SOLVING A PERMUTATION PROBLEM BY A FULLY POLYNOMIAL-TIME APPROXIMATION SCHEME

Stanislaw Gawiejnowicz, Wieslaw Kurc

AND

Lidia Pankowska

Adam Mickiewicz University
Faculty of Mathematics and Computer Science, Poland

e-mails: {stgawiej, wkurc, lpankow}@amu.edu.pl

Abstract

For a problem of optimal discrete control with a discrete control set composed of vertices of an n-dimensional permutohedron, a fully polynomial-time approximation scheme is proposed.

Keywords: combinatorial optimization, discrete control theory, fully polynomial-time approximation scheme.

2000 Mathematics Subject Classification: 90C27, 49N05, 68M20.

1. Introduction

In this paper, we consider a problem of an optimal discrete control with a discrete control set composed of vertices of an n-dimensional permutohedron. The problem can be formulated as follows. Let \( a^\circ = (a_1^\circ, a_2^\circ, \ldots, a_n^\circ) \) be a sequence of non-negative coefficients \( a_i^\circ > 1 \), where \( i = 1, 2, \ldots, n \). The control set \( \Pi(a^\circ) \) is composed of all permutations of the sequence \( a^\circ \). Given any control \( a = (a_1, a_2, \ldots, a_n) \in \Pi(a^\circ) \), the transition function is given by

\[
Z_i = a_i Z_{i-1} + 1 \quad \text{for} \quad i = 1, 2, \ldots, n \quad \text{with} \quad Z_0 = 1,
\]

and the aim is to minimize the goal function \( \sum_{i=0}^{n} Z_i \) over all \( a \in \Pi(a^\circ) \). For simplicity of further presentation, we will refer to the above problem as
to the problem \( (P) \). For this problem, we propose a fully polynomial-time approximation scheme (an FPTAS).

2. Preliminary results

Applying the matrix approach (Gawiejnowicz et al. [2, 3], Gawiejnowicz [1]), the problem \( (P) \) can be formulated as follows:

\[
(P) \quad \begin{cases}
\min W_P(a) \triangleq ||Z(a)||_1 \\
\text{s.t. } A(a)Z(a) = b, \ a \in \Pi(a^\circ),
\end{cases}
\]

where \( a = (a_1, a_2, \ldots, a_n) \), \( b = (b_0, b_1, \ldots, b_n)^\top \), with \( b_j = 1 \) for \( j = 0, 1, \ldots, n \), \( Z(a) = (Z_0, Z_1, \ldots, Z_n)^\top \) and

\[
A(a) = \begin{pmatrix}
1 & 0 & \ldots & 0 & 0 \\
-a_1 & 1 & \ldots & 0 & 0 \\
0 & -a_2 & \ldots & 0 & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & \ldots & -a_n & 1
\end{pmatrix}.
\]

Since

\[
A^{-1}(a) = \begin{pmatrix}
1 & 0 & \ldots & 0 & 0 \\
a_1 & 1 & \ldots & 0 & 0 \\
a_1a_2 & a_2 & \ldots & 0 & 0 \\
a_1a_2a_3 & a_2a_3 & \ldots & 0 & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
a_1a_2 \ldots a_n & a_2a_3 \ldots a_n & \ldots & a_n
\end{pmatrix}
\]

exists, \( Z(a) = A^{-1}(a)b \) and hence \( W_P(a) \triangleq ||Z(a)||_1 = ||A^{-1}(a)b||_1 = \sum_{j=0}^n \sum_{i=0}^j a_{i+1} \cdots a_j \), where an empty product is assumed to be equal to 1. In other words, \( W_P(a) \) is the sum of all elements of the matrix \( A^{-1}(a) \). Hence, \( ||Z(a)||_1 = ||Z(\overline{a})||_1 \), where \( \overline{a} = (a_n, a_{n-1}, \ldots, a_1) \).

Let us distinguish the sum \( Z_n \) of elements of the last row and the first column of \( A^{-1}(a) \), i.e., \( Z_n = Z_n(a) = \sum_{i=0}^n a_{i+1} \cdots a_n \) and \( Z_n(\overline{a}) = \sum_{j=0}^n a_1 \cdots a_j \).

