Parameterizing by the Number of Numbers

Michael R. Fellows[1], Serge Gaspers[2], and Frances A. Rosamond[1]

[1] PCRU, Office of DVC (Research), University of Newcastle, Callaghan, Australia.
{michael.fellows,frances.rosamond}@newcastle.edu.au
[2] CMM, University of Chile, Santiago, Chile. sgaspers@dim.uchile.cl

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 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.

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 Chapter 5 of his monograph [10]). For example, it was Valerie King in 1994 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.

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.

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 [11], there were not so many interesting applications of this fundamental result.

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

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 $\mathbf{Ax} \le \mathbf{b}$. ILPF was shown to be fixed-parameter tractable by Lenstra [8] and the running time has been improved by Kannan [6].

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 $[n] = \{1, \dots, n\}$.

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

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 λ denotes the empty letter. We denote the concatenation of two words $x_1, x_2 \in \Sigma^*$ by $x_1 x_2$. The i^{th} *power* of a word x is denoted x^i or $(x)^i$ and represents the word $\underbrace{xx \dots x}_{i \text{ times}}$.

3 Subset Sum and Partition

We start with two classic problems on multisets an 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 $\sum_{a \in X} a = \sum_{b \in A \setminus X} b$?

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

Proof. Given an instance (A, s) for *var*-SubSum, with $||A|| = k$, 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.

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

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 (the proof can be found in the appendix).

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

4 Numerical 3-Dimensional Matching

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 $x_{i,j,\ell}$, $i \in [k_1], j \in [k_2], \ell \in [k_3]$:

$$\begin{aligned}
x_{i,j,\ell} &= 0 && \text{for each } (i,j,\ell) \in ([k_1],[k_2],[k_3]) \\
& && \text{such that } a_i + b_j + c_\ell \neq s \\
\sum_{(j,\ell)\in([k_2],[k_3])} x_{i,j,\ell} &= m_{i,a} && \forall i \in [k_1] \\
\sum_{(i,\ell)\in([k_1],[k_3])} x_{i,j,\ell} &= m_{j,b} && \forall j \in [k_2] \\
\sum_{(i,j)\in([k_1],[k_2])} x_{i,j,\ell} &= m_{\ell,c} && \forall \ell \in [k_3]
\end{aligned}$$

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 $||C|| > ||\{a + b : a \in A, b \in B\}||$. Another well known numerical problem, very closely related to *var*-Num3-DM, 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 $C_1, \ldots, C_n \in (A, B, S)$, 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 (the proof is given in the appendix).

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.*

5 Mealy Machines

Mealy machines [9] are finite-state transducers, generating an output based on their current state and input. 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 symbol to the corresponding next state and output symbol.

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 symbol, there may be more than one possibility for the next state and output symbol. (Here $\mathcal{P}(X)$ denotes the powerset of a set X.)

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

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 machine $M = (S, s_0, \Gamma, \Sigma, T)$, and a census requirement $c : \Sigma \to \mathbb{N}$
Parameter:	$\|S\| + \|\Gamma\| + \|\Sigma\|$
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 [4] showing that Bandwidth is fixed-parameter tractable when parameterized my the maximum number of leaves in a spanning tree of the input graph. We need the following definition and lemma from [4].

In a digraph D, two directed paths 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 ([4]). *Any directed path 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 path Δ' from s to t, where Δ' has the form:*

(1) Δ' consists of an underlying directed path 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.*

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 $k = |S'| + |\Gamma'| + |\Sigma'|$. As M' might have multiple transitions from one state to another, we first subdivide each transition in order to obtain a 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, $S = S'$, $s_0 = s'_0$, $\Gamma = \Gamma' \cup \{\lambda\}$, and $\Sigma = \Sigma' \cup \{\lambda\}$; for each transition 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, λ) to (s_o, λ) to T. Clearly, there is at most one transition between every two states in M.

