langle \bar{\mathbf{e}},i,j\rangle$ or $\langle \bar{\mathbf{e}},j,i\rangle$.

Let us first verify that the parameter for *var*-GWMM is a function of k, and that there exists a function f such that the size of the instance for *var*-GWMM is $f(k) \cdot n^{O(1)}$, where n is the number of vertices of G. We have $|\Gamma| = k + 3 \cdot k \cdot (k-1)$, $|\Sigma| = 1 + 2 \cdot k \cdot (k-1)$, and $|S| = 1 + k \cdot (2 + 4 \cdot (k-1))$. The parameter of *var*-GWMM is thus bounded by a function of k. The length of x is $O(k^2 \cdot n^3)$. Now, we show that (M,x,c) is a YES-instance for *var*-GWMM if and only if (G,k) is a YES-instance for MCC.

First, suppose $(M = (S,s_0,\Gamma,\Sigma,T),x,c)$ is a YES-instance for *var*-GWMM.

We say that M *selects* a vertex $v_{i,p}$ if it makes a transition from state $s_{v,i}$ to state $s'_{v,i}$ reading $\langle i,k\rangle$ (respectively $\langle i,k-1\rangle$ if $i = k$) for the pth time. In other words, in the ith part of M, it reads $p \cdot (k-1) - 1$ letters of $x_{i,0}$, staying in state $s_{v,i}$ and outputs the letter $\langle \mathbf{l},i,r\rangle$ for each letter $\langle i,r\rangle$ it reads; then it transitions to state $s'_{v,i}$ on reading $\langle i,k\rangle$ (respectively $\langle k,k-1\rangle$ if $i = k$) and outputs $\langle \mathbf{l},i,k\rangle$ (respectively $\langle \mathbf{l},k,k-1\rangle$); in the state $s'_{v,i}$ it outputs the empty letter for each letter $\langle i,r\rangle$ it reads.

We say that M *selects* an edge $e(i,j,q)$ if it makes a transition from state $s^{(2)}_{e,i,j}$ to state $s^{(3)}_{e,i,j}$ after having read the letter $\langle \bar{\mathbf{e}},i,j\rangle$ of $x_{i,j,p}$ exactly q times, where

