7e Conférence Internationale de MOdélisation et SIMulation - MOSIM'08 - du 31 mars au 2 avril 2008 – Paris- France

« Modélisation, Optimisation et Simulation des Systèmes : Communication, Coopération et Coordination.».

ACCELERATION OF BENDERS DECOMPOSITION ALGORITHM USING MULTI-GENRATION OF CUTS

A. G.K. SAHARIDIS, B. M. IERAPETRITOU

Department of Chemical and Biochemical Engineering
Rutgers -The State University of New Jersey
98 Brett Road
Piscataway, NJ 08854-8058
A.sacharid@rci.rutgers.edu, B.marianth@sol.rutgers.edu

C. M. MINOUX

Laboratoire d'Informatique de Paris 6
Université Pierre @ Marie Curie
Place Jussieu
75005 Paris France
C. michel.minoux@lip6.fr

ABSTRACT: *In this article two new acceleration strategies for Benders algorithm have been proposed. In order to illustrate the efficiency of the algorithm two case studies are used ([G.K. Saharidis, 2006] and [M. Ierapetritou and C.A. Floudas, 1998]). These strategies are called Multi-Generation of High Density Cut Strategy (**MGHDCS)** and Maximization of the number of Active Constraints (**MAC**). The objective is to produce with systematic way a high density cut in each iteration of Benders algorithm. The MGHDCS and MAC strategies decrease the number of iterations leading to a better resolution time. A summary of the advantages and the limitations of the algorithm and suggestions for improving even more the resolution time are provided.*

KEYWORD : ***Benders decomposition, Mixed Integer Programming, Mutli-Generation of Cuts, Active Constraints***

1. INTRODUCTION

Decomposition techniques are based on the observation that we can decompose the decision variables of the initial problem (*P*) obtaining several smaller sub-problems: the slave problem (*SP*) which is obtained by fixing a number of decision variables of the initial problem (*P*) to a feasible value and the restricted master problem (*RMP*) which converges to the optimal solution after the addition of some valid cuts. These valid cuts are obtained by solving the *SP* in each iteration of the algorithm and sent to the restricted master problem (*RMP*) which converges to the optimal solution. If the optimality condition is not satisfied by the bounded solution obtained by *SP*, the *RMP* sends the updated information to the *SP* which produce a new cut for *RMP* and the algorithm continues until the optimality condition is satisfied [M. Minoux, 1986].

Over the years there have been a number of new techniques and strategies in order to speed-up the classical Benders decomposition approach which is a well known and used decomposition strategy for the MILP and for stochastic programming. Most of the proposed approaches have focused on either reducing the number of integer relaxed master problems being solved or on accelerating the solution of the restricted master problem using a better initialization scheme.

[D. McDaniel and M. Devine, 1977] presented the first such strategy for the acceleration of Benders algorithm. They proposed to relax the binary variables of the *RMP* program and generate a cut using this relaxed program. The authors proposed that useful information can be produced by adding the cuts derived from the continuous relaxation since any extreme ray or point of the dual of *SP* can generate a valid cut for *RMP*. The presented results showed that in some cases this new strategy decreases the resolution time of Benders algorithm reducing the iterations of the algorithm.

[T.L. Magnanti and R.T. Wong, 1981] proposed a new procedure to accelerate the Benders algorithm, using Pareto-optimal cuts. The characteristic of Pareto-optimal cut is that no other cut dominates it. The authors propose to add in each iteration of the Benders algorithm two types of cuts: the classical optimality or feasibility cut and also the Pareto-optimal cut. The obtained results showed a significant reduction of the convergence time of the algorithm. [W. Rei et al., 2006] presented a local branching strategy in order to accelerate the classical Benders decomposition algorithm. They showed that by applying local branching throughout the solution process, one can simultaneously improve both the lower and upper bounds of the algorithm. They also showed how Benders feasibility cuts can be strengthened or replaced with local branching constraints. To assess the performance of different algorithmic ideas, computational experiments were performed on a series of network design problems illustrating the benefits of the developed strategy.

Another group of strategies proposed to accelerate Benders are the hybrid strategies where more than one strategy is used at the same time. [V. Roy, 1983] proposed the cross decomposition strategy where the main idea was to use simultaneously primal decomposition which is Benders decomposition and its

dual which is Lagrangian relaxation. The author showed that the solution obtained for one of this problem can be used as a feasible solution for the other giving some useful information to the Benders algorithm. The author concluded by saying that the cross decomposition can improve the speed-up of the algorithm replacing the resolution of *RMP* by the resolution of the Lagrangian problem. The same idea has been developed by [K. Holmberg, 1994] who presented a complete study concerning the use of this technique. The authors showed that the bound obtained by the Lagrangian relaxation of the *RMP* is not better than the bound obtained by the Lagrangian dual relaxation of the initial problem *P*. They demonstrated that this is always true, even when all cuts produced by *SP* are added to *RMP*. Also, the authors claimed that when using Lagrangian relaxation in the *RMP*, a lack of controllability on the integer solution obtained may prevent the approach from converging.

