# Huge Unimodular N-Fold Programs 

Shmuel Onn * Pauline Sarrabezolles ${ }^{\dagger}$


#### Abstract

Optimization over $l \times m \times n$ integer 3 -way tables with given line-sums is NP-hard already for fixed $l=3$, but is polynomial time solvable with both $l, m$ fixed. In the huge version of the problem, the variable dimension $n$ is encoded in binary, with $t$ layer types. It was recently shown that the huge problem can be solved in polynomial time for fixed $t$, and the complexity of the problem for variable $t$ was raised as an open problem. Here we solve this problem and show that the huge table problem can be solved in polynomial time even when the number $t$ of types is variable. The complexity of the problem over 4 -way tables with variable $t$ remains open. Our treatment goes through the more general class of huge $n$-fold integer programming problems. We show that huge integer programs over $n$-fold products of totally unimodular matrices can be solved in polynomial time even when the number $t$ of brick types is variable.


## 1 Introduction

Consider the following optimization problem over 3 -way tables with given line-sums:

$$
\min \left\{w x: x \in \mathbb{Z}_{+}^{l \times m \times n}, \sum_{i} x_{i, j, k}=e_{j, k}, \sum_{j} x_{i, j, k}=f_{i, k}, \sum_{k} x_{i, j, k}=g_{i, j}\right\} .
$$

It is NP-hard already for $l=3$, see [3]. Moreover, every bounded integer program can be isomorphically represented in polynomial time for some $m$ and $n$ as some $3 \times m \times n$ table problem, see [4]. However, when both $l, m$ are fixed, it is solvable in polynomial time [2, 8, 10, and in fact, in time which is cubic in $n$ and linear in the binary encoding of $w, e, f, g$, see [7]. Assume throughout then that $l, m$ are fixed, and regard each table as a tuple $x=\left(x^{1}, \ldots, x^{n}\right)$ consisting of $n$ many $l \times m$ layers. The problem is called huge if the variable number $n$ of layers is encoded in binary. We are then given $t$ types of layers, where each type $k$ has its cost matrix $w^{k} \in \mathbb{Z}^{l \times m}$, column-sums vector $e^{k} \in \mathbb{Z}_{+}^{m}$, and row-sums vector $f^{k} \in \mathbb{Z}_{+}^{l}$. In addition, we are

[^0]given positive integers $n_{1}, \ldots, n_{t}, n$ with $n_{1}+\cdots+n_{t}=n$, all encoded in binary. A feasible table $x=\left(x^{1}, \ldots, x^{n}\right)$ then must have first $n_{1}$ layers of type 1 , next $n_{2}$ layers of type 2 , and so on, with last $n_{t}$ layers of type $t$. The special case of $t=1$ is the case of symmetric tables, where all layers have the same cost, row and column sums, and the classical (non-huge) table problem occurs as the special case of $t=n$ and $n_{1}=\cdots=n_{t}=1$. Note that for each $k$, the set of possible layers of type $k$ is
$$
\left\{z \in \mathbb{Z}_{+}^{l \times m}: \sum_{i} z_{i, j}=e_{j}^{k}, \sum_{j} z_{i, j}=f_{i}^{k}\right\}
$$
and may have cardinality which is exponential in the binary encoding of $e^{k}, f^{k}$. So it is not off hand clear how to even write down a single table, let alone optimize.

The huge table problem was recently considered in [11, where it was shown, combining results of [2, 8, 10] on Graver bases and results of [5, 6] on integer cones, that it can be solved in polynomial time for fixed $t$. The complexity of the problem for variable $t$ was raised as an open problem. Here we solve this problem and show that the huge table problem can be solved in polynomial time even when $t$ is variable.

Theorem 1.1 The huge 3-way table problem with a variable numbert of types can be solved in time which is polynomial in $t$ and in the binary encoding of $w^{k}, e^{k}, f^{k}, g, n_{k}$.

It was moreover shown in [11] that the huge $d$-table problem over $m_{1} \times \cdots m_{d-1} \times n$ tables with $m_{1}, \ldots m_{d-1}$ fixed and $n$ variable can also be solved in polynomial time for any fixed number $t$ of types. Interestingly, we do not know whether Theorem 1.1 could be extended to this more general situation, and the complexity of the huge $d$-way table problem with variable $t$ remains open, already for $3 \times 3 \times 3 \times n$ tables.

