# Column generation with a primal-dual method*

Jacek Gondzio† and Robert Sarkissian

Logilab, HEC Geneva, University of Geneva,
102 Bd Carl Vogt, CH-1211 Geneva 4, Switzerland
e-mail: gondzio@divsun.unige.ch and sarkissi@divsun.unige.ch

Logilab Technical Report 96.6
June 27, 1996, revised October 3, 1997

**Abstract**

A simple column generation scheme that employs an interior point method to solve underlying restricted master problems is presented. In contrast with the classical column generation approach where restricted master problems are solved exactly, the method presented in this paper consists in solving it to a predetermined optimality tolerance (loose at the beginning and appropriately tightened when the optimum is approached). An infeasible primal-dual interior point method which employs the notion of $\mu$-center to control the distance to optimality is used to solve the restricted master problem. Similarly to the analytic center cutting plane method, the present approach takes full advantage of the use of central prices. Furthermore, it offers more freedom in the choice of optimization strategy as it adaptively adjusts the required optimality tolerance in the master to the observed rate of convergence of the column generation process.

The proposed method has been implemented and used to solve large scale nonlinear multicommodity network flow problems. Numerical results are given to illustrate its efficiency.

**Key words.** Column generation, restricted master problem, primal-dual algorithm, analytic center.

*Supported by the Fonds National de la Recherche Scientifique Suisse, grant #12-42503.94.
†On leave from the Systems Research Institute, Polish Academy of Sciences, Newelska 6, 01-447 Warsaw, Poland.

# 1 Introduction

We are concerned in this paper with the solution of the following linear program:

$$\begin{aligned} \min \quad & \tilde{c}^T \tilde{x} \\ \text{s.t.} \quad & \tilde{A}\tilde{x} = b, \\ & \tilde{x} \geq 0, \end{aligned} \tag{1}$$

where $\tilde{c}, \tilde{x} \in \mathbb{R}^N$, $b \in \mathbb{R}^m$, $\tilde{A} \in \mathbb{R}^{m \times N}$. We assume that the number of variables $N$ is much larger than the number of constraints $m$. In particular, the matrix $\tilde{A}$ may not be given explicitly and $N$ does not have to be finite. This type of problems arise frequently in various real-life applications.

To solve (1) we employ a column generation scheme (see for example [3]). That is, we select a subset of $n$ columns of $\tilde{A}$ ($n \ll N$) to render the problem more tractable

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Ax = b, \\ & x \geq 0, \end{aligned} \tag{2}$$

where $c, x \in \mathbb{R}^n$ are subvectors of $\tilde{c}$ and $\tilde{x}$, respectively and $A \in \mathbb{R}^{m \times n}$ is a submatrix of $\tilde{A}$ built of $n$ columns chosen by now. We solve a *restricted master problem* (2) and employ its dual solution to improve the representation, i.e. we select a subset of new columns from $\tilde{A}$ that have to be appended to the restricted master problem. The process is continued until the representation becomes sufficiently close to $\tilde{A}$ in the sense that the optimal solution of the restricted master problem approaches the optimal solution to (1).

In the classical column generation scheme [22], every restricted master problem is solved to optimality. This may but does not have to be good strategy. For example, whenever a large number of restricted master problems have to be solved, reaching an exact optimal solution of the restricted master program may turn out to be a useless computational effort. This is, in particular, the case if model (2) is learning very slowly adding only one new column to the representation per each outer iteration (master reoptimization).

What we really need is a "good" dual solution to (2) so as to be able to make a wise choice of new column(s) to be appended to $A$. We would be satisfied with any dual solution that is sufficiently close to the optimality of the restricted master problem.

Consequently, in order to solve subsequent master problems (2) we propose to employ the infeasible primal-dual interior point method. This method, known as the most efficient interior point algorithm, offers perfect control of the distance to optimality through the notion of $\mu$-centers [20]. We propose to adjust the optimality tolerance required in subsequent master problems to the observed rate of convergence of the column generation scheme. The idea is to start from a rather loose tolerance to benefit from the use of central prices [14] and to build as fast as possible the first rough approximation to (1). Later on, we tighten the optimality tolerance required for the solution of the restricted master problem so as to improve our representation of (1) near its optimal solution.

Our approach is closely related to the analytic center cutting plane method of Goffin, Haurie and Vial [14], who were the first to employ interior point method (primal potential reduction algorithm) to solve the restricted master problems. The column generation scheme we propose lies somewhere between the optimal point strategy applied in a number of decomposition schemes [6, 4, 22] and the analytic center cutting plane method [14].

Compared with Goffin et al.'s approach, the method presented here reflects much of simplicity, both, in its presentation and in its implementation. Yet, it benefits of all the advantages resulting

from the use of central prices, i.e. the use of dual "central" point that lies in the relative interior of the dual polytope. The use of central prices can be traced back to the work of Levin [23] who was the first to suggest the choice of the center of gravity—one of several different strategies for the choice of a central point which have been proposed in the literature. The idea to use central points was later developed by Elzinga and Moore [10]. Next, Sonnevend [26] proposed to use the analytic center and Vaidya [27] advocated for the use of volumetric center.

The use of different centers naturally results in different properties of the cutting plane scheme. Although the choice of the center of gravity has been proved to generate the best possible cuts [25], it has little impact in computational practice because of the current lack of an efficient algorithm to find this center. On the other hand, the analytic center which is the basic notion of the interior point methodology can be computed efficiently by any interior point algorithm.

In the method of [14], analytic center is computed with the (feasible) primal projective algorithm. An advanced implementation of this approach [13, 18, 19, 9] rigorously follows the theory, which is quite a rare advantage in the optimization software. However, a requirement to keep close to the theory is sometimes a barrier of further development. We have realized this, in particular, when we were implementing techniques to control the accuracy in the feasible potential reduction algorithm.

The use of the primal-dual method to find an approximate analytic center seems to offer important advantages over primal projective algoritm. First, when applied to the general linear programs, primal-dual method is, on the average, more efficient. Secondly, primal-dual method is an infeasible algorithm hence the control of accuracy is much easier in it [2]. Replacing projective algorithm in the analytic center cutting plane method with the primal-dual algorithm seems thus to be a natural step of its development. Such a replacement could be done in the context of the code [18].

However, we decided to go much further and replace also the control of the distance to optimality with the one that uses $\mu$-centers defined in the primal-dual method.

