Convergence and Numerical Expirements with a Decomposition Algorithm

Nurminski, E.A.

IIASA Working Paper

WP-82-008

January 1982

Convergence and Numerical Experiments With a Decomposition Algorithm

E. Nurminski

January 1982
WP-82-8

Convergence and Numerical Experiments With a Decomposition Algorithm

E. Nurminski

January 1982
WP-82-8

Convergence and numerical experiments with a decomposition algorithm

E.Nurminski

International Institute for Applied Systems Analysis

Laxenburg, Austria

ABSTRACT

This paper gives a proof of convergence of a decomposition algorithm for solution of an optimization model consisting of two submodels. The submodels are represented by separate mathematical programming problems and are linked by dependence on common variables.

The method for coordinating the activities of submodels, in order to reach an overall optimum, is based on the approximation of the original problem which can be interpreted as the direct exchange of proposals between submodels. Computational improvements in comparison with the conventional master-subproblems schemas are shown.

1. Introduction

In the beginning of the 60's Dantzig and Wolfe proposed the widely known decomposition principle (Dantzig61a). The nature of this concept is to replace the large-scale problem by a sequence of smaller problems, each representing different sections of the initial one, with some coordinating master problem balancing the separate solutions of the subproblems.

However, computational experiments with this principle provided in some cases disappointment (Dantzig81a) In these cases the observed computational behavior of the Dantzig-Wolfe decomposition algorithm consisted of rather rapid improvement on the initial iterations of the optimization process, with slow convergence in the final stage. This resulted in many cycles between subproblems and the master problem, and this was the main source of disappointment for those who unsuccessfully tried to use the Dantzig-Wolfe decomposition principle.

Here we consider decomposition from the point of view of nondifferentiable programming. It results in a different theoretical analysis of some known ideas in decomposition algorithms as well as opening some new possibilities for improving their computational performance.

Throughout the paper we stay within the framework of convex analysis in finite dimensions. Also it is assumed that "sup" and "inf" operations traditionally used in convex analysis attain finite values unless it is clear from the context that it must be otherwise. Hence they are replaced by "max" and "min" to be in accordance with mathematical programming notational conventions.

We introduce also some additional notations.

The inner product of two vectors x and y is denoted by xy.

The interior of the set X is denoted by $int(X)$ and its convex hull by $co(X)$.

The subgradient set of the convex function $f(x)$ at x is denoted by $\partial f(x)$:

$$\partial f(x) = \{ g : f(y) \geq f(x) + g(y - x) \}$$

Directional derivative of the function $f(x)$ in the direction d is denoted by $f'(x,d)$. Of course, for the convex case

$$f'(x,d) = \max_{g \in \partial f(x)} g\, d$$

For some set G expressions like $G\ (\ y - x\ )$ denote the inner product $g(y-x)$ where g is *some* vector from G and the particular choice of this vector does not matter.

2. Formulation of the problem

Consider a two-block mathematical programming problem with linking variables:

$$\begin{aligned} &\min\ (\ c_A(z_A) + c_B(z_B)\ ) \\ &g_A(z_A, x) \leq 0 \\ &g_B(z_B, x) \leq 0 \end{aligned} \tag{1}$$

where z_A and z_B can be viewed as internal variables of subproblems or submodels

$$f_A(x) = \min_{g_A(z_A,x) \le 0} c_A(z_A)$$

$$f_B(x) = \min_{g_B(z_B,x) \le 0} c_B(z_B)$$

with the corresponding optimal values $f_A(x), f_B(x)$ being functions of linking variable x. Problem (1) can then be considered as the problem of finding

$$\min_{x} (f_A(x) + f_B(x)) = v^* \tag{2}$$

Under convexity conditions for c_A, c_B with respect to z_A, z_B and joint convexity of g_A, g_B with respect to the pairs (z_A, x) and (z_B, x) respectively, $f_A(x)$, $f_B(x)$ are convex functions and using duality, problem (2) can be converted into the dual problem

$$\min_{x} (f_A^*(-p) + f_B^*(p)) = -v^* \tag{3}$$

where f_A^* is the conjugate of function $f_A(x)$

$$f_A^*(p) = \max_{x} \{ p\, x - f_A(x) \}$$

and f_B^* is the conjugate of function $f_B(x)$

In fact

$$v^* = \min_{x} \{ f_A(x) + f_B(x) \} =$$

$$\min_{x_A, x_B} \sup_{p} \{ f_A(x_A) + f_B(x_B) + p(x_A - x_B) \} =$$

$$\max_{p} \{ \min_{x_A} \{ f_A(x_A) + p\, x_A \} + \min_{x_B} \{ f_B(x_B) - p\, x_B \} \} =$$

$$\max_{p} \{ -f_A^*(-p) - f_B^*(p) \} = -\min_{p} \{ f_A^*(-p) + f_B^*(p) \}$$

Dual variables p are customarily interpreted as prices for the linking variables x. Computation of the values $f_A^*(-p), f_B^*(p)$ can be interpreted as a local optimization of subproblems A and B for given prices p associated with the linking variables

$$- f_A^*(-p) = \min \{ c_A(z_A) + p\, x \} \qquad (4)$$
$$g_A(z_A, x) \leq 0$$

$$- f_B^*(-p) = \min \{ c_B(z_B) + p\, x \} \qquad (5)$$
$$g_B(z_B, x) \leq 0$$

Problems (2), (3) can be solved by a number of methods updating either primal variable x or prices p, using the values of the functions $f_A(x), f_B(x)$ or $f_A^*(-p), f_B^*(p)$ and their subgradients.

