Paper

A proposition to exploit the partially linear structure of the nonlinear multicommodity flow optimization problem

Paweł M. Białoń

Abstract — Optimization problems arising in telecommunications are often large-scale nonlinear problems. Usually their big size is generated mainly by their linear parts but the existence of small or medium nonlinear parts prevents us from directly tackling them with linear solvers, which are efficient. Instead, the author has proposed a method to decompose big nonlinear problems into nonlinear and linear parts. Its coordination procedure uses two auxiliary solvers: quadratic and pure nonlinear. The procedure falls in the class of projection methods. Special cuts proposed by the author allow to avoid an excessive zigzagging while not enormously increasing the complexity of both the parts. The validity of these cuts can be analyzed within the framework of obtuse cone model. Here the author summarizes the method and analyses its applicability to nonlinear multicommodity flow problems. The structure and particular sizes of this problem make the method useful. The considerations are illustrated by a numerical example with a multicommodity flow problem.

Keywords — multicommodity flow problem, projection methods, large nonlinear problems.

1. Introduction

Nonlinear multicommodity flow optimization problems have become a standard mathematical tool in the areas of networks design and flow control. Unfortunately, such problems are usually large. However, like in many other large nonlinear optimization problems, their large size is formed mainly by linear functions, equations, etc. This big linear part of a large nonlinear problem could be itself tackled with efficient linear programming techniques but one must take into account the existence of the small nonlinear part of the problem. Thus we can only think of solving the problem with the efficiency close to the efficiency with which its linear part alone would be solved.

For this sake the author has proposed in [3] a hierarchical optimization algorithm for large nonlinear optimization problems into a big linear part and a small nonlinear part. The obtained subproblems are: a large quadratic subproblem (with constraints from the linear part of the original problem) and a small nonlinear one (with constraints from the nonlinear part of the original problem). The nonlinear subproblem is computationally easy due to its small size; at least the same applies to the coordination procedure. Thus the efficiency of the whole algorithm depends on the efficiency of the quadratic solver applied to the quadratic subproblem and can reach a very high level due to the observed progress in quadratic programming, polynominal techniques etc.

The original author's proposition was not directed to telecommunication applications; it covered a quite general class of large nonlinear problems with big linear parts. However, four particular structural properties of the problems were needed to make the proposition work properly and efficiently. It turns out that these properties are possessed by nonlinear multicommodity flow (MCF) problems, thus making the proposition especially adequate for these problems. This adequateness is shown in this paper.

The coordination procedure of the author's method is a variant of projection methods for feasibility problems[1] [2, 5, 7] with accelerating cuts. The distinguishing features of the authors proposition are a technique of full cuts cumulation, and specially constructed cuts, so called Z-cuts, that allow to decrease the complication of sets shapes caused by cutting. The initial optimization problem can be reduced to a sequence of feasibility problems with the level control technique [8] and these can be then solved with the author's method.

In Section 2 of this paper the proposed method is first summarized, very briefly and with references to [3]. First, the class of large nonlinear feasibility problems solved by the method is defined. Then the idea of projection methods and accelerating them by cuts are sketched. Then follows the description of the author's proposition, involving: the definition of sets forming the upper-level feasibility problem, the realization of projections with optimization subproblems, the definitions of used cuts and the final algorithm statement.

In Section 3 the MCF problem (of a specific subclass) is defined and the suitability of the proposed method to its solving is indicated. The section ends with a numerical illustration with an artificially created MCF problem, aiming in understanding the proposition. In Section 4 the author gives some conclusions and argues his method can be taken into account as an element in a construction of algorithm solving a large MCF problem.

[1] A *feasibility problem* is a problem of finding a point satisfying a set of constraints.

2. The proposed method

2.1. Large nonlinear problem formulation

The initial optimization task is defined as follows:

$$\min_{x\in\mathbb{R}^{n_N}, y\in\mathbb{R}^{n_L}} f(x) \qquad f:\mathbb{R}^{n_N}\to\mathbb{R}$$

s.t. (subject to)

$$\tilde{g}(x)\le 0 \qquad \tilde{g}:\mathbb{R}^{n_N}\to\mathbb{R}^{m_N-1}$$
$$A(x^\top,y^\top)^\top\le b \qquad A \text{ is a matrix of size } m_{LI}\times n$$
$$B(x^\top,y^\top)^\top = d \qquad B \text{ is a matrix of size } m_{LE}\times n$$
$$x^{lo}\le x\le x^{up}, y^{lo}\le y\le y^{up}, \tag{1}$$

where functions f and $\tilde{g}_i$ are continuous, quasiconvex, x^{lo},x^{up},y^{lo},y^{up} are constant vector bounds. The above problem can be reduced [8] to a sequence of feasibility problems $F(Q)$ parametrized with a real number Q. Each problem $F(Q)$ consists in finding $(x^\top,y^\top)^\top$ that satisfies:

$$\begin{aligned} g(x)&\le 0\\ A(x^\top,y^\top)^\top&\le b\\ B(x^\top,y^\top)^\top&= d\\ x^{lo}\le x\le x^{up}, y^{lo}&\le y\le y^{up}, \end{aligned}\tag{2}$$

where function $g:\mathbb{R}^{n_N}\to\mathbb{R}^{m_N}$ was obtained from function $\tilde{g}$ by adding a new coordinate saying how much the goal function value exceeds Q, i.e. $g_i(\cdot)\stackrel{\text{def}}{=}\tilde{g}_i(\cdot)$, $i=1,\ldots m_N-1$, $g_{m_N}(\cdot)\stackrel{\text{def}}{=} f(\cdot)-Q$.