Our algorithm goes over all transition paths in M of length at most $|S|^2$ that start from s_0. There are at most $|S|^2!$ such transition paths and each such transition path has at most $|S|!$ 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 path and $L = (\ell_0, \ell_1, \ldots, \ell_{|L|})$ 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 path P, and executing x_i times each short loop ℓ_i, $0 \le i \le |L|$, 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 $\Sigma = \{\langle \ell, 1 \rangle, \langle \ell, 2 \rangle, \ldots, \langle \ell, |\Sigma| \rangle\}$. Define $m(i, j)$, $0 \le i \le |L|$, $1 \le j \le |\Sigma|$, to denote the number of times that M writes the letter $\langle \ell, j \rangle$ when it executes the loop ℓ_i once. Define $m(j)$, $1 \le j \le |\Sigma|$, 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 path P. Then, we only need to verify that there exist integers $x_1, x_2, \ldots, x_{|L|}$ such that

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

As the number of integer variables of this program is at most $|L| \le |S|! \le (|S'| + |T'|)! \le (|S'| + |S'|^2 \cdot |\Gamma'| \cdot |\Sigma'|)! \le (k + k^4)!$, and the number of transition paths that the algorithm considers is at most $|S|^2! \le (k + k^4)^2!$, *var*-EWMM is fixed-parameter tractable. $\square$

We note that the proof in [4] 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 $c : \Sigma \to \mathbb{N}$
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?

Our dynamic-programming algorithm proving the following theorem can be found in the appendix.

Theorem 6. *var*-GWMM *is in XP.*

To show that *var*-GWMM is $W[1]$-hard, we reduce from the MULTICOLORED CLIQUE problem, which is $W[1]$-hard [3].

MULTICOLORED CLIQUE (MCC)	
Input:	an integer k and a connected undirected graph $G = (V(1) \cup V(2) \ldots \cup V(k), E)$ 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.

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

Proof. Let $(k, G = (V(1) \cup V(2) \ldots \cup V(k), E))$ 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, $i,j \in [k]$. 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 q^{th} 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 $(q-1)^{\text{th}}$ 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 $\{\langle i \rangle, \langle i,j \rangle, \langle \bar{e}, i, j \rangle, \langle e, i, j \rangle : i, j \in [k], i \neq j\}$. M's output alphabet, Σ, is $\{\lambda\} \cup \{\langle \ell, i, j \rangle, \langle \ell, \bar{e}, i, j \rangle : i, j \in [k], i \neq j\}$. The word x is defined

$$
\begin{aligned}
x &:= x_1 x_2 \ldots x_k \\
x_i &:= x_{i,0} x_{i,1} \ldots x_{i,i-1} x_{i,i+1} x_{i,i+2} \ldots x_{i,k} \langle i \rangle && \forall i \in [k] \\
x_{i,0} &:= (\langle i,1 \rangle \langle i,2 \rangle \ldots \langle i,i-1 \rangle \langle i,i+1 \rangle \langle i,i+2 \rangle \ldots \langle i,k \rangle)^{|V(i)|} && \forall i \in [k] \\
x_{i,j} &:= \langle i,j \rangle x_{i,j,1} \langle i,j \rangle x_{i,j,2} \ldots \langle i,j \rangle x_{i,j,|V(i)|} \langle i,j \rangle && \forall i,j \in [k], i \neq j \\
x_{i,j,p} &:= \langle \bar{e}, i, j \rangle^{\mathsf{gap}(v_{i,p},j,1)} \langle e, i, j \rangle \langle \bar{e}, i, j \rangle^{\mathsf{gap}(v_{i,p},j,2)} \langle e, i, j \rangle \\
&\quad \ldots \langle \bar{e}, i, j \rangle^{\mathsf{gap}(v_{i,p},j,d_{V(j)}(v_{i,p}))} \langle e, i, j \rangle \langle \bar{e}, i, j \rangle^{\mathsf{gap}(v_{i,p},j,d_{V(j)}(v_{i,p})+1)}.
\end{aligned}
$$

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

$$
\begin{aligned}
c(\langle \ell, i, j \rangle) &:= |V(i)| + 1 \\
c(\langle \ell, i, j \rangle) &:= |V(i)| \\
c(\langle \ell, \bar{e}, i, j \rangle) &:= |E(i,j)|.
\end{aligned}
$$

The Mealy machine M consists of k parts. The i^{th} 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 k^{th} part, there is a transition to a final state): it reads the letter $\langle i \rangle$ and writes the letter λ. We set $\langle \ell', \bar{e}, i, j \rangle = \langle \ell, \bar{e}, j, i \rangle$ for all $i \neq j \in [k]$.