The use of (2) or (3) may depend on the structure of the problems, requirements for particular type of solutions etc... Formulation (3) has some advantages which are discussed in(Nurminski79a) One of them is that $f_A^*(-p)$, $f_B^*(p)$ are convex functions with subgradients $-x_A^*$, x_B^* equal to the x-components of the solutions of (4)-(5). In other words subgradients of the functions $f_A^*(-p), f_B^*(p)$ are proposals of the (local) subproblems, in terms of the Dantzig-Wolfe decomposition method.

From the point of view of nonlinear programming the Dantzig-Wolfe decomposition method can be interpreted as a cutting plane algorithm (Kelley60a) applied to the optimization of the nondifferentiable function $f(p) = f_A^*(p) + f_B^*(-p)$.

Conceptually the cutting plane method can be represented in following structural form:

BEGIN *CUTTING PLANE ALGORITHM*

Let $k=0$ and at initial point x^0 the value of the function f is $f(x^0)$ and its subgradient is $\partial f(x^0)$. Define the initial approximation $f^{-1}(x)$ as

$$f^{-1}(x) \equiv -\infty$$

While *(NOT SOLUTION)*

```
    BEGIN INNER LOOP
    Using $f(x^k), \partial f(x^k)$ update the approximation:
        $f^k(x) = \max \{ f^{k-1}(x) , f(x^k) + \partial f(x^k)(x - x^k) \}$
    Solve the auxiliary problem
        $\min_x f^k(x) = f^k( x^{k+1})$        (6)
    Set $k = k+1$ and compute $f(x^k), \partial f(x^k)$
    END INNER LOOP
END CUTTING PLANE ALGORITHM
```

The auxiliary problem (6) can of course be stated in a linear programming form:

$$\min v$$
$$v \geq f(p^k) + \partial f(p^k)(p - p^k), \; k = 1, \ldots, K$$

which is an attractive feature of the method.

Problem (6) corresponds to the master problem of the Dantzig-Wolfe decomposition method and problems (4)-(5) are the subproblems of that schemas reacting to prices provided by the master problem (6).

Some authors (Topkis70a) considered variants of the schemas with exclusion of some points from the set P which correspond to nonactive constraints in (6).

The cutting plane algorithm generally does not have a good reputation for computational efficiency. For instance the lower bound for the number of iterations in solving problem (3) is of the order of the number of linking variables which is too high for many applications. Every iteration involves the solution of subproblems (4)-(5) and the updating of the dual variables p which are then sent again to subproblems (4)-(5). In some situations only a small number of such cycles can be performed.

3. Conceptual framework of the algorithm

The nature of the decomposition approach is to replace the original problem (1) with problems (2) or (3) which are defined in terms of the separate solutions of subproblems, (4) and (5), for instance. Eventually it may serve as a model of distributed problem solving.

Using results of the section **2** we consider the problem

$$\min_{x} \{ f_A(x) + f_B(x) \} \tag{7}$$

having in mind that f_A , f_B may represent optimal values obtained in subproblems A and B either in dual or in primal form.

The implicit assumption behind the decomposition approach is that taken separately the problems

$$\min_{x} f_A(x) \tag{A}$$

$$\min_{x} f_B(x) \tag{B}$$

are easy to solve. The same is often true also if any of these problems (say B) is replaced by the problem

$$\min_{x} \{ h_A(x) + f_B(x) \}$$

were $h_A(x)$ might be some "simple" approximation of function $f_A(x)$. In the convex case of particular interest are piece-wise linear approximation with a "small" number of pieces.

Exploiting this idea we can present in the spirit of structured programming, a conceptual framework of the algorithm based on the piece-wise linear approximation of one of the functions in (2).

BEGIN *DECOMPOSITION ALGORITHM*

Let $k=0$ and at initial point x^0 the value of the function f_A is $f_A(x^0)$ and its subgradient is $\partial f_A(x^0)$. Define the initial approximation $f_A^{-1}(x)$ as

```
                    $f_A^{-1}(x) \equiv -\infty$
While ( NOT SOLUTION )
    BEGIN INNER LOOP
    Using $f_A(x^k), \partial f_A(x^k)$ update the approximation:
        $f_A^k(x) = \max\{ f_A^{k-1}(x) \, , \, f_A(x^k) + \partial f_A(x^k)(x-x^k) \}$
    Solve the auxiliary problem
        $\min_x \{ f_A^k(x) + f_B(x) \} = f_A^k( x^{k+1}) + f_B( x^{k+1})$        (P)
    Set $k=k+1$ and compute $f_A(x^k), \partial f_A(x^k)$
    END INNER LOOP
END DECOMPOSITION ALGORITHM
```

To simplify some technical details we assume that $f_A(x)$ and $f_B(x)$ are convex finite functions. We do not need smoothness assumptions and this makes the following theoretical analysis applicable to the decomposition approach.

To study the convergence of this algorithm let us introduce a few notions:

Definition. Let f_A^∞ be a function with epigraph

$$epi\,(f_A^\infty) = \bigcap_k epi\,(f_A^k)$$

From the definition it immediately follows that f_A^∞ is convex and

$$f_A^k \leq f_A^\infty \leq f_A$$

Since we assumed that f_A, f_B are finite functions also f_A^∞ is a finite convex function. Being finite and convex f_A^∞ is continuous.