The feasibility problem has n_N nonlinear variables[2], n_L linear variables, m_N nonlinear inequality constraints, m_{LI} linear inequality constraints, m_{LE} linear equality constraints. Let $m=m_N+m_{LI}+m_{LE}$, $n=n_L+n_N$. The better $m_N\ll m$ and $n_N\ll n$, are fulfilled, the more efficient will be the algorithm.

2.2. The idea of projection methods

Projection methods serve to solving the following *convex feasibility problem*:

Find

$$x\in S\stackrel{\text{def}}{=}\bigcap_{i=1,\ldots m} G_i\,, \tag{3}$$

where $G_i\subset\mathbb{R}^n$ are closed, convex sets. In practice G_i are often defined as sets of points allowed by some constraints. By now we assume that S is nonempty. In the description of the solving process we shall confine ourselves with the case of $m=2$.

For $x\in\mathbb{R}^n$ and a closed convex nonempty $C\subset\mathbb{R}^n$ we shall denote by P_Cx the orthogonal projection of x onto C, $P_Cx=\arg\min_{y\in C}\|x-y\|^2$. The *projection vector* for such a projection is P_Cx-x. It can be shown that such a projection is defined uniquely.

The simplest way to search for the solution consists in performing sequential alternate projections onto G_1 and G_2; i.e., given the starting point x^0, we produce a sequence

$$x^1=P_{G_1}x^0, x^2=P_{G_2}x^1, x^3=P_{G_1}x^2, \text{ etc.} \tag{4}$$

We assume such projections are easily realizable numerically.

The basic fact in convergence analysis of projection methods is that the projection operator possesses the Fejér contraction property.

Definition 1. *A finite or infinite sequence (x^i) of points in a Hilbert space H has the Fejér contraction property with respect to $C\subset H$ if*

$$\|x^i-c\|^2\ge\|x^{i+1}-c\|^2+\|x^{i+1}-x^i\|^2 \tag{5}$$

for each $c\in C$. Similarly, operator $O:H\to H$ has this property if for each $c\in C$ and $x\in H$ $\|x-c\|^2\ge\|Ox-c\|^2+$ $+\|Ox-x\|^2$.

Fact 1. *Projecting onto a nonempty closed convex set of points in $\mathbb{R}^n$ has Fejér contraction property with respect to this set, and, consequently, to each of its nonempty sets.*

For a proof of the above fact see calculations on page 228 in [11] with $t_{\min}=t_{\max}=1$.

After putting $C=S$ we see that with every projection performed in our algorithm (4) we decrease the squared norm from (any but fixed) point $c\in S$ by at least the square of the appropriate step (projection vector) length. Later it will suffice to assure certain lengths of steps to establish the convergence[3].

Alternatively, the Fejér contraction property of projections in our algorithm means that we approach each solution point with an acute angle.

Zigzagging often slows down projection methods: we may approach the solution with an angle less than but close to $\pi/2$, making the distance from a solution decrease very slowly. This happens in an example in Fig. 1; there, moreover, consecutive projection vectors form angles close to π.

Cuts serve as a standard remedy for zigzagging; a cut is an inequality of the form $\langle\cdot-a,b\rangle\ge\langle b,b\rangle\ge 0$ with fixed $a,b\in\mathbb{R}^n$; *its hyperplane* $H(a,b)$ is given as $\{x\in\mathbb{R}^n:\langle x-a,b\rangle=\langle b,b\rangle\}$, *its halfspace* – as $\{x\in\mathbb{R}^n:$ $\langle x-a,b\rangle\ge\langle b,b\rangle\}$.

Using cuts means replacing (4) with

$$x^1=P_{G_1'^1}x^0, x^2=P_{G_2'^2}x^1, x^3=P_{G_1'^3}x^2, \text{ etc.} \tag{6}$$

where sets $G_1'^k$ and $G_2'^k$ $(k=1,2,3,\ldots)$ are G_1 and G_2 narrowed by some cuts, i.e., they were obtained from G_1

[2] A *nonlinear variable* is a problem variable involved in at least one nonlinear function in the model formulation; the remaining variables will be called *linear*.

[3] Which is usually easy and is done with the notion of problem *regularity* [2].

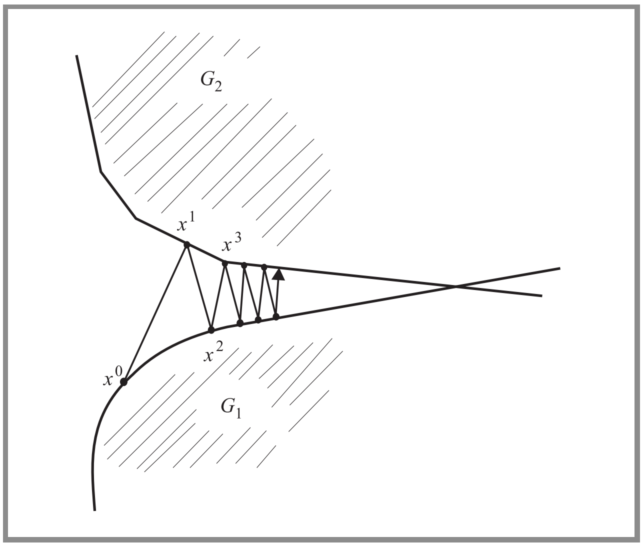

Fig. 1. Zigzagging.

and G_2 by intersecting G_1 and G_2 with halfspaces of some cuts.

