Speed-up Benders decomposition using maximum density cut (MDC) generation

Georgios K. D. Saharidis & Marianthi G. Ierapetritou

Annals of Operations Research

ISSN 0254-5330

Ann Oper Res
DOI 10.1007/s10479-012-1237-8

Springer

Ann Oper Res
DOI 10.1007/s10479-012-1237-8

Speed-up Benders decomposition using maximum density cut (MDC) generation

Georgios K.D. Saharidis · Marianthi G. Ierapetritou

Abstract The classical implementation of Benders decomposition in some cases results in low density Benders cuts. Covering Cut Bundle (CCB) generation addresses this issue with a novel way generating a bundle of cuts which could cover more decision variables of the Benders master problem than the classical Benders cut. Our motivation to improve further CCB generation led to a new cut generation strategy. This strategy is referred to as the Maximum Density Cut (MDC) generation strategy. MDC is based on the observation that in some cases CCB generation is computational expensive to cover all decision variables of the master problem than to cover part of them. Thus MDC strategy addresses this issue by generating the cut that involves the rest of the decision variables of the master problem which are not covered in the Benders cut and/or in the CCB. MDC strategy can be applied as a complimentary step to the CCB generation as well as a standalone strategy. In this work the approach is applied to two case studies: the scheduling of crude oil and the scheduling of multi-product, multi-purpose batch plants. In both cases, MDC strategy significant decreases the number of iterations of the Benders decomposition algorithm, leading to improved CPU solution times.

Keywords Benders decomposition · Mixed integer linear programming · Multi-generation of cuts · Covering cut bundle generation (CCB) · Active constraints

1 Introduction

Benders decomposition technique is based on the observation that we can decompose the decision variables of the initial problem (IP) obtaining several smaller and thus easier to

G.K.D. Saharidis (✉)
Department of Mechanical Engineering, University of Thessaly, Pedion Areos, 38334 Volos, Greece
e-mail: saharidis@gmail.com

M.G. Ierapetritou
Department of Chemical and Biochemical Engineering, Rutgers—The State University of New Jersey, 98 Brett Road, Piscataway, NJ 08854-8058, USA

Published online: 17 November 2012

solve sub-problems. These subproblems include the slave problem (SP) which is obtained by fixing a number of decision variables of IP to a feasible value and the restricted master problem (RMP) which converges to the optimal solution after the addition of some cuts. These cuts are obtained by solving the SP in each iteration of the algorithm. If the optimality condition is not satisfied, the RMP sends the updated information to the SP which produces a new cut and the algorithm continues until the optimality condition is satisfied (Benders 1962). Benders decomposition, as any decomposition, method was firstly introduced for the solution of problems with special structure, which after its application the IP is decomposed by blocks. Benders algorithm has been expanded beyond block decomposable problems, to address problems with different structure such as bi-level problems and non-convex mixed integer non-linear problems. Benders theory combines with global optimization in order to address the non-convexity nature of mixed integer non-linear problems (Zhu and Kuno 2003) and with Karush–Kuhn–Tucker optimality condition to solve the mixed integer bi-level linear problems (Saharidis and Ierapetritou 2009a).

Various techniques and strategies have been proposed to speed-up the classical Benders decomposition approach. McDaniel and Devine (1977) proposed the generation of cuts using the solution of the RMP relaxing the integrality constraint. Since any extreme ray or extreme point of the dual of SP can generate a valid cut for the RMP, one can hope to generate useful information for the integer case by adding the cuts derived from the continuous relaxation. The solution of the RMP with the integrality constraint is necessary for ensuring the convergence of the algorithm. The authors present some heuristic rules for determining when the integrality constraint is needed for the solution of RMP. The presented results for the modified Benders algorithm appear promising although in some cases the classical algorithm can be more efficient. Magnanti and Wong (1981) proposed a multi-cut generation procedure to accelerate the Benders algorithm, using what they refer to as Pareto-optimal cuts. A cut is defined as Pareto-optimal if no other cut dominates it. The authors propose to add in each iteration of the algorithm two types of cuts: the optimality or feasibility cut produced by the classical Benders procedure and also the Pareto-optimal cuts. The obtained results show a significant reduction of the convergence time of the algorithm.