Theorem 1.1 follows from broader results which we proceed to describe. The class of $n$-fold integer programming problems is defined as follows. The $n$-fold product of an $s \times d$ matrix $A$ is the following $(d+s n) \times(d n)$ matrix, with $I$ the $d \times d$ identity,

$$
A^{[n]}:=\left(\begin{array}{cccc}
I & I & \cdots & I \\
A & 0 & \cdots & 0 \\
0 & A & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & A
\end{array}\right)
$$

The classical $n$-fold integer programming problem is then the following:

$$
\begin{equation*}
\min \left\{w x: x \in \mathbb{Z}^{d n}, A^{[n]} x=b, l \leq x \leq u\right\} \tag{1}
\end{equation*}
$$

where $w \in \mathbb{Z}^{d n}, b \in \mathbb{Z}^{d+s n}$, and $l, u \in \mathbb{Z}_{\infty}^{d n}$ with $\mathbb{Z}_{\infty}:=\mathbb{Z} \uplus\{ \pm \infty\}$. For instance, optimization over multiway tables is an $n$-fold program, as is explained later on.

Our starting point is the following result on classical $n$-fold integer programming, established in [2, 8], building on results of [1, 2, ,12]. See the monograph [10] for a detailed treatment of the theory and applications of $n$-fold integer programming.

Proposition 1.2 For fixed matrix $A$, the classical n-fold integer programming problem (1) can be solved in time polynomial in $n$ and the binary encoding of $w, l, u, b$.

This result holds more generally if the identity $I$ in the definition of $A^{[n]}$ is replaced by another fixed matrix $B$. Moreover, recently, in [7], it was shown that the problem can be solved in time which is cubic in $n$ and linear in the binary encoding of $w, b, l, u$.

The vector ingredients of an $n$-fold integer program are naturally arranged in bricks, where $w=\left(w^{1}, \ldots, w^{n}\right)$ with $w^{i} \in \mathbb{Z}^{d}$ for $i=1, \ldots, n$, and likewise for $l, u$, and where $b=\left(b^{0}, b^{1}, \ldots, b^{n}\right)$ with $b^{0} \in \mathbb{Z}^{d}$ and $b^{i} \in \mathbb{Z}^{s}$ for $i=1, \ldots, n$. Call an $n$-fold integer program huge if $n$ is encoded in binary. More precisely, we are now given $t$ types of bricks, where each type $k=1, \ldots, t$ has its cost $w^{k} \in \mathbb{Z}^{d}$, lower and upper bounds $l^{k}, u^{k} \in \mathbb{Z}^{d}$, and right-hand side $b^{k} \in \mathbb{Z}^{s}$. Also given are $b^{0} \in \mathbb{Z}^{d}$ and positive integers $n_{1}, \ldots, n_{t}, n$ with $n_{1}+\cdots+n_{t}=n$, all encoded in binary. A feasible point $x=\left(x^{1}, \ldots, x^{n}\right)$ now must have first $n_{1}$ bricks of type 1 , next $n_{2}$ bricks of type 2 , and so on, with last $n_{t}$ bricks of type $t$. Classical $n$-fold integer programming occurs as the special case of $t=n$ and $n_{1}=\cdots=n_{t}=1$, and symmetric $n$-fold integer programming occurs as the special case of $t=1$. We show the following.

Theorem 1.3 Let $A$ be a fixed totally unimodular matrix and consider the huge $n$-fold program over $A$ with a variable number t of types. Then the optimization problem can be solved in time polynomial in $t$ and the binary encoding of $w^{k}, l^{k}, u^{k}, b^{k}, n_{k}$.

The rest of the article is organized as follows. In Section 2 we discuss the feasibility problem which is easier than the optimization problem and admits a more efficient algorithm. In Section 3 we discuss the optimization problem, using the results on feasibility. We conclude in Section 4 with further discussion of tables.