Given \( a \in \Pi(a^\circ) \), let us define the function \( L(a) \triangleq W_P(a) - (n + 1) \), in which the diagonal of \( A^{-1}(a) \) in the sum \( W_P(a) \) is omitted. Let us also
define \( M(a) \triangleq Z_n(a) - 1 \) and \( M(\bar{a}) \triangleq Z_n(\bar{a}) - 1 \). Notice that the goal function \( L(a) \) can be used interchangeably with \( W_P(a) \).

For the problem \((P)\), the following \( V\)-shape property (Mosheiov [5]) is known: if \( a^* \in \Pi(a^o) \) is an optimal solution to \((P)\), then \( a^* \) must be \( V\)-shaped, i.e., \(-a^*\) is unimodal with the maximum \(-a_k\) attained for some \( 1 \leq k \leq n \).

Since in subsequent sections we will consider \((1 + \epsilon)\)-approximation algorithms, we formulate now the following definition of the notion, assuming that only finite size instances of the problem \((P)\) will be considered.

\[ \text{Definition 1.} \quad \text{An algorithm} \ A_P \ \text{is called} \ (1 + \epsilon)\text{-approximation algorithm for the problem} \ (P), \ \text{if for each instance} \ a^o \ \text{of the problem} \ (P) \ \text{it delivers a feasible solution with objective value} \ A_P(a^o) \ \text{such that} \]
\[
|A_P(a^o) - W_P(a^*)| \leq \epsilon W_P(a^*),
\]
where \( \epsilon > 0 \) is an accuracy of solution, \( W_P \) is the objective function of the problem \((P)\) and \( a^* \) is the optimal solution to the problem \((P)\).

From Definition 1 it follows that
\[ A_P(a^o) \leq (1 + \epsilon)W_P(a^*). \]

The factor \( \rho = 1 + \epsilon \) is called the worst-case ratio for the algorithm \( A_P \). The next definition concerns a family of \((1 + \epsilon)\)-approximation algorithms.

\[ \text{Definition 2.} \quad \text{The family} \ \{A_P^\epsilon\}_\epsilon \ \text{of} \ (1 + \epsilon)\text{-approximation algorithms for the problem} \ (P) \ \text{is called a fully polynomial-time approximation scheme (an FPTAS), if for any} \ \epsilon > 0 \ \text{the time complexity of the algorithm} \ A_P^\epsilon \ \text{is polynomial in the input size} \ #a^o \ \text{and in} \ \frac{1}{\epsilon}. \]

Now, we introduce a number of formulae that will be applied in subsequent sections. The formulae concern concatenated sequences \( u = (u_1, u_2, \ldots, u_r) \) and \( v = (v_1, v_2, \ldots, v_s) \). Let \( u|v = (u_1, u_2, \ldots, u_r, v_1, v_2, \ldots, v_s) \) denote the concatenation of sequences \( u \) and \( v \) in the given order. Then
\[
M(u|v) = v_1v_2\cdots v_s(u_1u_2\cdots u_r + \ldots + u_r) + (v_1v_2\cdots v_s + \ldots + v_s) = M(v) + v_1v_2\cdots v_sM(u).
\]
Moreover, for $L(u|v) = \|Z(u|v)\|_1 - (r + s + 1)$, we have

$$L(u|v) = L(u) + L(v) + (u_1 \cdots u_r + \ldots + u_r)v_1 + \ldots + (u_1 \cdots u_r + \ldots + u_r)v_1v_2 \cdots v_s$$

$$= L(u) + L(v) + M(u)(v_1 + \ldots + v_1v_2 \cdots v_s).$$

Thus, if we denote $\pi(v) = v_1v_2 \cdots v_s$, we obtain the following result.

**Lemma 1.** There hold the following equalities:

(a) $M(u|v) = M(v) + \pi(v)M(u)$,

(b) $L(u|v) = L(u) + L(v) + M(u)M(\pi)$.

From Lemma 1 it follows the next result, concerning the case of concatenation of three sequences, $u|a|v$, where $u = (u_1, u_2, \ldots, u_r)$, $a = (a_1, a_2, \ldots, a_n)$ and $v = (v_1, v_2, \ldots, v_s)$.

**Lemma 2.** There hold the following equalities:

(a) $M(u|a|v) = M(v) + \pi(v)M(a) + \pi(a)\pi(v)M(u)$,

(b) $L(u|a|v) = L(u) + L(a) + L(v) + M(u)M(\pi) + M(a)M(\pi) + \pi(a)M(u)M(\pi)$.