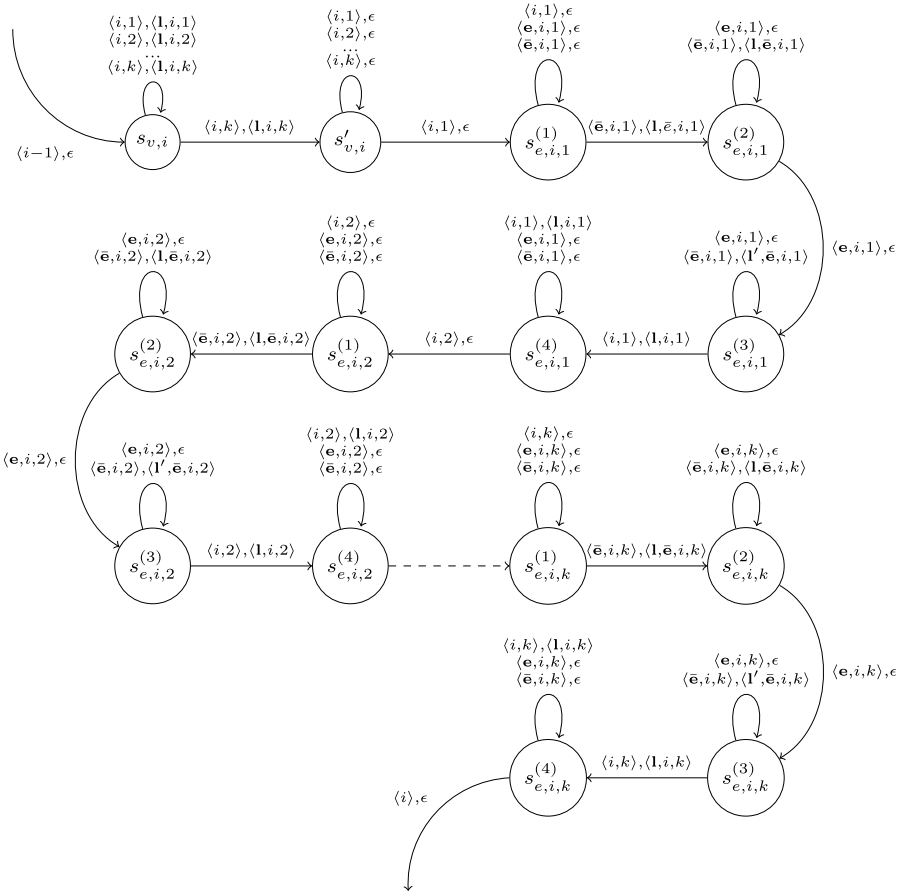

Fig. 1 The ith part of the Mealy machine M. It does not have the states $s^{(1)}_{e,i,i}, s^{(2)}_{e,i,i}, s^{(3)}_{e,i,i}$, and $s^{(4)}_{e,i,i}$; there is instead a transition from $s^{(4)}_{e,i,i-1}$ to $s^{(1)}_{e,i,i+1}$ reading $\langle i-1\rangle$ and writing ϵ, and there is a transition from $s^{(4)}_{e,k,k-1}$ to the last state reading $\langle k\rangle$ and writing ϵ. There are no transitions starting at the last state. (Drawing all this would have cluttered the figure too much)

$v_{i,p}$ is the vertex of color i that $e(i,j,q)$ is incident on. In other words, in the ith part of M, it transitions from the state $s^{(1)}_{e,i,j}$ to the state $s^{(2)}_{e,i,j}$ on reading the first letter of $x_{i,j,p}$ (if it did this transition any later, the census requirement of $\langle \mathbf{l}, \bar{\mathbf{e}}, i, j\rangle$ could not be met, as shown in the proof of Claim 2 below); then it stays in the state $s^{(2)}_{e,i,j}$ until it has read q times the letter $\langle \bar{\mathbf{e}}, i, j\rangle$ of $x_{i,j,p}$; then it transitions to the state $s^{(3)}_{e,i,j}$ on reading $\langle \mathbf{e}, i, j\rangle$; it stays in this state and outputs $\langle \mathbf{l}', \bar{\mathbf{e}}, i, j\rangle$ for each letter $\langle \bar{\mathbf{e}}, i, j\rangle$ it reads until transitioning to the state $s^{(4)}_{e,i,j}$ on reading the letter following $x_{i,j,p}$.

The following claims ensure that the edge-selection and the vertex-selection are compatible, i.e., that exactly one edge is selected from color i to color j, and that this edge is incident on the selected vertex of color i. □

Claim 1 *Let i be a color and let $v_{i,p}$ be the vertex selected in the ith part of M. In its ith part, M selects one edge incident to $v_{i,p}$ and to a vertex of color j, for each $j \in [k] \setminus \{i\}$.*

Proof After M has selected $v_{i,p}$, it has output p times each of the letters $\langle \mathbf{l}, i, 1\rangle$, $\langle \mathbf{l}, i, 2\rangle, \ldots, \langle \mathbf{l}, i, i-1\rangle, \langle \mathbf{l}, i, i+1\rangle, \langle \mathbf{l}, i, i+2\rangle, \ldots, \langle \mathbf{l}, i, k\rangle$. For each $j \in [k] \setminus \{i\}$, the only other transitions that output $\langle \mathbf{l}, i, j\rangle$ are the transition from $s^{(3)}_{e,i,j}$ to $s^{(4)}_{e,i,j}$ and a transition that loops on $s^{(4)}_{e,i,j}$. To meet the census requirement of $|V(i)|+1$ for $\langle \mathbf{l}, i, j\rangle$, M selects an edge while reading $x_{i,j,p}$. This edge is incident on $v_{i,p}$ by construction. □

The following claim makes sure that the edge selected from color i to color j is the same as the edge selected from color j to color i.