[G. Côté and M.A. Laughton, 1984] presented an algorithm generating only cuts associated with the solution obtained using just a feasible integer solution of *RMP* without solving it to optimality. This strategy has some problems concerning the convergence of the algorithm due to the proposed heuristic way in order to choose the iterations where the *RMP* will be solved to optimal do not guarantee the rate of the convergence of the algorithm. The same authors presented also a strategy using Lagrangian relaxation. They made an observation concerning the case where the solution space of the master problem has a special structure which can be exploited by different solution algorithms. In this special case the obtained cuts would prevent the use of specifically adapted methods for solving the *RMP*. The authors proposed the use of Lagrangian relaxation on these cuts and the generation of a special *RMP*. For specific values of Lagrangian multipliers, the problem can be solved efficiently with a method that exploits the structure of the feasible solution space of the master problem. However, the integer solution obtained may not be feasible for the *RMP*. [G. Côté and M.A. Laughton, 1984] proposed the use of sub-gradient optimization in order to modify the Lagrangian multipliers and resolve the problem. Unfortunately at the end may still not find an optimal solution to the *RMP* due to the duality gap. In order to solve this problem they proposed a branching strategy for the solution of *RMP* in order to bridge this gap.

In this paper we present two new strategies in order to produce high density cut accelerating classical Benders algorithm. The outline of the paper is as follows. In section 2 we briefly present the classical Benders algorithm and in section 3 the main inefficiencies of Benders Algorithm. In section 4, the first developed strategy named multi-generation of high density cut strategy (*MGHDCS*) is presented and compared with the classical Benders through some numerical examples. In the following section 5 the second strategy named maximization of active constraints (*MAC*), is presented and some numerical results illustrate the gain obtained by applying this strategy. Finally, in section 6, we present our conclusions and some perspectives for the suitability of the presented approaches.

2. BENBERS DECOMPOSITION

2.1. Introduction

Following the presentation in [M. Minoux, 1986], we can state a classical Mixed Integer linear Problem (MILP) as follows:

$$P\begin{cases} Min\, d\cdot x + f\cdot y \\ st.: \\ D\cdot x + F\cdot y = b \\ x \geq 0, y \in Y \subset R^m. \end{cases}$$

Given the structure of problem (*P*) the decision variables can be partitioned into two sets (x) and (y). For a fixed decision variable ($y = \bar{y}$), problem (*P*) takes the following form (Slave Problem):

$$SP(\bar{y})\begin{cases} Min\, d\cdot x + f\cdot \bar{y} \\ st.: \\ D\cdot x = b - F\cdot \bar{y} \\ x \geq 0. \end{cases}$$

To be able to apply this partition strategy, we can not choose the variables *y* in *Y* arbitrarily. It is at least necessary that the problem *SP* have non-empty solution set. To express this condition we shall use the lemma of *Farkas* and *Minkowski*. Associate with every constraints of *SP* a dual variable u_i (not sign-restricted) and denote by *u* the row vector of dual variables the lemma of *Farkas* and *Minkowski* states:

$SP(\bar{y})$ *has a solution* $x \geq 0$ *if and only if* $u^T \times (b - F \times y) \leq 0$ *for all u for which* $u^T \cdot D \leq 0$ *holds.*

Since the cone $U = \{u / u^T \cdot D \leq 0\}$ is polyhedral, it has a finite number of generators which we denote $u^1,\ldots,u^P$. The necessary and sufficient condition of *Farkas* and *Minkowski* lemma is then equivalent to the system of inequalities

$$(SI)\begin{cases} (u^1)^T \cdot (b - F\cdot y) \leq 0 \\ (u^2)^T \cdot (b - F\cdot y) \leq 0 \\ \ldots\ldots\ldots \\ (u^P)^T \cdot (b - F\cdot y) \leq 0 \end{cases}$$

In general this system contains an enormous number of inequalities, since this is the number of generators of the polyhedral cone *U*.

Let R be the set of vectors $y \in Y$ which satisfy the above system. The complete minimization problem can therefore be written as:

$$\underset{y \in R}{Min}\{ f^T \cdot y + \underset{x}{Min}\{ d^T \cdot x / D \cdot x = b - F \cdot y; x \geq 0\}\}$$

For a given $\bar{y} \in R$ the dual of $SP(\bar{y})$ is as follows:

$$SP^*(\bar{y}) \begin{cases} Max\, v \cdot (b - F \cdot \bar{y}) \\ st.: \\ v^T \cdot D \leq d \\ v\ of\ any\ sign. \end{cases}$$

The constraints polytope of $SP^*(y)$, $V = \{v / v^T \cdot D \leq 0\}$ does not depend on *y*, and the (u^1,… ,u^p) defined above are its extreme rays. Then by agreeing to assign the value $-\infty$ to the maximum of $SP^*(y)$ if it has no solution, and using the duality theorem, we can write the initial problem as

$$\underset{y \in R}{Min}\{ f^T \cdot y + Max\{ v^T \cdot (b - F \cdot y) / v^T \cdot D \leq c\}\}$$

The maximum of $SP^*(y)$ is obtained at a vertex of the polytope V. Assuming that V is not empty, we denote by v^1,… ,v^Q the vertices of the polytope *V*, then the initial problem can be written

$$\underset{y \in R}{Min}\{ f^T \cdot y + \underset{j=1,..,Q}{Max}\{ (v^j)^T \cdot (b - F \cdot y)\}\}$$

and this problem turns to be equivalent to the linear program

$$(MP) \begin{cases} Min\, z'' \\ st.: \\ f^T \cdot y + (v^j)^T \cdot (b - F \cdot y) - z'' \leq 0, \forall j \in [1,..,Q] \\ y \in R \end{cases}$$

By expressing the condition $y \in R$ by the system of inequalities, we deduce from it the equivalent of initial problem to the solution of the linear problem called Master Problem. The Master problem is:

$$(MP) \begin{cases} Min\, z'' \\ st.: \\ f^T \cdot y + (v^j)^T \cdot (b - F \cdot y) - z'' \leq 0, \forall j \in [1,Q] \\ (u^i)^T \cdot (b - F \cdot y) \leq 0, \forall i \in [1,P] \\ y \in R \end{cases}$$

where u^j are all the extreme points of the constraints polytope of $SP^*(y), V = \{v / v^T \cdot D \leq d\} (\forall j \in [1,Q])$ and u^i are the extreme rays $(\forall i \in [1,P])$.