As an application of the above formulae, let us consider two concatenations, $u|a|v$ and $u'|a|v'$. In view of part (b) of Lemma 2, we obtain the following general formula

$$L(u'|a|v') - L(u|a|v) = L(u') - L(u) + L(v') - L(v) + M(\pi)(M(u') - M(u)) + M(a)(M(\pi') - M(\pi)) + \pi(a)(M(u')M(\pi') - M(u)M(\pi)).$$

In particular, for $u' = v$ and $v' = u$ we obtain the following result.

**Lemma 3.** There holds the following equality:

$$L(v|a|u) - L(u|a|v) = M(\pi)(M(v) - M(u)) + M(a)(M(\pi) - M(\pi)) + \pi(a)(M(v)M(\pi) - M(u)M(\pi)).$$

If $u = (p)$ and $v = (q)$, by Lemma 3 we obtain the formula
Solving a permutation problem by a fully ... 195

(1) \[ L(q|a|p) - L(p|a|q) = (q - p)(M(\overline{a}) - M(a)) = (q - p)S^-(a), \]

where \( S^-(a) \equiv M(\overline{a}) - M(a) \) denotes the so-called signature (we refer the reader to Gawiejnowicz et al. [2, 3] for details).

We complete the section by a result that is an application of formula (1).

**Theorem 1** ([2, 3]). Let \( a^\dagger = (a_1, a_2, \ldots, a_n) \) be a non-decreasingly ordered sequence for the problem \((P)\) and let \( p = a_{2k-1}, \ q = a_{2k}, \) where \( 1 \leq k \leq \phi(n) \) for a suitable \( \phi. \) Let \((p|u|q)\) and \((q|u|p)\) denote concatenations of a partial V-shaped sequence \( u, \) composed of the elements of \( a^\dagger, \) with \( p \) and \( q \) in the given order, respectively. Then there hold implications:

(a) if \( S^-(u) \geq 0, \) then \( L(p|u|q) \leq L(q|u|p) \)

(b) if \( S^-(u) \leq 0, \) then \( L(p|u|q) \geq L(q|u|p). \)

Theorem 1 leads to a greedy algorithm (cf. [2, 3]) based on consecutive concatenations of elements of the sequence \( a^\dagger. \) The function \( \phi(n) \) will be defined in the next section.

3. Dynamic programming algorithms

In this section, following Woeginger [6], we formulate two dynamic programming algorithms for the problem \((P)\). Both these algorithms go through \( \phi(n) \) phases, where \( \phi(n) \) is a function that can be computed in a polynomial time with respect to \( n. \) The general idea of these algorithms is as follows.

The \( k \)-th phase, \( 1 \leq k \leq \phi(n), \) produces a set \( S_k \) of states \( S. \) Any state in \( S_k \in S \) is a vector \( S = [s_1, s_2, \ldots, s_\beta]^T \in \mathbb{Q}_+^\beta, \) where \( \mathbb{Q}_+ \) denote the set of positive rational numbers and \( \beta \geq 1 \) is a fixed natural number. In the problem \((P), \) the vectors \( S \) are related to partial V-shaped sequences \( a^S \) that concern the first \( k \) coefficients of a non-decreasing rearrangement \( a^\dagger \) of a given sequence \( a^\circ. \)

The sets of states \( S_1, S_2, \ldots, S_k, \ 1 \leq k \leq \phi(n), \) are constructed iteratively. Given an initial set \( S_0 \equiv \{S_0\}, \) the \( k \)-th set \( S_k \) is obtained from the set \( S_{k-1} \) by applying a fixed number of mappings \( F_1, F_2, \ldots, F_s \) which translate the states of the set \( S_{k-1} \) into the states of the set \( S_k. \) More precisely,

\[ S_k = \{F(X_k, S) : S \in S_{k-1}, F \in \mathcal{F} \equiv \{F_1, F_2, \ldots, F_s\}\}, \]

where \( 1 \leq k \leq \phi(n). \)
Non-negative vectors $X_1, \ldots, X_{\phi(n)}$, where $X_k = [x^k_1, x^k_2, \ldots, x^k_{\alpha}]^\top \in \mathbb{Q}_+^\alpha$ with a natural fixed $\alpha \geq 1$, are arranged in a prescribed way within the algorithm $DP$ for a given input data $a^\circ$. For our purposes, we will assume that $X_k = [a(k-1)\alpha + 1, \ldots, a_k]$, for $k = 1, 2, \ldots, \phi(n)$, where $\phi(n) = [\frac{n}{\alpha}] + 1$ if $n$ is not a multiple of $\alpha$, and $\phi(n) = \frac{n}{\alpha}$ otherwise. In the first case, $X_{\phi(n)}$ contains residual components of $a^\circ$.

