An improved general procedure for lexicographic bottleneck problems

Federico Della Croce, D.A.I., Politecnico di Torino[*], dellacroce@polito.it

Vangelis Th. Paschos[†], Alexis Tsoukias, LAMSADE, Université Paris Dauphine[‡], {paschos,tsoukias}@lamsade.dauphine.fr

Abstract

In combinatorial optimization, the bottleneck (or minmax) problems are those problems where the objective is to find a feasible solution such that its largest cost coefficient elements have minimum cost. Here we consider a generalization of these problems, where under a lexicographic rule we want to minimize the cost also of the second largest cost coefficient elements, then of the third largest cost coefficients and so on. We propose a general rule which leads, given the considered problem, to a vectorial version of the solution procedure for the underlying sum optimization (minsum) problem. This vectorial procedure increases by a factor of k (where k is the number of different cost coefficients) the complexity of the corresponding sum optimization problem solution procedure.

Keywords: lexicographic problem, bottleneck problem, combinatorial optimization.

1 Introduction

In most of the classical combinatorial optimization problems, the objective function is an additive function of the single variables. These problems are often denoted as minsum or sum optimization problems (SOP). Minmax or bottleneck optimization problems (BOP) are the easiest way to deal with scenarios where the variables can be related to each other under an ordinal scale rather than a cardinal one. In a BOP, the objective is to find a feasible solution where the largest cost coefficient elements have minimum cost. In many combinatorial optimization problems (e.g., the bottleneck assignment problem [10]), the minmax version is easier than the minsum version as it can be solved by tackling $\log k$ searches of feasible solutions of minsum problems where k is the number of distinct cost-coefficients, as the search for feasibility is often much easier than the search for optimality. Consider now a generalization of a minmax problem by requiring also the second largest cost coefficients elements in the solution to be minimum, then the third largest cost coefficient and so on. This kind of problems arrive, for instance (see, for example [11]) in the evaluation of fragmented alternatives in multicriteria decision aid. We will denote these problems in the remainder of the paper as lexicographic bottleneck optimization problems (LBOP). In this case, a straightforward solution procedure does not exist except for those problems (e.g., the shortest spanning tree) whose SOP, BOP and LBOP versions are all optimally solved at the same time by the greedy algorithm as they are optimization problems over the set of bases of a matroid [12]. The relevant literature on this topic is quite limited to our knowledge. Burkard et al. [2] considered a broad class of algebraic assignment problems including the lexicographic bottleneck one and derived a general solution procedure. A general framework for lexicographic bottleneck problems was presented by Burkard and Rendl [1] who proposed

[*]Corso Duca degli Abruzzi 24, 10129 Torino, Italy

[†]Author for correspondence

[‡]Place du Maréchal De Lattre de Tassigny, 75775 Paris Cedex 16, France

two different solution procedures, the first based on coefficients scaling and the second based on an iterative approach. The two procedures have comparable computational complexities, and the authors report that it is preferable to apply the first procedure for small k and the second procedure for large k. Both procedures create numbers growing very fast with k in such a way that they cannot practically handle medium-size problem-instances. Recently Calvete and Mateo [4] proposed a primal-dual algorithm for a generalized lexicographic multiobjective network flow problem. The interested reader may consider also the survey paper by Burkard and Zimmermann [3] on the algebraic versions of various optimization problems. Purpose of this work is to present a solution procedure which is a rearrangement of the first solution procedure of [1] based on a vectorial representation of the cost coefficients. We show that this procedure outperforms both their procedures in terms of computational complexity for all realistic problems where $k \leq n^2 \log k$, where n is the number of non-zero variables involved in the optimal solution, and, moreover, that the vectorial representation forbids the numbers explosion phenomenon for large k. The paper proceeds as follows. In section 2 we introduce the relevant definitions and notation. In section 3 we present the procedure and prove its optimality. In section 4 we illustrate the procedure on the LBOP versions of two well known combinatorial optimization problems, the shortest path problem and the assignment problem. Section 5 concludes the paper with final remarks.