In Benders decomposition framework two different problems are solved. The Restricted Master Problem (*RMP*) where only part of the extreme points and rays of $(SP^*(\bar{y}))$, are defined producing the constraints. The *RMP* is formed from the subsets $I \subset [1,P]$ and $J \subset [1,Q]$ of the constraints of the *MP*

$$(RMP) \begin{cases} Min\, z'' \\ st.: \\ f^T \cdot y + (v^j)^T \cdot (b - F \cdot y) - z'' \leq 0, (\forall j \in J) \\ (u^i)^T \cdot (b - F \cdot y) \leq 0, (\forall i \in I) \\ y \in R \end{cases}$$

A necessary and sufficient condition for $(\bar{y}, \bar{z})$ to be the optimal solution of problem *MP*, and hence *P*, is that $(\bar{y}, \bar{z})$ satisfy all the constraints of *MP* which are not explicitly stated in *RMP*. In order to test this condition we shall see that it is sufficient to solve simply, for $(y = \bar{y})$, the problem $(SP^*(y))$ where three cases can arise:

- *Case 1*: The optimal value of $SP^*(y)$ is unbounded. The simplex method, applied to $SP^*(y)$ then produces an extreme ray $\bar{u}$ of U such that :
  $\bar{u}^T \cdot (b - F \cdot \bar{y}) > 0$ and $\bar{u}^T \cdot D \leq 0$.
  Thus the constraint $\bar{u}^T \cdot (b - F \cdot \bar{y}) \leq 0$ does not hold for the current solution $(y = \bar{y})$ of *RMP*. This means that $(\bar{y}, \bar{z})$ is not a solution of *MP*. Then the constraint $\bar{u}^T \cdot (b - F \cdot y) \leq 0$ must be added to *RMP* to form a new augmented restricted master program.
- *Case 2*: The optimum $SP^*(y)$ of has a finite value and we have:
  $$f^T \cdot \bar{y} + \bar{v}^T \cdot (b - F \cdot \bar{y}) - \bar{z}'' \leq 0$$
  We can then write:
  $f^T \cdot \bar{y} + (v^j)^T \cdot (b - F \cdot \bar{y}) - z'' \leq 0, \forall j = 1,..., Q$.
  Moreover, we have $(u^i)^T \cdot (b - F \cdot \bar{y}) \leq 0$, for all extreme rays u^i, of *U*, $(\forall i \in [1,P])$. It follows that $(\bar{y}, \bar{z})$ is an optimal solution of *MP*, so that $\bar{y}$ is an optimal solution of *P* and the algorithm terminates.
- *Case 3*: The optimum of $SP^*(y)$ is bounded and obtained at a vertex $\bar{v}$, of *V*, but contrary to *case 2*, we have:
  $$f^T \cdot \bar{y} + \bar{v}^T \cdot (b - F \cdot \bar{y}) - \bar{z}'' > 0$$
  This shows that the constraint $f^T \cdot y + v^T \cdot (b - F \cdot y) - \bar{z}'' \leq 0$ is not satisfied by the current solution $(\bar{y}, \bar{z})$ of *RMP*. The constraint $f^T \cdot y + v^T \cdot (b - F \cdot y) - \bar{z}'' \leq 0$ must therefore be added to *RMP* to form a new augmented restricted master program.

As long as the optimality test is not satisfied (*Case 2*), we can add to the restricted master program the not satisfied constraints by the current solution $(\bar{y}, \bar{z})$. The new restricted master program is then solved for (y, z) and so on. If, at any stage, the *RMP* has no solution, then the problem *P* has no solution and the algorithm stops. Finally, if *V* is empty and if *P* has a solution, then we can conclude that *P* is unbounded, and the algorithm terminates. In figure 1, the complete Benders algorithm is presented.

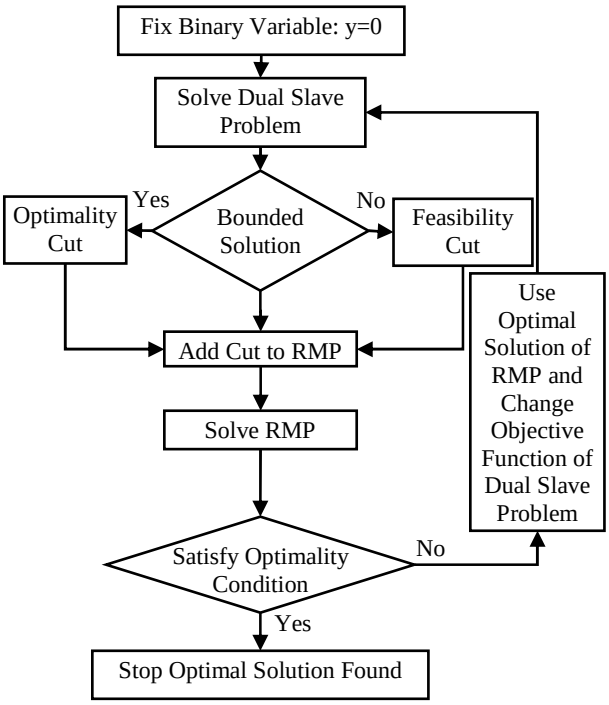

Figure 1. Benders Algorithm

To summarize the Benders algorithm we can say that Benders cut is a linear inequality based on Lagrange multipliers obtained from a solution of the dual sub-problem. The next set of values for the primary variables is obtained by solving the master problem, which contains all the Benders cuts so far generated. The process continues until the master problem and sub-problem converge in value.