Claim 2 *Suppose M selects the edge $e(i, j, q)$ in its ith part. Then, M selects the edge $e(j, i, q)$ in its jth part.*

Proof Before M selects $e(i, j, q)$, it has output $q' \leq q$ times the letter $\langle \mathbf{l}, \bar{\mathbf{e}}, i, j\rangle$. On selecting $e(i, j, q)$ it transitions to the state $s^{(3)}_{e,i,j}$, and after the selection it outputs $\langle \mathbf{l}', \bar{e}, i, j\rangle$ for every letter $\langle \bar{\mathbf{e}}, i, j\rangle$ of $x_{i,j,p}$ it reads. As it reads

$$\left(\sum_{r=1}^{d_{V(j)}(v_{i,p})+1} \mathsf{gap}(v_{i,p}, j, r)\right) - q = |E(i, j)| - q$$

times the letter $\langle \bar{\mathbf{e}}, i, j\rangle$ of $x_{i,j,p}$ after it has selected $e(i, j, q)$, the Mealy machine outputs $|E(i, j)| - q$ times the letter $\langle \mathbf{l}', \bar{e}, i, j\rangle$ in its ith part.

The only other transition where it outputs $\langle \mathbf{l}, \bar{e}, i, j\rangle = \langle \mathbf{l}', \bar{e}, j, i\rangle$ is the transition in the jth part of M looping on $s^{(3)}_{e,j,i}$ that reads $\langle \bar{\mathbf{e}}, j, i\rangle$ and outputs $\langle \mathbf{l}', \bar{\mathbf{e}}, j, i\rangle$. To meet the census requirement for $\langle \mathbf{l}, \bar{e}, i, j\rangle$, this transition must be used exactly $|E(i, j)| - q'$ times.

The only other transitions where it outputs $\langle \mathbf{l}', \bar{e}, i, j\rangle = \langle \mathbf{l}, \bar{e}, j, i\rangle$ are two transitions in the jth part of M: the transition from $s^{(1)}_{e,j,i}$ to $s^{(2)}_{e,j,i}$ and the transition looping on $s^{(2)}_{e,j,i}$, both reading $\langle \bar{\mathbf{e}}, j, i\rangle$ and writing $\langle \mathbf{l}, \bar{e}, j, i\rangle$. These transitions can be used at most q' times as the transition of the previous paragraph is used $|E(i, j)| - q'$ times. These transitions have to be used at least q times to meet the census requirement for $\langle \mathbf{l}', \bar{\mathbf{e}}, i, j\rangle$. Thus, these transitions are used exactly q times and $q = q'$.

Finally, the transition from $s^{(2)}_{e,j,i}$ to $s^{(3)}_{e,j,i}$ happens after having read q times the letter $\langle \bar{\mathbf{e}}, j, i\rangle$ of some vertex $x_{j,i,p'}$, $p' \in [|V(j)|]$, which means that M selects the edge $e(j, i, q)$ in its jth part.

By Claims 1 and 2, the k vertices that are selected by M form a multicolored clique. Thus, $(k, G = (V(1) \cup V(2) \cup \cdots \cup V(k), E))$ is a YES-instance for MCC.

Now, suppose that $(k, G = (V(1) \cup V(2) \cup \cdots \cup V(k), E))$ is a YES-instance for MCC.

Let $\{v_{1,p_1}, v_{2,p_2}, \ldots, v_{k,p_k}\}$ be a multicolored clique in G. We will construct a word y meeting c such that a computation of M on input x generates y. For two

adjacent vertices v_{i,p_i} and v_{j,p_j}, define $\mathsf{edge}(v_{i,p_i}, v_{j,p_j}) = t$ such that $e(i, j, t) = v_{i,p_i} v_{j,p_j}$. The word y is $y_1 y_2 \dots y_k$, where y_i, for $i \in [k]$, is