Other authors proposed hybrid methods combining different strategies at the same time. Van Roy (1983) proposed a new strategy of decomposition, called cross decomposition. The main idea is to use simultaneously primal decomposition, which corresponds to Benders decomposition and its dual, which corresponds to Lagrangian relaxation. He showed that the solution of the Lagrangian problem can act as a possible solution to the Benders master problem and vice versa. Therefore, the solutions of these two problems provides some useful information to Benders algorithm. The author concluded by saying that the cross decomposition can improve the speed of the algorithm replacing the solution of RMP by the solution of the Lagrangian problem. This strategy is also proposed by Holmberg (1994) who presented a complete study regarding the use of this technique. The author showed that the lower 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 (IP). This is always true, even when all cuts produced by SP are added to RMP. Also, the author observes that when using Lagrangian relaxation in the RMP, a lack of controllability on the integer solution obtained may prevent the approach from converging.

Cote and Laughton (1984) presented another approach for the acceleration of Benders algorithm. In the proposed algorithm the RMP is not solved to optimality but only the first integer feasible solution is used to generate the optimality or feasibility cut by SP. The main disadvantage of this strategy is that by generating only the cuts associated with the solution obtained using just a feasible integer solution, the algorithm can fail to converge. The authors proposed a heuristic way in order to choose the iterations where the RMP has to be solved to optimality. The same authors presented 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 various solution algorithms. In this special case the cuts obtained by SP would prevent the use of specifically adapted methods for solving the RMP. The authors proposed the use of Lagrangian relaxation on these constraints (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. Cote and 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 this case a branching strategy is proposed and used in RMP to bridge this gap.

Rei et al. (2006) investigated how local branching can be used to accelerate the classical Benders decomposition algorithm. By applying local branching throughout the solution process, one can simultaneously improve both the lower and upper bounds. They also show how Benders feasibility cuts can be strengthened or replaced with local branching constraints. To assess the performance of various algorithmic ideas, computational experiments were performed on a series of network design problems illustrating the benefits of this approach. Another interesting paper is presented by Zakeri et al. (1998) where in each iteration of the algorithm the cut is not computed from an optimal extreme point of the dual slave problem. When the slave problem is very large they determine the cuts by applying a primal-dual interior-point method to the current slave problem and terminating the solution procedure when it reaches a feasible dual solution. The authors show that these sub-optimal cuts named “inexact” cuts are computational less expensive and can produce very good results as they showed using a stochastic hydroelectric scheduling application. A new multi-generation of cuts algorithm is recently presented (Saharidis and Ierapetritou 2010) to improve the efficiency of Benders decomposition approach for the cases that optimality cuts are difficult to be achieved within the iterations of the algorithm. This strategy is referred to as maximum feasible subsystem (MFS) cut generation strategy. In MFS, in each iteration of the Benders algorithm an additional cut is generated that has the property to restrict the value of the objective function of the Benders master problem. Saharidis et al. (2010) presented a new strategy referred to as covering cut bundle (CCB) generation which implements in a novel way the multiple constraints generation idea. As the numerical results presented have shown, CCB algorithm significantly decreases the number of iterations of the Benders method leading to improved solution times. CCB strategy is presented in Sect. 2.2 in details.