A *geometric cut based on (constructed after) the projection* of $x \notin G$ onto close convex G, $G \supset S$ is defined as

$$\langle \cdot - x, P_G x - x\rangle \geq \langle P_G x - x, P_G x - x\rangle .$$

In Fig. 2, unlike in Fig. 1, point x^3 was obtained by projecting x^2 not onto G_2 but onto G_2 narrowed by the geometric cut constructed after projection of x^2 onto G_1. H is a hyperplane of this cut. We see that the step made is longer and we approach the solution with a smaller angle.

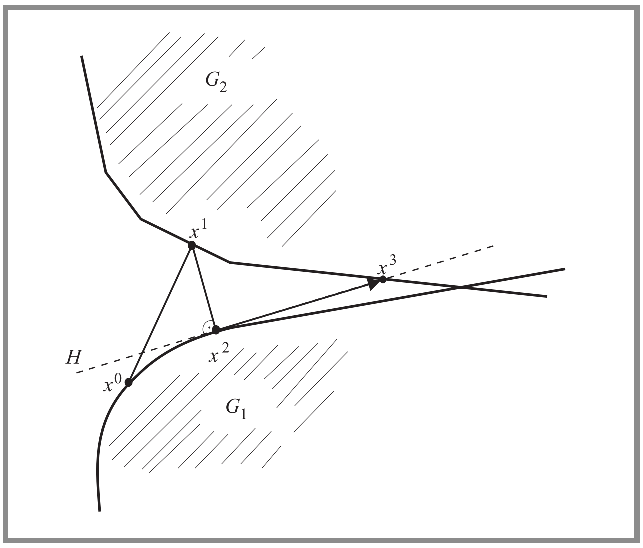

Fig. 2. A geometric cut reduces zigzagging.

A cut is called *valid* or *proper* if it is satisfied for each point in the solution set S. Validity is necessary to assure that projections on narrowed sets (i.e., $G_1'^{\,k}$ or $G_2'^{\,k}$) still possesses the Fejér contraction property with respect to S; moreover we do not want our method to degenerate by producing empty $G_1'^{\,k}$ or $G_2'^{\,k}$. Geometric cuts constructed after a projection of an $x \notin G$ onto nonempty, closed, convex $G \supset S$ can be easily shown to be proper.

We may narrow set G_1 or G_2 with only one cut but it may bring a profit in efficiency to narrow them with several cuts simultanously (i.e., to intersect G_1 or G_2 with the intersection of the halfspaces of several cuts). Various techniques for cuts cumulation are given in [4, 5, 9, 10, 13] and a specifically understood cumulation will be also used here.

2.3. The idea of the method

In order to solve our feasibility problem (2) we need to somehow transform it to the form of expression (3).

The following sets N and L will play the role of G_1 and G_2 in (3):

$$N = \{x \in \mathbb{R}^{n_N} : g(x) \leq 0 \wedge x^{lo} \leq x \leq x^{up}\}$$
$$L = \Big\{x \in \mathbb{R}^{n_N} : x^{lo} \leq x \leq x^{up} \wedge \exists_{y \in \mathbb{R}^{n_L}} \Big(y^{lo} \leq y \leq y^{up} \wedge \wedge A(x^\top, y^\top)^\top \leq b \wedge B(x^\top, y^\top)^\top = d\Big)\Big\}.$$

Notice that these are not actually the sets of points allowed by nonlinear and linear constraints but their orthogonal projections on the subspace of nonlinear variables. The projection method will be defined in this subspace.

The projection method of solving the feasibility problem of finding a common point of N and L will form the higher level of decomposition. The lower level will serve to realize the projections.

Finding the projection of point $z \in \mathbb{R}^{n_N}$ onto N may be realized as solving the nonlinear optimization problem

$$\min_{x \in \mathbb{R}^{n_N}} \frac{1}{2}\|x - z\|^2 \quad \text{s.t.} \quad x \in N . \tag{7}$$

Finding the projection of point $z \in \mathbb{R}^{n_N}$ onto L might be realized as solving the quadratic subproblem

$$\begin{aligned} &\min_{x \in \mathbb{R}^{n_N}, y \in \mathbb{R}^{n_L}} \frac{1}{2}\|x - z\|^2 \\ &\text{s.t.} \\ &A(x^\top, y^\top)^\top \leq b \\ &B(x^\top, y^\top)^\top = d \\ &x^{lo} \leq x \leq x^{up} \\ &y^{lo} \leq y \leq y^{up} . \end{aligned} \tag{8}$$

