Optim Lett (2011) 5:13–25
DOI 10.1007/s11590-010-0231-9

ORIGINAL PAPER

N-fold integer programming and nonlinear multi-transshipment

Raymond Hemmecke · Shmuel Onn · Robert Weismantel

Received: 12 May 2010 / Accepted: 25 August 2010 / Published online: 7 September 2010

Abstract The multi-transshipment problem is NP-hard already for two commodities over bipartite networks. Nonetheless, using our recent theory of n-fold integer programming and extensions developed herein, we are able to establish the polynomial time solvability of the problem in two broad situations. First, for any fixed number of commodities and number of suppliers, we solve the problem over bipartite networks with variable number of consumers in polynomial time. This is very natural in operations research applications where few facilities serve many customers. Second, for every fixed network, we solve the problem with variable number of commodities in polynomial time.

Keywords Integer programming · Transportation problem · Multicommodity flow · Combinatorial optimization · Nonlinear optimization · Convex optimization · Graver basis · N-fold product · Discrete optimization

1 Introduction

The multi-transshipment problem is a very broad problem which seeks minimum cost routing of several discrete commodities over a digraph subject to vertex demand and edge capacity constraints. The data for the problem is as follows. There is a digraph

R. Hemmecke
Technische Universität Munich, Munich, Germany

S. Onn (✉)
Technion, Israel Institute of Technology, Haifa, Israel
e-mail: onn@ie.technion.ac.il

R. Weismantel
ETH Zürich, Zurich, Switzerland

G with s vertices and t edges. There are l types of commodities. Each commodity has a demand vector $d^k \in \mathbb{Z}^s$ with d^k_v the demand for commodity k at vertex v (interpreted as supply when negative and consumption when positive). Each edge e has a capacity u_e (upper bound on the combined flow of all commodities on it). A *multi-transshipment* is a vector $x = (x^1, \ldots, x^l)$ with $x^k \in \mathbb{Z}^t_+$ for all k and x^k_e the flow of commodity k on edge e, satisfying the capacity constraint $\sum_{k=1}^{l} x^k_e \le u_e$ for each edge e and demand constraint $\sum_{e\in\delta^+(v)} x^k_e - \sum_{e\in\delta^-(v)} x^k_e = d^k_v$ for each vertex v and commodity k (with $\delta^+(v)$, $\delta^-(v)$ the sets of edges entering and leaving v). The cost of transshipment x is defined as follows. There are cost functions $f_e, g^k_e : \mathbb{Z} \to \mathbb{Z}$ for each edge and each edge-commodity pair. The transshipment cost on edge e is $f_e(\sum_{k=1}^{l} x^k_e) + \sum_{k=1}^{l} g^k_e(x^k_e)$ with the first term being the value of f_e on the combined flow of all commodities on e and the second term being the sum of costs that depend on both the edge and the commodity. The total cost, to be minimized, is

$$\sum_{e=1}^{t}\left(f_e\left(\sum_{k=1}^{l} x^k_e\right) + \sum_{k=1}^{l} g^k_e(x^k_e)\right).$$

Our results apply to cost functions which can be standard linear or convex such as $\alpha_e|\sum_{k=1}^{l} x^k_e|^{\beta_e} + \sum_{k=1}^{l} \gamma^k_e |x^k_e|^{\delta^k_e}$ for some nonnegative integers α_e, β_e, γ^k_e, δ^k_e, which take into account the increase in cost due to channel congestion when subject to heavy traffic or communication load (with the linear case obtained by $\beta_e = \delta^k_e = 1$).

A particularly useful special case of this problem is the *multi-transportation* problem, where the digraph is bipartite oriented from suppliers to consumers. But even deciding if a feasible multi-transportation exists is NP-complete already in the following two very special cases: first, with only $l = 2$ commodities over the complete bipartite digraphs $K_{m,n}$ [5,6]; and second, with variable number of commodities over the digraphs $K_{3,n}$ with only $m = 3$ suppliers on one side (see Sect. 4).

Yet, using the theory of n-fold integer programming recently introduced in [3,4,9] and extensions developed herein, we are able to establish the polynomial time solvability of these problems, with either standard linear costs or more general costs with *nonlinear convex* functions f_e, g^k_e, in two situations as follows.

First, for any fixed number l of commodities, we solve the multi-transportation problem over any bipartite subdigraph of $K_{m,n}$ with fixed number m of suppliers and variable number n of consumers in polynomial time. This is very natural in operations research applications where few facilities serve many customers. Here each commodity type k may have its own volume v_k per unit. Note again that if l is variable then the problem is NP-hard already for $m = 3$, so the following, main result of this article, is best possible (see Sect. 3 for the precise statement).

Theorem 1.1 *For any fixed l commodities and m suppliers, the (convex) multi-transportation problem with variable n consumers can be solved in polynomial time.*

Second, over any fixed digraph, we can solve the multi-transshipment problem with a variable number l of commodities (hence termed the *many-transshipment* problem). This problem may seem at a first glance very restricted: however, even for the single tiny bipartite digraph $K_{3,3}$, we are not aware as of yet of any solution method other

than the one provided herein; and as noted, the problem is NP-hard for the digraphs $K_{3,n}$. Our second result is the following (see Sect. 3 for the precise statement).