Convergence of this algorithm is based on a few facts:

Lemma 1

$$f_A^k(x^k) = f_A(x^k)$$

Proof. From the definition of f_A^k

$$f_A^k(x^k) = \max\{ f_A^{k-1}(x^k) \, , \, f_A(x^k) \} \geq f_A(x^k)$$

on the other hand

$$f_A^k(x^k) = \max\{\, f_A^{k-1}(x^k)\,,\, f_A(x^k)\,\} \leq \max\{\, f_A(x^k)\,,\, f_A(x^k)\,\} = f_A(x^k)$$

Lemma 2 If X^∞ is a set of accumulation points of the sequence $\{\, x^k \,\}$ generated by algorithm then for $x^*\in X^\infty$

$$f_A^\infty(x^*) = f_A(x^*)$$

Proof. It follows from continuity of f_A^∞ and monotonicity of $\{\, f_A^k \,\}$ that f_A^k converges to f_A^∞ locally uniformly. Then for $x^*\in X^\infty$ and $x^k \to x^*$

$$f_A^\infty(x^*) = \lim_{k\to\infty} f_A^\infty(x^k) = \lim_{k\to\infty} f_A^k(x^k) = \lim_{k\to\infty} f_A(x^k) = f_A(x^*)$$

Theorem 1. If f_A, f_B are finite convex functions and the sequence $\{\, x^k \,\}$ is bounded then

(i) $\lim\limits_{k\to\infty} \min\limits_{x} \{\, f_A^k(x) + f_B(x) \,\} = \min\limits_{x} \{\, f_A(x) + f_B(x) \,\}$

(ii) any limit point of $\{x^k\}$ belongs to the solution set of (7).

Proof. Let $x^*\in X^\infty$ and $x^k \to x^*$. Then

$$f_A(x^*) + f_B(x^*) = f_A^\infty(x^*) + f_B(x^*) = \lim_{k\to\infty} \{\, f_A^k(x^{k+1}) + f_B(x^{k+1}) \,\} =$$

$$\lim_{k\to\infty} \min_{x} \{\, f_A^k(x) + f_B(x) \,\} \leq \lim_{k\to\infty} \{\, f_A^k(x) + f_B(x) \,\} =$$

$$f_A^\infty(x) + f_B(x) \leq f_A(x) + f_B(x)$$

This completes the proof.

Notice that the proof did not make use of the particular type of approximation employed in the algorithm. The same result is true for any kind of approximation for which Lemmas 1 and 2 are valid.

In turn Lemmas 1 and 2 use only some general properties of the approximation in question and it is quite conceivable to imagine many different ways to construct approximations satisfying Lemmas 1 and 2.

4. Piece-wise linear case

The algorithm presented in the previous section is only a basic scheme which can be developed in many directions and Theorem 1 justifies only the theoretical validity of this approach without going into the details of its computational effectiveness. To study its practical significance one has to look at some specific cases singled out by additional assumptions about functions f_A, f_B.

An important case is the one when the functions f_A, f_B are piece-wise linear functions. It corresponds to the linearity of the underlying problems (4),(5) and in that case the assertion of the Theorem 1 can be strengthened.

The peculiar feature of piece-wise linear functions f_A, f_B is that they can be represented, in some neighborhood $U(x^*)$ of an arbitrary point x^*, in the following way:

$$f_A(x) = f_A(x^*) + \max_{g \in \partial f_A(x^*)} g(x - x^*) \tag{8}$$

$$f_B(x) = f_B(x^*) + \max_{g \in \partial f_B(x^*)} g(x - x^*) \tag{9}$$

which can be easily demonstrated.

Let $f(x)$ be a convex piece-wise linear function represented in the following way:

$$f(x) = \max_{i \in I} \{ a^i x + b_i \}$$

where $I = \{ 1,2,...,M \}$. Denote for any x

$$I(x) = \{ i: a^i x + b_i = f(x), i \in I \}$$

Then for any fixed point x^* due to finiteness of the set I and the upper semicontinuity of $I(x)$ as a set-valued mapping there is a neighborhood $U(x^*)$ such that

$$I(x) \subset I(x^*)$$

for all $x \in U(x^*)$ and consequently

$$\partial f(x) \subset \partial f(x^*) \tag{10}$$

Further on for all $x \in U(x^*)$

$$f(x^*) + \partial f(x^*)(x - x^*) \leq f(x) \leq f(x^*) + \partial f(x)(x - x^*)$$

or

$$\partial f(x^*)(x - x^*) \leq \partial f(x)(x - x^*)$$

which when combined with the inclusion above yields

$$f(x) = f(x^*) + \partial f(x^*)(x - x^*)$$

It is interesting to notice that also

$$f(x) = f(x^*) + \partial f(x)(x - x^*)$$

for $x \in U(x^*)$. This type of correspondence but in an asymptotic sense was used by R. Mifflin in the definition of semi-smooth (semi-convex) functions(Mifflin76a)

Theorem 2. If f_A, f_B are finite piece-wise linear functions and problem (7) has a unique solution x^* then the algorithm of Section 2 terminates in a finite number of steps.

Proof. First note that for any direction d

$$f'(x^*, d) \geq \delta \, ||d||$$

where $\delta > 0$.