3. BENDERS INNEFICIENCIES

One of the most important problems in Benders algorithm is the convergence of the algorithm. At the worst case Benders algorithm converges after the production of all the constraints using all the extreme points of the dual of slave problem. The finite convergence of the method can easily be established resulting from the fact that (*RMP*) has a finite number of constraints, and that the successively generated constraints are necessarily all different. For problems of very large size, the convergence could be very slow because of the inefficiency of cuts produced in each iteration. After studying the form of the cuts produced by Benders algorithm we can say that the cuts (optimality and feasibility cuts) are in some cases low density cuts.

Definition: A cut can be considered low density cut if it includes only a small portion of the decision variables of the restricted master problem.

If for example in each cut generation by the classical Benders algorithm only the 10% of decision variables of the master problem have been considered then the cut can be considered a low density cut.

At each iteration of classical Benders algorithm, given the current optimal solution of the *RMP*, we solve the corresponding (*SP*) and we produce the following cut:

$$(u^T \cdot D) \cdot y \le u^T \cdot b \text{ (feasibility cut) and}$$
$$(v^T \cdot D) \cdot y \le z'' + v^T \cdot b \text{ (optimality cut)}$$

where *u* or *v* are the current optimal solution of the dual (*SP*). The coefficient of y_j in the produced cut is equal to $u^T \cdot D^j$ or $v^T \cdot D^j$. In the application presented in [G.K. Saharidis, 2006] on average 90% of these coefficients are equal to zero that implies a low density cut. Two maim reasons are responsible for this phenomenon: the first one is that the matrix *D* is sparse, and the second one is that a big number of variable's (y_j) coefficients u^T or v^T are equal to zero. The first one happens due to the structure of the problem and the form of the functional constraints of the system under study and thus cannot be addressed for a specific problem formulation. Concerning the second one it happens because only a small number of dual variables are not equal to zero which means that only a small number of *SP* constraints are active. The basic idea studied here is the generation of a cut where the maximum number of variables is considered using the same optimal solution of *RMP*. In the following sections, we present two strategies to produce high density cut accelerating the convergence of the algorithm.

4. MULTIGENERATION OF CUTS

4.1. Introduction

To our knowledge nobody has developed a strategy in order to construct high density cut from the slave problem (*SP*). We define as high density cut to be the cut that is dense in terms of decision variables participation. In this section we present a new methodology to produce a group of low density cuts (multi-generation) at each iteration for the acceleration of Benders algorithm. The sum of these cuts results to a high density cut. This procedure, named multi-generation of high density cut strategy (*MGHDCS*), accelerates the Benders algorithm decreasing the number of iterations needed. Thus, the solution space of *RMP* will be restricted faster (multi-

generation of cuts), the algorithm will converge faster to the optimal solution.

4.2. Generation of cuts

Most of the resolution time in Benders algorithm is spent for the solution of *RMP* because this problem is usually an integer problem. In contrary the resolution of *SP* (slave problem) does not consume a lot of time since it usually corresponds to a continuous problem. In order to reduce the computational complexity of *RMP* and take full advantage of *SP* we have developed the *MGHDCS* strategy. The basic idea studied here is the generation of cuts where the maximum number of variables are considered. In the general case, some bounds have been added in each of these coefficients (of decision variable y_j) in order to generate the following cut. These bounds have the following form:

$$\alpha_j \le (u^T \cdot D^j) \le \beta_j \text{ or } \alpha_j \le (v^T \cdot D^j) \le \beta_j$$

where α_j, β_j correspond to the lower and upper bound, respectively that we want to impose in each coefficient of the variable decision (y_j) of *RMP*. These bounds give the following augmented slave problem where the number of inequalities added are equal to the number of decision variables of *RMP*. This program (for the case of feasibility cut) has the following form and it will be solved for different values of α_j and β_j:

$$(ADP)\begin{cases} Max\ Z = u^T \cdot (b - B \cdot \bar{y}) \\ st. \\ u^T \cdot D \ge 0 \\ -u^T \cdot e = 1 \\ -u^T \cdot D^j \ge -\beta_j \forall j \\ u^T \cdot D^j \ge \alpha_j \forall j \\ u \le 0 \end{cases}$$

Introducing two new set of variables ϑ_j and μ_j, for the additional inequalities, the corresponding auxiliary primal problem (*ADP*) takes the following form:

$$(APP)\begin{cases} MinZ' = -\sum_j \beta_j \cdot \theta_j + \sum_j \alpha_j \cdot \mu_j \\ st. \\ D \cdot x - \sum D^j \cdot \vartheta_j + \sum D^j \cdot \mu_j \le (b - B \cdot \bar{y}) \\ x, \theta, \mu \ge 0. \end{cases}$$

In the general form of *MGHDCS* algorithm, we solve problem *APP* for different values of α_j and β_j. We use the *APP* form because in order to produce the high density cut only the objective function must be changed using also the last optimal solution as a base for the generation of the next cut. In order to generate a cut where a decision variable of *RMP* (y_{j_0}), has been considered in the cut we solve the *APP* using the optimal solution $(\bar{y})$ of the *RMP* in current iteration and we fix the following values for the problem parameters:

$$\alpha_{j_0} = \beta_{j_0} = +1, \quad \alpha_{j_0} = -\eta, \quad \beta_{j_0} = +\eta, \quad \eta = M, \quad \forall j \ne j_0$$

Fixing these parameters we are making sure that at least the variable decision (y_{j_0}) will appear in the next cut since its coefficient has a nonzero value. This procedure produces valid cuts because in order to generate a cut with specific variables, we solve *ADP* in a sub-space of its original feasible region.