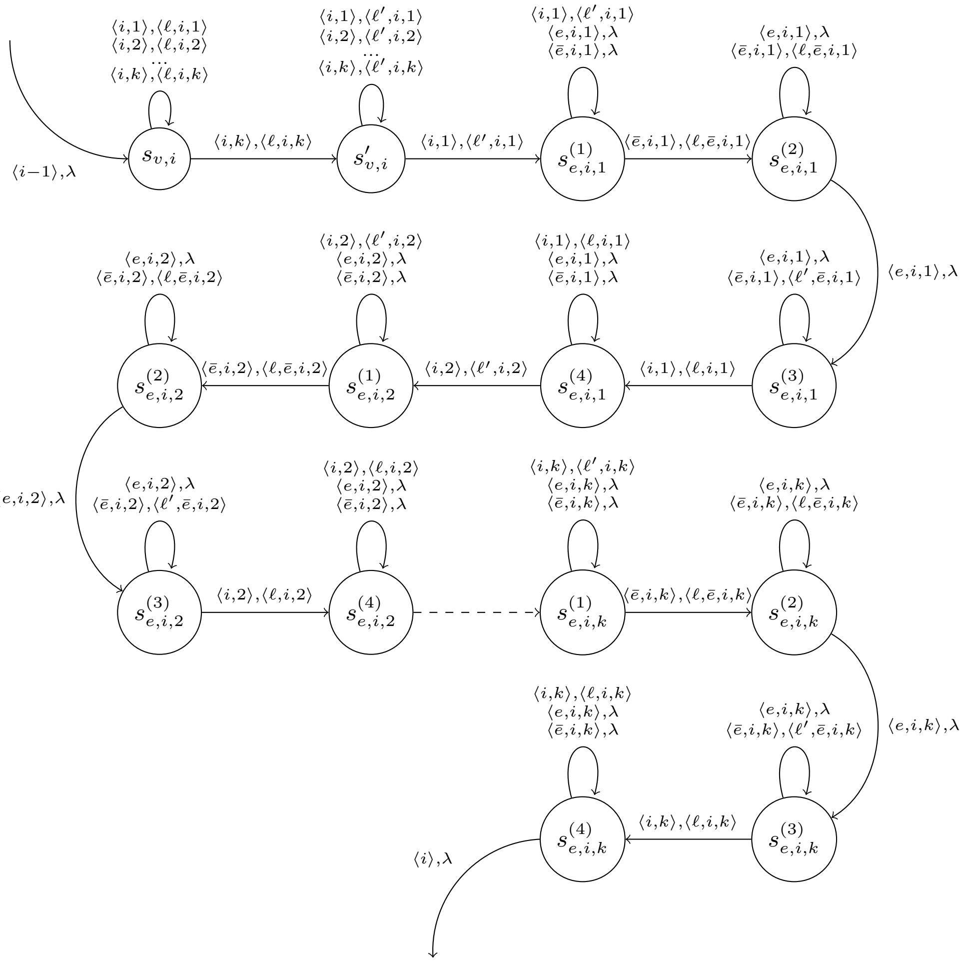

Fig. 1. The i^{th} part of the Mealy machine M. It does not have the states $s^{(1)}_{e,i,i}, s^{(2)}_{e,i,i}, s^{(3)}_{e,i,i}$, and $s^{(4)}_{e,i,i}$; there is instead a transition from $s^{(4)}_{e,i,i-1}$ to $s^{(1)}_{e,i,i+1}$ reading $\langle i-1\rangle$ and writing λ, and there is a transition from $s^{(4)}_{e,k,k-1}$ to a final state reading $\langle k\rangle$ and writing λ (drawing all this would have cluttered the figure too much).

First, suppose that $(M = (S, s_0, \Gamma, \Sigma, T), x, c)$ 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 p^{th} time. In other words, in the i^{th} 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 $\langle \ell, i, r\rangle$ for each letter $\langle i, r\rangle$ it reads; then it transitions to state $s'_{v,i}$ on reading $\langle i, k\rangle$ (respectively $\langle i, k-1\rangle$) and outputs $\langle \ell, i, k\rangle$ (respectively $\langle \ell, i, k-1\rangle$); in the state $s'_{v,i}$ it outputs the letter $\langle \ell', i, r\rangle$ 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 $\langle \bar{e}, i, j\rangle$ of $x_{i,j,p}$ exactly q times, where $v_{i,p}$ is the vertex of color i that $e(i, j, q)$ is incident on. In other words, in the i^{th} 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 $\langle \ell, \bar{e}, i, j \rangle$ 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 $\langle \bar{e}, i, j \rangle$ of $x_{i,j,p}$; then it transitions to the state $s^{(3)}_{e,i,j}$ on reading $\langle e, i, j \rangle$; it stays in this state and outputs $\langle \ell', \bar{e}, i, j \rangle$ for each letter $\langle \bar{e}, i, j \rangle$ 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 i^{th} part of M. In its i^{th} 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 $\langle \ell, i, 1 \rangle$, $\langle \ell, i, 2 \rangle, \ldots, \langle \ell, i, i-1 \rangle, \langle \ell, i, i+1 \rangle, \langle \ell, i, i+2 \rangle, \ldots, \langle \ell, i, k \rangle$. For each $j \in [k] \setminus \{i\}$, the only other transitions that output $\langle \ell, i, j \rangle$ 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 $\langle \ell, i, j \rangle$, 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 i^{th} part. Then, M selects the edge $e(j,i,q)$ in its j^{th} part.*