Due to Theorem 1 $\{ x^k \}$ converges to x^*, so without loss of generality one can consider the case

$$\{ x^k \} \subset U(x^*)$$

where representations (8) and (9) are valid. If so, then for any x^k

$$\partial f_A^k(x^k) \subset \partial f_A(x^*)$$

and for any $x \in U(x^*)$

$$f_A^k(x) = f_A(x^*) + \max_{g \in G_k \subset \partial f_A(x^*)} g(x - x^*)$$

for some subset G_k of $\partial f_A(x^*)$. Notice also that

$$G_k = \partial f_A^k(x^*)$$

It is easy to show that the sets G_k form an increasing sequence

$$G_{k+1} = \partial f_A^{k+1}(x^*) = co\, \{ \partial f_A^k(x^*),\, g^k \} = co\, \{ G_k,\, g^k \} \supset G_k$$

where $g^k \in \partial f_A(x^k)$. Then

$$\min_x \{ f_A^k(x) + f_B(x) \} = \min_x \{ f_A(x^*) + \max_{g \in G_k} g(x-x^*) + f_B(x^*) + \max_{g \in \partial f_B(x^*)} g(x-x^*) \} =$$

$$f_A(x^*) + f_B(x^*) + \min_x \max_{g \in G_k + \partial f_B(x^*)} g(x-x^*) = f_A(x^*) + f_B(x^*) + \max_{g \in G_k + \partial f_B(x^*)} g(x^{k+1} - x^*)$$

Set

$$D_k = G_k + \partial f_B(x^*)$$

Since the sets G_k are monotone the sets D_k also form a monotone sequence of convex sets. Since $x^{k+1} \in U(x^*)$, then

$$\min_x \max_{g \in D_k} g(x-x^*) = \max_{g \in D_k} g(x^{k+1}-x^*) = 0 \tag{10}$$

If $x^{k+1} = x^*$ then algorithm terminates and theorem 2 holds.

If however the sequence $\{ x^k \}$ is infinite with $x^k \neq x^*$ for all k , then (10) implies that

$$0 \in D_k$$

but

$$0 \notin int(D_k)$$

and $x^{k+1} - x^* \neq 0$ is a support vector of the set D_k at $\{ 0 \}$ (See Figure 1).

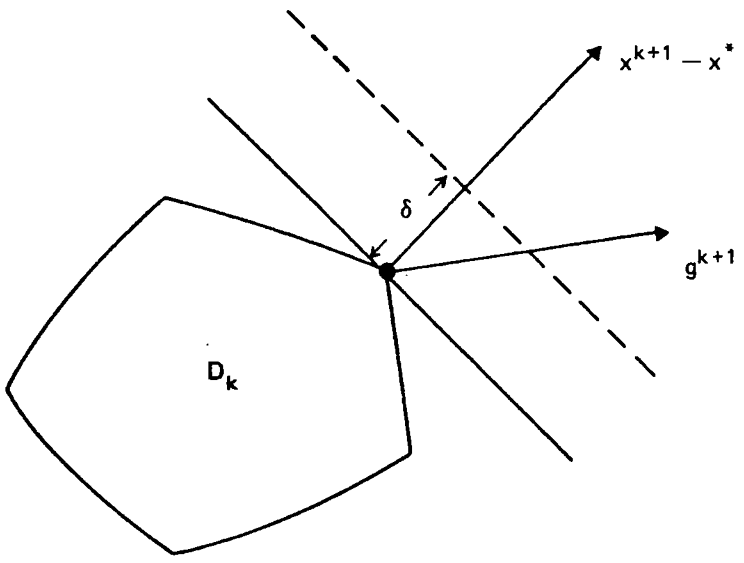

Figure 1.

Note that

$$\delta\,||x^{k+1} - x^*|| \le f_A'(x^*, x^{k+1} - x^*) + f_B'(x^*, x^{k+1} - x^*) \le$$

$$-f_A'(x^{k+1}, x^* - x^{k+1}) - f_B'(x^{k+1}, x^* - x^{k+1}) \le$$

$$-(\, f_A'(x^{k+1}, x^* - x^{k+1}) + f_B'(x^{k+1}, x^* - x^{k+1})\,) \le$$

$$-\max_{g \in \partial f_A(x^{k+1}) + \partial f_B(x^{k+1})} g(x^* - x^{k+1}) \le$$

$$-\,g^{k+1}(x^* - x^{k+1}) = g^{k+1}(x^{k+1} - x^*)$$

Deleting intermediate expressions it can be rewritten as

$$g^{k+1}(x^{k+1} - x^*) \ge \delta\,||x^{k+1} - x^*||$$

which is also illustrated on Figure 1.

Let

$$e^{k+1} \equiv \frac{x^{k+1} - x^*}{||x^{k+1} - x^*||}$$

and let e^∞ be an accumulation point of $\{ e^k \}$. By construction $e^{k+1} D_k \leq 0$ and it follows from monotonicity of the sequence $\{ D_k \}$ that

$$e^\infty D_k \leq 0$$

for any k and hence for any k

$$\max_{g \in D_k} e^\infty g \leq 0$$

Without loss of generality we may assume that $e^k \to e^\infty$ and then for $g \in D_k$

$$0 \geq \max_{g \in D_k} e^\infty g = \max_{g \in D_k} (e^\infty - e^k)g + e^k g \geq - \max_{g \in D_k} ||g||\,||e^\infty - e^k|| + \max_{g \in D_k} e^k g \geq$$

$$-||e^\infty - e^k||C + \max_{g \in D_k} e^k g \geq -||e^\infty - e^k||C + \delta \geq \frac{\delta}{2} > 0$$

for k large enough. This contradiction proves the theorem.