2 Notation and definitions

Let consider a combinatorial optimization problem involving m variables $x_1, \dots, x_m$ with cost coefficients or weights $w_1, \dots, w_m$. Let denote with $\overline{X}$ the set of feasible solutions to the problem. Let $W_1 = \max_{i:x_i>0}\{w_i\}$ be the largest active weight of a feasible solution $\overline{x} = \{x_1, \dots, x_m\}$ (we denote with *active* the weights corresponding to non-zero variables; hence, there exist n active weights) and correspondingly W_2 the second largest active weight and so on.

We say that

1. the solution $\overline{x^*} = \{x_1^*, \dots, x_m^*\}$ is optimal for a SOP if, $\forall \overline{x} = \{x_1, \dots, x_m\} \in \overline{X}$,

$$\sum_{i=1}^{m} w_i x_i \geq \sum_{i=1}^{m} w_i x_i^*; \tag{1}$$

2. the solution $\overline{x^*} = \{x_1^*, \dots, x_m^*\}$ is optimal for a BOP if, $\forall \overline{x} = \{x_1, \dots, x_m\} \in \overline{X}$,

$$W_1 \geq W_1^*; \tag{2}$$

3. the solution $\overline{x^*} = \{x_1^*, \dots, x_m^*\}$ is optimal for a LBOP if $\forall \overline{x} = \{x_1, \dots, x_m\} \in \overline{X}$ there does not exist any active weight W_l such that:

$$\begin{array}{ll} & \left[(W_l < W_l^*) \ \bigvee \ \left(W_l = W_l^* \ \bigwedge \ \sum\limits_{j:w_j=W_l} x_j < \sum\limits_{j:w_j=W_l^*} x_j^*\right)\right] \\ \bigwedge & \\ & \left(W_i = W_i^* \ \bigwedge \ \sum\limits_{j:w_j=W_i} x_j = \sum\limits_{j:w_j=W_i^*} x_j^*\right) \qquad \forall i < l \end{array} \tag{3}$$

Finally, as in [1], we denote by $T_{m,n}(\text{SOP})$ the worst-case running time to solve a m-variable SOP with n non-zero variables in the optimal solution where each elementary operation (e.g., addition) requires constant time.

3 The solution procedure: a vectorial approach

Given a LBOP, substitute each weight w_j with a vector of cost coefficients $\{c_1, c_2, \ldots, c_k\}$. The entries refer to the k different weights of the problem and are indexed in decreasing order of the weights values. All entries are set to 0 except the one that refers to the original weight w_j which is set to 1; for instance, the vector corresponding to the largest weight is $\{1, 0, 0, \ldots, 0, 0\}$. Consider now this problem as a SOP, where, as each weight is a vector of k components, each algebraic sum involving the weights becomes a vectorial sum of the corresponding k entries and each solution value is a vector of these k entries. We denote this problem with vectorial representation as VSOP (vectorial sum-optimization problem).

In order to apply a SOP solution procedure to a VSOP, let define how to compare two different solutions; recall that any solution value $f(\overline{X}) = \sum_{i=1}^{m} w_i x_i$ is now written as $f(\overline{X}) = \{\sum_{j:w_j=c_1} x_j, \sum_{j:w_j=c_2} x_j, \ldots, \sum_{j:w_j=c_k} x_j\}$.

We say that, given two solutions $\overline{X^\alpha}$ and $\overline{X^\beta}$,

$$f(\overline{X^\alpha}) < f(\overline{X^\beta}) \Longleftrightarrow \exists l, \left(\sum_{j:w_j=c_l} x_j^\alpha < \sum_{j:w_j=c_l} x_j^\beta \right) \bigwedge \left(\forall i < l, \sum_{j:w_j=c_i} x_j^\alpha = \sum_{j:w_j=c_i} x_j^\beta \right). \quad (4)$$

Theorem 1. *The optimal solution $\overline{X^*}$ of the VSOP computed by applying the vectorial version of the related sum optimization procedure is optimal also for the corresponding LBOP.*

Proof: Suppose by contradiction $\overline{X^*}$ to be non optimal for the LBOP. Then there exists a solution $\overline{X'}$ which dominates $\overline{X^*}$ according to (3). But then $f(\overline{X'}) < f(\overline{X^*})$ according to (4). ▮

Remark 1. The proposed approach follows the same lines of action of the first procedure of [1]. The weights vectorization plays here exactly the same role of the weights power raising in [1] avoiding however the numbers explosion. ▮

Corollary 1. *LBOP can be solved in $T_{m,n}(SOP)O(k)$ time.*

In fact, an elementary operation which requires in a SOP solution procedure constant time requires in the corresponding VSOP solution procedure $O(k)$ time as it involves k constant time SOP elementary operations.