Let $G$ be a non-negative function defined for the states $S = [s_1, s_2, \ldots, s_\beta]^\top$. Throughout the paper, we assume that $G(S) = s_1$, where $s_1 = L(a^S)$ and $a^S$ is a partial V-shaped sequence corresponding to $S$.

The above assumptions describe the untrimmed dynamic programming algorithm, called $DP$. The trimmed version of the algorithm $DP$, called $TDP$, uses an approximation procedure introduced by Ibarra and Kim [4]. The crucial point is the "trimming-the-state-space" technique (cf. [6]), which "clean up" and "thin out" the state spaces $S_k$ in a proper way.

The untrimmed and trimmed versions of these $DP$ algorithms can be formulated as follows.

Algorithm $DP$

$$
S_0 := \{S_0\};
\text{for } k := 1 \text{ to } \phi(n) \text{ do }
S_k := \emptyset;
\text{for every } S \in S_{k-1} \text{ and } F \in \mathcal{F} \text{ do }
\text{add } F(X_k, S) \text{ to } S_k;
\text{end}
\text{end}
\text{return } \min\{G(S) : S \in S_{\phi(n)}\}
$$

Algorithm $TDP$

$$
T_0 := S_0 := \{S_0\};
\text{for } k := 1 \text{ to } \phi(n) \text{ do }
T_k := \emptyset;
\text{for every } T \in T_{k-1} \text{ and } F \in \mathcal{F} \text{ do }
\text{add } F(X_k, T) \text{ to } T_k;
\text{end}
\text{end}
\text{compute a trimmed copy } T_k \text{ of } T_k;
\text{return } \min\{G(T) : T \in T_{\phi(n)}\}
$$

Let a given sequence $a^\circ$, with a non-decreasing rearrangement $a^\circ$, be fixed. Let $\hat{\alpha} = \max_{i=1,\ldots,n} \{a_i\}$. Let the parameters $\alpha$, $\beta$ and $s$ also be fixed. The mappings $F_i : \mathbb{Q}_+^\alpha \times \mathbb{Q}_+^\beta \rightarrow \mathbb{Q}_+^\beta$ from the set $\mathcal{F} = \{F_1, F_2, \ldots, F_s\}$ are given by formulae $S' = F_i(X_k, S)$, where $S' = [s'_1, s'_2, \ldots, s'_\beta]^\top$. (The particular form of $F_i(X_k, S)$ will be specified below.) The following two conditions, (C1) and (C2), will be satisfied by the dynamic programming algorithms for the problem $(P)$:

(C1) the formulae $F_i(X_k, S)$ can be evaluated in a polynomial time as functions of components of $X_k = [x^k_1, x^k_2, \ldots, x^k_{\alpha}]^\top$ and $S = [s_1, s_2, \ldots, s_\beta]^\top$;
(C2) for any state $\mathbf{S} = [s_1, s_2, \ldots, s_\beta]^{\top}$ and for each component $s_i$, there holds the estimation $0 < s_i \leq e^{p(n, \log \tilde{a})}$ for a certain polynomial $p(n, \log \tilde{a})$ of variable $n$ and natural logarithm $\log \tilde{a}$.

Given $\epsilon > 0$, let $\Delta = 1 + \frac{\epsilon}{2\phi(n)}$. Let $J$ be the smallest possible natural number such that $e^{p(n, \log \tilde{a})} \leq \Delta^J$. Without loss of generality, we can choose

$$J = \left\lceil \frac{p(n, \log \tilde{a})}{\log(\Delta)} \right\rceil \leq \left\lceil \left(1 + \frac{2\phi(n)}{\epsilon} \right)p(n, \log \tilde{a}) \right\rceil,$$

since the latter inequality follows from the inequality $\log(x) \geq \frac{1-x}{x}$ for $x \geq 1$.