Beside the quality of the produced Benders cut, the other main reason leading to slow convergence of Benders decomposition algorithm is that the lower bounds (in the case of minimization) obtained by the RMP in some cases is relatively weak. A lot of researchers have developed a series of valid inequalities for the restriction of RMP which successfully speed up the convergence of the algorithm because the RMP infeasible cases are a priori eliminated and the first bound obtained by RMP is significantly improved. Cordeau et al. (2006) introduce a new formulation of the logistics network design problem encountered in deterministic, single-country and single-period contexts where Benders decomposition algorithm is applied and a series of valid inequalities are presented as the part of the solution methodology. Cordeau et al. (2000) also use Benders decomposition to solve the locomotive and car assignment problem where three sets of valid inequalities are developed based on the special characteristic of the problem and added to RMP in order to enforce the part of the constraints that appear in the subproblem and accelerate convergence to the optimal solution. Andreas and Smith (2009) present a series of valid inequalities for the subtour elimination in order to ensure the existence of tree, each time that the subproblem is resolved. Saharidis et al. (2009) and Saharidis and Ierapetritou (2009b) have developed a series of valid inequalities for the optimal scheduling of crude oil in a refinery network where Benders decomposition algorithm was applied in order to solve large instances of the problem. Finally, Saharidis et al. (2011), present a series of valid inequalities which are the first attempt to generalize, the previously developed, valid inequalities to be applicable to the general fixed-charged network design problem.

The work presented in the literature has mainly focused on either reducing the number of integer relaxed master problems being solved or on accelerating the solution of the restricted master problem. In this paper, we present a new strategy for the production of extra cuts in order to improve the efficiency of CCB generation or generally of the Benders algorithm. This strategy offers a novel way of implementing the multiple constraints generation idea, where, in each iteration of the Benders algorithm, other cuts with specific characteristics are produced in addition to the classical Benders cut and CCB.

The outline of the paper is as follows. In Sect. 2, we briefly review the classic Benders decomposition algorithm as well as the CCB generation. In Sect. 3, we propose a strategy referred to as maximum density cut (MDC) generation strategy. In Sect. 4, we give the numerical results obtained by applying the classic Benders algorithm using CCB generation and the proposed approach for a series of test problems that correspond to two case studies. Finally, in Sect. 5, we present conclusions and some perspectives for further research.

2 Benders decomposition

2.1 Classical Benders cuts

We briefly recall the idea of the Benders algorithm (Benders 1962) considering, without loss of generality the following linear problem:

$$\begin{aligned}
&\text{Initial Problem (IP)}\\
&\text{Min}\, c^T x + d^T y\\
&\text{st.}\\
&Ax + By \le b\\
&x \in \Re^n_+,\ \ y \in Z^q_+
\end{aligned}$$

where $c \in \Re^n$, $d \in \Re^q$, $b \in \Re^m$, A and B are $m \times n$ and $m \times q$, matrices respectively. The decision variables are partitioned into two sets x and y. For fixed y $(y = \overline{y})$, IP takes the following form:

$$\begin{aligned}
&\text{Primal Slave Problem (PSP)}\\
&f(x) = \text{Min}\ c^T x + d^T \overline{y}\\
&\text{st.}\\
&Ax \le b - B\overline{y}\\
&x \in \Re^n_+
\end{aligned}$$

In Benders decomposition we decompose the IP into PSP, which is a restriction of IP and provides an upper bound (UB) in the case of minimization, and the following relaxation of IP, which is called the restricted master problem (RMP) and provides a lower bound (LB):

Restricted Master Problem (RMP)

$$F(y,z) = \text{Min } z$$

st.

$$\left.\begin{array}{l} v^{iT}(b - By) \le 0 \\ u^{jT}(b - By) + d^T y - z \le 0 \end{array}\right\} \text{Benders Cuts}$$

$$z \ge 0,\ \ y \in Z_+^q$$

where v^i is the vector that corresponds to the extreme ray i of the dual of PSP and u^j is the vector that corresponds to the extreme point j. In each iteration of the Benders algorithm, the PSP is solved for a different value of y $(y = \overline{y})$ which is updated by the optimal solution of RMP obtained in the previous iteration. Notice that in the first iteration of the algorithm an arbitrary value is given to y. In practice it is not the PSP which is solved in each iteration but the dual of PSP which has the following form:

Dual Slave Problem (DSP)

$$f'(u) = \text{Min } u^T(b - B\overline{y})$$

st.