In *MGHDCS* not only the *SP* is solved but we have also the successive resolution of *APP* using the same optimal solution of the current *RMP*. In each resolution of *APP*, the parameters $+n/-\eta$ change and a new cut is produced. The complete procedure of *MGHDCS* is as shown in the schema in figure 2:

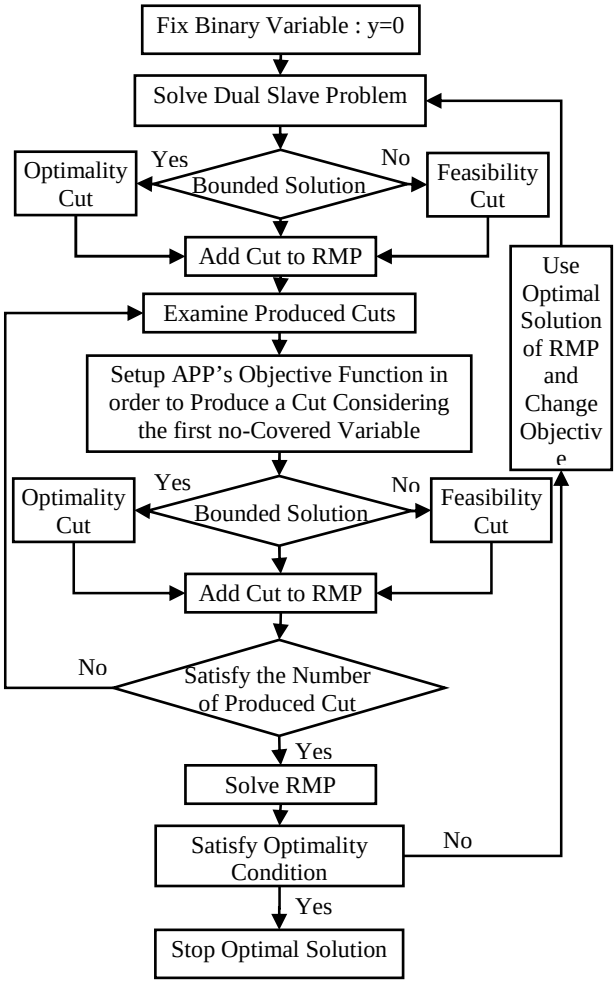

Figure 2. *MGHDCS* Strategy

Before resolving the *APP* the cut produced is added to *RMP* and for another not yet converted variable $y_{j_0'}$, the

parameters $\alpha_{j_0'}$ and $\beta_{j_0'}$ are fixed. At the same time the lower and the upper bound of $y_{j_0'}$ coefficient are re-initialized. A second cut is produced and the iterations continue. The procedure stops when the number of cuts determined a priori is generated or when all the decision variables of the *RMP* are covered. Before the production of multi cuts, the classical Benders algorithm produces a classical cut from the *SP*. After the generation of this cut a test is performed in order to determined which variable has been covered. The variables that are not covered are stored in a matrix in order to be used when we solve *APP*. The multi-generation of high density cut strategy can be considered as the procedure that generates cuts, towards all the directions by fixing the values of some *ADP* variables. In this way a maximum number of cuts is added to *RMP* defining a restrictive solution space, thus decreasing the number of Benders algorithm's iterations.

4.3. Case studies

All results presented in this paper correspond to the case study presented in [G.K. Saharidis, 2006] and [M. Ierapetritou and C.A. Floudas, 1998]. In [G.K. Saharidis, 2006] a general model for scheduling of crude oil in a refinery is presented. This model provides the optimal plan for the loading and unloading of crude oil to the tanks by ships and/or pipelines and to crude distillation unites (CDU) and/or another system of pipelines. The system under consideration consists of ports, storage tanks, CDUs and pipelines. It is very important not only for security reasons that the crude oil is loaded or unloaded continuously but also for another important reasons. The launching of a loading or unloading tanks requires a setup cost which is very important for the refinery. The system is further complicated due to different mixing strategies and different distillation options. The developed model for this application is a mixed integer linear program where the continuous variables have been introduced in order to describe flows in the pipelines and the inventory in the storage tanks. The binary variables describe the setup decision of loading or unloading and the decision to continue loading or unloading for the tanks. The data of the problem involve the arrival times of the crude oil in the refinery by ship or by pipeline, the quantities required by the CDUs, the type of preparation of mix and the distillation option. Finally, we know the dimension of the refinery: number of ports, storage tanks and CDUs and also storage capacity and pumping capacity. Concerning the problem constraints the model includes material balances to account from transfer between the port tanks and the CDUs, operational constraints expressing the system's specificities as for example the requirement that a tank can not be loaded and unloaded the same time period. The number of the decision variables for the examples considered vary between 640 and 1920 and the number of constraints is between 512 and 1860.