$$(\langle \mathbf{l}, i, 1\rangle \langle \mathbf{l}, i, 2\rangle \dots \langle \mathbf{l}, i, i-1\rangle \langle \mathbf{l}, i, i+1\rangle \langle \mathbf{l}, i, i+2\rangle \dots \langle \mathbf{l}, i, k\rangle)^{p_i}$$

$$y_{i,1} y_{i,2} \cdots y_{i,i-1} y_{i,i+1} y_{i,i+2} \cdots y_{i,k}$$

and $y_{i,j}$, for $i \neq j \in [k]$, is

$$\langle \mathbf{l}, \bar{e}, i, j\rangle^{\mathsf{edge}(v_{i,p_i}, v_{j,p_j})} \langle \mathbf{l}', \bar{\mathbf{e}}, i, j\rangle^{|E(i,j)| - \mathsf{edge}(v_{i,p_i}, v_{j,p_j})} \langle \mathbf{l}, i, j\rangle^{|V(i)| - p_i + 1}.$$

We note that y meets the census requirement c. Moreover, the computation of M on input x, which selects (as defined in the first part of the proof) exactly the vertices and edges of the multicolored clique $\{v_{1,p_1}, v_{2,p_2}, \dots, v_{k,p_k}\}$, outputs y. Thus $(M = (S, s_0, \Gamma, \Sigma, T), x, c)$ is a YES-instance for *var*-GWMM. □

The theorem holds if we restrict $\epsilon \notin \Gamma \cup \Sigma$. Indeed, $\epsilon \notin \Gamma$ in the target instance, and one can add a new letter e to Σ, which replaces ϵ and has census requirement $c(e) = |x| - \sum_{i,j \in [k], i \neq j} (c(\langle \mathbf{l}, i, j\rangle) + c(\langle \mathbf{l}, \bar{\mathbf{e}}, i, j\rangle)$. This instance is equivalent since the modified M outputs one letter for each letter in x.

6 Applications

In this section we sketch two examples that illustrate how number-of-numbers parameterized problems may reduce to census problems about Mealy machines, parameterized by the size of the machine. For another application, see [10].

Example 1 (Heat-Sensitive Scheduling) In a recent paper Chrobak et al. [5] introduced a model for the issue of temperature-aware task scheduling for microprocessor systems. The motivation is that different jobs with the same time requirements may generate different heat loads, and it may be important to schedule the jobs so that some temperature threshold is not breached.

In the model, the input consists of a set of jobs that are all assumed to be of unit length, with each job assigned a numerical heat level. If at time t the processor temperature is T_t, and if the next job that is scheduled has heat level H, then the processor temperature at time $t + 1$ is

$$T_{t+1} = (T_t + H)/2.$$

It is also allowed that perhaps no job is scheduled for time $t + 1$ (that is, *idle time* is scheduled), in which case $H = 0$ in the above calculation of the updated temperature.

The relevant decision problem is whether all of the jobs can be scheduled, meeting a specified deadline, in such a way that a given temperature threshold is never exceeded. This problem has been shown to be NP-hard [5] by a reduction from 3-DIMENSIONAL MATCHING. An image instance of the reduction, however, involves arbitrarily many distinct heat levels asymptotically close to $H = 2$, for a temperature threshold of 1.

Processor 1	4		3			3	
Processor 2		5		3	1		5
$x =$	4	1	2	1	1	1	4

Fig. 2 A winning game for the census: 1 (1), 3 (3), 4 (1), 5 (2)

In the spirit of the "deconstruction of hardness proofs" advocated by Komusiewicz et al. [19] (see also [4, 24]), one might regard this problem as ripe for parameterization by the number of numbers, for example (scaling appropriately), a model based on $2k$ equally-spaced heat levels and a temperature threshold of k. Furthermore, if the heat levels of the jobs are only roughly classified in this way, it also makes sense to treat the temperature transition model similarly, as:

$$T_{t+1} = \lceil (T_t + H)/2 \rceil.$$

The input to the problem can now be viewed equivalently as a census of how many jobs there are for each of the $2k + 1$ heat levels, with the available potential units of idle time allowed to meet the deadline treated as "jobs" for which $H = 0$. Because of the ceiling function modeling the temperature transition, the problem now immediately reduces to *var*-EWMM, for a machine on $k + 1$ states (that represent the temperature of the processor) and an alphabet of size at most $2k + 1$. By Theorem 5, the problem is fixed-parameter tractable.