Theorem 1.2 *For any fixed digraph G, the (convex) many-transshipment problem with variable number l of commodities over G can be solved in polynomial time.*

We also point out the following immediate corollary of Theorem 1.2.

Corollary 1.3 *For any fixed s, the (convex) many-transshipment problem with variable number l of commodities on any s-vertex digraph is polynomial time solvable.*

The complexity of the algorithm of Theorem 1.2 involves a term of $O\left(l^{g(G)}\right)$ where $g(G)$ is the *Graver complexity* of G, a fascinating new digraph invariant about which very little is known (even $g(K_{3,4})$ is as yet unknown), see discussion in Sect. 4.

We point out that the running time of our algorithms depends naturally on the *binary length* $\langle d_v^k, u_e \rangle$ of the numerical part of the data consisting of the demands and capacities (see Sect. 3), so our algorithms can handle very large numbers. To get such polynomial running time *even* in the much more limited situation when *both* the digraph *and* the number of commodities are fixed (where the number lt of variables becomes fixed) and where the cost functions are linear, one needs off-hand the algorithm of integer programming in fixed dimension [13]. However, Theorems 1.1 and 1.2 involve variable dimension and [13] does not apply.

We remark that multicommodity flows have been studied extensively in the literature with different emphasis. Let us mention in particular the important work of [12] which investigates max-flow min-cut properties and their use in approximation algorithms for multicommodity flow problems, and the references therein.

In Sect. 2 we briefly review the necessary background on n-fold integer programming, and establish a new theorem enabling the solvability of a generalized class of n-fold integer programs. In Sect. 3 we use the results of Sect. 2 and prove Theorem 1.1 and Theorem 1.2. We conclude in Sect. 4 with a short discussion.

2 N-fold integer programming

Linear integer programming is the following fundamental optimization problem,

$$\min \left\{ wx \ : \ x \in \mathbb{Z}^n \, , \ Ax = b \, , \ l \leq x \leq u \right\} , \tag{1}$$

where A is an integer $m \times n$ matrix, $b \in \mathbb{Z}^m$, and $l, u \in \mathbb{Z}_\infty^n$ with $\mathbb{Z}_\infty := \mathbb{Z} \uplus \{\pm\infty\}$. It is generally NP-hard, but polynomial time solvable in two fundamental situations: the dimension is fixed [13]; the underlying matrix is totally unimodular [10].

Recently, in [3], a new broad polynomial time solvable situation was discovered. We proceed to describe this class of so-termed *n-fold integer programs*.

An $(r, s) \times t$ *bimatrix* is a matrix A consisting of two blocks A_1, A_2, with A_1 its $r \times t$ submatrix consisting of the first r rows and A_2 its $s \times t$ submatrix consisting of

the last s rows. The *n-fold product* of A is the following $(r+ns)\times nt$ matrix,

$$A^{(n)} := \begin{pmatrix} A_1 & A_1 & \cdots & A_1 \\ A_2 & 0 & \cdots & 0 \\ 0 & A_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & A_2 \end{pmatrix}.$$

A function f is *separable convex* if $f(x)=\sum_i f_i(x_i)$ with each f_i univariate convex. In particular, any linear function $wx=\sum_i w_i x_i$ is separable convex. We denote by $\hat{f}$ the supremum of $|f(x)|$ over the feasible set. The following result of [9] asserts that separable convex n-fold integer programs are efficiently solvable. Throughout, as is customary (see e.g. [15]), we denote by $\langle a,b,c,\ldots\rangle$ the *binary length* of a finite list $a,b,c,\ldots$ of integer or rational numbers, vectors, matrices, or finite sets of such objects forming the input to an algorithm, that is, the number of bits in the binary encoding (with the binary length of any $\pm\infty$ entry being constant).

Theorem 2.1 [9] *For every fixed integer $(r,s)\times t$ bimatrix A, there is an algorithm that, given n, lower and upper bounds $l,u\in\mathbb{Z}^{nt}_{\infty}$, $b\in\mathbb{Z}^{r+ns}$, and separable convex function $f:\mathbb{Z}^{nt}\to\mathbb{Z}$ presented by an evaluation oracle, solves in time which is polynomial in n and $\langle l,u,b,\hat{f}\rangle$ the convex n-fold integer minimization problem*

$$\min\left\{f(x)\ :\ x\in\mathbb{Z}^{nt}\,,\ A^{(n)}x=b\,,\ l\le x\le u\right\}.$$

Some explanatory notes are in order. First, the dimension of an n-fold integer program is nt and is variable. Second, n-fold products $A^{(n)}$ are highly non totally unimodular: the n-fold product of the simple $(0,1)\times 1$ bimatrix with A_1 empty and $A_2:=2$ satisfies $A^{(n)}=2I_n$ and has exponential determinant 2^n. So this is indeed a class of programs which cannot be solved by methods of fixed dimension or totally unimodular matrices. Third, this class of programs turns out to be very natural and has numerous applications, the most generic being to integer optimization over multidimensional tables. In fact it is *universal*: the results of [6] imply that *any* integer program *is* an n-fold program over some simple bimatrix A, see Sect. 4.

We now provide a broad generalization of Theorem 2.1 which will be useful for the multi-transshipment applications to follow and is interesting on its own right.

