J Optim Theory Appl (2007) 134: 483–497
DOI 10.1007/s10957-007-9232-y

Primal-Dual Simplex Method for Multiobjective Linear Programming

M. Ehrgott · J. Puerto · A.M. Rodríguez-Chía

Published online: 12 July 2007

Abstract We develop a primal-dual simplex algorithm for multicriteria linear programming. It is based on the scalarization theorem of Pareto optimal solutions of multicriteria linear programs and the single objective primal-dual simplex algorithm. We illustrate the algorithm by an example, present some numerical results, give some further details on special cases and point out future research.

Keywords Multiobjective linear programming · Primal-dual simplex algorithm

1 Introduction

Multiobjective linear programming has been a topic of research since the 1960s. Naturally, early research focused on extensions of the simplex algorithm to deal with

Communicated by H.P. Benson.

The paper was written during a visit of the first author to the University of Sevilla financed by a grant of the Andalusian Consejería de Educación. The research of the first author was partially supported by University of Auckland Grant 3602178/9275. The research of the second and third authors was partially financed by Spanish Grants BFM2001-2378, BFM2001-4028, MTM2004-0909 and HA2003-0121.
We thank Anthony Przybylski for the implementation and making his results available. We thank the anonymous referees, whose comments have helped us to improve the presentation of the paper.

M. Ehrgott (✉)
Department of Engineering Science, University of Auckland, Auckland, New Zealand
e-mail: m.ehrgott@auckland.ac.nz

J. Puerto
Departamento de Estadística e Investigación Operativa, Universidad de Sevilla, Sevilla, Spain

A.M. Rodríguez-Chía
Departamento de Estadística e Investigación Operativa, Universidad de Cádiz, Cádiz, Spain

multiple objectives. Algorithms based on the simplex method have been proposed, see the brief review and the references in [1]. Steuer developed an implementation (called ADBASE [2]) of his multicriteria simplex method.

More recently, algorithms to solve multiobjective linear programs (MOLP) in objective space have been developed. These are motivated by the fact that the dimension of the objective space is usually much smaller than the one of the decision space, and therefore the number of efficient extreme points in objective space is much smaller than the number of Pareto optimal extreme points in decision space. Research along these lines is included in [3] and the references therein.

After the discovery of interior point algorithms to solve linear programs (LP) in polynomial time, efforts to apply such methods to deal with MOLP are evident, most notably [4], and other works which comprise both affine scaling [5] and primal-dual [6] interior point methods to find a most preferred Pareto optimal solution for a decision maker.

Another focus of interest in MOLP is based on scalarization. Recall that every Pareto optimal solution of an MOLP is an optimal solution of a scalar LP that minimizes a weighted sum of the objectives. This has drawn interest to weight space decomposition methods. Such decomposition can be done with respect to efficient bases of the MOLP or with respect to extreme points of the Pareto optimal set in decision space or the efficient set in outcome space [7, 8].

Duality theory of (single objective) linear programming is the basis of the primal-dual simplex algorithm. Duality theory for multiobjective linear programming has been studied (see, e.g., [9]) but as far as we know that has not lead to the development of dual or primal-dual simplex methods for multiobjective linear programming. One of the reasons for this might be that multiobjective duality theory cannot easily be used to develop an algorithm analogous to the single objective case, due to the different structure of the complementary slackness condition.

However, primal-dual simplex algorithms have proven to be very efficient for several classes of single objective linear programming problems, in particular those related to network optimization problems. These include the Hungarian method for assignment problems, the augmenting path method for minimum cost flow problems, Dijsktra's algorithm for shortest path problems, etc. (see, e.g., [10]).

In this paper we introduce a primal-dual simplex algorithm for MOLP in order to investigate whether these features can be extended to multiobjective problems. We develop this algorithm with a view to applications in network optimization problems. As in the single objective case, it cannot be expected that the primal-dual simplex algorithm is competitive with primal simplex algorithms for solving a general multiobjective linear program. But when solving an LP with a network structure, some steps of the primal-dual simplex algorithm can be carried out very efficiently. Once a primal-dual simplex algorithm for multiobjective linear programming is available, these issues can be investigated for the multiobjective case, too.

In this paper we use scalarization of MOLP and single objective duality theory to develop such an algorithm. In addition to finding efficient solutions of a multiobjective linear program our algorithm will also compute a weight space decomposition. This is extremely important for the development of algorithms to solve multiobjective combinatorial optimization problems. Most exact algorithms for these problems

are based on the two phase method (see, e.g., [11]). This approach implies finding solutions of weighted sum scalarizations of these problems in the first phase, yielding supported Pareto optimal solutions (or supported efficient points) and the associated weight space decomposition. The second phase uses the weight space decomposition to find Pareto optimal solutions that are not optimal for any weighted sum problem. At present, the two phase method has been very successfully used for biobjective combinatorial optimization problems, but not yet for more than two objectives. Thus, if our primal-dual simplex method can be applied to solve linear network problems with three criteria, an important advance in the solution of integer network problems with three criteria can be made. For this research to become possible, we develop a multicriteria primal-dual simplex algorithm in this paper.

2 Multiobjective Linear Programming

We consider the following multiobjective linear programming problem:

$$\begin{array}{lll} \text{(MOLP)} & \min & \left((c^1)^T x, \ldots, (c^k)^T x\right)^T, \\ & \text{s.t.} & Ax = b, \\ & & x \geq 0, \end{array}$$

where $c^i \in \mathbb{R}^n$ for $i = 1, \ldots, k$ are the objective functions, $A \in \mathbb{R}^{m \times n}$ is the constraint matrix, $b \in \mathbb{R}^m$ is the right hand side vector and $x \in \mathbb{R}^n$ is a vector of variables. The superscript T over a vector or matrix denotes the transpose of the corresponding vector or matrix. We shall denote the feasible set of the MOLP by X. In the following we assume, without loss of generality, that X is non empty. The objective function can be written as $C^T x$, where $C \in \mathbb{R}^{n \times k}$ has columns c^i.

A solution $x^* \in X$ of MOLP is (weakly) Pareto optimal if there is no $x \in X$ such that $C^T x \leq C^T x^*$ and $C^T x \neq C^T x^*$ ($C^T x < C^T x^*$). If x^* is (weakly) Pareto optimal, $C^T x^*$ is called (weakly) efficient. The set of Pareto optimal solutions is denoted by X_{Par} and the set of efficient points is denoted $Y_{\text{eff}} := C^T X_{\text{Par}}$.

A fundamental result of multiobjective linear programming states that (weakly) Pareto optimal solutions of MOLP can be characterized as optimal solutions of single objective linear programs with a convex combination of objectives $c^i, i = 1, \ldots, k$. Let $\Lambda = \{\lambda \in \mathbb{R}^k : \lambda \geq 0, \sum_{i=1}^k \lambda_i = 1\}$ and $\Lambda^0 = \{\lambda \in \mathbb{R}^k : \lambda > 0, \sum_{i=1}^k \lambda_i = 1\}$. We will refer to Λ respectively Λ^0 as parameter space or weight space.