Remark 2. The two procedures given by Burkard and Rendl [1] require respective running-times $T_{m,n}(\text{SOP})O(k \log n)$ and $T_{m,n}(\text{SOP})O(\max\{\log n, \log k\} \min\{n^2, k^2\})$, and are therefore dominated by the proposed procedure, for all realistic cases, i.e., the ones where $k \leq n^2 \log k$. Notice that any lexicographic problem is a VSOP which results in a complete and transitive binary relation on the vector space [7] and, as such, always admits an order-preserving numerical representation as a SOP [8] through a non always trivial process like in [1]. Indeed, our gain in efficiency is obtained by directly solving the VSOP instead of its numerical representation. ▮

4 Two illustrative applications

4.1 Lexicographic bottleneck shortest path problem

Consider the well known Dijkstra's algorithm [6] for the sum-optimization shortest path problem. Given a graph $G(N, A)$, let l_{ij} be the cost of the directed arc a_{ij} connecting node i to node j. Denote by $S \subseteq N$ (resp., $\overline{S} \subseteq N$) the set of *labeled* (resp., *unlabeled*) nodes. Let Γ_i br the set of successors of node i (i.e., $\Gamma_i = \{j : l_{ij} \neq 0\}$). Let $\pi(i)$ be the current shortest path value from the source node to node $i \in N$ and let $P(i)$ be the current predecessor of i in that path. A pseudocode of Dijkstra's algorithm, assuming node 1 to be the source node, is the following:

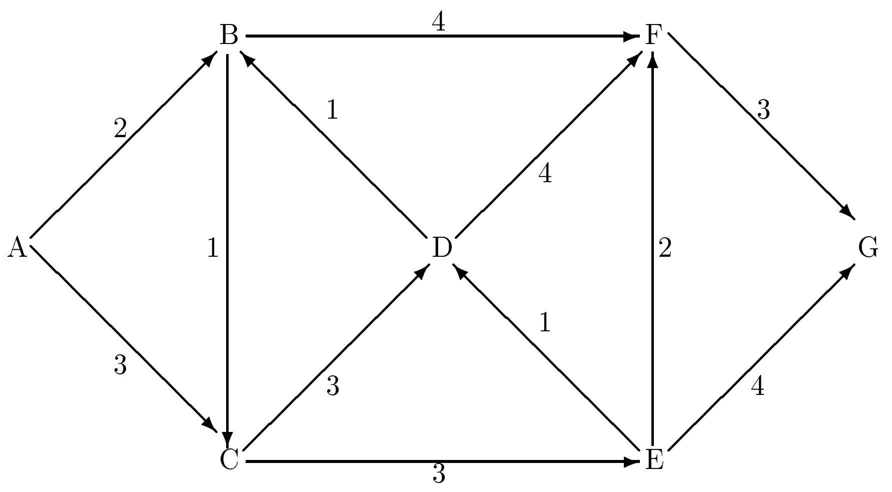

Figure 1: An instance of the lexicographic bottleneck shortest path problem.

```
Step 1:
    $S = \{1\},\ \overline{S} = \{2,\dots,N\};\ \pi(j) = \infty,\ \forall j \notin \Gamma_1;\ \pi(j) = l_{1j},\ P(j) = 1,\ \forall j \in \Gamma_1;$

Step 2:
    find $j \in \overline{S}$ such that $\pi(j) = \min_{i \in \overline{S}}\{\pi(i)\}$; set $S = S \bigcap \{j\},\ \overline{S} = S - \{j\};$
    IF $|\overline{S}| = 0$ RETURN $\pi(i)$;

Step 3:
    $\forall i \in \Gamma_j \bigcap \overline{S}$:   $\pi(i) = \min\{\pi(i), \pi(j) + l_{ji}\};\ P(i) = j$ IF $\pi(i) = \pi(j) + l_{ji};$
    GO TO Step 2;
```

Consider an instance of the lexicographic bottleneck shortest path problem shown in figure 1, where we want to find the shortest path from node A to node G. For this, we will apply Dijkstra's algorithm.