Note that if the solution $(x^{\star\top}, y^{\star\top})^\top$ of the later subproblem satisfies $x^\star \in N$ then it also solves the initial feasibility problem (2). Later one may use either the whole solution $(x^{\star\top}, y^{\star\top})^\top$ of (8) or only vector $x^\star$. The former is appropriate in communication with the user (the printout of final solution) while the later is more convenient in the algorithm description. The following consideration will be

in principle done in the subspace $\mathbb{R}^{n_N}$ of nonlinear variables. Placing the higher level of decomposition in this low-sized subspace will certainly increase the efficiency of calculations. This subspace can be really considered low-dimensional when $n_N \ll n$. The other reason why we required $n_N \ll n$ and also $m_N \ll m$ is connected with the nonlinear subproblem (7): both the inequalities make it easy by reducing the number of its variables and of its constraints, respectively.
For the generated sequence of points we shall use the following notation, slightly different from (6) and given in a recursive form:

$$\vec{x}^k = P_{L'^k}\breve{x}^{k-1},\ \breve{x}^k = P_{N'^k}\vec{x}^k \qquad k = 1, 2, \ldots \tag{9}$$

Sets N'^k and L'^k were obtained from L and N by narrowing with some (possibly by no) cuts.

2.4. Cuts

We shall measure zigzagging $Z_{(y^i)}(l)$ (with $k < l$) of a finite sequence $(y^i)_{i=0}^{l}$ of points in a Hilbert space as:

$$Z_{(y^i)}(k) = \frac{\sum_{i=k}^{l-1} \|y^{i+1} - y^i\|}{\|y^l - y^k\|}. \tag{10}$$

If x^i are points generated by some projection method we should try to keep $Z_{\cdot}(i)$ as small as possible. For this sake we shall introduce the *full cumulation* of geometric cuts. Namely, if a cut was constructed after the projection of point x^i onto some set (and x^{i+1} is the result of this projection) then this cut affects all the subsequent projections, which means that these projections are done onto sets narrowed by (maybe not only) this cut. In other words, all the subsequent points x^j must satisfy the cut. One of the alternatives for the full cuts cumulation is using noncumulated cuts: each cut affects only the nearest projections.
For the full cuts cumulation we can nicely assess the sequence zigzagging.

Theorem 1. *Let a sequence* $(x^i)_{i=0}^{n}$ *(where* $n \geq 1$*) of points in a Hilbert space satisfies the cumulated geometric cuts condition:*

$$\forall_{s,1\leq s\leq n-1}\,(x^s - x^{s-1})^\top (x^n - x^s) \geq 0. \tag{11}$$

Then the following assessment for the sequence zigzagging holds:

$$Z_{(x^i)}(n) \equiv \frac{\sum_{i=0}^{n-1} \|x^{i+1} - x^i\|}{\|x^n - x^0\|} \leq \sqrt{n}. \tag{12}$$

Proof. See the proof of Theorem 1 in [3].

The analysis of the the theorem proof convinces also that usually the zigzagging places below the above limit; inequality (12) is fulfilled as equality only for very particular configurations of points x^i.
We shall describe the firts two types of cuts present in the method. In kth iteration the following cuts are constructed:

1. $\langle \cdot - \vec{x}^k, \breve{x}^k - \vec{x}^k\rangle \geq \langle \breve{x}^k - \vec{x}^k, \breve{x}^k - \vec{x}^k\rangle$ – type A cuts. They are later used, once or many times[4], to narrow set L.

2. $\langle \cdot - \breve{x}^k, \vec{x}^k - \breve{x}^{k-1}\rangle \geq \langle \vec{x}^{k-1} - \breve{x}^{k-1}, \vec{x}^k - \breve{x}^{k-1}\rangle$ – type B cuts. They are later used, once or many times, to narrow set N.

   These are simply geometric cuts, but we distinguish the cuts made after projections onto N'^k (type A) and after projections onto L'^k (type B).

   It is possible to apply Theorem 1 to our algorithm. If we take sequence $\breve{x}^0$, $\vec{x}^1$, $\breve{x}^1$, $\vec{x}^1$, ... as sequence (x^i) in this theorem, cumulating both A-type and B-type cuts will assure the satisfaction of (11) for $n \geq 1$, thus the sequence will not zigzag too strongly.

   However, the cuts of both the types have their numerical drawbacks that increase in case of cumulation. A-cuts influence the definition of (subsequent) sets L'^i and thus complicate the quadratic optimization subproblem. The complication may consist in introducing nonzero elements in the sparse constraint matrix of this problem (approximately n_N ones per cut). Also, we cannot be certain that the subsequent cuts will not decrease the problem conditioning, e.g. by aligning almost in parallel. The main disadvantage of cumulating B-cuts origins from the small size of the nonlinear optimization subproblems: the relative complication introduced in these problems by many cuts may be large.

   Fortunately, it has turn out possible to resign cumulating cuts of one of the above types in the algorithm, while preserving the applicability of Theorem 1. With the trick described later, the user may resign generating (not only cumulating!) cuts of one of the types. The choice of the type should depend on particular problem properties. Due to the symmetry of the question, from now we shall only consider the case of giving up generating the B-type cuts.

   The trick consists in generating in kth iteration cuts of the third type:

3. $\langle \cdot - \vec{x}^k, \vec{x}^k - \vec{x}^{k-1}\rangle \geq \langle \vec{x}^k - \vec{x}^{k-1}, \vec{x}^k - \vec{x}^{k-1}\rangle$ – type Z cuts. They are later used to narrow set L.