Theorem 2.1 *A feasible point $x \in X$ is (weakly) Pareto optimal for MOLP if and only if there exists $\lambda \in \Lambda^0$ ($\lambda \in \Lambda$) such that x is an optimal solution of*

$$\begin{array}{lll} \text{(P}(\lambda)\text{)} & \min & c(\lambda)^T x := \displaystyle\sum_{i=1}^{k} \lambda_i (c^i)^T x, \\ & \text{s.t.} & Ax = b, \\ & & x \geq 0. \end{array}$$

A proof of this result can be found in [12]. In the following, we will refer to the components of the vector

$$c(\lambda)^T = (c_1(\lambda), \ldots, c_n(\lambda)),$$

with

$$c_j(\lambda) = \sum_{i=1}^{k} \lambda_i c_j^i, \quad \text{for } j = 1, \ldots, n.$$

Our algorithm is based on duality theory of P(λ). The dual D(λ) of P(λ) is given by

$$\begin{array}{lll} (\mathrm{D}(\lambda)) & \max & u^T b, \\ & \text{s.t.} & u^T A \leq c(\lambda)^T. \end{array}$$

From duality theory follows that the complementary slackness conditions between P(λ) and D(λ) in our case are: If x is primal feasible and u is dual feasible, then x and u are optimal for their respective problems if and only if $(u^T A - c(\lambda)^T)x = 0$. Combining Theorem 2.1 and the complementary slackness condition we get:

Theorem 2.2 *A feasible solution $x \in X$ is (weakly) Pareto optimal for MOLP if and only if there exists some $\lambda \in \Lambda^0$ ($\in \Lambda$) and u being a feasible solution of* D(λ) *such that $(u^T A - c(\lambda)^T)x = 0$.*

In order to develop the primal-dual multiobjective simplex algorithm it is necessary to have feasible dual problems. Notice that this is the same hypothesis made in the scalar case. For any $\lambda \in \Lambda$, dual feasibility implies boundedness of the objective value of P(λ). One condition to guarantee this is stated in the following lemma.

Lemma 2.1 *The problems* D(λ) *are feasible $\forall \lambda \in \Lambda$ if $\min\{c^T x : x \in X\}$ is bounded $\forall c$ in* cone(C), *where* cone(C) *is the cone generated by the columns of C.*

Corollary 2.1 *Let $\bar{c}$ be the componentwise minimum of the vectors c^i, $i = 1, \ldots, k$. Then,* D(λ) *is feasible for all $\lambda \in \Lambda$ if $\min\{\bar{c}^T x : x \in X\}$ is bounded.*

Special easy and interesting cases, because all the combinatorial optimization problems belong to this class, are those cases where all $c^i \geq 0$ for $i = 1, \ldots, k$ or X is a bounded set. In the former case, $u = 0$ is a feasible solution of D(λ) $\forall \lambda \in \Lambda$. In the latter case $\bar{c}$ guarantees feasibility, in fact, this condition is stronger because it ensures that there exists a common feasible point for D(λ) $\forall \lambda \in \Lambda$.

3 Primal-Dual Simplex Algorithm for Multiobjective Linear Programming

As pointed out in the introduction, it is well-known in the multiobjective linear programming literature that there exists a finite partition of the parameter space Λ such that any Pareto optimal solution of MOLP is associated with an element of this partition. Therefore, it is worth analyzing the structure of the primal-dual subsets of the

parameter space. In the description of the algorithm, we follow [13] and make adaptations where necessary.

Let us assume that we are given a set $\bar{\Lambda} \subset \Lambda$ and a dual solution $u_{\bar{\Lambda}}$ which is feasible for D(λ) for all $\lambda \in \bar{\Lambda}$ (we will briefly discuss how to deal with other cases in Sect. 5). For $\lambda \in \bar{\Lambda}$, we define the set

$$Q(\lambda) = \left\{ j : (u_{\bar{\Lambda}})^T a_j = c_j(\lambda) \right\}, \tag{1}$$

where a_j is the jth column of matrix A. Notice that there might be different sets $Q(\lambda)$ for different values of $\lambda \in \bar{\Lambda}$. We will say that $\hat{\Lambda} \subset \Lambda$ is maximal with respect to $Q(\lambda)$ if, for some $\hat{\lambda} \in \hat{\Lambda}$,

$$Q(\hat{\lambda}) = Q(\lambda), \quad \forall \lambda \in \hat{\Lambda} \quad \text{and} \quad Q(\hat{\lambda}) \neq Q(\lambda), \quad \forall \lambda \in \Lambda \setminus \hat{\Lambda}. \tag{2}$$

Therefore, for any maximal set $\hat{\Lambda}$ we denote $\mathcal{Q}(\hat{\Lambda}) = Q(\hat{\lambda})$ for some $\hat{\lambda} \in \hat{\Lambda}$ and set $u_{\hat{\Lambda}}(\lambda) = u_{\bar{\Lambda}}$ $\forall \lambda \in \hat{\Lambda}$. For this set $\mathcal{Q}(\hat{\Lambda})$, we define the restricted primal problem

$$\begin{array}{lll} (\mathrm{RP}(\hat{\Lambda})) & \min & \mathbf{1}^T y, \\ & \text{s.t.} & Ax + y = b, \\ & & x_i = 0, \quad i \notin \mathcal{Q}(\hat{\Lambda}), \\ & & x, y \geq 0, \end{array}$$

where $\mathbf{1}$ is a vector of ones. If the optimal objective value of RP($\hat{\Lambda}$) is zero, its optimal solution $\hat{x}$ is optimal for P(λ) for any $\lambda \in \hat{\Lambda}$. This follows easily from the single objective primal-dual algorithm and complementary slackness conditions, just as in the single objective case. If this, however, is not the case, we can formulate the dual of RP($\hat{\Lambda}$), called DRP($\hat{\Lambda}$),

$$\begin{array}{lll} (\mathrm{DRP}(\hat{\Lambda})) & \max & w^T b, \\ & \text{s.t.} & w^T a_j \leq 0, \quad j \in \mathcal{Q}(\hat{\Lambda}), \\ & & w \leq \mathbf{1}. \end{array}$$

Let $\hat{w}(\hat{\Lambda})$ be an optimal solution of DRP($\hat{\Lambda}$). If there is no $j \notin \mathcal{Q}(\hat{\Lambda})$ such that $\hat{w}(\hat{\Lambda})^T a_j > 0$ we conclude that P(λ) is infeasible for all $\lambda \in \hat{\Lambda}$, a fact which again follows easily from the single objective primal-dual algorithm and complementary slackness. In this case MOLP is infeasible. Otherwise we define, for $\lambda \in \hat{\Lambda}$,

$$\hat{\varepsilon}(\lambda) = \min_j \left\{ \frac{c_j(\lambda) - (u_{\hat{\Lambda}}(\lambda))^T a_j}{\hat{w}(\hat{\Lambda})^T a_j} : \hat{w}(\hat{\Lambda})^T a_j > 0 \right\}. \tag{3}$$

Notice that $\hat{\varepsilon}(\lambda)$ might have different definitions for different values of $\lambda \in \hat{\Lambda}$. We will say that $\Lambda^* \subset \hat{\Lambda}$ is maximal with respect to $\hat{\varepsilon}(\lambda)$ if for some j such that $\hat{w}(\hat{\Lambda})^T a_j > 0$ it satisfies

$$\hat{\varepsilon}(\lambda) = \frac{c_j(\lambda) - (u_{\hat{\Lambda}}(\lambda))^T a_j}{\hat{w}(\hat{\Lambda})^T a_j}, \quad \forall \lambda \in \Lambda^*, \tag{4a}$$

$$\hat{\varepsilon}(\lambda) \neq \frac{c_j(\lambda) - (u_{\hat{\Lambda}}(\lambda))^T a_j}{\hat{w}(\hat{\Lambda})^T a_j}, \quad \forall \lambda \in \hat{\Lambda} \setminus \Lambda^*. \tag{4b}$$