Proof. Before M selects $e(i,j,q)$, it has output $q' \leq q$ times the letter $\langle \ell, \bar{e}, i, j \rangle$. On selecting $e(i,j,q)$ it transitions to the state $s^{(3)}_{e,i,j}$, and after the selection it outputs $\langle \ell', \bar{e}, i, j \rangle$ for every letter $\langle \bar{e}, i, j \rangle$ 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 $\langle \bar{e}, i, j \rangle$ of $x_{i,j,p}$ after it has selected $e(i,j,q)$, it outputs $|E(i,j)| - q$ times the letter $\langle \ell', \bar{e}, i, j \rangle$ in its i^{th} part.

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

The only other transitions where it outputs $\langle \ell', \bar{e}, i, j \rangle = \langle \ell, \bar{e}, j, i \rangle$ are two transitions in the j^{th} 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 $\langle \bar{e}, j, i \rangle$ and writing $\langle \ell, \bar{e}, j, i \rangle$. 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 $\langle \ell', \bar{e}, i, j \rangle$. 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 $\langle \bar{e}, j, i\rangle$ of some vertex $x_{j,i,p'}, p' \in [|V(j)|]$, which means that M selects the edge $e(j,i,q)$ in its j^{th} part. □

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

Now, suppose that $(M = (S, s_0, \Gamma, \Sigma, T), x, c)$ is a NO-instance for *var*-GWMM. For the sake of contradiction, suppose that $(k, G = (V(1) \cup V(2) \ldots \cup V(k), E))$ 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 $y_1 y_2 \ldots v_k$, where $y_i, i \in [k]$ is

$$(\langle \ell, i, 1\rangle\langle \ell, i, 2\rangle \ldots \langle \ell, i, i-1\rangle\langle \ell, i, i+1\rangle\langle \ell, i, i+2\rangle \ldots \langle \ell, i, k\rangle)^{p_i}$$
$$(\langle \ell', i, 1\rangle\langle \ell', i, 2\rangle \ldots \langle \ell', i, i-1\rangle\langle \ell', i, i+1\rangle\langle \ell', i, i+2\rangle \ldots \langle \ell', i, k\rangle)^{|V(i)|-p_i}$$
$$y_{i,1} y_{i,2} \cdots y_{i,i-1} y_{i,i+1} y_{i,i+2} \cdots y_{i,k} \langle i\rangle$$

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

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

We note that there is a computation of M on input x that generates y and that y meets the census requirement c. This contradicts $(M = (S, s_0, \Gamma, \Sigma, T), x, c)$ being a NO-instance. □

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 [4].

Example 1: Heat-Sensitive Scheduling. In a recent paper Chrobak et al. [2] 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 (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 [2] 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.

In the spirit of the "deconstruction of hardness proofs" advocated by Komusiewicz et al. [7] (see also [1]), 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 [5] 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 [5] 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

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)

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 Section 6. We believe this line of attack may be widely applicable.

Finally, we remark that to date, there has been little attention to parameterized complexity in the context of cryptography. Number of numbers parameterization may provide some inroads into this underdeveloped area.