Let us mention that the use of the logaritmic barrier method within the column generation scheme has already been proposed by den Hertog et al. [8, 7]. The latter reference reported numerical results of its application to small but difficult geometric programming problems. Mitchell and Borchers [24] applied primal-dual cutting plane method to solve a set of real-world linear ordering problems. The structural properties of these problems implied that after every cut addition, the primal and dual feasible solutions to the new restricted master problem could be found. Consequently, the method did not have to bother about restoring feasibility and all its effort was concentrated on approaching well centered optimal solution. Note that such a situation appears rarely in the computational practice as, in general, there is no way to guess an interior primal and dual feasible solution to the restricted master problem after the addition of a set of new columns to it [17].

This paper is organized as follows. In Section 2 we address a general decomposition approach that leads to the column generation scheme. In particular, we introduce the notions of the restricted master problem and the subproblem — a procedure to generate new columns. In Section 3 we point out the advantages resulting from the application of the primal-dual interior point algorithm to solve the restricted master problems. We discuss the use of $\mu$-centers to control the distance to optimality within the primal-dual method and interpret its consequences in the framework of the column generation scheme. In Section 4 we recall the nonlinear multicommodity network flow problem. The method proposed in this paper has been successfully used to solve several large-scale examples of such problems. We discuss the numerical results of these runs in Section 5, and finally, in Section 6 we present our conclusions.

# 2 Column generation

## 2.1 Principles

In many presentations of the column generation scheme, the principles of the approach are mixed with the implementation details of the revised simplex method, applied to solve the underlying restricted master problems. We shall avoid this practice and give first a description of what we believe are the principles of column generation.

The problem (1) arises typically when one attempts to solve

$$\begin{aligned} \min \quad & d^T v && \text{(3a)} \\ \text{s.t.} \quad & Hv = g, && \text{(3b)} \\ & v \in X, && \text{(3c)} \end{aligned}$$

where $X$ is a polyhedral set, $H \in \mathbb{R}^{m \times n}$, $d, v \in \mathbb{R}^n$ and $b \in \mathbb{R}^m$. The problem is rendered difficult by the presence of constraint (3b). We assume that without (3b), the problem would be easily solved, taking full advantage of some special structure of $X$.

Any point of a polyhedral set can be represented as a linear combination of its extreme points and extreme rays, respectively. Let $p_1, \ldots, p_k$ and $r_1, \ldots, r_l$ be the extreme points and extreme rays of $X$, respectively. For any $v \in X$ there exist multipliers $\lambda_1, \ldots, \lambda_k$ and $\mu_1, \ldots, \mu_l$ such that

$$\begin{aligned} v &= \sum_{i=1}^{k} \lambda_i p_i + \sum_{j=1}^{l} \mu_j r_j && \text{(4a)} \\ \lambda_i &\geq 0, \quad i = 1, \ldots, k, && \text{(4b)} \\ \mu_j &\geq 0, \quad j = 1, \ldots, l, && \text{(4c)} \\ \sum_{i=1}^{k} \lambda_i &= 1. && \text{(4d)} \end{aligned}$$

Using (4a)-(4d) we transform the problem (3a)-(3c) to the one with variables $\lambda_i$ and $\mu_j$. We got the *full master program*

$$\begin{aligned} \min & \sum_{i=1}^{k} \lambda_i (d^T p_i) + \sum_{j=1}^{l} \mu_j (d^T r_j) \\ \text{s.t.} & \sum_{1 \leq i \leq k} \lambda_i (H p_i) + \sum_{1 \leq j \leq l} \mu_j (H r_j) = g, && \text{(5a)} \\ & \sum_{1 \leq i \leq l} \lambda_i = 1, && \text{(5b)} \\ & \lambda_i \geq 0, \quad 1 \leq i \leq k, && \text{(5c)} \\ & \mu_j \geq 0, \quad 1 \leq j \leq l. && \text{(5d)} \end{aligned}$$

This problem may have a very large number of variables and usually cannot be solved directly (it is often not even possible to formulate it explicitly). Instead, we may try to generate points $p_i$ and $r_j$ only when needed.

Consider now subsets $\{p_i : i \in I\}$ and $\{r_j : j \in J\}$ of the extreme points and extreme rays of $X$. If we restrict the full master program to those subsets, the resulting linear program, called

*restricted master program* (RMP), is a relaxation of (5a)-(5d). Its optimal solution thus provides an upper bound to the optimal solution of the full master program.

On the other hand, a dualization of constraint (3b) leads to the Lagrangian function

$$L(v,u) = d^T v + u^T(Hv - g), \quad v \in X.$$

Hence, for any $u$, the value

$$L(u) = \inf_{v \in X} \{d^T v + u^T(Hv - g)\} \tag{6}$$

is a lower bound of (3a)-(3c). Computing $L(u)$ is by assumption an "easy" task: we call it the *subproblem.*

As noted earlier, the optimal solution of the RMP provides a straightforward upper bound $z_u$ to (3a). A part of its dual optimal solution corresponding to constraint (5a) is passed to subproblem (6) which returns a lower bound $z_l$. Thus we know that the optimal solution of the original problem falls between the values $z_l$ and $z_u$. The difference $z_u - z_l$ can serve to measure the accuracy of the current solution. The algorithm terminates when this difference drops below a predetermined tolerance $\epsilon$, i.e. when $z_u - z_l \leq \epsilon$.

If a current solution does not yet satisfy the above stopping criteria, one can only conclude that the set $X$ is not well enough characterized: one new column $Hp_i$ or $Hr_j$ should then be introduced to the restricted master problem. This necessary information originates from the subproblem. Note that if $L(u) > -\infty$, there exists an extreme point p of X such that $L(u) = d^T p + u^T(Hp - g)$ and if $L(u) = -\infty$, there exist an extreme ray $r$ of X such that $d^T r + u(Hr - g) < 0$. Depending of the value of $L(u)$ (whether it is finite or not), one introduces a new column $Hp$ or $Hr$ to the restricted master problem.

### 2.2 Suboptimal solution of the restricted master problem

Recall that in the classical column generation scheme, the restricted master problem is solved to optimality. Therefore, its optimal solution generates an upper bound $z_u$ for the solution of the full master problem and its dual optimal solution $u$ can be applied by the subproblem (6) to obtain a lower bound $z_l$ to the solution of (5a)-(5d).

However, we do not have to solve RMP to optimality. All we need is:

- a reliable estimate of its optimal value, and
- a dual feasible point $u$ (with a guarantee of being close to optimality).