## 2 Feasibility

In this section we consider the feasibility problem for huge $n$-fold integer programs:

$$
\text { is }\left\{x \in \mathbb{Z}^{d n}: A^{[n]} x=b, l \leq x \leq u\right\} \text { nonempty ? }
$$

We begin with the case of symmetric programs, with one type, so that $t=1$, over an $s \times d$ totally unimodular matrix $A$. So the data here consists of the top right-hand side $a \in \mathbb{Z}^{d}$, and for all bricks the same lower and upper bounds $l$, $u \in \mathbb{Z}_{\infty}^{d}$ and same right-hand side $b \in \mathbb{Z}^{s}$. Then the set in question can be written as

$$
\begin{equation*}
\left\{x \in \mathbb{Z}^{d n}: \sum_{i=1}^{n} x^{i}=a, A x^{i}=b, l \leq x^{i} \leq u, i=1, \ldots, n\right\} \tag{2}
\end{equation*}
$$

We have the following lemma.
Lemma 2.1 Let $A$ be totally unimodular. Then the set in (2) is nonempty if and only if $A a=n b$ and $n l \leq a \leq n u$, and this can be decided in time that is polynomial in the binary encoding of $n, l, u, a, b$, even when $A$ is a variable part of the input.

Proof. Suppose first that the set in (2) contains a feasible point $x=\left(x^{1}, \ldots, x^{n}\right)$. Then $A a=A \sum_{i=1}^{n} x^{i}=\sum_{i=1}^{n} A x^{i}=n b$, and $n l \leq a=\sum_{i=1}^{n} x^{i} \leq n u$. For the converse we use induction on $n$. Suppose $a$ satisfies the conditions. If $n=1$ then $x^{1}:=a$ is a feasible point in (2). Suppose now $n \geq 2$. Consider the system

$$
l \leq y \leq u, \quad A y=b, \quad(n-1) l \leq a-y \leq(n-1) u
$$

in the variable vector $y$. Then $y=\frac{1}{n} a$ is a real solution to this system, and therefore, since $A$ is totally unimodular, there is also an integer solution $x^{n}$ to this system. In particular, $A x^{n}=b$ and $l \leq x^{n} \leq u$. Let $\bar{a}:=a-x^{n}$. Then $A \bar{a}=A\left(a-x^{n}\right)=(n-1) b$ and $(n-1) l \leq \bar{a}=a-x^{n} \leq(n-1) u$. It therefore now follows by induction that there is an integer solution $\left(x^{1}, \ldots, x^{n-1}\right)$ to the $(n-1)$-fold program

$$
\left\{x \in \mathbb{Z}^{d(n-1)}: \sum_{i=1}^{n-1} x^{i}=\bar{a}, A x^{i}=b, l \leq x^{i} \leq u, i=1, \ldots, n-1\right\}
$$

Then $\sum_{i=1}^{n} x^{i}=\bar{a}+x^{n}=a$ and therefore $x:=\left(x^{1}, \ldots, x^{n-1}, x^{n}\right)$ is a feasible point in (2). The statement about the computational complexity is obvious.

We proceed with the general case of $t$ types. So the data now consists of $b^{0} \in \mathbb{Z}^{d}$ and for $k=1, \ldots, t$, lower and upper bounds $l^{k}, u^{k} \in \mathbb{Z}_{\infty}^{d}$, right-hand side $b^{k} \in \mathbb{Z}^{s}$, and positive integer $n_{k}$, with $n_{1}+\cdots+n_{t}=n$. We denote by $I_{1} \uplus \cdots \uplus I_{t}=\{1, \ldots, n\}$ the natural partition with $\left|I_{k}\right|=n_{k}$. So the set in question can be now written as

$$
\begin{equation*}
\left\{x \in \mathbb{Z}^{d n}: \sum_{i=1}^{n} x^{i}=b^{0}, A x^{i}=b^{k}, l^{k} \leq x^{i} \leq u^{k}, k=1, \ldots, t, i \in I_{k}\right\} . \tag{3}
\end{equation*}
$$

We have the following theorem asserting that when $A$ is totally unimodular the feasibility problem is decidable in polynomial time even if the number $t$ of types is variable. The algorithm underlying the proof uses only classical $n$-fold integer programming and avoids the heavy results of [6] on integer cones used in [11].

Theorem 2.2 Let $A$ be a fixed totally unimodular matrix and consider the huge $n$-fold program over $A$ with variable number $t$ of types. Then it is decidable in time polynomial in $t$ and the binary encoding of $l^{k}, u^{k}, b^{k}, n_{k}$, if the set in (3) is nonempty.

Proof. Consider the following set of points of a classical $t$-fold integer program:

$$
\begin{equation*}
\left\{y \in \mathbb{Z}^{d t}: \sum_{k=1}^{t} y^{k}=b^{0}, A y^{k}=n_{k} b^{k}, n_{k} l^{k} \leq y^{k} \leq n_{k} u^{k}, k=1, \ldots, t\right\} \tag{4}
\end{equation*}
$$

We claim that (3) is nonempty if and only if (4) is nonempty, which can be decided within the claimed time complexity by Proposition 1.2 on classical $n$-fold theory.