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

References

1. Nadja Betzler, Michael R. Fellows, Jiong Guo, Rolf Niedermeier, and Frances A. Rosamond. Fixed-parameter algorithms for kemeny rankings. *Theoretical Computer Science*, 410(45):4554–4570, 2009.
2. Marek Chrobak, Christoph Dürr, Mathilde Hurand, and Julien Robert. Algorithms for temperature-aware task scheduling in microprocessor systems. In Rudolf Fleischer and Jinhui Xu, editors, *AAIM*, volume 5034 of *Lecture Notes in Computer Science*, pages 120–130. Springer, 2008.
3. Michael R. Fellows, Danny Hermelin, Frances Rosamond, and Stéphane Vialette. On the parameterized complexity of multiple-interval graph problems. *Theoretical Computer Science*, 410(1):53–61, 2009.
4. Michael R. Fellows, Daniel Lokshtanov, Neeldhara Misra, Matthias Mnich, Frances A. Rosamond, and Saket Saurabh. The complexity ecology of parameters: An illustration using bounded max leaf number. *Theory of Computing Systems*, 45(4):822–848, 2009.
5. Serge Gaspers, Mathieu Liedloff, Maya J. Stein, and Karol Suchan. Complexity of splits reconstruction for low-degree trees. *CoRR*, abs/1007.1733, 2010. Available on arXiv.org.
6. Ravi Kannan. Minkowski's convex body theorem and integer programming. *Mathematics of Operations Research*, 12(3):415–440, 1987.
7. Christian Komusiewicz, Rolf Niedermeier, and Johannes Uhlmann. Deconstructing intractability: A case study for interval constrained coloring. In Gregory Kucherov and Esko Ukkonen, editors, *CPM*, volume 5577 of *Lecture Notes in Computer Science*, pages 207–220. Springer, 2009.
8. Hendrik W. Lenstra. Integer programming with a fixed number of variables. *Mathematics of Operations Research*, 8(4):538–548, 1983.
9. George H. Mealy. A method for synthesizing sequential circuits. *Bell System Technical Journal*, 34(5):1045–1079, 1955.
10. Rolf Niedermeier. *Invitation to Fixed-Parameter Algorithms*, chapter 5: The Art of Problem Parameterization, pages 41–49. Oxford Lecture Series in Mathematics and Its Applications. Oxford University Press, Oxford, 2006.
11. Rolf Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Oxford Lecture Series in Mathematics and Its Applications. Oxford University Press, Oxford, 2006.

A Missing proofs

Proof (of Theorem 2). 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.

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

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 $\mathbf{Ax} \leq \mathbf{b}$. □

Proof (of Theorem 4). 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 $x_{i,j,\ell}$, $i, j, \ell \in [k]$:

$$\begin{aligned} x_{i,j,\ell} &= 0 && \text{for each } i, j, \ell \in [k] \\ & && \text{such that } a_i + a_j + a_\ell \neq s \\ \sum_{\substack{j,\ell \in [k] \\ j,\ell \neq i}} &(x_{i,j,\ell} + x_{j,i,\ell} + x_{j,\ell,i}) \\ +2 \cdot \sum_{\substack{j \in [k] \\ j \neq i}} &(x_{i,i,j} + x_{i,j,i} + x_{j,i,i}) \\ &+3 \cdot x_{i,i,i} = m_i && \forall i \in [k] \end{aligned}$$

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. □

Proof (of Theorem 6). Let $\Sigma = \{b_1, \ldots, b_{|\Sigma|}\}$. Our dynamic programming algorithm computes the entries of a boolean table A. The table A has an entry $A[s, c_1, \ldots, c_{|\Sigma|}, i]$ for each state $s \in S$, each $c_j \in \{0, \ldots, c(b_j)\}$, $j \in [|\Sigma|]$, and each index $i \in [|x|]$. The entry $A[s, c_1, \ldots, c_{|\Sigma|}, i]$ is set to `true` if there exists a computation of M reading the first i letters of x and outputting a word y in which the letter b_j occurs c_j times, for

each $j \in [|\Sigma|]$, and to `false` otherwise.
Set $A[s, c_1, \ldots, c_{|\Sigma|}, 0]$ to `true` if $s = s_0$ and $c_1 = \ldots = c_{|\Sigma|} = 0$, and to `false` otherwise. We compute the values of the table by increasing index i:

$$A[s, c_1, \ldots, c_{|\Sigma|}, i] = \bigvee_{\substack{s' \in S, b_j \in \Sigma: \\ T(s', x[i]) = (s, b_j)}} A[s', c_1, \ldots, c_{j-1}, c_j - 1, c_{j+1}, \ldots, c_{|\Sigma|}, i-1].$$

Finally, there exists an x-computation of M generating a word y that meets the census requirement if and only if $\bigvee_{s \in S} A[s, c(b_1), \ldots, c(b_{|\Sigma|}), |x|]$ is `true`.
The table has $|S| \cdot |x| \cdot \Pi_{j=1}^{|\Sigma|} c(b_j) \leq |S| \cdot |x|^{|\Sigma|+1}$ entries, and each entry can be computed in time $|S| \cdot |\Sigma|$. The running time of the algorithm is thus upper bounded by $O(n^{k+1} \cdot k^3)$, where n is the length of the description of an input instance, and k is the parameter. □