The former is needed to determine $z_u$ and the latter is needed by the subproblem to update $z_l$ and/or to generate new columns.

Let us assume that we are able to find an $\epsilon_0$-optimal solution $z_0$ to the restricted master problem as well as its corresponding primal and dual feasible solutions. Let $(u, u_0) \in \mathbb{R}^{m+1}$ be such a dual feasible solution. Then we can produce a reliable upper bound to the optimal solution of the full master problem

$$\tilde{z}_u = z_0 + \epsilon_0 |z_0|.$$

Moreover, due to the feasibility and the guaranteed small distance to the optimality in the RMP, its dual solution $u$ is a constructive proposition that can be used in (6) to generate a new lower bound $\tilde{z}_l$ and/or a new column.

These considerations lead to the column generation algorithm (Algorithm 2.1) that employs suboptimal solutions of the underlying restricted master problems.

**Algorithm 2.1** A prototype of the column generation scheme

**Parameter:**
$\epsilon =$ optimality tolerance;
**Initialize:**
$z_u = +\infty$, $z_l = -\infty$;
**while** $(z_u - z_l > \epsilon)$ **do**
**begin**
choose $\epsilon_0$;
find an $\epsilon_0$-optimal solution to RMP
(let $u$ be the part of its dual solution corresponding to constraint (5a));
update the upper bound:
$\tilde{z}_u = z_0 + \epsilon_0 |z_0|$,
$z_u = \min(z_u, \tilde{z}_u)$;
solve subproblem for the $u$;
update the lower bound:
$z_l = \max(z_l, \tilde{z}_l)$;
append new column to RMP;
**end**

Since the column generation scheme we propose uses inexact solutions of the restricted master problems, an important theoretical as well as practical question arises about the relative optimality tolerance $\epsilon_0$ which is required when solving the subsequent restricted master problems. An important characteristic of our algorithm is that the optimality tolerance $\epsilon_0$ is dynamically chosen. At the beginning of the optimization process, $\epsilon_0$ should be rather loose as we only aim at building the first approximation to the full master problem. The tolerance is then tightened when the optimum to (3a)-(3c) is approached. In other words, we adjust the accuracy required from the solution of the restricted master problem to the real need in the column generation scheme. In practice, we choose

$$\begin{aligned} \epsilon_0 &= \epsilon_0(z_u, z_l) \\ &= \delta \frac{|z_u - z_l|}{|z_u| + 1}, \end{aligned} \tag{7}$$

i.e. the relative optimality tolerance in the restricted master problem is a function of the relative optimality tolerance of the best available solution to the original problem.

In our implementation, $\epsilon_0$ is simply a fraction of the relative optimality tolerance of the best solution found by now. We have experimented with several values of parameter $\delta$ varying from 0.5 to 0.01. The numerical results reported in this paper correspond to the choice $\delta = 0.02$. Naturally, when the subproblem information is gathered very slowly (e.g., when only one column is appended after every call to the subproblem), larger $\delta$—closer to 1.0 might be better. In such a situation we would prefer to take more advantage of the regularizing central prices.

Let us point out that when the RMP is solved by the simplex method, it always has to be solved to optimality. Recall that the simplex method does not offer any measure of the distance of a given

feasible solution to optimality. The simplex method has to reach the optimality of both problems in order to produce the primal and dual feasible solutions and a reliable (perfect in this case) estimate of the optimal objective function value.

The situation appears different when one employs an interior point method to the solution of the restricted master problem. In particular a primal-dual method suits our needs perfectly. It usually provides a dual feasible solution before an exact optimal solution is found. Additionally, it also provides a primal feasible solution quite quickly and an excellent measure of the distance of these solutions to the optimality is therefore available. Consequently, the optimization process in the restricted master problem can be terminated at an early stage producing all the necessary information to ensure the convergence of the column generation algorithm.

# 3 Fundamentals of the primal-dual method

Consider the following pair of linear programs consisting in a primal problem

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Ax = b, \\ & x \geq 0, \end{aligned} \tag{8}$$

where $x, c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$ and $A \in \mathbb{R}^{m \times n}$ and its dual

$$\begin{aligned} \max \quad & b^T y \\ \text{s.t.} \quad & A^T y + s = c, \\ & s \geq 0, \end{aligned}$$

where $y \in \mathbb{R}^m$ and $s \in \mathbb{R}^n$. Having replaced the variable's nonegativity constraints with the logarithmic barrier function, we write down the following Lagrangian associated with (8)

$$L(x, y, s) = c^T x - \mu \sum_{j=1}^{n} \ln x_j - y(Ax - b),$$

where $\mu \geq 0$ is the barrier parameter. The associated first order optimality conditions give

$$Ax = b, \tag{9a}$$

$$A^T y + s = c, \tag{9b}$$

$$Xs = \mu e, \tag{9c}$$

where $X = diag\{x_j\} \in \mathbb{R}^{n \times n}$ and $e$ is a vector in $\mathbb{R}^n$ whose elements are all equal to one. If the primal and the dual regions have nonempty interiors then for any $\mu \geq 0$, there exists a unique point $(x(\mu), y(\mu), s(\mu))$ with $x \geq 0$ and $s \geq 0$ which satisfies (9a)-(9c). This point is called a $\mu$-center or simply an analytic center.

Note that a $\mu$-center defines a pair of primal and dual solutions that are primal and dual feasible, respectively. Consequently, a duality gap between these two solutions measures the distance to optimality

$$c^T x - b^T y = c^T x - (Ax)^T y = x^T (c - A^T y) = x^T s = n\mu. \tag{10}$$

Moreover, there exists a simple relation between this distance and the barrier parameter.

Classical primal-dual method for linear programming executes one damped Newton step towards the $\mu$-center and, regardless of how successful this step was, updates (usually by decreasing) the barrier parameter. All it's modern implementations (see for example survey in [2]) start from infeasible solutions and obtain feasibility as the optimality is approached. As the primal and the dual constraints are linear (in contrast to the nonlinear conditions $x_j s_j = \mu$ in (9c)) they are usually satisfied early in the optimization process [Newton's method works with a linear model of the system of equations (9a)-(9c), in which the first two equations are linear!] Consequently, the primal-dual algorithm reaches feasible solutions to the primal and the dual LP's at a relatively early stage.