   When we take sequence $(\vec{x}^i)$ as sequence (x^i) in the assumption of Theorem 1 and decide to cumulate Z-cuts, the theorem will limit the zigzagging of this sequence. However, we must prove that each Z-cut is proper. Fortunately, we can show the propriety of the Z-cut constructed in kth iteration, on condition that the A-cut constructed in $(k-1)$th iteration was taken into account in definition of L'^{k-1}. This Z-cut is shown to be proper as implied by two proper cuts: the mentioned A-cut constructed in iteratin $k-1$ and the B-cut that we might have (but have not) constructed in iteration k (see Fig. 3).

[4]Depending on our decision about cumulating the cuts.

Showing this implication exceeds the scope of this paper (see Theorem 2 in [3]). In the proof the conical cuts surrogating method [4, 5, 9] were used. Surrogating techniques enable showing the propriety of a certain constructed cuts (so called surogate cut) from the propriety of several other cuts.

The applied trick is also similar to the modification in Section 5 of [4].

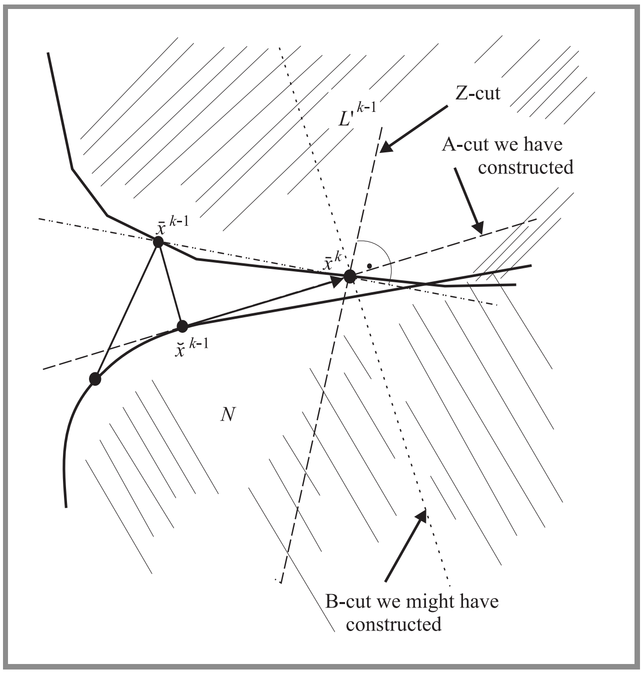

Fig. 3. Construction of Z-cut in kth iteration.

2.5. The algorithm

The algorithm will be given in its basic variant, in which B-cuts are absent, A-cuts are not cumulated, Z-cuts are cumulated.

Algorithm 1. **Parameters:** tolerance $t^N \geq 0$, starting point $\check{x}^0$.
We initialize the iteration counter k with 1.

1. Compute $\vec{x}^k = P_{L'^k}\check{x}^{k-1}$ with L'^k being L narrowed by some cuts constructed in earlier iterations:

$$L'^k = \{y \in L : \langle y - \vec{x}^{k-1}, \check{x}^{k-1} - \vec{x}^{k-1}\rangle \geq \langle \check{x}^{k-1} - \vec{x}^{k-1}, \check{x}^{k-1} - \vec{x}^{k-1}\rangle \wedge \wedge(\forall_{j\in K^k} \langle y - \vec{x}^{j-2}, \vec{x}^{j-1} - \vec{x}^{j-2}\rangle \geq \langle \vec{x}^{j-1} - \vec{x}^{j-2}, \vec{x}^{j-1} - \vec{x}^{j-2}\rangle\},$$

   where K^k equals to $\{3, \ldots k\}$, by solving the quadratic subproblem (8) with L replaced by L'^k and with the substitution $z \leftarrow \check{x}^{k-1}$. If $L'^k = \emptyset$ then STOP – report infeasibility.

2. Compute $\check{x}^k = P_N \vec{x}^k$ by solving the nonlinear subproblem (7) with the substitution $z \leftarrow \vec{x}^k$. If $N = \emptyset$ then STOP – report infeasibility. If $\|\check{x}^k - \vec{x}^k\| \leq t^N$ then STOP – return the last solution of the quadratic subproblem. Otherwise set $k := k+1$ and go to step 1.

A detailed convergence analysis of (a slightly more general) method is given in [3, Section 7]. It bases on Fejér contraction property and the regularity analysis of the problem and zigzagging; since the used cuts are proper, the Fejér contraction mechanism is not disturbed. The analysis conceives also the case of infeasibility: $L \cap N = \emptyset$. Based on the guaranteed sequence zigzagging, the moment of detection of infeasibility is assessed.

3. The applicability of the method to the multicommodity flow problem

3.1. The multicommodity flow problem

We shall formulate a variant of a problem of the well known class of multicommodity flow problems [15]. Let us represent a telecommunication network as a directed graph. Let the graph nodes be represented by integer numbers from the set $I \stackrel{\text{def}}{=} \{1, \ldots, N\}$, the directed arcs – as members of a set $E \subset I \times I$ (arc (i, j) will correspond to the unidirectional link from node i to node j).
Various commodities (various kinds of information) are to be transported through our network; let us number them with $1, \ldots K$. The demand on kth commodity in ith node is given by the real parameter $r^{i,k}$, while its negative value denotes that the node actually emits the commodity (in the amount of $|r^{i,k}|$). Define decision variables in our problem:

- $\phi_u^k \in \mathbb{R}$, $u \in E$, $k = 1 \ldots K$ – the kth commodity flow in arc u.
- $\psi_u \in \mathbb{R}$, $u \in E$ – total flow of all commodities in arc u.