So it remains to prove the claim. First, suppose $x$ is in (3). Define $y$ by setting $y^{k}:=\sum_{i \in I_{k}} x^{i}$ for $k=1, \ldots, t$. Then we have $\sum_{k=1}^{t} y^{k}=\sum_{i=1}^{n} x^{i}=b^{0}, A y^{k}=$ $A \sum_{i \in I_{k}} x^{i}=n_{k} b^{k}$, and $n_{k} l^{k} \leq y^{k}=\sum_{i \in I_{k}} x^{i} \leq n_{k} u^{k}$, so $y$ is in (4). Conversely, suppose $y$ is in (4). For $k=1, \ldots, t$ consider the symmetric $n_{k}$-fold program

$$
\left\{\left(x^{i}: i \in I_{k}\right) \in \mathbb{Z}^{d n_{k}}: \sum_{i \in I_{k}} x^{i}=y^{k}, A x^{i}=b^{k}, l^{k} \leq x^{i} \leq u^{k}, i \in I_{k}\right\}
$$

Since $y$ is in (4) we have that $A y^{k}=n_{k} b^{k}$ and $n_{k} l^{k} \leq y^{k} \leq n_{k} u^{k}$. Therefore, by Lemma 2.1, this program is feasible and has a solution $\left(x^{i}: i \in I_{k}\right)$. Let $x=\left(x^{1}, \ldots, x^{n}\right)$ be obtained by combining the solutions of these $t$ programs. Then we have $\sum_{i=1}^{n} x^{i}=\sum_{k=1}^{t} y^{k}=b^{0}$ and $A x^{i}=b^{k}$ and $l^{k} \leq x^{i} \leq u^{k}$ for $k=1, \ldots, t$ and $i \in I_{k}$, so $x$ is in (3). This completes the proof of the claim and the theorem.

## 3 Optimization

In this section we consider the optimization problem for huge $n$-fold programs:

$$
\min \left\{\sum_{k=1}^{t} \sum_{i \in I_{k}} w^{k} x^{i}: x \in \mathbb{Z}^{d n}, \sum_{i=1}^{n} x^{i}=b^{0}, A x^{i}=b^{k}, l^{k} \leq x^{i} \leq u^{k}, k=1, \ldots, t, i \in I_{k}\right\} .
$$

The optimization problem is harder than the feasibility problem in that we need to actually produce an optimal solution if one exists. Since the problem is huge, meaning that $n$ is encoded in binary, we cannot explicitly even write down a single point $x \in \mathbb{Z}^{d n}$ in polynomial time. But it turns out that we can present $x$ compactly as follows. For $k=1, \ldots, t$ the set of all possible bricks of type $k$ is the following

$$
S^{k}:=\left\{z \in \mathbb{Z}^{d}: A z=b^{k}, l^{k} \leq z \leq u^{k}\right\}
$$

We assume for simplicity that $S^{k}$ is finite for all $k$, which is the case in most applications, such as in multiway table problems. Let $\lambda^{k}:=\left(\lambda_{z}^{k}: z \in S^{k}\right)$ be a nonnegative integer tuple with entries indexed by points of $S^{k}$. Each feasible point $x=\left(x^{1}, \ldots, x^{n}\right)$ gives rise to $\lambda^{1}, \ldots, \lambda^{t}$ satisfying $\sum\left\{\lambda_{z}^{k}: z \in S^{k}\right\}=n_{k}$, where
$\lambda_{z}^{k}$ is the number of bricks of $x$ of type $k$ which are equal to $z$. Let the support of $\lambda^{k}$ be $\operatorname{supp}\left(\lambda^{k}\right):=\left\{z \in S^{k}: \lambda_{z}^{k} \neq 0\right\}$. Then a compact presentation of $x$ consists of the restrictions of $\lambda^{k}$ to $\operatorname{supp}\left(\lambda^{k}\right)$ for all $k$. While the cardinality of $S^{k}$ may be exponential in the binary encoding of the data $b^{k}, l^{k}, u^{k}$, it turns out that a compact presentation of polynomial size always exists. The following theorem was shown in [11] using the recent computationally heavy algorithm of [6] which builds on [5].