Denote

$$\hat{\varepsilon}_{\Lambda^*}(\lambda) = \hat{\varepsilon}(\lambda), \quad \lambda \in \Lambda^*.$$

Then, define

$$u_{\Lambda^*}(\lambda) = u_{\hat{\Lambda}}(\lambda) + \hat{\varepsilon}_{\Lambda^*}(\lambda)\hat{w}(\hat{\Lambda}), \quad \forall \lambda \in \Lambda^*. \tag{5}$$

In this way we arrive at a refinement of $\bar{\Lambda}$ into a partition of smaller sets $\Lambda^* \subset \bar{\Lambda}$, for each of which we know an improved dual feasible solution $u_{\Lambda^*}(\lambda)$ that is feasible for D(λ) for all $\lambda \in \Lambda^*$. We can now treat these sets Λ^* as new sets $\bar{\Lambda}$ and continue in the same way.

Notice that the restricted primal and dual problems RP($\hat{\Lambda}$) and DRP($\hat{\Lambda}$) depend on $\hat{\Lambda}$ only in the way the columns from A are chosen according to $\mathcal{Q}(\hat{\Lambda})$. However, the coefficients of those problems are independent of any λ values so that $\hat{w}(\hat{\Lambda})$ is well defined. The previous description leads to the formulation of Algorithm 3.1 as a branching scheme reflecting refinements of the partition of $\hat{\Lambda}$ according to the computation of $\mathcal{Q}(\hat{\Lambda})$ and $\hat{\varepsilon}_{\Lambda^*}(\lambda)$.

Algorithm 3.1 (Primal-dual simplex method for MOLP)

Step 1 Choose $u_{\bar{\Lambda}}$, a dual feasible solution of D(λ) for all $\lambda \in \bar{\Lambda}$ and compute the partition $\{\hat{\Lambda}_i : i \in I_0\}$ of $\bar{\Lambda}$ in the sense of (2).

Step 2 For each $i \in I_0$, compute $\mathcal{Q}(\hat{\Lambda}_i)$ as in (1), set $u_{\hat{\Lambda}_i}(\lambda) = u_{\bar{\Lambda}}$ for all $\lambda \in \hat{\Lambda}_i$ and $\mathcal{L} = \mathcal{L} \cup \{(\hat{\Lambda}_i, u_{\hat{\Lambda}_i}(\lambda)\}$.

Step 3 While $\mathcal{L} \neq \emptyset$, choose $(\hat{\Lambda}, u_{\hat{\Lambda}}(\lambda)) \in \mathcal{L}$ and solve RP($\hat{\Lambda}$).

- If the optimal value is 0, stop consideration of $\hat{\Lambda}$. An optimal solution of P(λ) for all $\lambda \in \hat{\Lambda}$ is found. Set $\mathcal{L} = \mathcal{L} \setminus \{(\hat{\Lambda}, u_{\hat{\Lambda}}(\lambda))\}$.
- Otherwise solve DRP($\hat{\Lambda}$) and let $\hat{w}(\hat{\Lambda})$ be an optimal solution.
  - If there is no $j \notin \mathcal{Q}(\hat{\Lambda})$ such that $\hat{w}(\hat{\Lambda})^T a_j > 0$, P($\lambda$) is infeasible for all $\lambda \in \hat{\Lambda}$ and the multiobjective linear program is infeasible.
  - Otherwise, compute the partition $\{\Lambda_l^* : l \in I^*\}$ of $\hat{\Lambda}$, where each Λ_l^* is maximal in the sense (4). For each $l \in I^*$, compute $\hat{\varepsilon}_{\Lambda_l^*}(\lambda)$ according to (3) and update $u_{\Lambda_l^*}(\lambda)$ according to (5). Compute $\mathcal{Q}(\Lambda_l^*)$ and set $\mathcal{L} = \mathcal{L} \cup \{(\Lambda_l^*, u_{\Lambda_l^*}(\lambda))\}$. Set $\mathcal{L} = \mathcal{L} \setminus \{(\hat{\Lambda}, u_{\hat{\Lambda}}(\lambda))\}$.

In order to prove finiteness and correctness of the algorithm, we first prove that u is always an affine function of λ.

Lemma 3.1 *At each stage of the algorithm, $u_{\hat{\Lambda}}(\lambda)$ is an affine function of $\lambda \in \hat{\Lambda}$.*

Proof At step 1, $u_{\hat{\Lambda}}(\lambda) = u_{\bar{\Lambda}}$ is constant. Now assume $u_{\hat{\Lambda}}(\lambda)$ is an affine function of λ for $\lambda \in \hat{\Lambda}$. We show that it remains an affine function of λ after the update (5) in step 3. For any $\Lambda^* \subset \hat{\Lambda}$ as in the algorithm $u_{\Lambda^*}(\lambda) = u_{\hat{\Lambda}}(\lambda) + \hat{\varepsilon}_{\Lambda^*}(\lambda)\hat{w}(\hat{\Lambda})$ $\forall \lambda \in \Lambda^*$.

Here $u_{\hat{\Lambda}}(\lambda)$ is an affine function of λ by assumption, $\hat{w}(\hat{\Lambda})$ is constant and $\hat{\varepsilon}_{\Lambda^*}(\lambda)$ is of the form $(c_j(\lambda) - u_{\hat{\Lambda}}(\lambda)^T a_j)/\hat{w}(\hat{\Lambda})^T a_j$. By assumption on $u_{\hat{\Lambda}}(\lambda)$, because $\hat{w}(\hat{\Lambda})$ is constant, and because $c_j(\lambda)$ is affine in λ this is an affine function of λ. □

Theorem 3.1 *Let the MOLP be nondegenerate. Then, Algorithm 3.1 is finite and at termination the output gives an optimal solution of* P(λ) *for each* $\lambda \in \Lambda$.

Proof To prove finiteness of the algorithm we proceed in two steps.

(i) The number of branches from each node is finite.

To show this, we need to show that there is a finite number of subsets $\hat{\Lambda}$ of Λ (in Step 2) and Λ^* of $\hat{\Lambda}$ (in Step 3) considered. Observe that these refinements depend on the calculation of $\mathcal{Q}(\hat{\Lambda})$ and $\hat{\varepsilon}_{\Lambda^*}(\lambda)$ respectively.

For the computation of $\mathcal{Q}(\hat{\Lambda})$ we consider the equations $u(\lambda)^T a_j = c_j(\lambda)$ (see (1)) which are linear equations in λ for $\lambda \in \hat{\Lambda}$ according to Lemma 3.1. There is a finite number of systems of equalities of this type, choosing from the n columns of A. The solution set of each of these is a polyhedron and therefore a finite number of polyhedral subsets $\hat{\Lambda} \subset \Lambda$ has to be considered.

For the computation of $\hat{\varepsilon}_{\Lambda^*}(\lambda)$ note that finding the minimum in (3) amounts to the computation of the lower envelope of affine functions which is defined by a finite number of those functions. So, again, a finite number of subsets of $\Lambda^* \subset \hat{\Lambda}$ has to be considered.