Let us divide the cube $[0, \Delta^J]^{\beta}$ into $(J + 1)^{\beta}$ boxes along lines that are perpendicular to respective axis at the points $\Delta^j$, where $j = 0, 1, 2, \ldots, J$. These boxes will be called $\Delta$-boxes.

**Definition 3 ([6])**. The states $\mathbf{S} = [s_1, s_2, \ldots, s_\beta]^{\top}$ and $\mathbf{S}' = [s'_1, s'_2, \ldots, s'_\beta]^{\top}$ are said to be $\Delta$-close, if $s_i\Delta^{-1} \leq s'_i \leq s_i\Delta$ for $i = 1, 2, \ldots, \beta$.

Notice that if $\mathbf{S}$ and $\mathbf{S}'$ are in the same $\Delta$-box, then for $s_i$ and $s'_i$ there holds $\Delta^{-1} \leq s'_i, s_i \leq \Delta$ for some $j$. Hence $\Delta^{-1} \leq s'_i/s_i \leq \Delta$.

The trimming is defined as follows.

**Definition 4 ([6])**. If the state sets $\mathcal{U}$ and $\mathcal{T}$ belong to $[0, \Delta^J]^{\beta}$, then $\mathcal{T}$ is said to be a trimmed copy of $\mathcal{U}$, if (i) $\mathcal{T} \subset \mathcal{U}$ and (ii) for every $\Delta$-box $B$ with $B \cap \mathcal{U} \neq \emptyset$ the set $\mathcal{T}$ contains exactly one state $S \in B \cap \mathcal{U}$.

Clearly, each state $S$ from condition (ii) of Definition 4 is $\Delta$-close to each element of $B \cap \mathcal{U}$.

### 4. Fully polynomial-time approximation scheme

In this section, we prove that for any fixed $\epsilon > 0$ the trimming procedure added to the untrimmed dynamic algorithm algorithm $\text{DP}$ leads to such a solution which can be only $(1 + \epsilon)$-times worst than the original one.

In order to do this, we need to know that the problem $(P)$ is $\text{DP}$-simple (cf. [6]). This means that the optimal solution to the problem $(P)$, with the criterion $L(a)$, is equal to $G(S^*)$, where $S^* \in \mathcal{S}_{\phi(n)}$ is determined by the algorithm $\text{DP}$. This, in turn, requires an additional knowledge concerning the formulae $F_i(X_k, S)$ in the algorithm $\text{DP}$. Hence, now we describe precisely how to obtain the copy of $\mathcal{U}_k$ in the case $\alpha = 1$. 

In the $k$-th phase, $X_k = [a_k]^T$ and it consists of the $k$-th element from non-decreasing rearrangement $a^\uparrow = (a_1, a_2, \ldots, a_n)$ of the sequence $a$. In this case, $\phi(n) = n$. Let $a = a^S$ be any $(k - 1)$-element V-shaped sequence obtained at the $(k - 1)$-th phase, with the corresponding state vector $S = [L(a), M(a), M(\overline{a}), \Pi(a)]^T$ from $T_{k-1}$ for TDP and from $S_{k-1}$ for DP, respectively. Then, the new set of states is given by

$$ U_k = \bigcup_{T \in T_{k-1}} \{ F_1([a_k]^T, T), F_2([a_k]^T, T) \} $$

for TDP and

$$ S_k = \bigcup_{S \in S_{k-1}} \{ F_1([a_k]^T, S), F_2([a_k]^T, S) \} $$

for DP, respectively, where

$$ F_1([a_k]^T, S) = F_1([a_k]^T, T) = [L(a_k|a), M(a_k|a), M(\overline{a}|a), \pi(a_k|a)]^T, $$

$$ F_2([a_k]^T, S) = F_2([a_k]^T, T) = [L(a|a_k), M(a|a_k), M(\overline{a}|a_k), \pi(a|a_k)]^T. $$

The concatenation formulae for $L(p|a), L(a|p), M(p|a)$ and $M(a|p)$, given in Lemmata 1–3, can be applied in order to obtain formulae for computing the components of new states in a polynomial time. In particular, denoting the above mentioned state $S$ by $[s_1, s_2, s_3, s_4]^T$, we obtain new states $S' = [s'_1, s'_2, s'_3, s'_4]^T$ in $U_k$ or $S_k$, respectively, computed by means of mappings from $\mathcal{F} = \{ F_1, F_2 \}$, where

$$ F_1([a_k]^T, S) = [s_1 + a_k(s_3 + 1), s_2 + a_k s_4, a_k(s_3 + 1), a_k s_4]^T, $$

$$ F_2([a_k]^T, S) = [s_1 + a_k(s_2 + 1), a_k(s_2 + 1), s_3 + a_k s_4, s_4 a_k]^T. $$