Proposition 3.1 For fixed $d$ and $t$, the huge $n$-fold integer optimization problem with $t$ types, over an $s \times d$ matrix $A$ which is part of the input, can be solved in polynomial time. That is, in time polynomial in the binary encoding of $A, l^{k}, u^{k}, b^{k}, n_{k}$, it can either be asserted that the problem is infeasible, or a compact presentation $\lambda^{1}, \ldots, \lambda^{t}$ of an optimal solution with $\left|\operatorname{supp}\left(\lambda^{k}\right)\right| \leq 2^{d}$ for $k=1, \ldots, t$ be computed.

We now show that for a totally unimodular matrix, we can solve the huge problem even for variable $t$, extending both the above result and classical $n$-fold theory.
Theorem 1.3 Let $A$ be a fixed totally unimodular matrix and consider the huge $n$-fold program over $A$ with a variable number t of types. Then the optimization problem can be solved in time polynomial in $t$ and the binary encoding of $w^{k}, l^{k}, u^{k}, b^{k}, n_{k}$.

Proof. Consider the following classical $t$-fold integer optimization problem:
$\min \left\{\sum_{k=1}^{t} w^{k} y^{k}: y \in \mathbb{Z}^{d t}, \sum_{k=1}^{t} y^{k}=b^{0}, A y^{k}=n_{k} b^{k}, n_{k} l^{k} \leq y^{k} \leq n_{k} u^{k}, k=1, \ldots, t\right\}$.
By Proposition 1.2 on classical $n$-fold theory we can either assert the problem is infeasible, or obtain an optimal solution $y$, within the claimed time complexity. As shown in the proof of Theorem[2.2, if this problem is infeasible, then so is the original program, and we are done. So assume we have obtained an optimal solution $y$.

For $k=1, \ldots, t$ consider the symmetric $n_{k}$-fold program
$\min \left\{\sum_{i \in I_{k}} w^{k} x^{i}:\left(x^{i}: i \in I_{k}\right) \in \mathbb{Z}^{d n_{k}}, \sum_{i \in I_{k}} x^{i}=y^{k}, A x^{i}=b^{k}, l^{k} \leq x^{i} \leq u^{k}, i \in I_{k}\right\}$.
As shown in the proof of Theorem 2.2, this program is feasible. Since this is a huge symmetric program, that is, with a single type, by Proposition 3.1 we can compute in polynomial time a compact presentation $\lambda^{k}$ with $\left|\operatorname{supp}\left(\lambda^{k}\right)\right| \leq 2^{d}$ of an optimal solution $\left(x^{i}: i \in I_{k}\right) \in \mathbb{Z}^{d n_{k}}$. (In fact, any point in that program has the same objective function value $\sum_{i \in I_{k}} w^{k} x^{i}=w^{k} y^{k}$ and is optimal to that program.) Then $\lambda^{1}, \ldots, \lambda^{t}$ obtained from all these programs provide a compact presentation of a point $x=\left(x^{1}, \ldots, x^{n}\right)$ feasible in the original program. We claim this $x$ is optimal. Suppose indirectly there is a better point $\bar{x}$ and define $\bar{y}$ by $\bar{y}^{k}=\sum_{i \in I_{k}} \bar{x}^{i}$ for all $k$.

Then we have

$$
\sum_{k=1}^{t} w^{k} \bar{y}^{k}=\sum_{k=1}^{t} \sum_{i \in I_{k}} w^{k} \bar{x}^{i}<\sum_{k=1}^{t} \sum_{i \in I_{k}} w^{k} x^{i}=\sum_{k=1}^{t} w^{k} y^{k}
$$

contradicting the optimality of $y$ in the $t$-fold program. So indeed $\lambda^{1}, \ldots, \lambda^{t}$ provide a compact presentation of an optimal solution of the given huge $n$-fold program.

We make the following remark. The algorithm of Theorem 2.2 for the feasibility problem involves only one application of the classical $n$-fold integer programming algorithm of Proposition 1.2. In contrast, the algorithm of Theorem 1.3 is much heavier, and in addition to one application of classical $n$-fold integer programming, uses $t$ times the algorithm of Proposition 3.1 for huge $n$-fold integer programming with one type, which in turn uses the heavy algorithm for integer cones of [6].