Example 2 (A Problem in Computational Chemistry) The parameterized problem of WEIGHTED SPLITS RECONSTRUCTION FOR PATHS that arises in computational chemistry [15] reduces to a special case of *var*-GWMM. The input to the problem is obtained from time-series spectrographic data concerning molecular weights. The problem as defined in [15] is equivalent to the following two-processor scheduling problem. The input consists of

- a sequence x of positive integer *time gaps* taken from a set of positive integers Γ, and
- a census requirement c on a set of positive integers Σ of *job lengths*.

The question is whether there is a "winning play" for the following one-person two-processor scheduling game. At each step, first, *Nature* plays the next positive integer "gap" of the sequence of time gaps x—this establishes the next *immediate deadline*. Second, the *Player* responds by scheduling on one of the two processors, a job that begins at the last stop-time on that processor, and ends at the immediate deadline. The *Player* wins if there is a sequence of plays (against x) that meets the census requirement c on job lengths. Figure 2 illustrates such a game.

This problem easily reduces to a special case of *var*-GWMM. Whether this special case is also $W[1]$-hard remains open.

7 Concluding Remarks

The practical world of computing is full of computational problems where inputs are "weighted" in a realistic model—weighted graphs provide a simple example relevant to many applications. Here we have begun to explore parameterizing on the *numbers*

of numbers as a way of mitigating computational complexity for problems that are numerically structured. One might view some of the impulse here as *moving approximation issues into the modeling*, as illustrated by Example 1 in Sect. 6. We believe this line of attack may be widely applicable.

To date, there has been little attention to parameterized complexity issues in the context of cryptography, control theory, and other numerically structured areas of application. Number of numbers parameterization may provide some inroads into these underdeveloped areas.

Our main FPT result, Theorem 5, has a poor worst-case running-time guarantee. Can this be improved—at least in important special cases?

Acknowledgement We thank Iyad Kanj for stimulating conversations about this work.