Standard primal-dual methods for LP continue iterating until the relative duality gap drops below a prescribed optimality tolerance $\epsilon_{opt}$, i.e.

$$\frac{|c^T x - b^T y|}{|b^T y| + 1} < \epsilon_{opt}, \tag{11}$$

where $\epsilon_{opt}$ is usually set between $10^{-8}$ and $10^{-6}$. In the approach presented in this paper, the algorithm terminates earlier as the optimality tolerance $\epsilon_0$ chosen in (7) will not be that small except near the end of the optimization procedure.

In efficient implementations of the general purpose primal-dual method, subsequent iterates are not exact $\mu$-centers although it is preferable [16] that they lie in the vicinity of the central path. During the solution of the restricted master problem we thus do not pay special attention to the centrality of iterates. However, the quality of the final solution is checked by measuring the ratio between the largest and the smallest complementarity product

$$v_{max} = \max_j x_j s_j \quad \text{and} \quad v_{min} = \min_j x_j s_j. \tag{12}$$

If this ratio is excessively large, e.g., when

$$v_{max} > 10\mu \quad \text{or} \quad v_{min} < 0.1\mu,$$

then an additional recentering primal-dual step is made. That way we ensure that the new proposition $u$ sent to subproblem (6) is relatively well centered as we hope to take advantage of the use of central prices like in the analytic center cutting plane method [14].

At this point, the question of the choice of the final barrier parameter $\mu$ naturally arises. We want this parameter to correspond to the $\epsilon_0$-optimality in the restricted master problem. Thus, once we obtain only a rough estimate of the optimal objective value $z_0 = c^T x$, e.g., when (11) is satisfied for $\epsilon_{opt} = 10^{-1}$, we can fix the barrier

$$\mu = \epsilon_0 \frac{z_0}{n},$$

and believe that the corresponding $\mu$-center is an optimal solution with the relative tolerance $\epsilon_0$ (cf. eqn. (10)).

We note that the requirements imposed on the quality of the centrality of $u$ could be relaxed in the approach we presented in this paper. This is in contrast to the implementation of the analytic center cutting plane method [18] in which the high quality of the last central prices $u$ is a necessary condition for an efficient warm start when solving the subsequent relaxed master problem.

The warm start procedure of Gondzio [17], which is applied to re-optimize the restricted master problems with the primal-dual method, makes a distinction between an advanced initial point for subsequent problem and the constructive central dual prices needed by the column generation algorithm (see [17] for details).

# 4 Nonlinear multicommodity network flow

In order to illustrate the approach presented in this paper, we consider a well-known problem in the literature, the nonlinear multicommodity network flow problem. It consists in shipping different commodities that have to share a common facility in a network. Each commodity has to satisfy a mass-balance equation. Let $G = (V, E)$ be a directed graph and $N$ its incidence matrix. We assume that there are $L$ commodities and that each commodity $l$ has exactly one source $s_l$ and one sink $t_l$, and that its demand volume is $\gamma^l$. Let $b^l \in \mathbb{R}^{|V|}$ be the vector defined by

$$b_i^l = \begin{cases} \gamma^l & \text{if } i = s_l, \\ -\gamma^l & \text{if } i = t_l, \\ 0 & \text{otherwise.} \end{cases}$$

Let $X^l = \{x \in \mathbb{R}^{|E|} : Nx = b^l, x \geq 0\}$ be the set of feasible flows for commodity l. A vector $x$ of $X$ has a component $x_{ij}$ for each arc $(i, j)$. Let us introduce now a convex cost function $\phi_{ij} : \mathbb{R} \longrightarrow \mathbb{R}$ for each arc $(i, j)$. An example of cost function [11] is given by $\phi_{ij}(y) = \frac{y}{C_{ij}-y}$, where $C_{ij}$ is the capacity of arc $(i, j)$. A flow formulation of the problem is

$$\begin{aligned} \min \quad & \sum_{(i,j)\in E} \phi_{ij}(y_{ij}) \\ \text{s.t.} \quad & \sum_{1\leq l\leq L} x_{ij}^l \leq y_{ij}, \quad \forall (i,j) \in E, && \text{(13a)} \\ & x^l \in X^l, \quad 1 \leq l \leq L, && \text{(13b)} \\ & y_{ij} \geq 0, \quad \forall (i,j) \in E. && \text{(13c)} \end{aligned}$$

Polyhedral considerations allow us to formulate (13a)-(13c) in an alternative way: Let $P \in \mathcal{P}^l$ be the set of paths from the source $s_l$ to the sink $t_l$ of commodity $l$. One can check that the set $X^l$ is a bounded polyhedra and that its extreme points are feasible solutions that lie along a path $P \in \mathcal{P}^l$.

Since the points of a bounded polyhedra can always be represented as a convex combination of its extreme points, the flow of $X^l$ can be decomposed into elementary flows along paths. The flow component along a path $P$ will be denoted $f(P)$. For $(i, j) \in E$, let $\delta_{ij}(P)$ be equal to 1 if the arc $(i, j)$ belongs to $P$ and 0 otherwise. The equivalent formulation of (13a)-(13c) is then

$$\begin{aligned} \min \quad & \sum_{(i,j)\in E} \phi_{ij}(y_{ij}) \\ \text{s.t.} \quad & \sum_{1\leq l\leq L} \sum_{P\in\mathcal{P}^l} f(P)\delta_{ij}(P) \leq y_{ij}, \quad \forall (i,j) \in E, \\ & \sum_{P\in\mathcal{P}^l} f(P) = \gamma^l, \quad 1 \leq l \leq L, \\ & f(P) \geq 0, \quad \forall P \in \mathcal{P}^l, \quad 1 \leq l \leq L. \end{aligned}$$