## 4 Tables

We now return to tables. Consider first 3-way $l \times m \times n$ tables. Index each table as $x=\left(x^{1}, \ldots, x^{n}\right)$ with $x^{i}=\left(x_{1,1}^{i}, \ldots, x_{l, m}^{i}\right)$. Then the table problem is the $n$-fold program with matrix $A_{l, m}^{[n]}$ with $A_{l, m}$ the vertex-edge incidence matrix of the bipartite graph $K_{l, m}$. Indeed, then $\sum_{i=1}^{n} x^{i}=g$ provides the vertical line-sum equations, and $A_{l, m} x^{i}=b^{k}$ with $b^{k}=\left(e^{k}, f^{k}\right)$ provides the column and row sum equations for $i \in I_{k}$. Since $A_{l, m}$ is totally unimodular, Theorem 1.3 implies our following claimed result.
Theorem 1.1 The huge 3-way table problem with a variable number $t$ of types can be solved in time which is polynomial in $t$ and in the binary encoding of $w^{k}, e^{k}, f^{k}, g, n_{k}$. In particular, deciding if there is a huge table with variable number of types is in $P$.

Let us continue with 4 -way $k \times l \times m \times n$ tables. Index each table as $x=\left(x^{1}, \ldots, x^{n}\right)$ with each $x^{k}$ an $k \times l \times m$ layer. Then the table problem is the $n$-fold program with matrix $A^{[n]}$ where $A=A_{k, l}^{[m]}$. Now, unfortunately, for $k, l, m \geq 3$, the matrix $A$ is not totally unimodular. Therefore, the results of the previous sections do not apply, and we remain with the results of [11], which are as follows.

Proposition 4.1 The huge 4-way table problem with fixed number $t$ of types is solvable in time polynomial in the binary encoding of $w^{k}, n_{k}$ and the line sums. Moreover, deciding if there is a huge table with $t$ variable is in NP intersect coNP.

The contrast between Theorem 1.1 and Proposition 4.1 motivates the following.
Open problem. What is the complexity of deciding feasibility of huge 4 -way tables with a variable number of types ? In particular, is it in P for $3 \times 3 \times 3 \times n$ tables ?

## Acknowledgments

The research of the first author was partially supported by a VPR Grant at the Technion and by the Fund for the Promotion of Research at the Technion. The research of the second author was partially supported by a Bourse d'Aide à la Mobilité Internationale from Université Paris Est.

## References

[1] Aoki, S., Takemura, A.: Minimal basis for connected Markov chain over $3 \times 3 \times K$ contingency tables with fixed two-dimensional marginals. Australian and New Zealand Journal of Statistics 45:229-249 (2003)
[2] De Loera, J., Hemmecke, R., Onn, S., Weismantel, R.: N-fold integer programming. Discrete Optimization 5:231-241 (2008)
[3] De Loera, J., Onn, S.: The complexity of three-way statistical tables. SIAM Journal on Computing 33:819-836 (2004)
[4] De Loera, J., Onn, S.: All linear and integer programs are slim 3-way transportation programs. SIAM Journal on Optimization 17:806-821 (2006)
[5] Eisenbrand, F., Shmonin, G.: Carathéodory bounds for integer cones. Operations Research Letters 34:564-568 (2006)
[6] Goemans, M.X., Rothvoß, T.: Polynomiality for Bin Packing with a Constant Number of Item Types. Symposium on Discrete Algorithms 25:830-839 (2014)
[7] Hemmecke, R., Onn, S., Romanchuk, L.: N-fold integer programming in cubic time. Mathematical Programming 137:325-341 (2013)
[8] Hemmecke, R., Onn, S., Weismantel, R.: A polynomial oracle-time algorithm for convex integer minimization. Mathematical Programming 126:97-117 (2011)
[9] Hoşten, S., Sullivant, S.: Finiteness theorems for Markov bases of hierarchical models. Journal of Combinatorial Theory Series A 114:311-321 (2007)
[10] Onn, S.: Nonlinear Discrete Optimization. Zurich Lectures in Advanced Mathematics, European Mathematical Society (2010), available online at: http://ie.technion.ac.il/~onn/Book/NDO.pdf
[11] Onn, S.: Huge multiway table problems. Discrete Optimization 14:72-77 (2014)
[12] Santos, F., Sturmfels, B.: Higher Lawrence configurations. Journal of Combinatorial Theory Series A 103:151-164 (2003)


[^0]:    *Technion - Israel Institute of Technology, Haifa, Israel. Email: onn@ie.technion.ac.il
    ${ }^{\dagger}$ Ecole des Ponts, Paris, and Technion, Haifa. Email: pauline.sarrabezolles@gmail.com