In the case of linearity of the underlying optimization problems an auxiliary optimization problem (P) in the algorithm description can also be stated as a linear programming problem just slightly more complex than the subproblems themselves.

In dual formulation, if

$$f_A^*(-p) = -\min \{ c_A(z_A) + p\, x \}$$
$$g_A(z_A, x) \leq 0$$

and the solution of this problem for some p is the pair (z_A, x_A) then an approximation of f_A^* after performing K iterations may be defined as

$$f_A^*(-p)^K = \max_{k=1,\ldots,K} \{ f_A^*(-p^k) + x_A^k(p-p^k) \}$$

and subproblem (P) is

$$\min_p \{ f_A^*(-p)^K + f_B^*(p) \}$$

which can be further transformed in the following way:

$$\min_p \{ f_A^*(-p)^K + f_B^*(p) \} = \min_{v \geq f_A^*(-p^k) + x_A^k p^k - x^k p} \{ v + f_B^*(p) \} =$$

$$\min_{v,p} \max_{\lambda_k \geq 0} \{ v + f_B^*(p) + \sum \lambda_k (f_A^*(-p^k) + x_A^k p^k - x^k p - v) \} =$$

$$\max_{\lambda_k \geq 0} \{ \inf_v v(1 - \sum \lambda_k) + \min_p \{ f_B^*(p) - p \sum \lambda_k x_A^k \} - \sum \lambda_k (f_A^*(-p^k) - x_A^k p^k) \} =$$

$$\max_{\lambda_k \geq 0} \{ \inf_v v(1 - \sum \lambda_k) + \min_p \{ f_B^*(p) - p \sum \lambda_k x_A^k \} - \sum \lambda_k v_A^k \} =$$

$$\max_{\lambda_k \geq 0, \sum \lambda_k = 1} \{ - \max_p \{ p \sum \lambda_k x_A^k - f_B^*(p) \} - \sum \lambda_k v_A^k \} =$$

$$\max_{\lambda_k \geq 0, \sum \lambda_k = 1} \{ - f_B(\sum \lambda_k x_A^k) - \sum \lambda_k v_A^k \} =$$

$$- \min_{\lambda_k \geq 0, \sum \lambda_k = 1} \{ f_B(\sum \lambda_k x_A^k) + \sum \lambda_k v_A^k \}$$

where summation is assumed to take place over the range $k = 1,\ldots,K$ of corresponding indices.

In linear programming formulation the latter problem may be stated as the following problem in variables z_B, $\lambda_k, k = 1,\ldots,K$:

$$- \min \{ c_B z_B + \sum \lambda_k v_A^k \} \tag{11}$$

$$A_B z_B + B_B \sum \lambda_k x_A^k \leq b_B \tag{12}$$

$$\lambda_k \geq 0, \sum \lambda_k = 1 \tag{13}$$

Problem (11)-(13) can be interpreted as a direct exchange of proposals between subproblems B and A.

On the other hand it can also be interpreted in a more traditional way as forming the master problem of the Dantzig-Wolfe decomposition method from subproblem B and convex hull of some extreme points of the constraint set of problem (4), corresponding to the solutions $x_A^k, k=1,\ldots,K$ generated so far.

Problem (11)-(13) is then a restricted master problem which can be used to generate through dual multipliers related to (12) new reduced cost coefficients in (11) and on a new cycle - new solution of the A-subproblem to enter the restricted master.

This interpretation says little however about the computational effectiveness of this idea and the existing literature provides a differing views on how subproblems and master problem should be formed to improve computational performance.

One general argument in favor of (P) might be the consideration that the objective function in (P) is closer to the objective function of the ultimate problem (7) and therefore likely to provide better convergence. Unfortunately worst-case counterexamples are not too difficult to construct even if they look rather artificial.

More detailed analysis of computational performance of the algorithm of section 2 depends on the properties of the functions $f_A(x)$, $f_B(x)$ in the vicinity of the optimal point x^* and will be the subject of future study. Here we demonstrate the computational performance of the algorithm on a few test problems.

5. Examples

Applied projects of International Institute for Applied Systems Analysis present a wealth of problems from which many examples of the formulation suitable for testing this approach can be drawn. Two of them, with a reasonable degree of complexity, were selected.

The algorithm was implemented using MINOS (Murtug77a) as a mean to solve the auxiliary linear problems. Unfortunately, MINOS does not have utilities (subroutines) to modify internal representation of the data when parameters of the problem are changed or when additional rows/columns are added. For this reason the formulation and updating of the auxiliary subproblems have been done via modification of the input files in the external format.

It is surely the most inefficient way to implement the algorithm but at this stage the main concern was about the number of major iterations and not the computational effectiveness as a whole.

One additional advantage was the small amount of programming efforts needed to supply codes for generating updated input files. Some UNIX (Ritchie78a) utilities came in very handy.

The chosen mode of implementation resulted also in some loss of accuracy which showed up when comparing the decomposed solution with the solution of integrated model.