Obviously, this formulation does not fit our problem of interest since it is nonlinear. Assuming that for each $(i, j) \in E$ a family $\Gamma_{ij}$ of feasible points $y_{ij}^k$ indiced with $k \in K_{ij}$ is given, we know that for a $y_{ij}$ that lies in the convex hull of $\Gamma_{ij}$, there exist multipliers $\lambda_{ij}^k \geq 0$, $k \in K_{ij}$ such that

$$y_{ij} = \sum_{k\in K_{ij}} \lambda_{ij}^k y_{ij}^k,$$

and

$$\sum_{k \in K_{ij}} \lambda_{ij}^k = 1.$$

Because of the convexity of $\phi_{ij}$, the convex combination $\sum_{k \in K_{ij}} \lambda_{ij}^k \phi_{ij}(y_{ij}^k)$ is an upper bound of $\phi_{ij}(y_{ij})$. Note that the $\lambda_{ij}^k$ may not be unique. It is therefore quite natural to approximate $\phi_{ij}(y_{ij})$ with the smallest possible convex combination of $\Gamma_{ij}$. Hence, we have the problem

$$\begin{aligned}
\min \quad & \sum_{(i,j) \in E} \sum_{k \in K_{ij}} \lambda_{ij}^k \phi_{ij}(y_{ij}^k) \\
\text{s.t.} \quad & \sum_{1 \le l \le L} \sum_{P \in \mathcal{P}^l} f(P) \delta_{ij}(P) - \sum_{k \in K_{ij}} \lambda_{ij}^k y_{ij}^k \le 0, \quad \forall (i,j) \in E, \\
& \sum_{P \in \mathcal{P}^l} f(P) = \gamma^l, \quad 1 \le l \le L, \\
& f(P) \ge 0, \quad \forall P \in \mathcal{P}^l, 1 \le l \le L, \\
& \lambda_{ij}^k \ge 0, \quad \forall (i,j) \in E, \forall k \in K_{ij}.
\end{aligned}$$

Now if the sets $\Gamma_{ij}$ cover all the feasible $y$, the above linear problem is equivalent to the nonlinear program. It has a matrix whose columns represent the paths $P \in \mathcal{P}^l$ and the boundary points of the epigraphs $epi(\phi_{ij})$. We are now in the usual framework of Section 2.

In this context, the equation (13a), which introduces competition among commodities for the same capacity $y$ is the "difficult" constraint. If we dualize it, we get the Lagrangian function

$$\begin{aligned}
L(x^1, \ldots, x^L, y, u) &= \sum_{(i,j) \in E} \phi_{ij}(y_{ij}) + \sum_{(i,j) \in E} u_{ij} \Big( \sum_{1 \le l \le L} x_{ij}^l - y_{ij} \Big) \\
&= \sum_{(i,j) \in E} (\phi_{ij}(y_{ij}) - u_{ij} y_{ij}) + \sum_{1 \le l \le L} \sum_{(i,j) \in E} u_{ij} x_{ij}^l.
\end{aligned}$$

The subproblem consists in finding, for a given $u$, the solution of

$$L(u) = \sum_{(i,j) \in E} \min_{y_{ij} \ge 0} (\phi_{ij}(y_{ij}) - u_{ij} y_{ij}) + \sum_{1 \le l \le L} \min_{x^l \in X^l} \sum_{(i,j) \in E} u_{ij} x_{ij}^l.$$

This is not a difficult task since it simply involves two types of rather simple problems. The first one is

$$\min_{y_{ij} \ge 0} (\phi_{ij}(y_{ij}) - u_{ij} y_{ij}),$$

which may be computed analytically or may represent a trivial problem. The second one is

$$\min_{x^l \in X^l} \sum_{(i,j) \in E} u_{ij} x_{ij}^l, \tag{14}$$

and may be computed by a shortest path algorithm, for example using Dijkstra's algorithm (see [1]).

In practice, one often needs to solve undirected multicommodity flow problems. A simple transformation allows to reformulate an undirected problem as a directed one [1]. One can then show that solving undirected multicommodity flow problem with the column generation method is essentially the same as solving directed problem. In fact, the only change lies in the function (14): it needs to find the shortest path in an undirected graph instead of a directed one.

| Problem | Nodes | Arcs | Comm. | Subprobs |
|---|---|---|---|---|
| NDO22 | 14 | 22 | 23 | 45 |
| NDO148 | 61 | 148 | 122 | 270 |
| Random12 | 300 | 600 | 1000 | 1600 |
| Random16 | 300 | 1000 | 1000 | 2000 |
| Random20 | 400 | 1000 | 5000 | 6000 |
| Random31 | 700 | 2000 | 2000 | 4000 |
| Random32 | 1000 | 3000 | 2000 | 5000 |
| Random33 | 1000 | 4000 | 2000 | 6000 |
| Random41 | 1000 | 5000 | 2000 | 7000 |
| Random42 | 1000 | 5000 | 3000 | 8000 |
| Random51 | 1000 | 2000 | 6000 | 8000 |
| Random52 | 1000 | 2000 | 8000 | 10000 |
| Random53 | 1000 | 2000 | 10000 | 12000 |
| Random54 | 1000 | 3000 | 6000 | 9000 |
| Random55 | 1000 | 3000 | 8000 | 11000 |

Table 1: Problems statistics.

# 5 Numerical results

The method proposed in this paper has been implemented within the context of the public domain primal-dual code HOPDM [15]. The HOPDM program is written in FORTRAN but the column generation routines are written in C. All computations were performed on a Power PC workstation (66MHz, 64MB of RAM, type 7011, model 25T). The program was compiled by the IBM FORTRAN and C compilers `xlf` and `xlc`, respectively. Compilation options `-O -qarch=ppc` were used.

Our experimental implementation of the primal-dual column generation method has been applied to various large scale nonlinear multicommodity network flow problems. Two of these problems are the well known examples NDO22 and NDO148 from [12] and the remainder are randomly generated undirected problems. A description of our (public domain) generator can be found in [13]. The problem statistics (the numbers of nodes, arcs and commodities, respectively) are summarized in Table 1 which also gives the number of subproblems in order to facilitate the determination of the size of the restricted master problem. In the formulation of nonlinear multicommodity network flow problems the number of coupling constraints equals to the number of arcs while the number of subproblems equals to the sum of numbers of arcs and commodities.

Table 1 does not reflect well the difficulty of the problems to be solved. The linear version of the Random33 problem, for example, in an equivalent compact LP formulation, would involve 1000 blocks of 2000 constraints of commodity flow balance at each node and 4000 coupling constraints of total flow capacity on the arcs. This formulation comprises $1000*2000+4000 = 2004000$ constraints and $4000 * 2000 = 8000000$ variables. When such a problem is decomposed, the restricted master problem becomes a linear program of quite a considerable size: the number of its constraints equals to the sum of numbers of arcs and subproblems while the number of columns is continuously increasing (and approaches 100,000).