We need to review some material from [3,9]. We make use of a partial order $\sqsubseteq$ on $\mathbb{R}^n$ defined as follows. For two vectors $x,y\in\mathbb{R}^n$ we write $x\sqsubseteq y$ if $x_i y_i\ge 0$ and $|x_i|\le|y_i|$ for $i=1,\ldots,n$, that is, x and y lie in the same orthant of $\mathbb{R}^n$ and each component of x is bounded by the corresponding component of y in absolute value. Thus, the order $\sqsubseteq$ is the natural extension of the standard coordinate-wise partial order $\le$ on the nonnegative orthant $\mathbb{R}^n_+$ to the entire space $\mathbb{R}^n$. A classical lemma of Gordan implies that every subset $Z\subseteq\mathbb{Z}^n$ has finitely-many $\sqsubseteq$-minimal elements, that is, elements $x\in Z$ such that no other $y\in Z$ satisfies $y\sqsubseteq x$.

The following fundamental object was introduced in [8].

Definition 2.2 The *Graver basis* of an integer matrix A is defined to be the finite set $\mathcal{G}(A) \subset \mathbb{Z}^n$ of $\sqsubseteq$-minimal elements in $\{x \in \mathbb{Z}^n \,:\, Ax = 0,\ x \neq 0\}$.

For instance, the Graver basis of the 1×3 matrix $A := (1\ \ 2\ \ 1)$ consists of 8 vectors,

$$\mathcal{G}(A) \ = \ \pm\{(2,-1,0),(0,-1,2),(1,0,-1),(1,-1,1)\};$$

note that the first six elements are so-called *circuits* of the matrix A, that is, primitive linear dependencies on its columns, but the last two, $\pm(1,-1,1)$, are not.

The following lemma from [9] demonstrates the usefulness of the Graver basis.

Lemma 2.3 *There is an algorithm that, given an integer $m \times n$ matrix A, its Graver basis $\mathcal{G}(A)$, $l, u \in \mathbb{Z}^n_\infty$, $b \in \mathbb{Z}^m$, and separable convex function $f : \mathbb{Z}^n \to \mathbb{Z}$ presented by an evaluation oracle, solves in time polynomial in $\langle A, \mathcal{G}(A), l, u, b, \hat{f}\rangle$, the program*

$$\min\{f(x) \,:\, x \in \mathbb{Z}^n\,,\ Ax = b\,,\ l \leq x \leq u\}. \tag{2}$$

Outline of proof and algorithm: First, assume that we have an initial feasible point x to begin with. Then we iterate the following procedure as long as it provides a better point: for each Graver direction $g \in \mathcal{G}(A)$, we solve the univariate problem

$$\min\{f(x+\lambda g) \,:\, \lambda \in \mathbb{Z}_+\,,\ l \leq x + \lambda g \leq u\} \tag{3}$$

of finding a best step size λ and update the feasible point to the best point $x := x + \lambda g$ among all Graver directions. It is shown in [9] that at each iteration, problem (3) can be solved in polynomial time, and that after polynomially many iterations, an optimal solution of problem (2) is obtained. Second, it is shown in [9] that by applying the same procedure to a suitable auxiliary separable convex integer program, an initial feasible point can be obtained. See [9] for further details.

One may wonder whether Lemma 2.3 can be extended to any convex function. Unfortunately, as we next show, it is unlikely even for convex quadratics of rank 1.

Proposition 2.4 *It is NP-hard to even determine the optimal value of the problem*

$$\min\left\{(vx - v_0)^2 \,:\, x \in \mathbb{Z}^n,\ Ax = b,\ l \leq x \leq u\right\} \tag{4}$$

with convex quadratic of rank 1*, where $v \in \mathbb{Z}^n$ and $v_0 \in \mathbb{Z}$, even when $\mathcal{G}(A)$ is given.*

Proof Let $v \in \mathbb{Z}^n_+$ and $v_0 \in \mathbb{Z}_+$ be input to the *subset sum* problem of deciding if there exists $x \in \{0,1\}^n$ with $vx = v_0$. Let $A := 0$ be the zero $1 \times n$ matrix, whose Graver basis $\mathcal{G}(A) = \{\pm\mathbf{1}_i \,:\, i = 1,\ldots,n\}$ consists of the n unit vectors and their negations. Let $b_i := 0, l_j := 0$ and $u_j := 1$ for all i and j. Then problem (4) becomes

$$\min\left\{(vx - v_0)^2 \,:\, x \in \{0,1\}^n\right\},$$

whose optimal value is 0 if and only if there is a subset sum, proving the claim. □

The Graver basis is typically exponential and cannot be written down, let alone computed, in polynomial time. However, as the next lemma from [3] demonstrates, Graver bases of n-fold products are better behaved.