(ii) The length of each branch is finite.

After the computation of $\mathcal{Q}(\hat{\Lambda})$, we have that $\hat{w}(\hat{\Lambda})^T a_j \leq 0$ for all $j \in \mathcal{Q}(\hat{\Lambda})$. If there is no $j \in \{1, \ldots, n\} \setminus \mathcal{Q}(\hat{\Lambda})$ such that $\hat{w}(\hat{\Lambda})^T a_j > 0$ the vector $u_{\hat{\Lambda}}(\lambda) + \varepsilon \hat{w}(\hat{\Lambda})$ is feasible for the dual D(λ), $\lambda \in \hat{\Lambda}$ for all $\varepsilon \geq 0$. As ε increases this yields an unbounded solution to the dual D(λ), i.e. an infeasible primal P(λ) and therefore an infeasible MOLP.

If, on the other hand $\hat{w}(\hat{\Lambda})^T a_j > 0$ for at least one j, $u_{\hat{\Lambda}}(\lambda) + \varepsilon \hat{w}(\hat{\Lambda})$ is feasible for the dual D(λ), $\lambda \in \hat{\Lambda}$ for small positive ε. We increase $\varepsilon(\lambda)$ to the first point where one of inequalities $\hat{w}^T(\lambda) a_j < c_j(\lambda)$, $j \notin \mathcal{Q}(\hat{\Lambda})$ becomes an equality. This determines $\hat{\varepsilon}_{\Lambda^*}(\lambda)$ for $\lambda \in \Lambda^* \subseteq \hat{\Lambda}$ and an index $k(\Lambda^*)$. The new $u_{\Lambda^*}(\lambda)$ vector corresponds to an increased value of the dual objective $u_{\Lambda^*}(\lambda)^T b = u_{\hat{\Lambda}}(\lambda)^T b + \hat{\varepsilon}_{\Lambda^*}(\lambda) \hat{w}(\hat{\Lambda})^T b$. In addition, the corresponding new set $\mathcal{Q}(\Lambda^*)$ now includes the index $k(\Lambda^*)$. Any other index i that corresponded to a positive value of x_i in the associated restricted primal is in the new set $\mathcal{Q}(\Lambda^*)$, because by complementary slackness $\hat{w}(\hat{\Lambda})^T a_i = 0$ for such an i and thus $u_{\Lambda^*}(\lambda)^T a_i = u_{\hat{\Lambda}}(\lambda)^T a_i + \hat{\varepsilon}_{\Lambda^*}(\lambda) \hat{w}(\hat{\Lambda})^T a_i = c_i(\Lambda^*)$. This means that the old optimal solution is feasible for the new restricted primal and that if $x_{k(\Lambda^*)}$ enters the basis the value of the associated restricted primal will decrease.

In summary, it has been shown that at each step either an improvement in the associated primal is made or an infeasibility condition is detected. Assuming non-degeneracy, this implies that no basis of the associated primal is repeated and since there are only a finite number of possible bases, an optimal solution for a subset of Λ (which is possibly a subset of Λ^*), is reached in a finite number of steps. Optimality of the algorithm follows, because we consider partitions of the set Λ, and the fact that the single objective primal-dual algorithm is correct (i.e. produces an optimal solution of P(λ) for each $\lambda \in \bar{\Lambda}$). □

Notice that the nondegeneracy assumption is purely technical and the result holds as long as a mechanism that prevents cycling among a set of bases of the primal is employed—those problems for which the single objective primal-dual simplex

method is most successful are always degenerate, a fact that does not cause any problem for the resulting combinatorial algorithms.

In order to effectively apply the algorithm the computation of $\mathcal{Q}(\hat{\Lambda})$ and of $\hat{\varepsilon}_{\Lambda^*}$ must be described in more detail.

Since $u_{\bar{\Lambda}}$ and a_j do not depend on λ, we have that $u_{\bar{\Lambda}}^T a_j = c_j(\lambda)$ defines a hyperplane $h_j(\lambda) = \{\lambda : u_{\bar{\Lambda}}^T a_j = c_j(\lambda)\}$ for all $j = 1, \ldots, n$. Consider the family of hyperplanes $\mathcal{H} = \{h_j(\lambda), j = 1, \ldots, n\}$. $\mathcal{H}$ induces an arrangement on $\bar{\Lambda}$. This arrangement of hyperplanes is the partition of $\bar{\Lambda}$ into the connected pieces induced by the hyperplanes in $\mathcal{H}$. Notice that any face $\hat{\Lambda}_i$ of this arrangement with dimension not greater than $k-2$ is maximal in the sense (2), $\mathcal{Q}(\hat{\Lambda}_i)$ being the set of indices of the hyperplanes which determine this face. According to [14] the construction of an arrangement of n hyperplanes in dimension $k-1$ (the dimension of Λ) can be done in $O(n^{k-1})$ time. Therefore, the complexity of performing the steps before the while loop of Algorithm 3.1 is bounded above by $O(n^{k-1})$.

The following lemma shows that often we do not even have to compute the arrangement of hyperplanes.

Lemma 3.2 *Assume that the entries of C are such that* $c_j(\lambda) > 0$ *for all* $j = 1, \ldots, n$ *and all* $\lambda \in \bar{\Lambda}$. *Then,* $u_{\bar{\Lambda}} = 0$ *is feasible for* $\mathrm{D}(\lambda)$ *for all* $\lambda \in \bar{\Lambda}$ *and the partition computed in Step* 1 *consists of* $\bar{\Lambda}$ *only.*

Proof As mentioned after Corollary 2.1, under the assumptions of the lemma $u_{\bar{\Lambda}} = 0$ is feasible for $\mathrm{D}(\lambda)\, \forall \lambda \in \bar{\Lambda}$. Since the left hand side of the constraints in $\mathrm{D}(\lambda)$ becomes 0, but the right hand side is strictly positive, we have $Q(\lambda) = \emptyset\, \forall \lambda \in \bar{\Lambda}$. □

Lemma 3.3 *In Step* 3 *of Algorithm* 3.1, *we have*

$$\mathcal{Q}(\Lambda_l^*) = \left\{ s : \hat{\varepsilon}_{\Lambda_l^*}(\lambda) = \frac{c_s(\lambda) - u_{\hat{\Lambda}}(\lambda)^T a_s}{\hat{w}(\hat{\Lambda})^T a_s} \forall \lambda \in \Lambda_l^* \text{ with } \hat{w}(\hat{\Lambda})^T a_s > 0 \right\} \cup \left\{ s \in \mathcal{Q}(\hat{\Lambda}) : \hat{w}(\hat{\Lambda})^T a_s = 0 \right\}.$$

Proof We have that $s \in \mathcal{Q}(\Lambda_l^*)$ if and only if $u_{\Lambda_l^*}(\lambda)^T a_s = c_s(\lambda)$ for all $\lambda \in \Lambda_l^*$. Moreover, since $u_{\Lambda_l^*}(\lambda) = u_{\hat{\Lambda}}(\lambda) + \hat{\varepsilon}_{\Lambda_l^*}(\lambda)\hat{w}(\hat{\Lambda})$, we have that $u_{\Lambda_l^*}(\lambda)^T a_s = u_{\hat{\Lambda}}(\lambda)^T a_s + \hat{\varepsilon}_{\Lambda_l^*}(\lambda)\hat{w}(\hat{\Lambda})^T a_s$. Thus,

(i) If $s \in \mathcal{Q}(\hat{\Lambda})$ then $u_{\hat{\Lambda}}(\lambda)^T a_s = c_s(\lambda)$ for all $\lambda \in \hat{\Lambda}$. Thus, since $\Lambda^* \subseteq \hat{\Lambda}$, we have that $s \in \mathcal{Q}(\Lambda_l^*)$ if and only if $\hat{w}(\hat{\Lambda})^T a_s = 0$.