The flow ψ_u in arc u costs $\Phi(\psi_u)$, where $\Phi : \mathbb{R}_+ \mapsto \mathbb{R}_+$ is an increasing function. The cost can have various real-world interpretations. For example, it can represent the cost of reconstruction of link u to the capacity of ψ_u or it can be a certain measure of slowness of the link.
The multicommodity flow optimization problem consists in minimizing the total cost of network flow and is formulated as follows:

$$\min_{\phi_u^k} \sum_{u\in E} \Phi(\psi_u) \quad (13)$$

s.t.

$$\psi_u = \sum_{k=1}^{K} \phi_u^k \quad \text{for } u \in E \quad (14)$$

$$\sum_{(i,j)\in E} \phi_{(j,i)}^k - \sum_{(i,j)\in E} \phi_{(i,j)}^k = r^{i,k} \quad \text{for } i \in I, k = 1, \ldots K \quad (15)$$

$$\phi_u^k \geq 0 \quad \text{for } u \in E, k = 1, \ldots K. \quad (16)$$

Equation (13) defines the total cost of network flow, (14) defines the total flow in each arc u, (15) expresses the Kirchoff law for each node i and, finally, (16) reflects the unidirectional character of arcs.
The resulting feasibility problem $F(Q)$ takes the form

$$\begin{aligned} &\text{Find } \phi_u^k \in \mathbb{R}, \psi \in \mathbb{R} \\ &\quad \text{satisfying} \\ &\sum_{u \in E} \Phi(\psi_u) \le Q \text{ and} \\ &\quad (14)-(16). \end{aligned} \tag{17}$$

3.2. The applicability of the proposed method

Let us summarize the important properties of the feasibility problem (2) important for the proper and efficient work of our algorithm:

1. The nonlinear equality constraints are absent.
2. The nonlinear inequality constraint functions must be quasiconvex.
3. $n_N \ll n$, where n_N is the number of nonlinear variables, n is the number of all variables, is appreciated.
4. $m_N \ll m$, where m_N is the number of nonlinear constraints, m is the number of all constraints, is appreciated.

The key observation in this paper is that these properties are possessed by problem (17):

Ad 1. Obviously.

Ad 2. The increasing character of Φ implies its quasiconvexity and thus the quasiconvexity of $\sum_{u \in E} \Phi(\psi_u)$ treated as a vector function of ψ_us. It should be stressed that continuous, increasing but concave Φ, typical in practice due to the economy-of-scale phenomenon, are acceptable[5].

Ad 3. Note that $n_N = |E|$, $n = K \cdot |E|$.

Ad 4. Note that $m_n = 1$, $m = K \cdot N$.

Observe also that the last two properties are the better fulfilled the greater is the number K of commodities.

3.3. Numerical illustration

The method was applied to an artificial multicommodity flow problem of class (13)–(16). The aim of experiments was to show the relations between particular sizes of the problem (and subproblems) we can deal with, to simply validate the method by analyzing its results and to investigate how much iterations do the coordination procedure of our method as well as the level control loop (the costs of optimization of subproblems were not investigated, since they depend on many technical details: used solvers, restarting techniques, etc.).
The bidirectional ring network, shown in Fig. 4 was used in computations.

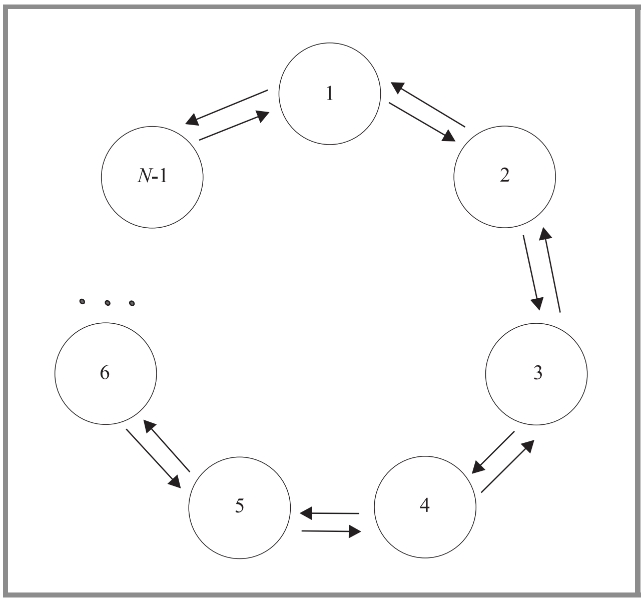

Fig. 4. The example network. Circles represent nodes, arrows represent links, numbers in circles – numbers of nodes.