Lemma 2.5 *For every fixed integer bimatrix A there is an algorithm that, given n, obtains the Graver basis $\mathcal{G}(A^{(n)})$ of the n-fold product of A in time polynomial in n.*

Outline of proof and algorithm: Let A be any integer $(r, s) \times t$ bimatrix. We naturally index elements x in the Graver basis $\mathcal{G}(A^{(n)})$ as $x = (x^1, \ldots, x^n)$ and call each $x^k \in \mathbb{Z}^t$ a *brick* of x. It has been shown in [1,11,14] (in increasing generality) that there is a finite nonnegative integer $g(A)$, the so-termed *Graver complexity* of the bimatrix A, such that for any n, any element in $\mathcal{G}(A^{(n)})$ has at most $g(A)$ nonzero bricks independent of n. This is used in [3] to compute the Graver basis $\mathcal{G}(A^{(n)})$ in polynomial time $O(n^{g(A)})$ as follows. Let $\mathcal{G}(A^{(g(A))})$ be the Graver basis of the $g(A)$-fold product of A. Note that it has constant size so can be built into the algorithm. Now, from each element $h \in \mathcal{G}(A^{(g(A))}) \subset \mathbb{Z}^{g(A)t}$ create $\binom{n}{g(A)}$ elements $g \in \mathbb{Z}^{nt}$ by padding h with $n - g(A)$ additional zero bricks in all possible ways. The resulting multiset of $|\mathcal{G}(A^{(g(A))})|\binom{n}{g(A)}$ elements contains $\mathcal{G}(A^{(n)})$. The latter can be obtained by removing all non $\sqsubseteq$-minimal elements. See [3] for further details.

Lemmas 2.3 and 2.5 together imply Theorem 2.1 mentioned above. In particular, the algorithm of Theorem 2.1 incorporates the computation of $\mathcal{G}(A^{(n)})$ by the algorithm of Lemma 2.5 which take $O(n^{g(A)})$ time. So the Graver complexity of A affects the exponent of the running time. See further discussion in Sect. 4.

We proceed with two new lemmas needed in the proof of our generalized theorem. In the next lemma, as before, $\hat{f}$ and $\hat{g}$ denote the supremum values of $|f(Wx)|$ and $|g(x)|$ over the feasible set. We also allow to incorporate inequalities on Wx in addition to the lower and upper bound inequalities on x.

Lemma 2.6 *There is an algorithm that, given an integer $m \times n$ matrix A, an integer $d \times n$ matrix $W, l, u \in \mathbb{Z}^n_\infty, \hat{l}, \hat{u} \in \mathbb{Z}^d_\infty, b \in \mathbb{Z}^m$, the Graver basis $\mathcal{G}(B)$ of*

$$B := \begin{pmatrix} A & 0 \\ W & I \end{pmatrix},$$

and separable convex functions $f : \mathbb{Z}^d \to \mathbb{Z}, g : \mathbb{Z}^n \to \mathbb{Z}$ presented by evaluation oracles, solves in time polynomial in $\langle A, W, \mathcal{G}(B), l, u, \hat{l}, \hat{u}, b, \hat{f}, \hat{g}\rangle$, the program

$$\min\{f(Wx) + g(x) \ : \ x \in \mathbb{Z}^n, \ Ax = b\,, \ \hat{l} \le Wx \le \hat{u}, \ l \le x \le u\}.$$

Proof Define $h : \mathbb{Z}^{n+d} \to \mathbb{Z}$ by $h(x, y) := f(-y) + g(x)$ for all $x \in \mathbb{Z}^n$ and $y \in \mathbb{Z}^d$. Clearly, h is separable convex since f, g are. Now, our problem can be rewritten as

$$\min\left\{h(x, y) : (x, y) \in \mathbb{Z}^{n+d}, \ \begin{pmatrix} A & 0 \\ W & I \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} b \\ 0 \end{pmatrix}, \ l \le x \le u, -\hat{u} \le y \le -\hat{l}\right\},$$

and the statement follows at once by applying Lemma 2.3 to this problem. □

Lemma 2.7 *For every fixed integer $(r, s) \times t$ bimatrix A and $(p, q) \times t$ bimatrix W, there is an algorithm that, given any positive integer n, computes in time polynomial in n, the Graver basis $\mathcal{G}(B)$ of the following $(r + ns + p + nq) \times (nt + p + nq)$ matrix,*

$$B := \begin{pmatrix} A^{(n)} & 0 \\ W^{(n)} & I \end{pmatrix}.$$

Proof Let D be the $(r + p, s + q) \times (t + p + q)$ bimatrix whose blocks are defined by

$$D_1 := \begin{pmatrix} A_1 & 0 & 0 \\ W_1 & I_p & 0 \end{pmatrix}, \quad D_2 := \begin{pmatrix} A_2 & 0 & 0 \\ W_2 & 0 & I_q \end{pmatrix}.$$

Apply the algorithm of Lemma 2.5 and compute in polynomial time the Graver basis $\mathcal{G}(D^{(n)})$ of the n-fold product of D, which is the following matrix:

$$D^{(n)} = \left(\begin{array}{ccc|ccc|c|ccc} A_1 & 0 & 0 & A_1 & 0 & 0 & \cdots & A_1 & 0 & 0 \\ W_1 & I_p & 0 & W_1 & I_p & 0 & \cdots & W_1 & I_p & 0 \\ \hline A_2 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ W_2 & 0 & I_q & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & A_2 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & W_2 & 0 & I_q & \cdots & 0 & 0 & 0 \\ \hline \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & \cdots & A_2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \cdots & W_2 & 0 & I_q \end{array}\right).$$