It is essential to keep the number of columns the smallest possible so as to preserve the sparsity of the Cholesky matrix in the interior point method. We reach this goal by applying the technique of eliminating useless (inactive) cuts. The elimination relies on an analysis of the "central" solution returned from the restricted master problem. All complementarity products $x_j s_j$ of this solution

| Problem | ACCPM - No Elimination | | | | ACCPM - With Elimination | | | |
|---|---|---|---|---|---|---|---|---|
| | O | Its | Cuts | Time | O(S) | Its | Cuts | Time |
| NDO22 | 17 | 84 | 420 | 1.1 | 12(19) | 47(2) | 241 | 0.7 |
| NDO148 | 13 | 83 | 2230 | 14.3 | 13(18) | 80(6) | 1875 | 15.6 |
| Random12 | 19 | 257 | 16519 | 737 | 17(30) | 246(30) | 10730 | 599 |
| Random16 | 21 | 259 | 26494 | 1448 | 20(34) | 260(23) | 14920 | 822 |
| Random20 | 28 | 577 | 64241 | 11689 | 24(43) | 497(83) | 35352 | 6624 |
| Random31 | 22 | 434 | 57700 | 17652 | 20(39) | 379(43) | 33772 | 9423 |
| Random32 | 20 | 389 | 73795 | 42477 | 19(34) | 319(18) | 44493 | 17615 |
| Random33 | 20 | 382 | 95902 | 71429 | 24(42) | 401(37) | 59253 | 27990 |
| Random41 | 26 | 466 | 148214 | 63453 | 24(40) | 385(35) | 71842 | 31585 |
| Random42 | 24 | 534 | 149791 | 116726 | 24(36) | 411(38) | 76918 | 45038 |
| Random51 | 26 | 640 | 92503 | 61387 | 22(38) | 489(38) | 55025 | 33758 |
| Random52 | 30 | 688 | 115151 | 78527 | 24(44) | 562(56) | 62432 | 46489 |
| Random53 | 32 | 929 | 132816 | 99485 | 25(47) | 635(50) | 75479 | 57214 |
| Random54 | 26 | 593 | 122243 | 114765 | 22(41) | 501(46) | 66698 | 57035 |
| Random55 | 28 | 750 | 149046 | 141534 | 24(44) | 632(62) | 81030 | 91384 |

Table 2: Advantages of the use of cut elimination.

remain close to the final barrier parameter $\mu$ (cf. eqn. (12)). We identify columns $j$ such that

$$x_j \leq p\mu^{1/2} \text{ and } s_j \geq (1/p)\mu^{1/2}, \tag{15}$$

(with some $p < 1$) and assume that the corresponding variables $x_j$ approach zero in the optimal solution. Hence they can safely be eliminated from the restricted master problem.

This column elimination technique was first implemented in the approach presented in this paper and remarkably contributed to its overall performance (the method was on the average more than two times faster than the analytic center cutting plane method [13, 19] which did not eliminate inactive cuts). Clearly, such an elimination technique can be incorporated into any central cutting plane method. We have thus implemented it within the analytic center cutting plane method so that the following comparison of the new approach with this method could be more informative.

Goffin et al.'s method [14] uses feasible interior point algorithm to find an analytic center so after the elimination, a special step has to be made to restore primal feasibility. Next, few additional interior point iterations are needed to recenter the iterate, which may nonnegligibely contribute to the solution time. Thus, to keep the method efficient, we decided to perform only few (3 to 5) very aggressive eliminations with a large value of parameter $p$. The first elimination is done with $p = 0.25$ when a rough approximation of the solution is determined. The later eliminations use smaller values of $p$: $p_2 = 0.1$, $p_3 = 0.05$, $p_4, p_5 = 0.01$.

In Table 2, we show the advantages of the use of cut elimination in the analytic center cutting plane method. For two variants of this method we report: the number of outer iterations (O), the number of inner iterations, i.e. interior point iterations needed to compute the analytic center (Its), the number of cuts generated (Cuts) and the CPU time in seconds needed to reach 6-digit accurate optimum on a Power PC computer. In the number of inner iterations we distinguish in parenthesis the number of special iterations made to restore the centrality after cut elimination. Let us also note that these results have been obtained with an advanced implementation of the method that exploits special supersparse structure of cuts [13] and takes advantage of the particular structure of sparse LP constraint matrix in the master problem [19]. Additionally, knowing that the subproblem in the nonlinear multicommodity network flow is extremely cheap, we have enabled the additional

| Problem | DWCPM | | | | ACCPM with elimination | | | | PDCGM | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | O | Its | Cuts | Time | O | Its | Cuts | Time | O | Its | Cols | Time |
| NDO22 | 15 | 334 | 675 | 0.5 | 12 | 47 | 241 | 0.7 | 17 | 99 | 359 | 2.7 |
| NDO148 | 15 | 3009 | 4320 | 9.8 | 13 | 80 | 1875 | 15.6 | 16 | 94 | 1940 | 37.3 |
| Random12 | 17 | 15210 | 27200 | 387 | 17 | 246 | 10730 | 599 | 18 | 159 | 10941 | 637 |
| Random16 | 28 | 44763 | 56000 | 2136 | 20 | 260 | 14920 | 822 | 21 | 191 | 15276 | 1002 |
| Random20 | 17 | 49722 | 102000 | 5112 | 24 | 497 | 35352 | 6624 | 19 | 236 | 28420 | 3419 |
| Random31 | 31 | 326399 | 124000 | 16825 | 20 | 379 | 33772 | 9423 | 20 | 213 | 28543 | 6092 |
| Random32 | 34 | 1086443 | 170000 | 56259 | 19 | 319 | 44493 | 17615 | 21 | 196 | 40797 | 13654 |
| Random33 | 38 | $> 4e6$ | 228000 | 250000 | 24 | 401 | 59253 | 27990 | 25 | 242 | 60356 | 24574 |
| Random41 | 32 | $> 3e6$ | 224000 | 398192 | 24 | 385 | 71842 | 31585 | 27 | 265 | 70089 | 30325 |
| Random42 | 35 | $> 3e6$ | 280000 | 597879 | 24 | 411 | 76918 | 45038 | 21 | 214 | 62017 | 33244 |
| Random51 | 17 | 70507 | 136000 | 10007 | 22 | 489 | 55025 | 33758 | 18 | 230 | 40930 | 19314 |
| Random52 | 18 | 91951 | 180000 | 17612 | 24 | 562 | 62432 | 46489 | 19 | 258 | 51006 | 25825 |
| Random53 | 17 | 82859 | 204000 | 22989 | 25 | 635 | 75479 | 57214 | 18 | 270 | 58301 | 34389 |
| Random54 | 36 | 364287 | 324000 | 54417 | 22 | 501 | 66698 | 57035 | 19 | 249 | 59051 | 36393 |
| Random55 | 22 | 271321 | 242000 | 44844 | 24 | 632 | 81030 | 91384 | 22 | 306 | 66149 | 51841 |

Table 3: Comparison of three different column generation schemes.

call to it at the promising point lying close to the optimum of the current relaxed master problem (cf. [9] for details). Consequently, the number of outer iterations in the method with elimination is different from the number of subproblem calls. The latter number is given in parenthesis.