$K = N$ commodities were distinguished. Each kth commodity had a single source node, namely node k, and a single collector node, namely node $((k+1) \mod N)+1$, so each commodity flew clockwise between two consecutive nodes. The flow of kth commodity amounted to the value of $1.5 \cdot k/N$. Precisely, there was:

$$\begin{aligned} &r^{k,k} = 1.5 \cdot k/N: && \text{for } k = 1, \ldots N \\ &r^{k,((k+1) \mod N)+1} = -r^{k,k} && \text{for } k = 1, \ldots N \\ &r^{i,j} = 0 && \text{for remaining pairs } (i,j). \end{aligned}$$

The number of nodes N was the parameter of the problem, and the problem structure implied the remaining sizes: $|E| = 2$ and, as said above, $K = N$.
Table 1 shows the particular sizes of the problem, seen as an instance of optimization problem (1). The same sizes are adequate also for the resulting problem (2). The number m_N of nonlinear constraints equals 1. The sizes of optimization subproblems from the decomposition scheme can be also reconstructed from this table: the nonlinear subproblem has n_N variables and $m_N = 1$ constraints and the quadratic subproblem has n variables and m constraints.
The cost function Φ was defined as $\Phi(\psi) = (1+\psi^2)^{0.4} - 1$. For small arguments this function behaves like a convex function, whereas for large arguments – like a concave one. Such a choice was aimed to show the broadness of the class of functions Φ acceptable by our method; also it introduces the speciffic phenomena in the optimized network flow (see later). It can have the following real-world interpretation: the cost of reconstruction of a link should be in principle

[5]It remains to explain why our method required property 2. It was simply necessary to make the level sets of the constraint functions convex and thus the projection methodology applicable.

given by a concave function, to acomodate the economy-of-scale phenomenon. However, for small flows, a serious reconstruction of a link might be not necessary, it perhaps suffices to make small improvements. Thus for small arguments Φ should be rather flat.

Table 1
The problems and their sizes

Problem	A	B	C	D
N	3	10	20	30
Number of variables (n_N)	25	220	840	1860
Number of constraints (m)	15	120	440	960
Number of nonlinear variables (n_N)	6	20	40	60

The reduction of optimization problem (1) to a sequence of feasibility problems (2) was done with the level control scheme [8]. This method can be viewed as a method for finding the optimal value of the problem (1). It is based on bisection of a certain interval. The initial left end L and right end U of the interval are given by the user: L and U are lower and upper bounds for the optimal value. The value Q for current feasibility problem Q is chosen somewhere in the midle on the current interval. The interval is narrowed with the following techniques:

- Values of f in feasible points generated during the algorithm course are used to update the lower bound (left end of the interval).
- Infeasibility of the feasibility problem $F(Q)$ allows to update the current upper bound (right end of the interval) to the value of Q. The infeasibility is detected by encountering that the sum of squares of made steps exceeds the square of R, the user-given diameter of a ball containing all the points generated by the algorithm[6].

The applied method varied from the original method of level control in the following aspects:

- The cuts were present when solving the feasibility problems.
- Infeasibility of a feasibility subproblem was detected much quicker by encountering the emptyness of N or L'^k (which, in turn, was detected as an infeasibility of one of the optimization subproblems), similarly as in [6].

The simple structure of the problem allows to quess its optimal value (the minimal cost). Each commodity can be reasonably sent between its source and its collector (the consecutive nodes) only in two ways: clockwise (through a single link) or clock-counterwise (through a path of length $N-1$). Since the later way engages much more links, it probably generates a bigger cost. Thus we can transport all the commodity clockwise. For such a solution the total flow cost will be certainly equal to $\sum_{k=1}^{N} \Phi(r^{k,k})$, and it will be refered later as the *heuristic optimal value of the problem or heuristic* $f^\star$, since we obtained it with a heuristic reasoning.

The method was implemented in the C++ language. The LP-DIT library [14] implementing sparse matrices and realizing linear problems storage was used.

The solver from IAC-DIDASN++ system (see e.g. [12]) and HOPD [1] were used as auxiliary solvers: respectively nonlinear and quadratic.

The parameters of the level control scheme were set as follows: $L = 0$, $U = 5N$, $R = 50N$, $\theta = 0.75$ (θ is a bisection parameter – see op.cit.). Tolerance t^N was set to $1e-4$. The results of experiments are given in Table 2, where $f^\star$ denotes the optimal value of a problem, "nfp" – the number of feasibility problems generated by the level control scheme and "total iterations" – the total number of iterations the method did in solving all feasibility problems.

Table 2
Results of experiments

Problem	A	B	C	D
Heuristic $f^\star$	1.52605	5.76155	11.81417	17.86702
Computed $f^\star$	1.37162	5.73391	11.8015	17.8588
nfp/total iterations	9/34	12/31	11/31	12/26

Experiments are commented in Section 4, here we shall only show why the comuted optimal values seem reasonable. They are slightly lower than their "heuristic" variants. This explains in the following way. While the flow in clock-counterwise is very small, the derivative of Φ at the point corresponding to such a flow is also very small. Thus it pays to send a small fraction of each demand a clock-counterwise-way, which is cheaper and which we did not take into account in our heuristic reasoning. The analysis of the values of decision variables obtained by the method supports this hypothesis. However, with the grow of N, the length of the clock-counterwise path becomes larger, the costs of sending flows clock-counterwise – larger, and the described phenomenon vanishes. This we see in Table 2: the gap between the heuristic and computed optimal values clearily vanishes with the growth of N.

4. Conclusions

The goal of the author was to show that multicommodity flow problems are a very interesting case of large nonlinear optimization problems that seem especially created for his method, mainly due to their particular sizes and absence of nonlinear equality constraints.