Suitable row and column permutations applied to $D^{(n)}$ give the following matrix:

$$C := \left(\begin{array}{cccc|cccc|ccc} A_1 & A_1 & \cdots & A_1 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ A_2 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ 0 & A_2 & \cdots & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & A_2 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ \hline W_1 & W_1 & \cdots & W_1 & I_p & I_p & \cdots & I_p & 0 & 0 & \cdots & 0 \\ W_2 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & I_q & 0 & \cdots & 0 \\ 0 & W_2 & \cdots & 0 & 0 & 0 & \cdots & 0 & 0 & I_q & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & W_2 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & I_q \end{array}\right).$$

Obtain the Graver basis $\mathcal{G}(C)$ in polynomial time from $\mathcal{G}(D^{(n)})$ by permuting the entries of each element of the latter by the permutation of the columns of $\mathcal{G}(D^{(n)})$ that is used to get C (the permutation of the rows does not affect the Graver basis).

Now, note that the matrix B can be obtained from C by dropping all but the first p columns in the second block. Consider any element in $\mathcal{G}(C)$, indexed, according to the block structure, as $(x^1, x^2, \ldots, x^n, y^1, y^2, \ldots, y^n, z^1, z^2, \ldots, z^n)$. Clearly, if $y^k = 0$ for $k = 2, \ldots, n$ then the restriction $(x^1, x^2, \ldots, x^n, y^1, z^1, z^2, \ldots, z^n)$ of this element is in the Graver basis of B. On the other hand, if $(x^1, x^2, \ldots, x^n, y^1, z^1, z^2, \ldots, z^n)$ is any element in $\mathcal{G}(B)$ then its extension $(x^1, x^2, \ldots, x^n, y^1, 0, \ldots, 0, z^1, z^2, \ldots, z^n)$ is clearly in $\mathcal{G}(C)$. So the Graver basis of B can be obtained in polynomial time by

$$\mathcal{G}(B) := \Big\{(x^1, \ldots, x^n, y^1, z^1, \ldots, z^n) : (x^1, \ldots, x^n, y^1, 0, \ldots, 0, z^1, \ldots, z^n) \in \mathcal{G}(C)\Big\}.$$

□

We can now provide our new theorem on generalized n-fold integer programming. As before, $\hat{f}, \hat{g}$ denote the supremum values of $|f(Wx)|, |g(x)|$ over the feasible set.

Theorem 2.8 *For every fixed integer $(r, s) \times t$ bimatrix A and integer $(p, q) \times t$ bimatrix W, there is an algorithm that, given $n, l, u \in \mathbb{Z}_\infty^{nt}, \hat{l}, \hat{u} \in \mathbb{Z}_\infty^{p+nq}, b \in \mathbb{Z}^{r+ns}$, and separable convex functions $f : \mathbb{Z}^{p+nq} \to \mathbb{Z}, g : \mathbb{Z}^{nt} \to \mathbb{Z}$ presented by evaluation oracles, solves in time polynomial in n and $\langle l, u, \hat{l}, \hat{u}, b, \hat{f}, \hat{g}\rangle$, the generalized problem*

$$\min\Big\{f(W^{(n)}x) + g(x) \ : \ x \in \mathbb{Z}^{nt}\,,\ A^{(n)}x = b\,,\ \hat{l} \le W^{(n)}x \le \hat{u}\,,\ l \le x \le u\Big\}.$$

Proof First use the algorithm of Lemma 2.7 to compute the Graver basis $\mathcal{G}(B)$ of

$$B := \begin{pmatrix} A^{(n)} & 0 \\ W^{(n)} & I \end{pmatrix}.$$

Now use the algorithm of Lemma 2.6 to solve the problem in polynomial time. □

3 Nonlinear multi-transshipment

We begin with our main theorem, on nonlinear multi-transportation. We may and do assume that the underlying digraph is $K_{m,n}$ with edges oriented from suppliers to consumers (the problem over any subdigraph G of $K_{m,n}$ reduces to that over $K_{m,n}$ by simply forcing 0 capacity on all edges not present in G). It is convenient here to change the labeling of the data a little bit as follows. We now denote edges by pairs (i, j) where $1 \le i \le m$ is a supplier and $1 \le j \le n$ is a consumer. The demand vectors are now replaced by (nonnegative) supply and consumption vectors: each supplier i has a supply vector $s^i \in \mathbb{Z}_+^l$ with s_k^i its supply in commodity k, and each consumer j has a consumption vector $c^j \in \mathbb{Z}_+^l$ with c_k^j its consumption in commodity k. In addition, each commodity k may have its own volume $v_k \in \mathbb{Z}_+$

per unit flow. A multi-transportation is now indexed as $x = (x^1, \ldots, x^n)$ with $x^j = (x^j_{1,1}, \ldots, x^j_{1,l}, \ldots, x^j_{m,1}, \ldots, x^j_{m,l})$, where $x^j_{i,k}$ is the flow of commodity k from supplier i to consumer j. The capacity constraint on edge (i, j) is $\sum_{k=1}^{l} v_k x^j_{i,k} \le u_{i,j}$ and the cost is $f_{i,j}\left(\sum_{k=1}^{l} v_k x^j_{i,k}\right) + \sum_{k=1}^{l} g^j_{i,k}\left(x^j_{i,k}\right)$ with $f_{i,j}, g^j_{i,k} : \mathbb{Z} \to \mathbb{Z}$ convex. As before, $\hat{f}, \hat{g}$ denote the maximum absolute values of f, g over the feasible set. It is usually easy to determine an upper bound on these values from the problem data (for instance, in the special case of linear cost functions f, g, bounds which are polynomial in the binary length of the data readily follow from Cramer's rule).