(ii) If $s \notin \mathcal{Q}(\hat{\Lambda})$, then $u_{\Lambda_l^*}(\lambda)^T a_s < c_s(\lambda)^T$. Thus, we have that $s \in \mathcal{Q}(\Lambda_l^*)$ if and only if $\hat{w}(\hat{\Lambda})^T a_s \neq 0$. Moreover, since $u_{\hat{\Lambda}}(\lambda)^T a_s < c_s(\lambda)^T$ and $\hat{\varepsilon}_{\Lambda_l^*}(\lambda) > 0$ then $\hat{w}(\hat{\Lambda})^T a_s$ must be greater than 0. Therefore, isolating $\hat{\varepsilon}_{\Lambda_l^*}(\lambda)$ in the expression defining $u_{\Lambda_l^*}(\lambda)$, we have that $s \in \mathcal{Q}(\Lambda_l^*)$ if and only if

$$s \in \left\{ r : \hat{\varepsilon}_{\Lambda_l^*}(\lambda) = \frac{c_r(\lambda) - u_{\hat{\Lambda}}(\lambda)^T a_r}{\hat{w}(\hat{\Lambda})^T a_r} \forall \lambda \in \Lambda^* \text{ with } \hat{w}(\hat{\Lambda})^T a_r > 0 \right\}. \qquad \square$$

Notice that, in Step 3, the computation of $\mathcal{Q}(\Lambda^*)$ does not induce a refinement of Λ_l^* because the set of indices which determine the value of $\hat{\varepsilon}_{\Lambda_l^*}(\lambda)$ is constant in Λ_l^* (by the definition of Λ_l^*) and $\hat{w}(\hat{\Lambda})^T a_s$ does not depend on λ. Thus, within the while loop, the refinement of $\hat{\Lambda}$ is only produced by the computation of $\hat{\varepsilon}_{\Lambda_l^*}(\lambda)$.

Moreover, the computation of $\hat{\varepsilon}_{\Lambda_l^*}(\lambda)$ for $\lambda \in \Lambda_l^*$ in step 3 reduces to determining the lower envelope of the finite set of affine hyperplanes

$$\left\{ \frac{c_j(\lambda) - u_{\hat{\Lambda}}(\lambda)^T a_j}{\hat{w}(\hat{\Lambda})^T a_j} \text{ with } \hat{w}(\hat{\Lambda})^T a_j > 0 \right\}.$$

The lower envelope of a finite set of hyperplanes is a convex polyhedron. This polyhedron can be computed using geometric duality: Identify each hyperplane with the dual point; then the lower envelope of the original hyperplanes coincides with the convex hull of the dual points (in the dual space). Convex hulls of n points in dimension k can be computed in $O(n^k)$ (see [14]). Hence, transforming back the convex hull to the primal space, the lower envelope of the original set of hyperplanes is obtained with the same complexity. With this we have proven the following lemma.

Lemma 3.4 *All sets Λ^* considered in the algorithm are (open or closed) polyhedra.*

Some further results are useful for implementation of the algorithm.

Lemma 3.5 *Let $\Lambda_1,\ \Lambda_2 \subseteq \Lambda$ such that $\mathcal{Q}(\Lambda_1) \subseteq \mathcal{Q}(\Lambda_2)$. Additionally, assume that x is an optimal solution of* P(λ) $\forall \lambda \in \Lambda_1$. *Then, x is also optimal for* P(λ), $\forall \lambda \in \Lambda_2$.

Proof Because x is optimal for P(λ) for all $\lambda \in \Lambda_1$ we know that $x, y = 0$ is an optimal solution of RP(Λ_1). Therefore $x, y = 0$ is feasible (and thus optimal) for RP(Λ_2) because $\mathcal{Q}(\Lambda_1) \subseteq \mathcal{Q}(\Lambda_2)$ and hence x is optimal for P(λ) for all $\lambda \in \Lambda_2$. □

A special case where the situation of Lemma 3.5 occurs is shown in Lemma 3.6.

Lemma 3.6 $\mathcal{Q}(\Lambda') \subseteq \mathcal{Q}(cl(\Lambda'))$ *for $\Lambda' \subseteq \Lambda$, where cl denotes the closure of a set.*

Proof This follows because in the equations $u(\lambda)^T a_j = c_j(\lambda)$ both $u(\lambda)$ and $c_j(\lambda)$ are affine and continuous functions of λ. □

In the algorithm the distinction between open and closed sets and the choice of Q using the open sets only may be used to distinguish Pareto optimal and weakly Pareto optimal solutions. We also observe that if in the course of the algorithm we have two subsets Λ_1, Λ_2 with a common optimal solution x of P(λ) then this solution is also optimal in P(λ) for all $\lambda \in conv(\Lambda_1 \cup \Lambda_2)$ and any subsets of this can be eliminated from further investigation.

4 Illustrative Example and Numerical Results

We first illustrate the algorithm using an example from [15, page 385].

$$\begin{array}{ll} \min & (-x_1, -x_2, -x_3), \\ \text{s.t.} & x_1 + x_2 + x_3 + s_1 = 5, \\ & x_1 + 3x_2 + x_3 + s_2 = 9, \\ & 3x_1 + 4x_2 + s_3 = 16, \\ & x_1, x_2, x_3, s_1, s_2 \geq 0. \end{array}$$

The feasible set has seven Pareto optimal extreme points that define two Pareto optimal facets, see Fig. 1.

We apply the algorithm with $\bar{\Lambda} = \Lambda^0 = \{\lambda \in \mathbb{R}^3 : 0 < \lambda_i < 1, \lambda_1 + \lambda_2 + \lambda_3 = 1\}$.

For illustration purposes we will eliminate λ_3 from inequalities defining subsets of Λ^0. We shall also apply Lemmas 3.5 and 3.6 throughout and only consider open sets. Clearly, $u_{\bar{\Lambda}} = (-1, 0, 0)$ is feasible for D(λ) for all $\lambda \in \Lambda^0$ and $Q(\lambda) = \{5, 6\}$ for all $\lambda \in \Lambda^0$. Therefore, we can initialize $\mathcal{L} = \{\Lambda^0\}$ and enter the while loop.

(Λ^0) The optimal objective value of RP(Λ^0) is 5 and the optimal solution of the restricted dual DRP(Λ^0) is $\hat{w}(\Lambda^0) = (1, 0, 0)$. Thus, $\hat{\varepsilon}(\lambda) = \min\{-\lambda_1 + 1, -\lambda_2 + 1, -\lambda_3 + 1\}$. Computing the lower envelope of these three functions in Λ^0 yields the refinement of Λ^0 into Λ^1, Λ^2, and Λ^3 and the dual solutions shown in Table 1. Thus, $\mathcal{L} = \{\Lambda^1, \Lambda^2, \Lambda^3\}$ (see Fig. 2(a)).