As one can see in figure 1, there are four different cost coefficients (we consider that absence of arc can be considered as arc of cost coefficient ∞), hence the vectorial representation of the costs matrix is as follows (by $\overline{\infty}$ we denote the string $\langle\infty,\infty,\infty,\infty\rangle$):

	A	B	C	D	E	F	G
A	$\overline{\infty}$	$0,0,1,0$	$0,1,0,0$	$\overline{\infty}$	$\overline{\infty}$	$\overline{\infty}$	$\overline{\infty}$
B	$\overline{\infty}$	$\overline{\infty}$	$0,0,0,1$	$\overline{\infty}$	$\overline{\infty}$	$1,0,0,0$	$\overline{\infty}$
C	$\overline{\infty}$	$\overline{\infty}$	$\overline{\infty}$	$0,0,0,1$	$0,1,0,0$	$\overline{\infty}$	$\overline{\infty}$
D	$\overline{\infty}$	$0,0,0,1$	$\overline{\infty}$	$\overline{\infty}$	$\overline{\infty}$	$1,0,0,0$	$\overline{\infty}$
E	$\overline{\infty}$	$\overline{\infty}$	$\overline{\infty}$	$0,0,0,1$	$\overline{\infty}$	$0,0,1,0$	$1,0,0,0$
F	$\overline{\infty}$	$\overline{\infty}$	$\overline{\infty}$	$\overline{\infty}$	$\overline{\infty}$	$\overline{\infty}$	$0,1,0,0$
G	$\overline{\infty}$	$\overline{\infty}$	$\overline{\infty}$	$\overline{\infty}$	$\overline{\infty}$	$\overline{\infty}$	$\overline{\infty}$

The solution procedure works as follows:

```
Step 1:
    $S = \{A\};\ \overline{S} = \{B,C,D,E,F,G\};\ \pi(D) = \pi(E) = \pi(F) = \pi(G) = [\overline{\infty}];$
    $\pi(B) = [0,0,1,0];\ \pi(C) = [0,1,0,0];\ P(B) = P(C) = A;$
```

Step 2:
    $\mathtt{min}_{\mathtt{i}\in\overline{\mathtt{S}}}\{\pi(\mathtt{i})\} = \mathtt{B}$; $\mathtt{S} = \{\mathtt{A},\mathtt{B}\}$; $\overline{\mathtt{S}} = \{\mathtt{C},\mathtt{D},\mathtt{E},\mathtt{F},\mathtt{G}\}$;

Step 3:
    $\Gamma_B \bigcap \overline{S} = \{C, F\}$:
    $\pi(\mathtt{C}) = \mathtt{min}\{\pi(\mathtt{C}), \pi(\mathtt{B}) + [0,0,0,1]\} = [0,0,1,1]$;
    $\pi(\mathtt{F}) = \mathtt{min}\{\pi(\mathtt{F}), \pi(\mathtt{B}) + [1,0,0,0]\} = [1,0,1,0]$; $\mathtt{P(C)} = \mathtt{P(F)} = \mathtt{B}$;

Step 2:
    $\mathtt{min}_{\mathtt{i}\in\overline{\mathtt{S}}}\{\pi(\mathtt{i})\} = \mathtt{C}$; $\mathtt{S} = \{\mathtt{A},\mathtt{B},\mathtt{C}\}$; $\overline{\mathtt{S}} = \{\mathtt{D},\mathtt{E},\mathtt{F},\mathtt{G}\}$;

Step 3:
    $\Gamma_{\mathtt{C}} \bigcap \overline{\mathtt{S}} = \{\mathtt{D},\mathtt{E}\}$:
    $\pi(\mathtt{D}) = \mathtt{min}\{\pi(\mathtt{D}), \pi(\mathtt{C}) + [0,0,0,1]\} = [0,0,1,2]$;
    $\pi(\mathtt{E}) = \min\{\pi(\mathtt{E}), \pi(\mathtt{C}) + [0,1,0,0]\} = [0,1,1,1]$; $\mathtt{P(D)} = \mathtt{P(E)} = \mathtt{C}$;