5.1. Agricultural model

First tests have been done with the decomposition of a part of Polish agricultural model developed for the Food and Agriculture program at IIASA by A. Jozwiak, T. Wollodko, L. Wisniewski, J. Rajtar, J. Gomulka. The detailed structure of the model is described elsewhere (Jozwiaknga) and here we present only a very brief description of the model from the point of view of application of the results of the previous sections.

The whole production model of the agricultural sectors is composed of 4 submodels including the following submodels of agriculture:

- State sector
- Private part-time sector
- Private traditional sector
- Private developing sector

Each of the technological matrices describing submodels includes about 250 variables, 170 rows and 2200 non-zero coefficients.

The construction of the models permits the study of the reactions of the respective sub-sectors and the private sector as well as the whole agriculture to economic incentives (prices) and non-economic means of control (limits of allocation of production inputs, goals imposed etc...).

The sectorial models may be linked into the model of private agriculture and eventually into the model of the whole agricultural sector. This linkage is performed by replacing 11 groups of local constraints concerning production inputs into global ones and adding 2 additional constraints related to labor force balance and animals turnover (Makowskinga).

For test purposes two submodels of the private sector were linked using the algorithm of section 2. These submodels are private traditional (MIT) and private part time (MID) submodels.

Statistics for the MID submodel:

	total	normal	free	fixed	bounded
rows	174	46	9	115	4
columns	249	231	0	1	17

no. of matrix elements	2331	density	5.380

Statistics for the MIT submodel:

	total	normal	free	fixed	bounded
rows	171	57	7	105	2
columns	240	219	0	1	20

no. of matrix elements	2096	density	5.107

These two problems are interlinked by a group of 11 constraints which represent either distribution of common recourses between submodels or balance of certain flows between these submodels.

The linking constraints were transformed into linking variables by adding specially introduced linking variables each one corresponding to the value of a linking row.

In one experiment the subproblem MIT was used as the pricing part with proposals comming from subproblem MID. In another experiment subproblem MID was used as the pricing device and subproblem MIT was used as a generator of proposals.

The stopping criteria for these two experiments was a generation of the same price or proposal in any of the auxiliary subproblems.

The results of both experiments are shown in Table 1 which also contains the results of using conventional Dantzig-Wolfe decomposition method (DWD). In the table the arrows show which submodel is sending prices to which.

For each of the subproblems the total number of iterations performed in it during the experiment is shown in the appropriate columns. The number of local iterations per major iteration was maximal during the first cycle (major iteration) when initial infeasibilities had to be resolved and then decreased rapidly.

For Dantzig-Wolfe decomposition method the pricing problem corresponds to the conventional master problem, and subproblems are placed under the heading 'Proposing problem". The number of iterations shown there corresponds to the total number of iterations performed in both subproblems.

The value of objective function in pricing problem approximates the optimal solution from above. For Dantzig-Wolfe decomposition method the lower estemate of the solution is also available as a correspondent value in the proposing subproblem which substitutes here two local subproblems of the test problem.

Table 1

Agricultural model

general information		pricing problem		proposing problem	
test	major it.	iter	objective	iter	objective
MID -> MIT	7	371	-1.5871d+05	553	-
MIT -> MID	39	1424	-1.5882d+05	501	-
DWD	49	180	-1.5724d+05	4548	-1.6238d+05

For Dantzig-Wolfe decomposition method the solution process was stopped after 49 cycles between subproblems and master problem (major iterations) and the final results are shown in Table 1.

The rate of convergence is shown on Figure 2.

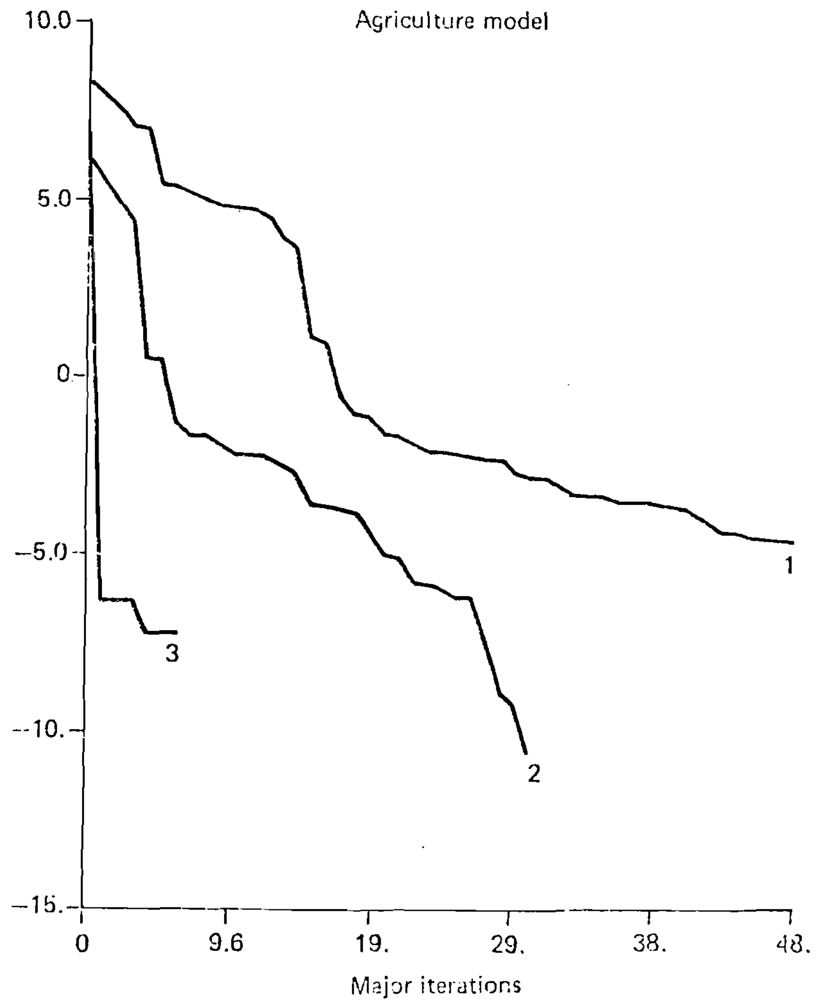

Figure 2.