(Λ^1) We obtain RP(Λ^1), which has an optimal value of 0 and yields the Pareto optimal solution $(x_{\Lambda^1}, s) = (5, 0, 0, 0, 4, 1)$. Thus, Λ^1 can be discarded and $\mathcal{L} = \{\Lambda^2, \Lambda^3\}$.

(Λ^2) RP(Λ^2) has optimal value 2 and the optimal solution of DRP(Λ^2) is $\hat{w}(\Lambda^2) = (1, -1/3, 0)$. Thus, $\hat{\varepsilon}(\lambda) = \min\{-(3/2)(\lambda_1 - \lambda_2), -(3/2)(\lambda_3 - \lambda_2), \lambda_2\}$ and computing the lower envelope of these three functions in Λ^2 yields the refinement Λ^4, Λ^5, Λ^6 shown in Table 1. This leaves us with $\mathcal{L} = \{\Lambda^3, \Lambda^4, \Lambda^5, \Lambda^6\}$.

(Λ^3) RP(Λ^3) has optimal value 0 with $(x_{\Lambda^3}, s) = (0, 0, 5, 0, 4, 16)$, so that Λ^3 is discarded and $\mathcal{L} = \{\Lambda^4, \Lambda^5, \Lambda^6\}$ (see Fig. 2(b)).

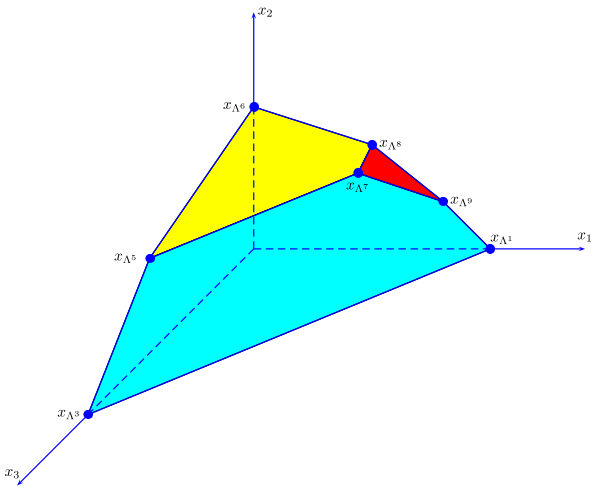

Fig. 1 The feasible set of the example. The Pareto optimal set is shaded

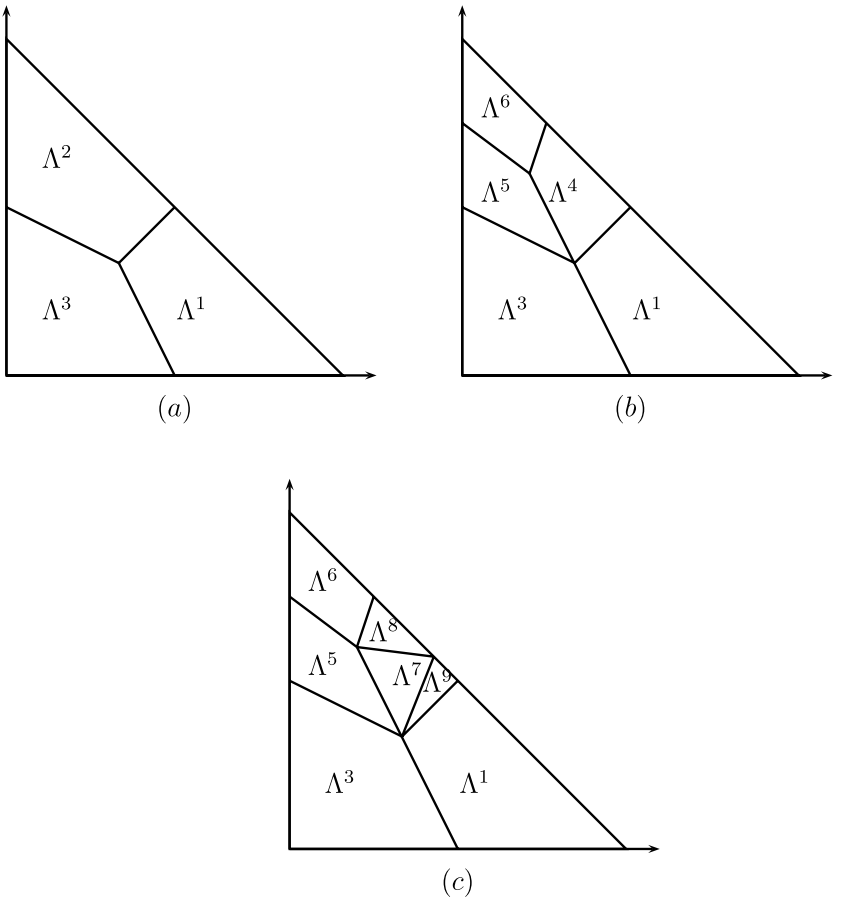

Fig. 2 The partition of Λ

(Λ^4) RP(Λ^4) has optimal value 2/5 and the optimal solution of DRP(Λ^4) is (1, 1/5, $-2/5$). This yields $\hat{\varepsilon}(\lambda) = \min\{(5/6)(\lambda_1 - \lambda_3), (1/2)(3\lambda_1 - \lambda_2), (5/2)(\lambda_2 - \lambda_1)\}$. Computing the lower envelope of the three functions yields the partition of Λ^4 into Λ^7, Λ^8, and Λ^9 shown in Table 1. Hence, we have $\mathcal{L} = \{\Lambda^5, \ldots, \Lambda^9\}$.

(Λ^5) We find that RP(Λ^5) has optimal value 0 with $(x_{\Lambda^5}, s) = (0, 2, 3, 0, 0, 8)$, so that we can discard Λ^5 and $\mathcal{L} = \{\Lambda^6, \ldots, \Lambda^9\}$.

(Λ^6) The reduced primal RP(Λ^6) has optimal value 0 for $(x_{\Lambda^6}, s) = (0, 3, 0, 2, 0, 3)$. We discard Λ^6 and have $\mathcal{L} = \{\Lambda^7, \Lambda^8, \Lambda^9\}$ (see Fig. 2(c)).

(Λ^7) The reduced primal RP(Λ^7) has optimal value 0 with $(x_{\Lambda^7}, s) = (8/3, 2, 1/3, 0, 0, 0)$ and we are left with $\mathcal{L} = \{\Lambda^8, \Lambda^9\}$.

(Λ^8) The reduced primal RP(Λ^8) has optimal value 0 with $(x_{\Lambda_8}, s) = (11/5, 12/5, 0, 2/5, 0, 0)$ and we are left with $\mathcal{L} = \{\Lambda^9\}$.

(Λ^9) The reduced primal RP(Λ^9) has optimal value 0 with $(x_{\Lambda^9}, s) = (4, 1, 0, 0, 1, 0)$ and the algorithm terminates. The final partition of Λ is shown in Fig. 2(c).