Step 2:
    $\mathtt{min}_{\mathtt{i}\in\overline{\mathtt{S}}}\{\pi(\mathtt{i})\} = \mathtt{D}$; $\mathtt{S} = \{\mathtt{A},\mathtt{B},\mathtt{C},\mathtt{D}\}$; $\overline{\mathtt{S}} = \{\mathtt{E},\mathtt{F},\mathtt{G}\}$;

Step 3:
    $\Gamma_{\mathtt{D}} \bigcap \overline{\mathtt{S}} = \{\mathtt{F}\}$: $\pi(\mathtt{F}) = \mathtt{min}\{\pi(\mathtt{F}), \pi(\mathtt{D}) + [1,0,0,0]\} = [1,0,1,0]$;

Step 2:
    $\mathtt{min}_{\mathtt{i}\in\overline{\mathtt{S}}}\{\pi(\mathtt{i})\} = \mathtt{E}$; $\mathtt{S} = \{\mathtt{A},\mathtt{B},\mathtt{C},\mathtt{D},\mathtt{E}\}$; $\overline{\mathtt{S}} = \{\mathtt{F},\mathtt{G}\}$;

Step 3:
    $\Gamma_{\mathtt{E}} \bigcap \overline{\mathtt{S}} = \{\mathtt{F},\mathtt{G}\}$:
    $\pi(\mathtt{F}) = \mathtt{min}\{\pi(\mathtt{F}), \pi(\mathtt{E}) + [0,0,1,0]\} = [0,1,2,1]$;
    $\pi(\mathtt{G}) = \mathtt{min}\{\pi(\mathtt{G}), \pi(\mathtt{E}) + [1,0,0,0]\} = [1,1,1,1]$; $\mathtt{P(F)} = \mathtt{P(G)} = \mathtt{E}$;

Step 2:
    $\mathtt{min}_{\mathtt{i}\in\overline{\mathtt{S}}}\{\pi(\mathtt{i})\} = \mathtt{F}$; $\mathtt{S} = \{\mathtt{A},\mathtt{B},\mathtt{C},\mathtt{D},\mathtt{E},\mathtt{F}\}$; $\overline{\mathtt{S}} = \{\mathtt{G}\}$;

Step 3:
    $\Gamma_{\mathtt{F}} \bigcap \overline{\mathtt{S}} = \{\mathtt{G}\}$: $\pi(\mathtt{G}) = \mathtt{min}\{\pi(\mathtt{G}), \pi(\mathtt{F}) + [0,1,0,0]\} = [0,2,2,1]$; $\mathtt{P(G)} = \mathtt{F}$;

Step 2:
    $\mathtt{min}_{\mathtt{i}\in\overline{\mathtt{S}}}\{\pi(\mathtt{i})\} = \mathtt{G}$; $\mathtt{S} = \{\mathtt{A},\mathtt{B},\mathtt{C},\mathtt{D},\mathtt{E},\mathtt{F},\mathtt{G}\}$; $\overline{\mathtt{S}} = \{\}$; RETURN A-B-C-E-F-G;

The lexicographic bottleneck shortest path from A to G is $A-B-C-E-F-G$ and its solution value is $[0,2,2,1]$.

4.2 Lexicographic bottleneck assignment problem

Consider the following lexicographic bottleneck assignment problem. The costs matrix is as follows:

	1	2	3	4
A	9	5	12	9
B	12	6	7	6
C	5	7	6	8
D	8	5	8	9

There are six distinct cost coefficients, hence the vectorial representation of the costs matrix is as follows:

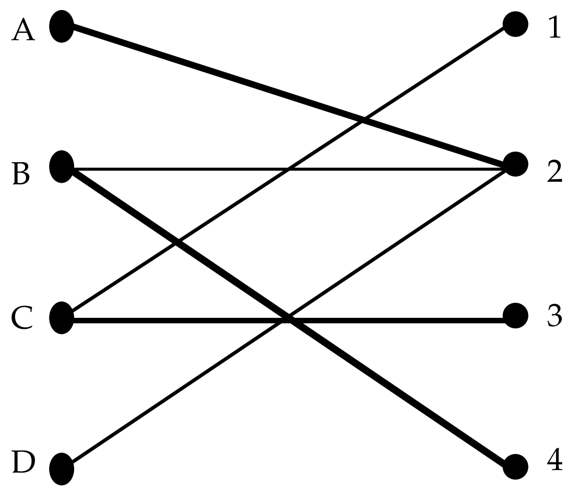

Figure 2: First application of the matching procedure.

	1	2	3	4
A	0,1,0,0,0,0	0,0,0,0,0,1	1,0,0,0,0,0	0,1,0,0,0,0
B	1,0,0,0,0,0	0,0,0,0,1,0	0,0,0,1,0,0	0,0,0,0,1,0
C	0,0,0,0,0,1	0,0,0,1,0,0	0,0,0,0,1,0	0,0,1,0,0,0
D	0,0,1,0,0,0	0,0,0,0,0,1	0,0,1,0,0,0	0,1,0,0,0,0

Consider the application of the Hungarian method [9] to solve the problem (we will use the matrix form version of the algorithm as it is described in [5] pp. 374-375 and its notation). The initial step is the construction of the reduced cost matrix C' by first subtracting to each row the minimum cost element and then doing the same to each column. The rows reduction leads to the following cost matrix:

	1	2	3	4
A	0,1,0,0,0,-1	0,0,0,0,0,0	1,0,0,0,0,-1	0,1,0,0,0,-1
B	1,0,0,0,-1,0	0,0,0,0,0,0	0,0,0,1,-1,0	0,0,0,0,0,0
C	0,0,0,0,0,0	0,0,0,1,0,-1	0,0,0,0,1,-1	0,0,1,0,0,-1
D	0,0,1,0,0,-1	0,0,0,0,0,0	0,0,1,0,0,-1	0,1,0,0,0,-1

Then, the columns reduction leads to the reduced cost matrix C' which is as follows:

	1	2	3	4
A	0,1,0,0,0,-1	0,0,0,0,0,0	1,0,0,0,-1,0	0,1,0,0,0,-1
B	1,0,0,0,-1,0	0,0,0,0,0,0	0,0,0,1,-2,1	0,0,0,0,0,0
C	0,0,0,0,0,0	0,0,0,1,0,-1	0,0,0,0,0,0	0,0,1,0,0,-1
D	0,0,1,0,0,-1	0,0,0,0,0,0	0,0,1,0,-1,0	0,1,0,0,0,-1

A maximum matching is now solved (by means of a labelling procedure which iteratively searches for for augmenting paths) on the bipartite graph G' where an arc (i, j) is present if the corresponding $c'_{i,j}$ entry of the reduced cost matrix C' is zero (in the vectorial representation this corresponds to $c_{i,j} = [0, 0, 0, 0, 0, 0]$). The bipartite graph G' and its maximum matching are shown in figure 2 (the arcs belonging to the matching are depicted in bold).

A perfect matching (no exposed vertices) would correspond to the optimal assignment. As the matching is not perfect, the solution is not optimal and, hence, a hungarian tree has been

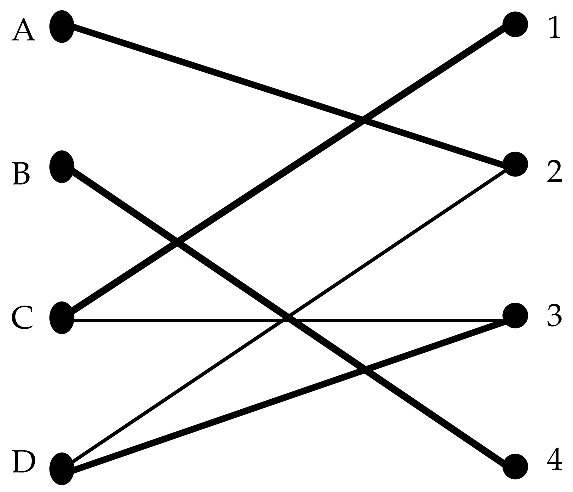

Figure 3: Second application of the matching procedure.