In order to further assess the efficiency of the developed strategies, we also considered a second case study [M. Ierapetritou and C.A. Floudas, 1998]. This also concerns a scheduling problem for multi-product, multi-purpose batch plants. The problem considers the optimization of the plant capacity to improve a given economic objective. Usually profit, cost, or production makespam is considered. The decision variables involves the assignment of tasks to units, the amount of material in each tank at each time instant and the starting and finishing time of each task. Given is the production recipe (what is needed to produce certain products and the correct task sequence that has to be followed), the unit maximum capacities and minimum requirements for operation, the processing time for the specific tasks. The constraints involve material balance throughout the plan, the assignment constraints to ensure that each task is assigned to one unit, capacity constraints to ensure that the limits of the units are not violated and a set of timing constraints to ensure that the tasks are executed according to the correct sequence and timing limitations and that all tasks are scheduled before the determined time horizon. This model is general to describe scheduling problems in a variety of industrial sectors including pharmaceutical and chemicals. The problem size can be very substantial. The examples considered here involve numbers of variables in the range of 876 to 7301 and number of constraints between 7636 and 9176.

4.4. Numerical results applying *MGHDCS*

In order to accelerate Benders decomposition algorithm the most important task is to reduce the number of *RMP* that should be solved. Decreasing the algorithm's iterations, usually results in reduction of resolution time but this is not always the case since the resolution time is not only a function of the number of iterations but also depends on the resolution of the *APP* problems. Incorporating additional cuts in each iteration result in reducing the number of iterations that leads to a significant reduction due to smaller number of *RMP* that have to be solved. However more time is required to build the additional cuts. In table 1, we present some examples where we show that the use of *MGHDCS* helps Benders algorithm to converge significantly faster. In some examples a reduction up to 98% is observed concerning the resolution time and the number of iterations. The computer used was a Pentium (R) 4, CPU 2.40 GHz, RAM 1 GB and the optimization software was CPLEX 9.0.

	Ex.	1	2	3	4	5	6
Number of	Var.	1920	768	640	1280	1700	1280
	Con.	1860	744	512	1210	1420	1210
Classical Benders	Time	15h	25m	24h	28m	12h	27m
	Iter.	2297	814	3500	817	3317	228
MGHDCS Strategy	Time	13m	20m	22m	2m	5m	5m
	Iter.	36	42	44	65	106	27
Difference	Time	98%	20%	98%	92%	98%	76%
	Iter.	98%	94%	98%	92%	96%	88%

Table 1. Benders vs *MGHDCS*

Although these results are encouraging, *MGHDCS* strategy does not give every time good results and in some cases fails to accelerate the convergence of the classical Benders algorithm. In table 2 some examples for the case study presented in [M. Ierapetritou and C.A. Floudas, 1998] are given. The comparison between classical Bender's and *MGHDCS* shows the deficiency of *MGHDCS* in some cases. In these five examples we choose to produce 16 additional cuts per iteration in order to show the advantages and disadvantages of *MGHDCS*. In example 7 classical Benders is more efficient than *MGHDCS* that requires 20 CPU sec compared to 5 CPU sec needed by classical Benders. However although resolution time can increase, an important reduction concerning the number of iterations is always observed. This is also true in the examples presented in table 3 where the effect of number of cuts is examined.

	Ex.	7	8	9	10	11
Number of	Var.	7301	876	876	1022	1022
	Con.	7636	9176	9176	9176	9176
Classical Benders	Time	5s	13m	5m	50m	2m
	Iter.	41	683	378	1093	172
MGHDCS Strategy	Time	20s	13m	2m	39m	2m
	Iter.	39	573	105	712	96
Difference	Time	-300%	0%	60%	22%	20%
	Iter.	5%	16%	72%	25%	44%

Table 2. Benders vs *MGHDCS* (16 cuts)

In all the results obtained, the application of the *MGHDCS* to Benders algorithm results in an important reduction in terms of the total number of iterations. For the example presented in table 3 the classical Benders algorithm finds the optimal solution in 38 seconds after 180 iterations. Applying the *MGHDCS* and generating 4 cuts in each iteration, a reduction of 44% of the resolution time and 71% of the number of iterations is observed (cf. table 3). In the same example a minimum reduction of 27% (130 iterations) is obtained using 22 cuts in each iteration. With 22 cuts the convergence of the algorithm is slower due to the time spent in order to generate all the cuts. We should point out that even if the resolution time is worse than in classical Benders algorithm, the number of iteration continues to be smaller. In table 3, we can see that in some cases decreasing the number of additional cuts result in a decrease in a number of iterations. When the number of cuts is equal to 32, *MGHDCS* converge in 52 iterations. We can decrease even more the number of iterations, for example with 16 cuts the algorithm converges in only 48 iterations which is the minimum number of iterations observed in this example.

Number of cuts	Time	Iterations
1	25s	88
4	21s	64
7	33s	62
10	1m 21s	62
13	2m	71
16	1m 24s	48
19	1m 24s	54
22	7m 16s	130
25	2m 25s	57
28	2m 30s	55
32	2m 40s	52

Table 3. Effect of number of cuts generated in each iteration

The numerical examples show that the choice of the number of cuts produced in each iteration is very important since as it is observed increasing the number of cuts can have a negative effect in the performance of the algorithm. Thus we must select the number of generated cuts in order to find the minimum number required to give the maximum reduction in the number of iterations. Generally the *MGHDCS* is a good strategy in order to accelerate the convergence of Benders algorithm. The only deficiency that appears is the time spent to solve the *APP* problems. In order to accelerate more the *MGHDCS*, we developed also an additional strategy where we produce a cut where the maximum of constraints are active. In the following section this strategy named Maximization of Active Constraints (*MAC*) is presented and some numerical results concerning the performance are given.