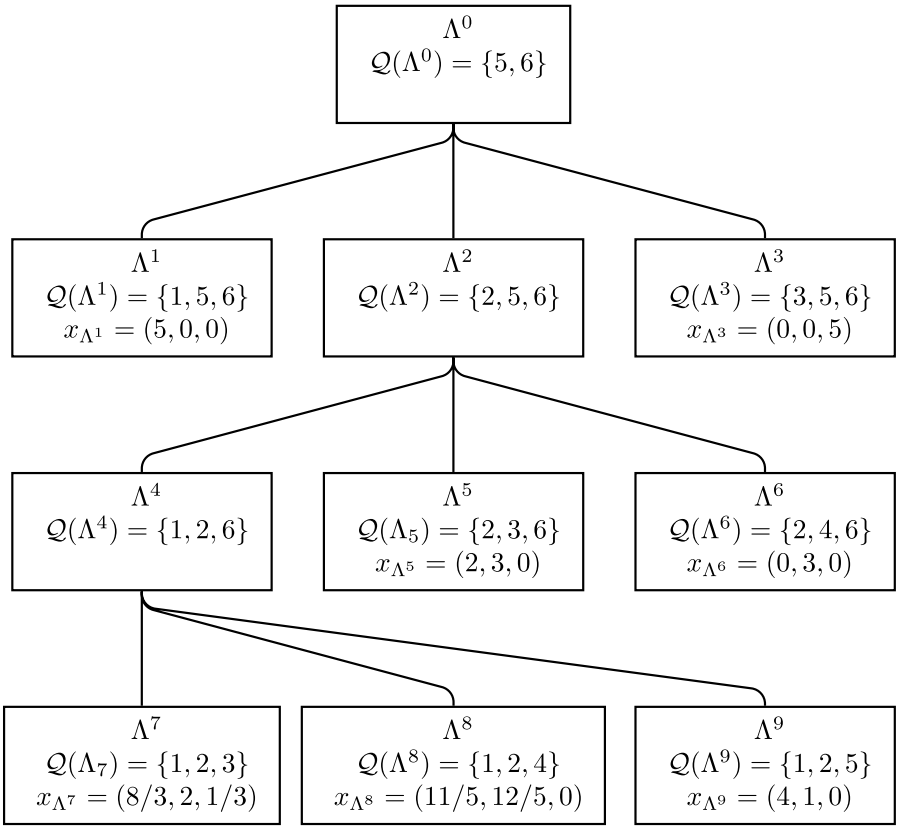

Fig. 3 The branching scheme of the primal-dual algorithm

The total effort to find a weight space decomposition and a Pareto optimal solution for each subset of the weight space consisted of the solution of 10 restricted primals (7 of which just confirm optimality), three restricted duals, and three lower envelopes of affine functions of two variables.

Figure 3 shows a schematic summary of the algorithm for this example.

The algorithm has been implemented for the assignment problem with three objectives [16]. Five instances with a size ranging from 5×5 to 25×25 with a step of 5 have been generated and solved. The objective function coefficients have been randomly selected from the integers $\{0, \ldots, 20\}$. The computer used for the experiments is equipped with a PowerPC G4 1 Ghz processor with 512 MB of RAM, and runs under the OS X operating system. The algorithm has been implemented in C. The binary has been obtained using the compiler gcc with the optimizer option -O3. For the computation of the convex hull we have used the software qhull 3.1 [17], available at www.qhull.org. It was not possible to solve problems of larger size due to numerical imprecision in the qhull code. The number of Pareto optimal solutions and the CPU times are shown in Table 2.

Table 1 Refinement of Λ^0

Λ^i	$\mathcal{Q}(\Lambda^i)$	$u_{\Lambda^i}(\lambda)$
$\Lambda^1 = \{\lambda \in \Lambda^0 : \lambda_1 > \lambda_2, \lambda_1 > \lambda_3\}$	$\{1, 5, 6\}$	$(-\lambda_1 + 1, 0, 0)$
$\Lambda^2 = \{\lambda \in \Lambda^0 : \lambda_2 > \lambda_1, \lambda_2 > \lambda_3\}$	$\{3, 5, 6\}$	$(-\lambda_2 + 1, 0, 0)$
$\Lambda^3 = \{\lambda \in \Lambda^0 : \lambda_3 > \lambda_1, \lambda_3 > \lambda_2\}$	$\{2, 5, 6\}$	$(-\lambda_3 + 1, 0, 0)$
$\Lambda^4 = \{\lambda \in \Lambda^2 : \lambda_2 > 1 - 2\lambda_1, \lambda_2 < 3\lambda_1\}$	$\{1, 2, 6\}$	$((1/2)(\lambda_2 - 3\lambda_1), (1/2)(\lambda_1 - \lambda_2), 0)$
$\Lambda^5 = \{\lambda \in \Lambda^2 : \lambda_2 < 1 - 2\lambda_1, \lambda_2 < (3/4)(1 - \lambda_1)\}$	$\{2, 3, 6\}$	$((1/2)(\lambda_2 - 3\lambda_3), (1/2)(\lambda_3 - \lambda_2), 0)$
$\Lambda^6 = \{\lambda \in \Lambda^2 : \lambda_2 > 3\lambda_1, \lambda_2 > (3/4)(1 - \lambda_1)\}$	$\{2, 4, 6\}$	$(0, -(1/3)\lambda_2, 0)$
$\Lambda^7 = \{\lambda \in \Lambda^4 : \lambda_2 < (1/8)(5 - \lambda_1), \lambda_2 > (1/2)(5\lambda_1 - 1)\}$	$\{1, 2, 3\}$	$((1/6)(3\lambda_2 - 4\lambda_1 - 5\lambda_3), (1/6)(4\lambda_1 - 3\lambda_2 - \lambda_3), (1/3)(\lambda_3 - \lambda_1))$
$\Lambda^8 = \{\lambda \in \Lambda^4 : \lambda_2 > (1/8)(5 - \lambda_1)\}$	$\{1, 2, 4\}$	$(0, (1/5)(4\lambda_1 - 3\lambda_2), 1/5(\lambda_2 - 3\lambda_1))$
$\Lambda^9 = \{\lambda \in \Lambda^4 : \lambda_2 < (1/2)(5\lambda_1 - 1)\}$	$\{1, 2, 5\}$	$(3\lambda_2 - 4\lambda_1, 0, \lambda_1 - \lambda_2)$

Table 2 Numerical results obtained by Algorithm 3.1 for the assignment problem with three objectives

Instance size	Pareto optimal solutions	CPU time (sec.)
5×5	10	0.01
10×10	43	0.28
15×15	81	2.63
20×20	146	28.15
25×25	245	121.21

5 Special Cases

In this section we first summarize some remarks concerning special cases.

If our algorithm is applied to a closed set $\bar{\Lambda}$, e.g. the closed parameter space Λ, it may find weakly Pareto optimal solutions which are not Pareto optimal. These can be eliminated after termination of the algorithm. In addition, if $\bar{\Lambda} = \Lambda$ or $\bar{\Lambda} = \Lambda^0$ at termination of the algorithm, the final maximal subsets Λ_l^* define the complete partition of the parameter space.

As a consequence of Theorem 3.1, the list of Pareto optimal solutions found after termination (if $\bar{\Lambda} = \Lambda^0$ or $\bar{\Lambda} = \Lambda$) can be used to completely describe the efficient set $Y_{\text{eff}} = C^T X_{\text{Par}}$, although not the complete set X_{Par}. However, the list of Pareto optimal points at termination yields a set of optimal bases of P(λ) for $\lambda \in \Lambda^0$ which can be used to generate the complete set X_{Par} by known procedures, e.g., [18].