detected: the sets of labelled (unlabelled) rows I^+ (I^-) and columns K^+ (K^-) are the following: $I^+ = \{A, D\}$, $I^- = \{B, C\}$, $K^+ = \{2\}$, $K^- = \{1, 3, 4\}$. Hence, $\Delta = \min_{i \in I^+, k \in K^-}\{c'_{i,k}\} = c'_{D,3} = [0, 0, 1, 0, -1, 0]$. The reduced cost matrix is then updated by setting $c'_{i,k} = c'_{i,k} - \Delta$, for $i \in I^+$ and $k \in K^-$, $c'_{i,k} = c'_{i,k} + \Delta$, for $i \in I^-$ and $k \in K^+$, $c'_{i,k}$ remains unchanged otherwise. The updated reduced cost matrix is as follows:

	1	2	3	4
A	0,1,-1,0,1,-1	0,0,0,0,0,0	1,0,-1,0,0,0	0,1,-1,0,1,-1
B	1,0,0,0,-1,0	0,0,1,0,-1,0	0,0,0,1,-2,1	0,0,0,0,0,0
C	0,0,0,0,0,0	0,0,1,1,-1,-1	0,0,0,0,0,0	0,0,1,0,0,-1
D	0,0,0,0,1,-1	0,0,0,0,0,0	0,0,0,0,0,0	0,1,-1,0,1,-1

The updated G' and its maximum matching are depicted in figure 3.

As the matching is perfect, the assignment $A - 2$, $B - 4$, $C - 1$ and $D - 3$ is optimal.

5 Final remarks

The proposed produre was sought to handle LBOP. However, it can be applied to any problem for which the following conditions hold: ordinality of the scale associated to the cost-vector and existence of at least a weak order on any subset of feasible solutions (which is always the case in a lexicographic order). The following steps may be considered in a future research:

- verify if it is possible to weaken the above necessary conditions without increasing the time-complexity of the proposed procedure;
- devise exact or approximation algorithms for problems where there exists either a partial order or a simple reflexive binary relation on the solution set.

Acknowledgement. Many thanks to an anonymous referee for very pertinent and helpful comments and suggestions.

References

[1] R.E. Burkard, F. Rendl - "Lexicographic bottleneck problems" - *Operations Research Letters* 10 (1991), 303-308.

[2] R.E. Burkard, W. Hahn, U. Zimmermann - "An algebraic approach to assignment problems" - *Mathematical Programming* 12 (1977), 318-327.

[3] R.E. Burkard, U. Zimmermann - "Combinatorial Optimization in Linearly Ordered Semimodules: a survey" in Modern Applied Mathematics: Optimization and Operations Research, *B. Korte (ed.)*, **North Holland**, 1982.

[4] H.I. Calvete, P.M. Mateo - "Lexicographic optimization in generalized network flow problems" - *Journal of the Operational Research Society*, 49(5) (1998), 519-529.

[5] N. Christophides - "Graph Theory an algorithmic approach" - **Academic Press**, 1975.

[6] E.W. Dijkstra - "A note on two problems in connection with graphs" - *Numerische Mathematik* 1 (1959), 269-271.

[7] P.C. Fishburn - "Lexicographic orders, utilities and decision rules: a survey" - *Management Science* 20 (1974), 1442-1471.

[8] D. Scott, P. Suppes - "Foundational Aspects of Theories of Measurement" - *Journal of Symbolic Logic*, 23 (1958), 113-128.

[9] H.W. Kuhn - "The Hungarian method for the assignment problem" - *Naval Research Logistics Quarterly* 2 (1955), 83-97.

[10] A.P. Punnen, K.P.K. Nair - "Improved complexity bound for the maximum cardinality bottleneck bipartite matching problem" - *Discrete Applied Mathematics* 55 (1994), 91-93.

[11] Ph. Vincke - *Multicriteria decision aid* - **J. Wiley**, 1992.

[12] U. Zimmermann - "Some partial orders related to Boolean optimization and the greedy algorithm" - *Annals of Discrete Mathematics* 1 (1977), 539-550.