Theorem 1.1 *For any fixed l commodities, m suppliers, and volumes $v_k \in \mathbb{Z}_+$, there is an algorithm that, given n consumers, supplies and demands $s^i, c^j \in \mathbb{Z}^l_+$, capacities $u_{i,j} \in \mathbb{Z}_+$, and convex costs $f_{i,j}, g^j_{i,k} : \mathbb{Z} \to \mathbb{Z}$ presented by evaluation oracles, solves in time polynomial in n and $\langle s^i, c^j, u, \hat{f}, \hat{g}\rangle$, the nonlinear multi-transportation problem*

$$\min \sum_{i,j}\left(f_{i,j}\left(\sum_{k} v_k x^j_{i,k}\right) + \sum_{k=1}^{l} g^j_{i,k}\left(x^j_{i,k}\right)\right)$$

$$\text{s.t. } x^j_{i,k} \in \mathbb{Z}\,,\ \sum_{j} x^j_{i,k} = s^i_k\,,\ \sum_{i} x^j_{i,k} = c^j_k\,,\ \sum_{k=1}^{l} v_k x^j_{i,k} \le u_{i,j}\,,\ x^j_{i,k} \ge 0.$$

Proof Construct bimatrices A and W as follows. Let D be the $(l, 0) \times l$ bimatrix with first block $D_1 := I_l$ and second block D_2 empty. Let V be the $(0, 1) \times l$ bimatrix with first block V_1 empty and second block $V_2 := (v_1, \ldots, v_l)$. Let A be the $(ml, l) \times ml$ bimatrix with first block $A_1 := I_{ml}$ and second block $A_2 := D^{(m)}$. Let W be the $(0, m) \times ml$ bimatrix with first block W_1 empty and second block $W_2 := V^{(m)}$. Let b be the $(ml + nl)$-vector $b := (s^1, \ldots, s^m, c^1, \ldots, c^n)$.

Let $f : \mathbb{Z}^{nm} \to \mathbb{Z}$ and $g : \mathbb{Z}^{nml} \to \mathbb{Z}$ be the separable convex functions defined by $f(y) := \sum_{i,j} f_{i,j}(y_{i,j})$ with $y_{i,j} := \sum_{k=1}^{l} v_k x^j_{i,k}$ and $g(x) := \sum_{i,j} \sum_{k=1}^{l} g^j_{i,k}(x^j_{i,k})$.

Now note that $A^{(n)}x$ is an $(ml + nl)$-vector, whose first ml entries are the flows from each supplier of each commodity to all consumers, and whose last nl entries are the flows to each consumer of each commodity from all suppliers. Therefore the supply and consumption equations are encoded by $A^{(n)}x = b$. Next note that the nm-vector $y = (y_{1,1}, \ldots, y_{m,1}, \ldots, y_{1,n}, \ldots, y_{m,n})$ satisfies $y = W^{(n)}x$. So the capacity constraints become $W^{(n)}x \le u$ and the cost function becomes $f(W^{(n)}x) + g(x)$. Therefore, the problem is precisely the following n-fold integer program,

$$\min\left\{f\left(W^{(n)}x\right) + g\,(x)\ :\ x \in \mathbb{Z}^{nml}\,,\ A^{(n)}x = b\,,\ W^{(n)}x \le u\,,\ x \ge 0\right\}.$$

By Theorem 2.8 this program can be solved in polynomial time as claimed. □

We proceed with our theorem on nonlinear many-transshipment. Again, $\hat{f}, \hat{g}$ are the maximum absolute values of the objective functions f, g over the feasible set.

Theorem 1.2 *For every fixed digraph G there is an algorithm that, given l commodity types, demand $d_v^k \in \mathbb{Z}$ for each commodity k and vertex v, edge capacities $u_e \in \mathbb{Z}_+$, and convex costs $f_e, g_e^k : \mathbb{Z} \to \mathbb{Z}$ presented by evaluation oracles, solves in time polynomial in l and $\langle d_v^k, u_e, \hat{f}, \hat{g}\rangle$, the nonlinear many-transshipment problem*

$$\min \sum_e \left(f_e\left(\sum_{k=1}^{l} x_e^k\right) + \sum_{k=1}^{l} g_e^k(x_e^k)\right)$$
$$\text{s.t. } x_e^k \in \mathbb{Z}\,, \quad \sum_{e\in\delta^+(v)} x_e^k - \sum_{e\in\delta^-(v)} x_e^k = d_v^k\,, \quad \sum_{k=1}^{l} x_e^k \le u_e\,, \quad x_e^k \ge 0.$$