$$A^T u \ge c$$

$$u \in \Re_-^m$$

In each iteration the objective function of DSP is updated using the optimal solution of RMP obtained in the previous iteration. Notice that the solution space of DSP is always the same. In each iteration the Benders algorithm produces a cut called the Benders cut which is added to RMP. The cut is produced from the optimal extreme point (or extreme ray) of DSP solution space. Two different types of cuts can be produced in classic Benders algorithm:

- Case 1: If the optimal value of DSP is unbounded then the following feasibility cut is added to RMP: $v^{iT}(b - By) \le 0$ where v^i is the vector that corresponds to extreme ray i.
- Case 2: If the optimal value of DSP is bounded and the optimality condition is not satisfied then the following optimality cut is added to RMP: $u^{jT}(b - By) + d^T y - z \le 0$ where u^j is the vector that corresponds to extreme point j.

If we produce all possible Benders cuts using all the extreme points and extreme rays of DSP solution space then the resulting augmented RMP is an equivalent version of IP and its solution gives the same optimal solution as IP. The total number of Benders cuts, which equals the number of extreme points and extreme rays of DSP, is generally enormous. However, we know that at the optimum the number of RMP active constraints will never exceed the number of RMP decision variables. The main idea of the Benders algorithm is based on the observation that the algorithm is going to converge, satisfying the optimality condition before the addition of all Benders cuts. The convergence criterion is satisfied when the difference between the UB obtained by the best optimal solution of PSP and the LB obtained by the solution of the last RMP is less than or equal to the parameter ε $(UB - LB \le \varepsilon)$, where ε is a very small number (e.g. $\varepsilon = 0.01 * LB$). Notice that the finite convergence of the algorithm results from the fact that DSP has a finite number of extreme points and extreme rays. More details about the Benders decomposition algorithm are given in a number of references including the papers of Benders (1962) and Minoux (1986).

Although in many cases Benders Algorithm has been shown to perform very well, there are instances where convergence is poor due to the produced Benders cuts. Thus the question in those cases is how to best design the cuts that are used in each iteration so that convergence can be improved. If it was possible to define a priori the extreme points and extreme rays of the solution space of DSP, which can produce only the cuts which correspond to active constraints, then we could produce all these cuts simultaneously, add them to RMP and solve the augmented RMP only once. The solution of this RMP would give the optimal solution of IP. The development of such a procedure is as difficult as developing a procedure to define a priori which is the optimal base in the beginning of the simplex method. Even if we could not produce a priori only those cuts which correspond to active constraints, we can at least produce in each iteration more than one cut using the same solution of RMP in order to further restrict its solution space. The objective is to produce, in addition to Benders cut, other "good" cuts which can help the convergence of the algorithm. A "good" cut is defined as a cut which significantly restricts the solution space of RMP. As previously noted, we have the possibility, from the first iteration of the Benders algorithm, to produce all the possible cuts because the solution space of the DSP is not affected by the solution of RMP; it is the objective function which changes in each iteration. Using all extreme points and rays we obtain an equivalent version of IP which is more complicated to be solved to optimality. On the one hand Benders could converge in only one iteration if we produce all Benders cuts at once, but the CPU solution time would be significantly longer than the direct solution of IP. On the other hand, if we produce only one cut in each iteration, a large number of iterations is likely to be needed, resulting in a longer CPU solution time. A good strategy in order to converge to optimality faster than the classic algorithm is to maintain a balance between the number of iterations and the number of cuts produced in each iteration. This balance should be based on the idea that increasing the number of cuts decreases the number of iterations, but RMP becomes more complicated to be solved to optimality and extra time is needed to generate the additional cuts.

In this paper a new approach is developed as alternative strategy which can be used separately or in combination with any other strategy already developed for speeding-up the Benders decomposition algorithm. This new strategy is referred to as the maximum density cut (MDC) generation. In Sects. 3 and 4, MDC strategy is presented and applied in two case studies in order to illustrate their efficiency. Notice that all results presented in this paper have been obtained on Pentium (R) 4, CPU 2.40 GHz, RAM 1 GB and CPLEX 9.0 using a C++ implementation of the proposed approach.