To be more clear, if we have $G(S) = s_1$, where $s_1 = L(a^S)$ in the state $S \in S_{k-1}$, then for $S' \in S_k$ we get $G(S') = s_1 + a_k(s_3 + 1)$ in the case of $L(a_k|a)$ and $G(S') = s_1 + a_k(s_2 + 1)$ in the case of $L(a|a_k)$. We proceed in this way for $k = 1, 2, 3, \ldots, n$, starting with $T_0 := S_0 \triangleq \{[1, 1, 1, 1]^T\}$ and the empty sequence $a^{S_0} \triangleq ()$. 

**Lemma 4.** Let $\bar{a} = \max_{i=1 \ldots n} \{ a_i \}$ for a given sequence $a^\circ$ for the problem (P). Then the algorithm DP satisfies conditions (C1) and (C2), with the polynomial $p(n, \log \bar{a}) = \frac{1}{2} n(n + 1) + n \log \bar{a} - 1$. 
Proof. The condition (C1) follows from the formulae for $F_1([a_k] \top, S)$ and $F_2([a_k] \top, S)$ given above. To prove that the condition (C2) is satisfied, notice that $L(a_k|a), M(a_k|a), M(a_k|a), a_k\pi(a)$, with $a = a^S$ for $S \in S_{k-1}$, are less or equal to $L(a^S)$ with $S \in S_n$. Let us write for simplicity that $a^S = (a_1, \ldots, a_n)$. Since $L(a^S)$ is the sum of all elements of $A^{-1}$ except the diagonal, then majorizing these $a_i$ by $\hat{a}$ it is not difficult to show that

$$L(a^S) \leq n \cdot \hat{a} + (n - 1) \cdot \hat{a}^2 + \ldots + \hat{a}^n \leq \frac{n(n + 1)}{2} \cdot \hat{a}^n \leq e^{\frac{n(n+1)}{2}} \cdot \hat{a}^n.$$  

Thus, it is sufficient to require that $p(n, \log \hat{a}) = \frac{1}{2} n(n+1) + n \log \hat{a} - 1$. 

Lemma 5. Any resulting sequence $a^S$ generated by algorithms DP and TDP is V-shaped.

Proof. Since we take $a_k$ from non-decreasing rearrangement $a^\uparrow$ of the initial sequence $a^\circ$, then by induction one can show that concatenations $a_k|u$ and $u|a_k$ lead from the V-shaped sequence $u$ to V-shaped sequences.

Lemma 6. The problem (P) is DP-simple, i.e., if $a^*$ is a solution of the problem (P), then $W_P(a^*) = L(a^*) + (n + 1)$ with $L(a^*) = G(S^*)$, for some $S^* \in S_n$ such that $G(S^*) = \min\{G(S) : S \in S_n\}$ and $a^* = a^{S^*}$.

Proof. Recall that any optimal solution $a^*$ to the problem (P) must be V-shaped. In the final state space $S_n$, each state $S = [s_1, s_2, s_3, s_4]$ is such that $G(S) = s_1$, where $s_1 = L(a^S)$ and $a^S$ runs over all possible V-shaped resulting concatenations. Clearly, some of these V-shaped concatenations coincide with the optimal one $a^*$, since it is also V-shaped.

Theorem 2. Given any $\epsilon > 0$, then for each final state $S \in S_n$ there exists a trimmed state $T \in T_n$, determined by the algorithm TDP, such that

$$G(T) \leq (1 + \epsilon)G(S)$$

or, with corresponding V-shaped sequences $a^T$ and $a^S$,

$$W_P(a^T) \leq (1 + \epsilon)W_P(a^S).$$

Proof. We will prove that for each $S = [s_1, s_2, s_3, s_4] \top \in S_k$ there exists state $T = [t_1, t_2, t_3, t_4]$ $\in T_k$ such that $T \leq \Delta^k S$ coordinatewise for $k = 0, 1, \ldots, n$. In this case $\beta = 4$. We will proceed by induction.
For \( k = 0 \) we have \( T_0 := S_0 := \{S_0\} \), with \( S_0 := [1, 1, 1, 1]^{\top} \), so inequalities (2) and (3) are satisfied.