The analysis of results collected in Table 2 confirms that cut elimination based on our inexpensive test is a useful computational technique. It brought a considerable reduction of cuts that simplified the restricted master problems and saved an important number of inner iterations. These savings translated into the average 50% reduction of the CPU time.

In Table 3 we report the results of solving the same set of problems with the method proposed in this paper (PDCGM) and we also compare them with other methods: the Dantzig-Wolfe decomposition (DWCPM) [6] and the Analytic Center Cutting Plane Method (ACCPM) [14, 13, 19]. For all methods we state: the number of outer iterations (O), the number of inner iterations (Its), the number of columns generated (Cuts/Cols), and the CPU time in seconds needed to reach a 6-digit accurate optimum on a Power PC computer. Note that both "central" methods: ACCPM and PDCGM use cut elimination based on inexpensive test mentioned earlier. PDCGM employs an infeasible interior point algorithm. It then deals easily with a small infeasibility resulting from columns elimination, so there was no need to limit the number of eliminations in it. However, these eliminations are significantly less aggressive than in ACCPM (in PDCGM, parameter $p$ is always kept equal to 0.01).

Table 3 illustrates the advantages of the primal-dual column generation method. For instance, when large problems are solved (with at least 2000 arcs), PDCGM almost always needs less outer iterations than ACCPM. Regardless the size of the problems solved, the number of outer iterations in PDCGM seldom exceeds 20, which is an appreciable result. It can be explained by the strategy of the choice of accuracy required in the solution of subsequent restricted master problems (cf. eqn. (7)). This choice does not enforce an exaggerated and needless accuracy (as is the case in the optimal point strategy applied in DWCPM) but dynamically adjusts it at any given stage to the real needs of the optimization process.

The second observation stemming from the analysis of these results is a remarkable reduction of the number of inner iterations when compared with ACCPM. This reduction varies for large

| Problem | DWCPM | ACCPM | PDCGM | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | $\delta = 1.1$ | $\delta = 1.2$ | $\delta = 1.3$ | $\delta = 1.5$ | $\delta = 2.0$ | $\delta = 3.0$ | $\delta = 5.0$ |
| NDO22 | 540 | 145 | 142 | 122 | 119 | 126 | 138 | 155 | 177 |
| NDO148 | $> 10000$ | 866 | 754 | 708 | 733 | 821 | 999 | 1234 | 1599 |

Table 4: Comparison of column generation schemes: aggregate cuts.

problems between 40% and 50%. There are at least two reasons for this reduction. One is a general better performance of the primal-dual method over the primal projective algorithm applied in ACCPM. The second reason resides in the use of an efficient warm start procedure [17] to re-optimize. Naturally, the reduction in the number of interior point iterations has obvious consequences on the overall efficiency of the method: PDCGM is, on the average, 10% and 40% faster than ACCPM on very large problems. This is regardless the fact that PDCGM uses a general purpose LP code [15] and not a sophisticated structure exploiting implementation of the interior point method [13, 19] as is the case of ACCPM.

Finally, we can observe that the Dantzig-Wolfe decomposition sometimes gets into trouble when applied to the nonlinear multicommodity network flow problems. This is in contrast with its very good behavior for linear multicommodity network flow problems [21]. Although our experimental implementation of the Dantzig-Wolfe decomposition is based on an efficient simplex code—Cplex 3.0 [5], it is unreliable when applied to the solution of large scale problems.

One can see from the results collected in Table 3 that DWCPM approach is very much sensitive to the number of arcs in the network (recall that the nonlinear term in the objective penalizes for flows approaching the arc capacity $\phi_{ij}(y) = \frac{y}{C_{ij}-y}$). DWCPM runs efficiently for problems with up to 2000 arcs; in fact, it is the most efficient approach on problems Random51, Random52 and Random53. However, it starts to suffer when the number of arcs is further increased. This approach is still able to solve problems with 3000 arcs but it fails for larger networks. On the problem Random33 with 4000 arcs DWCPM reached the relative accuracy of 0.0005 after about 180000 CPU seconds and stalled. We let the program run till 250000 CPU seconds, (10 times longer than the solution of this problem with PDCGM) before it has been interrupted. DWCPM failed when applied to problems with 5000 arcs. It had reached the relative accuracy of 0.059 and 0.023 on the problems Random41 and Random42, respectively before Cplex reported numerical difficulties.

**Aggregated cuts**

Our last experiment shows the behavior of the primal-dual column generation method in a case when only one cut is added at every call to the oracle: this cut is the aggregation of all cuts generated by the independent subproblems. Again we compare the approach proposed in this paper with the Dantzig-Wolfe decomposition and with the analytic center cutting plane method. In Table 4 we present the results of solving two NDO problems with the three approaches compared: we report the number of calls to the subproblems, i.e., the number of outer iterations. As there is exactly one cut generated at every call to the subproblem, this number indicates also the total number of cuts. To give more insight into the role of centrality in the nondifferentiable optimization, we have run PDCGM for several different values of $\delta$ (cf. (7)) varying from 1.1 to 5.

From the results collected in Table 4 one can conclude that the optimal point strategy does not perform well with the single cut approach (actually, it fails for NDO148). Both central point strategies converge to the optimal solution in a competitive number of steps. PDCGM with the

conservative strategy $\delta \in [1.1, 1.5]$ (such a small value of $\delta$ promotes "centrality" over "optimality") is the most efficient approach. It is worth to note that increasing $\delta$ above 1.5 causes a loss of efficiency in PDCGM. This is not surprizing because for larger $\delta$ PDCGM strategy approaches the optimal point strategy of DWCPM. However, unlike DWCPM primal-dual column generation is still able to reach the optimal solution even for large values of $\delta$. This effect confirms that interior point methods are excellent tools to resolve degeneracy in column generation schemes.