5. MAXIMIZATION OF THE ACTIVE CONSTRAINTS (MAC)

5.1. Introduction

The primal form of a linear program in von Neumann symmetric form is as follows:

$$(\text{Primal})\begin{cases} Min\ z = c^T \cdot x \\ st. \\ A \cdot x \geq b, (A: m \times n) \\ x \geq 0 \end{cases}$$

and the dual problem has the following form :

$$(\text{Dual})\begin{cases} Min\ z = b^T \cdot y \\ st. \\ A^T \cdot y \leq c, (A: m \times n) \\ y \geq 0 \end{cases}$$

Introducing slack variables $x_s = (x_{n+1,} x_{n+2}, ..., x_{n+m})^T \geq 0$, $y_s = (y_{m+1,} y_{m+2}, ..., y_{m+n})^T \geq 0$ to the primal and dual problem, respectively, we can measure the extent of constraint satisfaction. By complimentary slackness theorem we know that for the optimal solution of the primal and dual systems, whenever the slack variable of the *kth* constraint of either system is positive, the *kth* variable of its dual is zero and if the *kth* variable is positive in either system, the *kth* relation of its dual is tight which means that the value of the slack variable is zero. We can describe this relation by the following equations:

$$x_k \cdot y_{m+k} = 0 \text{ for } (k=1,..,n) \text{ and}$$

$$y_k \cdot x_{n+k} = 0 \text{ for } (k=1,..,m)$$

When Benders algorithm produces a less dense cut a big number of dual variables of slave problem take a value equal to zero. That means that only a few number of the slave problem's constraints are active. In order to produce a high density cut we should be able to find the solution of *SP* where the maximum number of constraints becomes active. In general, the classical Benders algorithm does not provide cuts with the maximum number of active constraints, and this is the main reason that a low density cut can be obtained. For the example presented in [G.K. Saharidis, 2006] the resolution of *SP* gives a small number of active constraints. The maximization of the number of active constraints using the Extended Slave Problem presented in the following subsection leads to a significant reduction of the number of not active constraints which is about 30% compared with the solution taken by the *SP* that has not active constraints in 25% on average. This result shows that high density cuts can be produced maximizing the active constraints and using a different (from the Benders optimal) extreme point of the solution space of Slave problem. The basic idea of *MAC* strategy is to find the optimal solution of the slave problem between the points that have the maximum number of active constraints.

5.2. Generation of high density cut by MAC strategy

For the needs of MAC strategy's presentation, we introduce the following slave problem:

$$(\text{Slave Problem}) \begin{cases} MinZ = c^T \cdot x \\ st. \\ A_1 \cdot x \geq b_1 (A_1 : \ (m-k) \times n) \\ A_2 \cdot x \geq b_2 (A_2 : \ k \times n) \\ x \geq 0 \end{cases}$$

In order to find the extreme point of the solution space of the (*Slave Problem)*, where the maximum number of constraints are active the *Slave Problem* is extended as follows:

$$\text{Extended Slave Problem} \begin{cases} MinZ = Binary_1 + Binary_2 \\ A_1 \cdot x - Slack_1 = b_1, A_2 \cdot x - Slack_2 = b_2 \\ Slack_1 \leq M \cdot Binary_1, Slack_2 \leq M \cdot Binary_2 \\ x \geq 0, Slack_1, slack_2 \geq 0 \\ Binary_1, Binary_2 \in \{0,1\}. \end{cases}$$

Solving the Extended Slave Problem (*ESP*), we can find the maximum number of active constraints. Obtaining these results by *ESP* we build the New Slave Problem (*NSP*), which corresponds to the *SP* with the active constraints identified by problem *ESP*. For example if $Binary_1=0$, which means that the first group of constraints is active then the *NSP* is :

$$(\text{New Slave Problem}) \begin{cases} MinZ = c^T \cdot x \\ st. \\ A_1 \cdot x = b_1 \\ A_2 \cdot x > b_2 \\ x \geq 0. \end{cases}$$

We solve the *NSP* and we produce a cut in the same way as in the classical Benders algorithm. The cut produced by this problem has the maximum number of positive coefficients and is added to the *MRP*. It should be noticed that in order to optimize the results of the *NSP*, the value of Big-M in *ESP* should be as small as possible in order to restrict the bounds of each constraint. In order to use in each constraints the same value of Big-M we can replace it with the maximum value of the constraints which can be calculated a priori.

We should pay attention to the fact that the above procedure of maximizing the number of active constraints involves the solution of a *MILP* formulation (*ESP*) in order to produce the linear program *NSP*. Generally the resolution time of *ESP* is not so long even if it is a *MILP*. But if we need to develop a more simple version of the *ESP*, we can replace the objective function of the *ESP* by the following form: $Min\ z = Slack_1 + Slack_2$ and eliminate the use of binary variables. This procedure leads to a linear Extended Slave Problem. In order to give equal weight to all slack variables, we multiply each slack variable by a coefficient equal to the mean value of the right hand side of the constraint. Since the value of the right hand side change in each iteration, it is better to chose the mean value of the right hand side. This procedure is not an exact procedure in order to find the maximum of active constraints but an approximation which in general gives a solution where the number of active constraints is bigger than in the case of initial slave problem.

5.3. Numerical results applying MAC

Applying the *MAC* strategy, the produced cut involves the maximum number of decisions variable of *MRP* and thus improves the convergence of the Benders algorithm. We should notice that in both of these new strategies, the classical Benders cut should be produced and added to the *RMP*. In the following table we present some numerical results where we compare *MGHDCS* strategy with and without the *MAC* strategy.