References

1. Alon, N., Azar, Y., Woeginger, G.J., Yadid, T.: Approximation schemes for scheduling on parallel machines. J. Sched. **1**, 55–66 (1998)
2. Bard, G.V.: Algebraic Cryptanalysis. Springer, Berlin (2009)
3. Bazgan, C.: Schémas d'approximation et complexité paramétrée. Master's thesis, Université Paris Sud (1995)
4. Betzler, N., Fellows, M.R., Guo, J., Niedermeier, R., Rosamond, F.A.: Fixed-parameter algorithms for Kemeny rankings. Theor. Comput. Sci. **410**(45), 4554–4570 (2009)
5. Chrobak, M., Dürr, C., Hurand, M., Robert, J.: Algorithms for temperature-aware task scheduling in microprocessor systems. In: Fleischer, R., Jinhui, X. (eds.) AAIM 2008. Lecture Notes in Computer Science, vol. 5034, pp. 120–130. Springer, Berlin (2008)
6. Downey, R.G., Fellows, M.R.: Parameterized Complexity. Springer, New York (1999)
7. Fellows, M.R., Gaspers, S., Rosamond, F.A.: Parameterizing by the number of numbers. In: Raman, V., Saurabh, S. (eds.) IPEC 2010. Lecture Notes in Computer Science, vol. 6478, pp. 123–134. Springer, Berlin (2010)
8. Fellows, M.R., Hermelin, D., Rosamond, F., Vialette, S.: On the parameterized complexity of multiple-interval graph problems. Theor. Comput. Sci. **410**(1), 53–61 (2009)
9. Fellows, M.R., Koblitz, N.: Fixed-parameter complexity and cryptography. In: Cohen, G.D., Mora, T., Moreno, O. (eds.) AAECC 1993. Lecture Notes in Computer Science, vol. 673, pp. 121–131. Springer, Berlin (1993)
10. Fellows, M.R., Lokshtanov, D., Misra, N., Mnich, M., Rosamond, F.A., Saurabh, S.: The complexity ecology of parameters: An illustration using bounded max leaf number. Theory Comput. Syst. **45**(4), 822–848 (2009)
11. Fellows, M.R., Lokshtanov, D., Misra, N., Rosamond, E.A., Saurabh, S.: Graph layout problems parameterized by vertex cover. In: Hong, S.-H., Nagamochi, H., Fukunaga, T. (eds.) ISAAC 2008. Lecture Notes in Computer Science, vol. 5369, pp. 294–305. Springer, Berlin (2008)
12. Fiala, J., Golovach, P.A., Kratochvíl, J.: Parameterized complexity of coloring problems: Treewidth versus vertex cover. Theor. Comput. Sci. **412**(23), 2513–2523 (2011)
13. Flum, J., Grohe, M.: Parameterized Complexity Theory. Texts in Theoretical Computer Science. An EATCS Series, vol. XIV. Springer, Berlin (2006)
14. Frank, A., Tardos, É.: An application of simultaneous diophantine approximation in combinatorial optimization. Combinatorica **7**(1), 49–65 (1987)
15. Gaspers, S., Liedloff, M., Stein, M.J., Suchan, K.: Complexity of splits reconstruction for low-degree trees. In: Kratochvíl, J. (ed.) WG 2011. Lecture Notes in Computer Science, vol. 6986 pp. 167–178. Springer, Berlin (2011)
16. Gramm, J., Niedermeier, R., Rossmanith, P.: Fixed-parameter algorithms for closest string and related problems. Algorithmica **37**(1), 25–42 (2003)
17. Kannan, R.: Minkowski's convex body theorem and integer programming. Math. Oper. Res. **12**(3), 415–440 (1987)

18. Knuth, D.E.: The Art of Computer Programming, Volume III: Sorting and Searching. Addison-Wesley, Reading (1973)
19. Komusiewicz, C., Niedermeier, R., Uhlmann, J.: Deconstructing intractability—a multivariate complexity analysis of interval constrained coloring. J. Discrete Algorithms **9**(1), 137–151 (2011)
20. Lenstra, H.W.: Integer programming with a fixed number of variables. Math. Oper. Res. **8**(4), 538–548 (1983)
21. Mealy, G.H.: A method for synthesizing sequential circuits. Bell Syst. Tech. J. **34**(5), 1045–1079 (1955)
22. Munro, I., Spira, P.M.: Sorting and searching in multisets. SIAM J. Comput. **5**(1), 1–8 (1976)
23. Niedermeier, R.: Invitation to Fixed-Parameter Algorithms. Oxford Lecture Series in Mathematics and Its Applications. Oxford University Press, Oxford (2006)
24. Niedermeier, R.: Reflections on multivariate algorithmics and problem parameterization. In: Marion, J.-Y., Schwentick, T. (eds.) STACS 2010. LIPIcs, vol. 5, pp. 17–32. Schloss Dagstuhl—Leibniz-Zentrum fuer Informatik, Dagstuhl (2010)
25. Parikh, R.J.: On context-free languages. J. ACM **13**(4), 570–581 (1966)
26. Pietrzak, K.: On the parameterized complexity of the fixed alphabet shortest common supersequence and longest common subsequence problems. J. Comput. Syst. Sci. **67**(4), 757–771 (2003)
27. Roche, E., Schabes, Y.: Finite-state language processing. The MIT Press, Cambridge (1997)
28. Savage, J.E.: Models of Computation—Exploring the Power of Computing. Addison-Wesley, Reading (1998)
29. Sen, S., Gupta, N.: Distribution-sensitive algorithms. Nord. J. Comput. **6**, 194–211 (1999)
30. Sontag, E.: Mathematical Control Theory: Deterministic Finite Dimensional Systems, 2nd edn. Springer, Berlin (1998)