Assume now that for the \((k - 1)\)-th step of the algorithm TDP there holds the inequality \( T \leq \Delta^{k-1} S \), where state \( S = [s_1, s_2, s_3, s_4]^{\top} \in S_{k-1} \) and state \( T = [t_1, t_2, t_3, t_4]^{\top} \in T_{k-1} \). According to the formulation of the algorithm TDP, we apply the mappings \( F_1 \) and \( F_2 \) from \( \mathcal{F} \) in order to obtain the new state space \( U_k \) by attaching \( F_1(X_k, T) \) and \( F_2(X_k, T) \), and the state space \( S_k \) by attaching \( F_1(X_k, S) \) and \( F_2(X_k, S) \), where \( X_k = [a_k]^{\top} \) and \( a_k \) is not yet considered element of \( a^1 \).

For \( F_1(X_k, S) \), we get
\[
S' = [s_1', s_2', s_3', s_4']^{\top} = [s_1 + a_k(s_3 + 1), s_2 + a_k s_4, a_k(s_3 + 1), a_k s_4]^{\top} \in S_k
\]
and for \( F_1(X_k, T) \) we get
\[
R = [r_1', r_2', r_3', r_4']^{\top} = [t_1 + a_k(t_3 + 1), t_2 + a_k t_4, a_k(t_3 + 1), a_k t_4]^{\top} \in U_k.
\]

By the trimming procedure we obtain a trimmed state
\[
T' = [t_1', t_2', t_3', t_4']^{\top} \in T_k,
\]
which is \( \Delta \)-close to the state \( R \), i.e., \( \Delta^{-1} R \leq T' \leq \Delta R \).

To prove that \( T' \leq \Delta^k S' \), we apply first that \( \Delta^{-1} R \leq T' \leq \Delta R \) for \( R \in U_k \). Then we apply the induction assumption to get the inequalities
\[
t_1 + a_k(t_3 + 1) \leq \Delta^{k-1} s_1 + a_k(\Delta^{k-1} s_3 + 1),
\]
\[
t_2 + a_k t_4 \leq \Delta^{k-1} s_2 + a_k \Delta^{k-1} s_4,
\]
\[
a_k(t_3 + 1) \leq a_k(\Delta^{k-1} s_3 + 1)
\]
and
\[
a_k t_4 \leq a_k \Delta^{k-1} s_4.
\]

Now, since \( \Delta^{k-1} > 1 \), we have
\[
\Delta R \leq \Delta^k [s_1 + a_k(s_3 + 1), s_2 + a_k s_4, a_k(s_3 + 1), a_k s_4]^{\top} = \Delta^k S'.
\]

Collecting all this together, we get \( T' \leq \Delta^k S' \) as desired.
Proceeding further by induction, we obtain that the same is true for the final phase, i.e., for $k = n$. Thus, in view of the formulation of the algorithm $TDP$, for any state $S$ in $\mathcal{S}_n$ there holds the inequality $T \leq \Delta^n S$. In particular, since $G(S) = s_1$, we have that $G(T) \leq \Delta^n G(S)$. But $\Delta^n = (1 + \frac{\epsilon}{2\delta(n)})^n = (1 + \frac{\epsilon}{2n})^n \leq 1 + \epsilon$. Thus, $G(T) \leq (1 + \epsilon)G(S)$. \hfill \blacksquare

As a corollary from Theorem 2 we obtain the following result.

**Theorem 3.** For the problem (P) there exists a fully polynomial-time approximation scheme (an FPTAS).

**Proof.** By Theorem 2, for each $\epsilon > 0$ and for each state $S \in \mathcal{S}_n$ there exists a trimmed state $T \in \mathcal{T}_n$ determined by the algorithm $TDP$ such that $G(T) \leq (1 + \epsilon)G(S)$. For simplicity of further presentation, we will refer to the algorithm $TDP$ as to the algorithm $A_P$ (cf. Definition 2). In view of Lemma 6, the problem (P) is DP-simple. Recall that this means that the dynamic programming algorithm returns the optimal solution to (P).