Proof Assume G has s vertices and t edges and let D be its $s \times t$ vertex-edge incidence matrix. Let $f : \mathbb{Z}^t \to \mathbb{Z}$ and $g : \mathbb{Z}^{lt} \to \mathbb{Z}$ be the separable convex functions defined by $f(y) := \sum_{e=1}^{t} f_e(y_e)$ with $y_e := \sum_{k=1}^{l} x_e^k$ and $g(x) := \sum_{e=1}^{t}\sum_{k=1}^{l} g_e^k(x_e^k)$. Let $x = (x^1, \ldots, x^l)$ be the vector of variables with $x^k \in \mathbb{Z}^t$ the flow of commodity k for each k. Then the problem can be rewritten in vector form as

$$\min\left\{ f\left(\sum_{k=1}^{l} x^k\right) + g\,(x) \;:\; x \in \mathbb{Z}^{lt}\,,\; Dx^k = d^k\,,\; \sum_{k=1}^{l} x^k \le u\,,\; x \ge 0\right\}.$$

Let A be the $(0, s) \times t$ bimatrix with first block A_1 empty and second block $A_2 := D$. Let W be the $(t, 0) \times t$ bimatrix with first block $W_1 := I_t$ the $t \times t$ identity matrix and second block W_2 empty. Let $b := (d^1, \ldots, d^l)$. Then the problem is precisely the following l-fold integer program,

$$\min\left\{ f\left(W^{(l)}x\right) + g\,(x) \;:\; x \in \mathbb{Z}^{lt}\,,\; A^{(l)}x = b\,,\; W^{(l)}x \le u\,,\; x \ge 0\right\}.$$

By Theorem 2.8 this program can be solved in polynomial time as claimed. □

Here is a simple example of a nonlinear transshipment problem and its solution.

Example 3.1 Consider the following data. The digraph G has set of vertices $V = \{1, 2, 3, 4\}$ and set of (directed) edges $E = \{12, 13, 14, 23, 24, 34\}$. There are $l = 2$ commodities. Each of vertices $1, 2, 3$ supplies one unit of each commodity, that is, $d_v^k = -1$ for $v = 1, 2, 3$ and $k = 1, 2$, whereas vertex 4 consumes three units of each commodity, that is, $d_4^1 = d_4^2 = 3$. The capacity of each edge e is $u_e = 3$. The cost over edge 34 is linear $f_{34}(x_{34}^1 + x_{34}^2) = x_{34}^1 + x_{34}^2$, whereas the cost of every other edge e is quadratic $f_e(x_e^1 + x_e^2) = (x_e^1 + x_e^2)^2$. All costs g_e^k are zero. Following the algorithm of Theorem 1.2 incorporating the algorithm of Theorem 2.8, we need to construct the

Graver basis $\mathcal{G}(B)$ of the following matrix

$$B = \left(\begin{array}{cccccc|cccccc|cccccc}
-1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1
\end{array}\right).$$

The Graver basis $\mathcal{G}(B)$ has 186 elements in $\mathbb{Z}^{18}$ such as

$$(1,-1,0,0,1,-1,0,1,-1,0,0,1,-1,0,1,0,-1,0),$$
$$(1,0,-1,1,0,1,-1,-1,2,0,-1,-1,0,1,-1,-1,1,0).$$

We naturally index feasible points of the relevant integer program as

$$x = (x^1_{12}, x^1_{13}, x^1_{14}, x^1_{23}, x^1_{24}, x^1_{34}, x^2_{12}, x^2_{13}, x^2_{14}, x^2_{23}, x^2_{24}, x^2_{34}, y_{12}, y_{13}, y_{14}, y_{23}, y_{24}, y_{34}).$$

We start with the obvious natural feasible point x where each of vertices 1, 2, 3 sends one unit of each commodity directly to vertex 4, given by

$$x = (0,0,1,0,1,1,0,0,1,0,1,1,0,0,-2,0,-2,-2),$$

with cost $f(x) = (1+1)^2+(1+1)^2+(1+1) = 10$. Applying one step of the Graver iterative procedure outlined in Lemma 2.3 and solving (3), we obtain $\lambda = 1$ and

$$g = (0,1,-1,0,0,1,0,0,0,0,0,0,0,-1,1,0,0,-1),$$

and update

$$x := x + \lambda g = (0,1,0,0,1,2,0,0,1,0,1,1,0,-1,-1,0,-2,-3),$$

turning out to be the optimal solution with cost $f(x) = 1^2+1^2+(1+1)^2+(2+1) = 9$. So the optimal flows of commodities 1, 2 respectively are

$$x^1 = (x^1_{12}, x^1_{13}, x^1_{14}, x^1_{23}, x^1_{24}, x^1_{34}) = (0,1,0,0,1,2)$$
$$x^2 = (x^2_{12}, x^2_{13}, x^2_{14}, x^2_{23}, x^2_{24}, x^2_{34}) = (0,0,1,0,1,1).$$

4 Discussion

We conclude with a short discussion of the universality for integer programming of the multi-transshipment problem and of the complexity of our algorithms.

Consider the following special form of the n-fold product. For an integer $s \times t$ matrix D, let $D^{[n]} := A^{(n)}$ where A is the $(t, s) \times t$ bimatrix A with first block $A_1 := I_t$ the $t \times t$ identity matrix and second block $A_2 := D$. We consider such n-fold products of the 1×3 matrix $(1\ 1\ 1)$. Note that $(1\ 1\ 1)^{[n]}$ is precisely the $(3 + n) \times 3n$ incidence matrix of the complete bipartite graph $K_{3,n}$. For instance,

$$(1\ 1\ 1)^{[2]} = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}.$$

The following theorem was proved in [6] building on results of [5]. (For further details and consequences for privacy in statistical databases see [6,7].)