The rate of convergence of decomposition methods. The relative accuracy of the solution reached is shown in a natural logarithmic scale as a function of the number of major iterations. Curve 1 corresponds to Dantzig-Wolfe decomposition method, 2 - to the computations in the case when submodel MIT was used as a pricing problem, 3 - when subproblem MID was used as a pricing problem.

These results lend itself to some preliminary conclusions that computational performance of the algorithm strongly depends on which of the functions in (7) is approximated, or in other words, which subsystem is used as a pricing subproblem and which is used for proposal generation.

The algorithm seems to be rather sensitive to the accuracy of the intermediate results. It was not able to reach relative accuracy more then 10^{-4}. This of course could also be caused by the rather crude round-off of the intermediate results.

It is also clear from the results that the conventional Dantzig-Wolfe algorithm is the slowest for this problem. That could be caused by a particular way of implementation, and more advanced implementation like (Loute81a) would perform certainly better.

5.2. Energy model

Another example is a simplified version of the energy model currently under further development in Energy project at IIASA. It is a new version of MESSAGE - Model for Energy Supply Systems Alternatives and their General Environmental impact (Schrattenhol81a) More detailed information is contained in (Messnernga)

This model is a dynamic linear programming model which is intended to describe a transition process from one pattern of energy production to another, depending on the availability of certain resources and environmental effects.

The model from which the test problem was derived describes a process of energy generation starting from some raw material and meeting the final demand specified elsewhere.

It can be considered as consisting of 2 submodels.

The first submodel (CENTR) describes the production of different kinds of final energy from sources such as fossil and nuclear fuels, hydro, solar, geothermal energy and some others. The final energy produced is electricity, district heat, hydrogen, coal, liquid and gaseous fuels.

The second one (END) relates to further transformations of final energy into useful energy. The final energy flows then go through different stages of transportation, distribution and on-site conversion to meet the demand of end-users.

The linking variables of this model are flows of final energy between subproblems and the model generator allows for the different variants to be specified. These variants differ in the number of time periods, number of technologies represented, etc. For this test, the number of links between subsystems was chosen to be 42 which correspond to 7 time periods. Different simplifications have been made in the structure of the subproblems to cut down the size of the blocks. Full-scale experiment with this model is planned for the near future.

Here are some statistics for subproblems CENTR and END.

CENTR:

	total	normal	free	fixed	bounded
rows	246	182	22	42	0
columns	202	192	0	4	6

no. of matrix elements 963 density 1.938

END:

	total	normal	free	fixed	bounded
rows	157	102	13	42	0
columns	139	126	0	3	10

no. of matrix elements 520 density 2.383

As a test-bed this problem is characterized by larger connectivity than the previous one, if it is characterized by the ratio between the numbers of internal and linking variables.

Experiments with this model were also conducted in two ways: first - the subproblem CENTR was used as the pricing part of the algorithm and subproblem END generated proposals, second - the subproblem END defined prices and the subproblem CENTR generated proposals.

A summary of the results is shown in the Table 2.

Table 2

Energy model

general information		pricing problem		proposing problem	
test	major it.	iter	objective	iter	objective
CENTER -> END	9	220	9.2562d+ 03	175	8.3817d+ 03
END -> CENTER	50	915	1.5033d+ 04	1998	-2.7079d+ 04

which show the clear superiority of one way over the other. Calculations were simply stopped after 50 major iterations in the experiment with END as the pricing part.

The rate of convergency for these two runs is shown on Figure 3 in the same way as in Figure 2.