The algorithm can also handle problems where D(λ) is infeasible for λ in some subset $\Lambda' \subset \bar{\Lambda}$ by simply considering as initial parameter space $\bar{\Lambda} \setminus \Lambda'$. In this case,

Λ' would have to be known in advance, and several initial dual feasible solutions for several (possibly disconnected) subsets of $\bar{\Lambda}$ are needed at the start.

Further improvements can be made in the case of two objective functions, consider $(\lambda_1, \lambda_2) = (\lambda_1, 1 - \lambda_1)$. Thus, we can identify subsets $\bar{\Lambda}$, $\hat{\Lambda}$ and Λ^* with intervals in $[0, 1]$. In this case computations of $\mathcal{Q}(\hat{\Lambda})$ and $\hat{\varepsilon}_{\Lambda^*}(\lambda)$ for $\lambda \in \Lambda^*$ can be done very efficiently. Computation of $\mathcal{Q}(\hat{\Lambda})$ involves intersection of at most n intervals. This can be done in $O(n \log n)$ by ordering according to the left endpoint. In addition, the computation of $\hat{\varepsilon}_{\Lambda^*}(\lambda)$ is also doable in $O(n \log n)$ using Hershberger and Shuri's algorithm [19] for the lower envelope of affine functions of one variable.

6 Concluding Remarks

In this paper we have presented a primal-dual simplex algorithm for multiobjective linear programming. Because we base the algorithm on duality of scalarized linear programs P(λ), we can compute in parallel both Pareto optimal solutions and the partition of the parameter space. We also avoid degeneracy in the primal representation of vertices. The algorithm allows to use parallel computation to generate Pareto optimal solutions for each parameter $\lambda \in \Lambda$ by exploiting the branching structure obtained through partition of the parameter space.

It also gives another interpretation of Pareto optimal solutions as the sensitivity analysis of several right hand side vectors in scalar linear programming problems.

The major disadvantage of the algorithm is that we may have to explore exponentially many nodes. But notice that this cannot be avoided by any algorithm because of the possibly exponentially many extreme points of the efficient set.

Finally, we would like to discuss applications to combinatorial problems. Single objective primal-dual algorithms have been very successfully applied to combinatorial problems with totally unimodular constraint matrix such as flow, path, assignment, and matching problems [10]. Using our algorithm we can generate all Pareto optimal solutions of these problems which are optimal solutions for some scalarization P(λ) of the multiobjective combinatorial problem (supported Pareto optimal solutions). However, in multiobjective combinatorial optimization it is known that there exist Pareto optimal solutions which are not optimal for any scalarization with $\lambda \in \Lambda$, even if the constraint matrix A is totally unimodular.

Our algorithm can be used to find supported Pareto optimal solutions. In this context it is important to note that RP(λ) and DRP(λ) are not parametric problems (i.e., their coefficients do not depend on λ). Thus the same methods for their solution can be used that have been so efficiently used in the single objective case.

Most algorithms that guarantee to find unsupported Pareto optimal solutions proceed in two phases. First, supported Pareto optimal solutions and a weight space decomposition are computed. Then unsupported Pareto optimal solutions are found, using the weight space decomposition and other techniques [20]. This technique has been applied to biobjective network flow [21], knapsack [22] and assignment [11] problems.

However, no effective implementation for the same problems with three objectives is available. This is due to the lack of methods to compute supported Pareto optimal solutions and a weight space decomposition simultaneously. With our algorithm this problem can be overcome. Such extensions are currently under investigation.

References

1. Ehrgott, M., Wiecek, M.M.: Multiobjective programming. In: Figueira, J., Greco, S., Ehrgott, M. (eds.) Multiple Criteria Decision Analysis: State of the Art Surveys. International Series in Operations Research and Management Science, vol. 76, pp. 667–708. Springer, New York (2005)
2. Steuer, R.E.: ADBASE: A MOLP solver. Technical report, Terry College of Business, University of Georgia, Athens, GA (2000)
3. Benson, H.P.: An outer approximation algorithm for generating all efficient extreme points in the outcome set of a multiple objective linear programming problem. J. Glob. Optim. **13**, 1–24 (1998)
4. Abhyankar, S.S., Morin, T.L., Trafalis, T.: Efficient faces of polytopes: interior-point algorithms, parametrization of algebraic varieties, and multiple objective optimization. Contemp. Math. **114**, 319–341 (1990)
5. Arbel, A.: Anchoring points and cones of opportunities in interior multiobjective linear programming. J. Oper. Res. Soc. **45**, 83–96 (1994)
6. Arbel, A.: An interior multiobjective primal-dual linear programming algorithm based on approximated gradients and efficient anchoring points. Comput. Oper. Res. **24**, 353–365 (1997)
7. Zeleny, M.: Linear Multiobjective Programming. Lecture Notes in Economics and Mathematical Systems, vol. 95. Springer, Berlin (1974)
8. Benson, H.P., Sun, E.: Outcome space partition of the weight set in multiobjective linear programming. J. Optim. Theory Appl. **105**, 17–36 (2000)
9. Hamel, H., Heyde, F., Löhne, A., Tammer, C., Winkler, K.: Closing the duality gap in linear vector optimization. J. Convex Anal. **11**, 163–178 (2004)
10. Ahuja, R., Magnanti, T., Orlin, J.: Network Flows: Theory, Algorithms, and Applications. Prentice-Hall, Englewood Cliffs (1994)
11. Ulungu, E.L., Teghem, J.: The two-phases method: an efficient procedure to solve biobjective combinatorial optimization problems. Found. Comput. Decis. Sci. **20**, 149–165 (1994)
12. Geoffrion, A.: Proper efficiency and the theory of vector maximization. J. Math. Anal. Appl. **22**, 618–630 (1968)
13. Luenberger, D.G.: Linear and Nonlinear Programming. Addison-Wesley, Reading (1989)
14. Edelsbrunner, H.: Algorithms in Combinatorial Geometry. Springer, Berlin (1987)
15. Steuer, R.E.: Multiple Criteria Optimization: Theory, Computation, and Application. Krieger Publishing Company, Malabar (1989)
16. Przybylski, A.: Application of primal-dual simplex method for MOLP to the MO assignment problem. Technical Report, University of Nantes, Nantes, France (2005)
17. Barber, C.B., Dobkin, D.P., Huhdanpaa, H.: The quickhull algorithm for convex hulls. ACM Trans. Math. Softw. **22**, 469–483 (1996)
18. Isermann, H.: The enumeration of the set of all efficient solutions for a linear multiple objective program. Oper. Res. Q. **28**, 711–725 (1977)
19. Hershberger, J., Shuri, S.: Offline maintenance of planar configurations. J. Algorithms **21**, 453–475 (1996)
20. Ehrgott, M., Gandibleux, X.: Approximative solution methods for multiobjective combinatorial optimization. Top **12**, 1–85 (2004)
21. Sedeño-Noda, A., González-Martín, C.: An algorithm for the biobjective integer minimum cost flow problem. Comput. Oper. Res. **28**, 139–156 (2001)
22. Visée, M., Teghem, J., Pirlot, M., Ulungu, E.L.: Two-phases method and branch-and-bound procedures to solve the biobjective Knapsack problem. J. Glob. Optim. **12**, 139–155 (1998)