	MGHDCS		MGHDCS with MAC		Difference	
Ex.	Iter.	Time	Iter.	Time	Iter.	Time
1	36	13	31	12	13.8%	7.7%
2	42	20	36	18	14.3%	10%
3	44	22	39	21	11.4%	4.5%
4	65	2	62	2	4.6%	0%
5	106	5	96	4.5	9.4%	10%
6	27	5	26	5	3.7%	0%

Table 4. *MGHDCS* vs. *MGHDCS* with *MAC*

The results show an additional important reduction in the number of iterations of the algorithm. The cut produced by *NSP* result in some cases in a reduction of resolution times of *RMP* and give an additional reduction on the number of iterations which fluctuates between 4% and 14%. This reduction is very important because the *MGHDCS* had already decreased in a critical way the number of iterations. For example for example 1 the classical Benders algorithm converged after 2297 iterations. Using *MGHDCS* resulted in 98% reduction and the algorithm converged after 36 iterations. Adding the cut produced by *NSP* the algorithm converged in 31 iterations which means an additional 13.9% reduction. Of course we should also take into account the extra time required to produce the additional cuts solving problems *ESP* and *NSP* (the average time spent for the resolution of ESP and NSP per iteration is only some seconds but as the number of iteration increase as the sum of this time becomes important).

We should notice that the *MAC* strategy can be applied alone without the *MGHDCS*, accelerating in a significant way the convergence of the algorithm. However, in order to have the best results we should apply these strategies together although additional time is required to solve the series of *ADP*, *ESP*, and *NSP* problems.

6. CONCLUSIONS AND FUTURE DIRECTIONS

Two acceleration strategies for Benders decomposition algorithm have been presented in this paper namely the multi-generation of high density cut strategy (*MGHDCS*) and maximization of number of active constraints (*MCA*). The presented examples illustrate the applicability and efficiency of these strategies. All the presented examples illustrate that both strategies result in a significant decrease of the number of iterations and resolution time of the Benders algorithm. Adopting multi-generation of high density cut strategy the number of iterations always decrease resolving in general a better resolution time since the *RMP* has to be solved less number of times. In some cases the resolution time increased and this deficiency appears because of the extra time needed in order to solve the *ADP* problem. Adopting also the *MAC* strategy an additional reduction is achieved but the algorithm becomes slower because of the extra time needed for the resolution of *ESP* and *NSP* problems. In order to eliminate this problem some ideas form parallel optimization literature can be used. The basic idea is take advantage of the fact that the resolution of *RMP* takes more time than the sum of time spent to solve the three auxiliary problems (*ADP*, *ESP* and *NSP*). All these cuts can be constructed, using the last optimal solution of *RMP*, simultaneously with the solution of *RMP*. In addition parallel optimization can be used in order to exchange sub-optimal information between the dual *SP*, *ADP*, *NSP* and *RMP* problems. A new strategy can thus be developed where the four problems communicate with each other and exchange information about the feasible solution that they find in each iteration of simplex algorithm for the linear problems and branch and bound algorithm for the mixed integer linear *RMP* problem. Results of this algorithm will be reported in future publication.

ACKNOWLEDGEMENDS

M. Ierapetritou would like to rate fully acknowledge financial support from the National Science Foundation under the NSF CTS 0625515 grant and also the USEPA-funded Environmental Bioinformatics and Computational Toxicology Center under the GAD R 832721-010 grant.

REFERENCES

Codato,G., and Fischetti, M., 2004. Combinatorial Benders cuts for mixed-integer linear programming. *Lectures Notes in Computer Science: Springer-Verlag*, p. 178-195.

Côté, G., and Laughton, M.A., 1984. Large-scale mixed integer programming: Benders-type heuristics. *European Journal of Operational Research*, 16, p. 327-333.

Holmberg, K., 1994. On using approximations of the Benders master problem. *European Journal of Operational Research*, 77, p. 111-125.

Hooker, J.N., 2000. Logic-Based Methods for Optimization: Combining Optimization and Constraint Satisfaction. *John Wiley and Sons*.

Ierapetritou M.G., and Floudas C.A. 1998. Effective Continuous-Time Formulation for Short-Term Scheduling: I. Multipurpose Batch Processes. Ind. & Eng. Chem. Res 37(11) p. 4341-4359.

Magnanti, T.L. and Wong, R.T., 1981. Accelerating Benders decomposition algorithmic enhancement and model selection criteria. *Operations Research*, 29, p. 464-484.

McDaniel, D. and Devine, M. 1977. A modified benders' partitioning algorithm for mixed integer programming. Management Science 24 p. 312-319.

Minoux, M. 1986. Mathematical Programming Theory and Algorithms. *Wiley-Interscience Series in Descrete Mathematics and Optimization*.

Saharidis, G.K., 2006. Pilotage de production à moyen et à court terme : contribution aux problématique d'optimisation globale vs locale et à l'ordonnancement dans les raffineries. Thèse de Doctorat-22, Ecole Centrale Paris, France.

Van, Roy, 1983. Cross decomposition for mixed integer programming. *Mathematical Programming*, 25, p. 46-63.

Walter, Rei, Michel, Gendreau, Jean-Francois, Cordeau and Patrick Soriano, 2006. Accelerating Benders decomposition by local branching. *Hybrid methods and branching rules in combinatorial optimization, Montréal, September 2006*, p. 18-22.