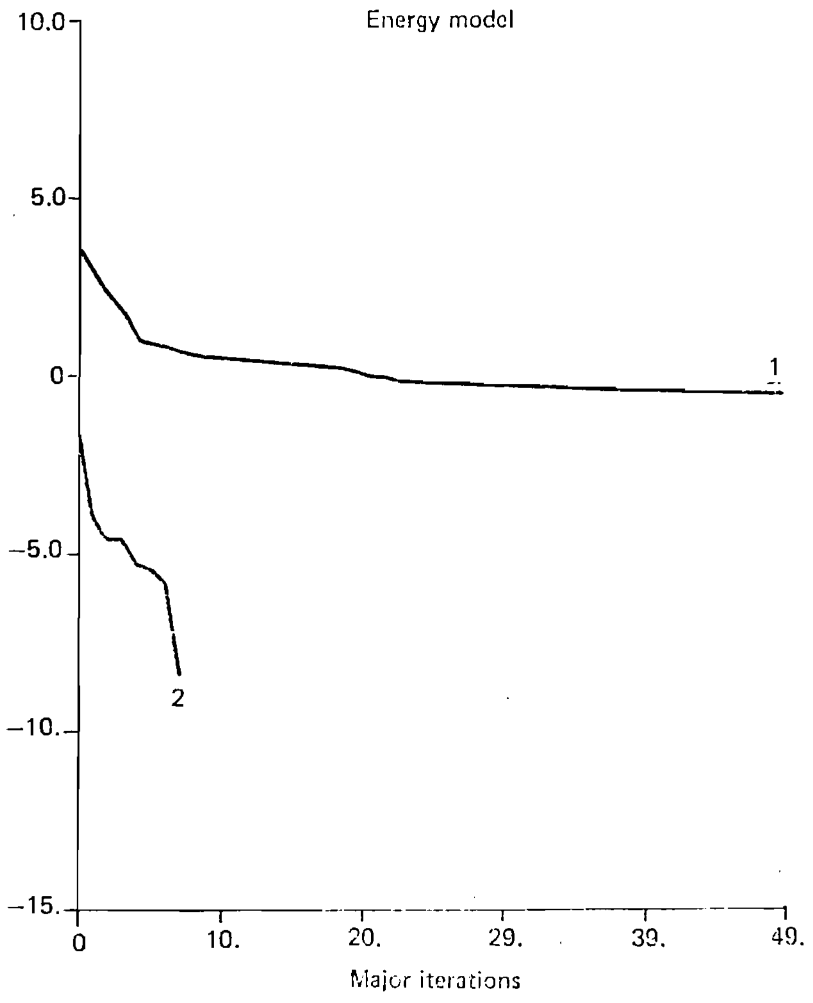

Figure 3.

The rate of convergence of decomposition method. The relative accuracy of the reached solution is shown in a natural logarithmic scale as a function of the number of major iterations. Curve 1 corresponds to the computations in the case when submodel END was used as a pricing problem, 2 - when subproblem CENTR was used as a pricing problem.

Comparative calculations with this problem treated as a whole produced the results agreeing up to 4 digits accuracy with the decomposed solution. Dantzig-Wolfe decomposition algorithm stopped after 28 major iterations still far away from solution and is not included into the table. Seeming disconvergence of this algorithm could be caused by round-off errors that occur when the results of the solution of one auxiliary problem are used to form the input file for another problem.

6. Conclusions

The view of decomposition principle as an aggregation of large-scale structured problems into nonlinear nondifferentiable framework allows concise general description of the approach.

Specific decomposition algorithms can be considered as particular ways to construct computationally tractable approximations of the resulting problem.

Numerical experiments with a decomposition algorithm based on this idea show that for the same level of implementation, considered algorithm seems to be essentially faster than the conventional Dantzig-Wolfe decomposition method.

Acknowledgments

The author wishes to thank Roger Wets for very helpful comments and discussions.

References

Dantzig61a. G.B. Dantzig and P. Wolfe, "The decomposition algorithm for linear programming," *Econometrica* **29** pp. 767-778 (1961).

Dantzig81a. G.B. Dantzig, "Time-Staged Methods in Linear Programming. Comments and Early History," pp. 4-16 in *Large-Scale Linear Programming. Proceedings of IIASA workshop.*, ed. G.B. Dantzig, M.A.H. Dempster and M.J. Kallio, (1981).

Jozwiaknga. A. Jozwiak, T. Wollodko, L. Wisniewski, J. Rajtar, and J. Gomulka, "Production Model of Polish Agriculture," Technical report, International Institute for Applied Systems Analysis, Laxenburg, Austria (forthcoming).

Kelley60a. J.E. Kelley, "The Cutting Plane Method for Solving Convex Programs," *Journal of the Society for Industrial and Applied Mathematics* **8**(4) pp. 703-712 (1960).

Loute81a. E. Loute and J.K. Ho, "An Advance Implementation of the Dantzig-Wolfe Decomposition Algorithm for Linear Programming," pp. 425-460 in *Large-Scale Linear programming, Proceedings of a IIASA Workshop,2-6 June 1980, Volume 1,* ed. G.B. Dantzig, M.A.H. Dempster and M.J. Kallio, (1981).

Makowskinga. M. Makowski and J. Sosnowski, "MERGE," Technical report, International Institute for Applied Systems Analysis, Laxenburg, Austria (forthcoming).

Messnernga. S. Messner, "Users Guide for Message II," Technical report, International Institute for Applied Systems Analysis, Laxenburg, Austria (forthcoming).

Mifflin76a. R. Mifflin, "Semismooth and Semiconvex Functions in Constrained Optimization," RR-76-21, IIASA, Laxenburg (1976). [Also in SIAM Journal on Control and Optimization, Vol.15(6), pp.959-972. (1977)]

Murtug77a. B.A. Murtug and M.A. Saunders, "MINOS. A Large-Scale Nonlinear Programming System (for Problems with Linear Constraints)," Technical report SOL 77-9, Stanford University (1977).

Nurminski79a E. Nurminski, "Some theoretical considerations on linkage problems," WP-79-117, IIASA (December 1979).

Ritchie78a D.M. Ritchie and K. Thompson, "The UNIX Time-Sharing System," *The Bell System Technical Journal* **57**(6) pp. 1905-1931 (1978).

Schrattenhol81a L. Schrattenholzer, "The Energy Supply Model MESSAGE," RR-81-31, International Institute for Applied Systems Analysis, Laxenburg, Austria (1981).

Topkis70a D.M. Topkis, "Cutting plane method without nested constraint set," *Operation Research* **18**(3) pp. 404-413 (1970).