# 6 Conclusions

We have shown in this paper the computational advantages originating from the use of the primal-dual method in the column generation approach. The proposed method shares the advantages of the analytic center cutting plane method in the sense that it uses central prices.

It is more efficient than ACCPM when applied to solve difficult large scale multicommodity network flow problems. This is due in part to the high overall efficiency of the primal-dual interior point method and in another part to the successful strategy which consists in dynamically adjusting the accuracy (required to solve subsequent restricted master problems) to the current needs of the column generation scheme. The latter strategy is controlled with the notion of $\mu$-centers and this control is done in a particularly easy and comprehensive way.

The computational results given in this paper were restricted to a single class of problems. It is not obvious that the comparison of ACCPM with the proposed approach would always favorize the latter to such an extent when applied to other classes of problems. However, these preliminary results certainly show that our new approach is very promising.

**Acknowledgement**
We would like to thank Professor Jean-Philippe Vial for many suggestions that helped us to improve the paper.

# References

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows*, Prentice-Hall, New York, 1993.

[2] E. Andersen, J. Gondzio, C. Meszaros, and X. Xu, *Implementation of interior point methods for large scale linear programming*, in Interior Point Methods in Mathematical Programming, T. Terlaky, ed., Kluwer Academic Publishers, 1996, pp. 189–252.

[3] M. S. Bazaraa, J. J. Jarvis, and H. F. Sherali, *Linear Programming and Network Flows*, John Wiley & Sons, New York, second ed., 1990.

[4] J. F. Benders, *Partitioning procedures for solving mixed-variables programming problems*, Numerische Mathematik, 4 (1962), pp. 238–252.

[5] Cplex Optimization Inc., *Using the Cplex Callable Library*, Incline Village, NV 89451-9436, 1995.

[6] G. B. Dantzig and P. Wolfe, *The decomposition algorithm for linear programming*, Econometrica, 29 (1961), pp. 767–778.

[7] D. den Hertog, J. Kaliski, C. Roos, and T. Terlaky, *A logarithmic barrier cutting plane method for convex programming*, Annals of Operations Research, 58 (1995), pp. 69–98.

[8] D. den Hertog, C. Roos, and T. Terlaky, *A built-up variant of the path-following method for LP*, Operations Research Letters, 12 (1992), pp. 181–186.

[9] O. du Merle, J.-L. Goffin, and J.-P. Vial, *On the comparative behavior of Kelley's cutting plane method and the analytic center cutting plane method*, tech. rep., Logilab, University of Geneva, 102 Bd Carl-Vogt, CH-1211, March 1996. To appear in *Computational Optimization and Applications*.

[10] J. Elzinga and T. Moore, *A central cutting plane algorithm for the convex programming problem*, Mathematical Programming, 8 (1973), pp. 134–145.

[11] E. M. Gafni and D. P. Bertsekas, *Two-metric projection method for constrained optimization*, SIAM Journal on Control and Optimization, 22 (1984), pp. 936–964.

[12] J.-L. Goffin, *The ellipsoid method and its predecessors*, in Recent Advances in System Modelling and Optimization, L. Contesse, R. Correa, and A. Weintraub, eds., Lecture Notes in Control and Information Sciences 87, Springer-Verlag, Berlin, 1987, pp. 127–141.

[13] J.-L. Goffin, J. Gondzio, R. Sarkissian, and J.-P. Vial, *Solving nonlinear multicommodity flow problems by the analytic center cutting plane method*, Mathematical Programming, 76 (1997), pp. 131–154.

[14] J.-L. Goffin, A. Haurie, and J.-P. Vial, *Decomposition and nondifferentiable optimization with the projective algorithm*, Management Science, 38 (1992), pp. 284–302.

[15] J. Gondzio, *HOPDM (version 2.12) – a fast LP solver based on a primal-dual interior point method*, European Journal of Operational Research, 85 (1995), pp. 221–225.

[16] ——, *Multiple centrality corrections in a primal-dual method for linear programming*, Computational Optimization and Applications, 6 (1996), pp. 137–156.

[17] ——, *Warm start of the primal-dual method applied in the cutting plane scheme*, tech. rep., Logilab, University of Geneva, 102 Bd Carl-Vogt, CH-1211, May 1996, revised in April 1997. To appear in *Mathematical Programming*.

[18] J. Gondzio, O. du Merle, R. Sarkissian, and J.-P. Vial, *ACCPM - a library for convex optimization based on an analytic center cutting plane method*, European Journal of Operational Research, 94 (1996), pp. 206–211.

[19] J. Gondzio, R. Sarkissian, and J.-P. Vial, *Using an interior point method for the master problem in a decomposition approach*, European Journal of Operational Research, 101 (1997), pp. 577–587.

[20] C. C. Gonzaga, *Path–following methods for linear programming*, SIAM Review, 34 (1992), pp. 167–224.

[21] K. L. Jones, I. J. Lustig, J. M. Farvolden, and W. B. Powell, *Multicommodity network flows: the impact of formulation on decomposition*, Mathematical Programming, 62 (1993), pp. 95–117.

[22] J. E. Kelley, *The cutting plane method for solving convex programs*, Journal of the SIAM, 8 (1960), pp. 703–712.

[23] A. Y. Levin, *On an algorithm for the minimization of convex functions over convex functions*, Soviet Mathematical Doklady, 6 (1965), pp. 286–290.

[24] J. E. Mitchell and B. Borchers, *Solving real-world linear ordering problems using a primal-dual interior point cutting plane method*, Annals of Operations Research, 62 (1996), pp. 253–276.

[25] A. Nemirovskii, *Lecture notes.* Technion, The Israel Institute of Technology, Faculty of Industrial Engineering & Management, 1994.

[26] G. Sonnevend, *New algorithms in convex programming based on a notion of "centre" (for systems of analytic inequalities) and on rational extrapolation*, in Trends in Mathematical Optimization : Proceedings of the 4th French–German Conference on Optimization in Irsee, Germany, April 1986, K. H. Hoffmann, J. B. Hiriat-Urruty, C. Lemarechal, and J. Zowe, eds., vol. 84 of International Series of Numerical Mathematics, Birkhäuser Verlag, Basel, Switzerland, 1988, pp. 311–327.

[27] P. M. Vaidya, *A new algorithm for minimizing a convex function over convex sets*, in Proceedings of the 30th Annual IEEE Symposium on Foundations of Computer Science, Los Alamitos, CA, USA, 1989, IEEE Computer Society Press, pp. 338–343.