The Universality Theorem [6] *Every rational polytope* $\{x \in \mathbb{R}^d_+ : Bx = b\}$ *stands in polynomial time computable integer preserving bijection with some polytope*

$$\left\{ x \in \mathbb{R}^{3nl}_+ : (1\ 1\ 1)^{[n][l]} x = a \right\}. \tag{5}$$

In particular, every integer program can be lifted in polynomial time to a program over a matrix $(1\ 1\ 1)^{[n][l]}$ completely determined by two parameters n and l.

It follows from the proof of Theorem 1.2 (adding an extra slack commodity) that the integer points in (5) are precisely the feasible points of some l-commodity multi-transshipment problem over $K_{3,n}$. So any integer program can be lifted to an l-commodity transshipment problem over some $K_{3,n}$ with fixed $m = 3$ suppliers. Hence this problem is *universal* for integer programming and in particular NP-hard.

Our algorithms involve two major tasks: the construction of the Graver basis of a suitable n-fold product in Lemmas 2.5 and 2.7, and the iterative use of this Graver basis to solve the underlying (convex) integer program in Lemmas 2.3 and 2.6. The polynomial time solvability of these tasks is established in [3,9]. Here we only briefly discuss the complexity of the first task in the special case of a digraph, which is relevant for the complexity of the many-transshipment application.

Let D be the $s \times t$ incidence matrix of a digraph G. Consider l-fold products $D^{[l]}$ of the special form defined above. The *type* of an element $x = (x^1, \ldots, x^l)$ in the Graver basis $\mathcal{G}(D^{[l]})$ is the number of nonzero bricks $x^k \in \mathbb{Z}^t$ of x. It turns out that for any digraph G there is a finite nonnegative integer $g(G)$ which is the largest type of any element of any $\mathcal{G}(D^{[l]})$ independent of l. We call this new digraph invariant $g(G)$ the *Graver complexity* of G. The complexity of computing $\mathcal{G}(D^{[l]})$ is $O(l^{g(G)})$ (see [3]) and hence the importance of $g(G)$. Unfortunately, our present understanding of the Graver complexity of a digraph is very limited and much more study is required.

Very little is known even for the complete bipartite digraphs $K_{3,n}$ (oriented from one side to the other): while $g(K_{3,3}) = 9$, already $g(K_{3,4})$ is unknown. See [2] for more details and a lower bound on $g(K_{3,n})$ which is exponential in n.

Acknowledgments The work of Shmuel Onn on this article was partly supported by the ISF and partly done while he was delivering the Nachdiplom Lectures at ETH Zürich. He would like to thank Komei Fukuda and Hans-Jakob Lüthi for related stimulating discussions. The authors thank the referees for several valuable suggestions which improved the presentation of this article.

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. Aust. N. Z. J. Stat. **45**, 229–249 (2003)
2. Berstein, Y., Onn, S.: The Graver complexity of integer programming. Ann. Comb. **13**, 289–296 (2009)
3. De Loera, J., Hemmecke, R., Onn, S., Weismantel, R.: N-fold integer programming. Disc. Optim. **5**, 231–241 (2008) (volume in memory of George B. Dantzig)
4. De Loera, J., Hemmecke, R., Onn, S., Rothblum, U.G., Weismantel, R.: Convex integer maximization via Graver bases. J. Pure Appl. Algebra **213**, 1569–1577 (2009)
5. De Loera, J., Onn, S.: The complexity of three-way statisticaltables. SIAM J. Comp. **33**, 819–836 (2004)
6. De Loera, J., Onn, S.: All linear and integer programs are slim 3-way transportation programs. SIAM J. Optim. **17**, 806–821 (2006)
7. De Loera, J., Onn, S.: Markov bases of three-way tables are arbitrarily complicated. J. Symb. Comp. **41**, 173–181 (2006)
8. Graver, J.E.: On the foundation of linear and integer programming I. Math. Prog. **9**, 207–226 (1975)
9. Hemmecke, R., Onn, S., Weismantel, R.: A polynomial oracle-time algorithm for convex integer minimization. Math. Prog. (To appear)
10. Hoffman, A.J., Kruskal, J.B.: Integral boundary points of convex polyhedra. In: Linear Inequalities and Related Systems. Ann. Math. Stud., vol. 38, pp. 223–246. Princeton University Press, Princeton (1956)
11. Hoşten, S., Sullivant, S.: Finiteness theorems for Markov bases of hierarchical models. J. Comb. Theory Ser. A **114**, 311–321 (2007)
12. Leighton, T., Rao, S.: Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. J. Assoc. Comp. Mach. **46**, 787–832 (1999)
13. Lenstra, H.W. Jr.: Integer programming with a fixed number ofvariables. Math. Oper. Res. **8**, 538–548 (1983)
14. Santos, F., Sturmfels, B.: Higher Lawrence configurations. J. Comb. Theory Ser. A **103**, 151–164 (2003)
15. Schrijver, A.: Theory of Linear and Integer Programming. Wiley, Chichester (1986)