[6]Which, roughly speaking, contradicts to the behavior implied by the Fejér contraction property.

The simple example numerical presented above was only an illustration. It does not conceive the complexity of practical models, with their additional structural elements, some hierarchical structure necessary to account for extremely huge sizes, etc. However, the basic information for a constructor of a software solving MCF refers to the behavior of the projection method itself and is following: the method did not do many iterations, and their number did not grow with the increase of the problem size. Moreover, the size of the subspace in which the projection method operates, as well as the sizes of nonlinear subproblems were really small. Thus embedding the method in such a software thus seems worth considering. However, many technical details ought to be dealt with. The first one will perhaps concern warm restarts. Optimization subproblems, especially quadratic ones, are very similar each to other and should not be solved independently, but in each subproblem some information (e.g., some matrix factorization) from an earlier instance of the subproblem, should be preserved in order not to repeat similar computations. However, not all the quadratic solvers allow for warm restarts (e.g., the version of the quadratic solver used by the author).

Using projection methods in the presented way gives also some light to the question of how big the complication introduced to big linear MCF problems by the addition of a small nonlinear part is. This complication expressed here mainly with a number several tens of iterations in the higher level of our decomposition, and with the necessity of taking into account quadratic goal functions. Both these aspects of complication may turn out to be modest by the current and future growth of computers power and progress in quadratic programming techniques; moreover, techniques like warm restarting can decrease their meaning.

Finally, we state that, despite the precise convergence analysis given in [3], the simple example showed the validity of the method in the sense of finding a proper solution.

References

[1] A. Altman and J. Gondzio, "Regularized symmetric indefinite systems in interior point methods for linear and quadratic optimization", *Optim. Meth. Softw.*, vol. 11–12, pp. 275–302, 2000.

[2] H. Bauschke and J. Borwein, "On projection algorithms for solving convex feasibility problems", *SIAM Rev.*, vol. 38, no. 3, pp. 367–426, 1996.

[3] P. Białoń, "Large-scale nonlinear projection algorithm using projection methods", *Discus. Math. Differ. Inclus., Contr. Optim.*, vol. 20, no. 2, pp. 171–194, 1999.

[4] A. Cegielski, "A method of projection onto an acute cone with level control in convex minimization", *Math. Progr.*, vol. 85, pp. 469–490, 1999.

[5] A. Cegielski, *Relaxation Methods in Convex Optimization Problems, Higher College of Engineering*. Series Monographies, no. 67. Zielona Góra, Poland (in Polish).

[6] R. Dylewski, "Numerical behavior of the method of projection onto an acute cone with level control in convex optimization", *Discus. Math. Differ. Inclus., Contr. Optim.*, vol. 20, 1999.

[7] S. Flåm and J. Zowe, "Relaxed outer projections, weighted averages and convex feasibility", *BIT*, vol. 30, pp. 289–300, 1999.

[8] S. Kim, A. Hyunsil, and C. Seong-Cheol, "Variable target value subgradient method", *Math. Progr.*, vol. 49, pp. 356–369, 1991.

[9] K. Kiwiel, "Monotone Gram matrices and deepest surrogate inequalities in accelerated relaxation methods for convex feasibility problems", *Linear Algeb. Appl.*, vol. 215, pp. 27–33, 1997.

[10] K. Kiwiel, "The efficiency of subgradient projection methods for convex optimization", Part I: "General level methods", *SIAM Contr. Optim.*, vol. 34, no. 2, pp. 660–676, 1996.

[11] K. Kiwiel, "Block-iterative surrogate projection methods for convex feasibility problems", *Linear Algeb. Appl.*, vol. 15, pp, 225–259, 1995.

[12] T. Kręglewski, J. Granat, and A. Wierzbicki, *IAC-DIDAS-N – A Dynamic Interactive Decision Analysis and Support System for Multicriteria Analysis of Nonlinear Models, v. 4.0*. Laxenburg (Austria): International Institute for Applied Systems Analysis, June 1991, collabor. paper, CP-91-101.

[13] C. Lemaréchal, A. S. Nemirovskii, and Yu. Nesterov, "New variants of bundle methods", *Math. Progr.*, vol. 69, pp. 111–147, 1995.

[14] M. Makowski, *LP-DIT Data Interchange Tool for Linear Programming Problems (Version 1.20)*. Laxenburg (Austria): International Institute for Applied Systems Analysis, 1994, work. paper, WP-94-36.

[15] M. Minoux, "Network synthesis and optimum network design problems: moodels, solution methods and applications", *Network*, vol. 19, pp. 313–360, 1989.

[16] M. Shchepakin, "On a modification of a class of algorithms for mathematical programming", *Zh. Vychisl. Mat. i Mat. Fiz.*, vol. 19, pp. 1387–1395, 1979 (in Russian).

Paweł M. Białoń was born in Warsaw in 1971. He received his M.Sc. in computer science from Warsaw University of Technology in 1995. Currently he is employed by National Institute of Telecommunications in Warsaw. His research focuses on nonlinear optimization methods and on decision support and is directed towards the Ph.D. degree. He has participated in several projects applying decision support methods in the telecommunications, agricultural and environmental areas.
e-mail: P.Bialon@itl.waw.pl
National Institute of Telecommunications
Szachowa st 1
04-894 Warsaw, Poland