Let $G(S^*) = \min\{G(S) : S \in \mathcal{S}_n\}$ and let $a^{S^*}$ be the corresponding final V-shaped sequence for the state $S^*$ in $\mathcal{S}_n$. Then, according to the above inequality, we have $G(T^*) \leq (1 + \epsilon)G(S^*)$ for some $T^* \in \mathcal{T}_n$ returned by $A_P$, i.e., by the algorithm $TDP$. Let $a^{T^*}$ be the corresponding final V-shaped sequence for the state $T^*$ in $\mathcal{T}_n$. Taking into account that $L(a^{S^*}) = G(S^*)$ and $L(a^{T^*}) = G(T^*)$, we have $L(a^{T^*}) \leq (1 + \epsilon)L(a^{S^*})$. Clearly, $a^{S^*} \in \Pi(a^0)$ and $a^{T^*} \in \Pi(a^0)$. Since $W_P(a) = L(a) + (n + 1)$, we get that $W_P(a^{T^*}) \leq (1 + \epsilon)W_P(a^{S^*})$ for the final V-shaped sequences $a^{T^*}$ and $a^{S^*}$. Since $\epsilon > 0$ is arbitrary, we have constructed an approximation scheme.

To end the proof, it is sufficient to show that algorithms $A_P$, i.e., $TDP$-type algorithms with arbitrary $\epsilon > 0$, are polynomial with respect to the size of instances $a^0$ of the problem (P) and with respect to $\frac{1}{\epsilon}$. According to the formulation of the algorithm $TDP$, we proceed by $\phi(n)$ phases, where $\phi(n)$ polynomially depends on $n$. At each of these phases, we compute in polynomial time formulae $F_i(X_k, T)$ for every $T \in \mathcal{T}_{k-1}$ and $F_i \in \mathcal{F}$, obtaining a new state space $\mathcal{U}_k$. The size of $\mathcal{F}$ is finite and constant. It remains to obtain an estimation for the cardinality $\#T_k$ of $\mathcal{T}_k$. It is clear that $\#T_k$ equals the number of $\Delta$-boxes having, according to the formulation of the trimming procedure, a non-empty intersection with $\mathcal{U}_k$ in $[0, \Delta^J]^3$. By Lemma 4, each coordinate of $S = [s_1, s_2, s_3, s_4]^\top$ satisfies the inequality $0 < s_i \leq e^{p(n, \log \bar{a})}$.
Thus, each $S$ belongs to some of $(J + 1)\beta$ $\Delta$-boxes. In consequence,

$$\#T_k \leq (J + 1)^\beta \leq \left[ (1 + \frac{2\phi(n)}{\epsilon})p(n, \log \hat{a}) \right]^\beta,$$

where $p(n, \log \hat{a})$, by Lemma 4, is a polynomial with respect to $\log \hat{a}$ and $n$. We complete the proof by noticing that $\hat{a}$ has a finite encoding and the dependence on $\frac{1}{\epsilon}$ is linear.

Notice that from Theorem 3 it immediately follows that the problem $(P)$ cannot be $NP$-hard in the strong sense.

The time complexity of the proposed FPTAS can be found by estimation of the number $m$ of the $\Delta$-boxes, where $m = (J + 1)^\beta$ with $\beta = 4$. For the actual form of the polynomial $p(n, \log \hat{a})$, we have $m = n^{12\epsilon^{-4}}$. Hence, for a given instance $a^\alpha$, the complexity is $O(n^{13\epsilon^{-4}})$. However, by considering the polynomial in the form of $p(n, \log \hat{a}) = (n + 1)\log \hat{a} - 2\log(\hat{a} - 1)^{-2}$, where $\hat{a} = \min_{1 \leq i \leq n} \{a_i\}$, one can decrease the complexity to $O(n^{9\epsilon^{-4}})$.

5. Conclusions

In the paper we considered the problem $(P)$ of optimal discrete control with a discrete control set composed of vertices of an $n$-dimensional permutohedron. We have shown that for this problem there exist a fully polynomial-time approximation scheme. Thus, though the time complexity of the problem is still unknown, the problem can be at most $NP$-hard in the ordinary sense.

Future research may concern the following open problems. The first one is to apply the presented approach for $\alpha \geq 2$. The second one is to consider a generalization of the DP algorithms to the problem $(P)$ with $b_j \neq 1$ for $0 \leq j \leq n$. Finally, an interesting problem is to investigate relations between the DP algorithms and greedy algorithms presented in [2, 3].

References


Solving a permutation problem by a fully …


Received 30 November 2009