2.2 Covering cut bundle generation

The heuristic or exact methods for the acceleration of Benders algorithm presented in the literature focused on the complexity of the subproblems (RMP and DSP), eliminating the integrality constraints in the RMP problem (McDaniel and Devine 1977) or using a sub-optimal solution of DSP or RMP in order to produce an alternative cut (Cote and Laughton 1984; Rei et al. 2006; and Zakeri et al. 1998). To the best of our knowledge only Magnanti and Wong (1981), Gabrel et al. (1999), Minoux (2001), Saharidis et al. (2010) and Saharidis and Ierapetritou (2010) introduced the concept of multi-generation of cuts in a general framework.

Saharidis et al. (2010) presented the CCB strategy which has as an objective to study the density of Benders cuts. A low (high) density cut is a cut covering a small (large) number of decision variables of RMP. A decision variable is covered in a cut if its coefficient is not equal to zero (or near to zero relative to the other coefficients). Low (high) density cut is defined as a cut where less (greater or equal) than n % of decision variables are involved as α-covered.

Definition 1 In a feasibility cut, a variable y_k is said to be α-covered in the cut of the form $\sum_k (v^T B)_k y_k \geq v^T b$ if the kth row of the matrix $(v^T B)$ is greater than or equal to α % of the coefficient with the maximum absolute value in the current cut: $|(v^T B)_k| \geq 10^{-2}\alpha \,\mathrm{Max}_{\forall k}\{|(v^T B)_k|\}$ where α is a given parameter chosen in [0, 100 %].

Definition 2 We call α-covering cut bundle, a set of cuts such that each variable y_k, is α-covered in some cut of the bundle.

The above definitions are used in the case of feasibility cut. In the case of optimality cut the vector $u \in U$ is used instead of the vector $v \in V$.

The CCB generation proceeds by generating in each iteration a bundle of cuts instead of a single cut. The CCB algorithm implies the generation of a bundle of cuts by an auxiliary problem (based on the slave problem solved in current iteration) using the values obtained by the solution of the last RMP. The produced bundle of cuts is intended to involve most decision variables of RMP. Due to this feature one can expect that the solution space of RMP is restricted faster due to addition of this bundle of cuts and the algorithm converges faster to an optimal solution.

In CCB strategy the objective is to examine (in each iteration) the cut produced by the classical Benders algorithm and find which variables are not "α-covered" by this. Next we generate a second cut "covering" at least one of those variables, update the set of variables not covered yet and continue this procedure until the moment that all decision variables of RMP are covered or a predefined maximum number of covered decision variables is reached. At each iteration of classic Benders decomposition algorithm, given the current optimal solution $(\overline{y})$ of the RMP, we solve the corresponding SP and we produce one of the following cuts:

$$v^{iT}(b - By) \leq 0 \quad \text{or} \quad u^{jT}(b - By) + d^T y - z \leq 0$$

where the extreme ray v or the extreme point u is the current optimal dual solution. The coefficient of y_j in the produced cut is equal to $(v^T B)_j$ or $(u^T B)_j$. In the two applications presented in Saharidis et al. (2010), on average 70 % of these coefficients are equal to zero which confirms that the cuts tend to have low density. In order to generate the α-covering cut bundle, we will consider bounds inducing constraints on v (or u) and add them to DSP. These constraints have the following form:

$$LB_j \leq \left(v^T B\right)_j \leq UB_j \quad \text{or} \quad LB_j \leq \left(u^T B\right)_j \leq UB_j$$

where the parameter LB_j is the lower bound on the coefficient of the variable decision y_j and UB_j is the upper bound on this coefficient. Without loss of generality, assuming that the current PSP is infeasible (case of feasibility cut) and including these additional constraints in DSP (dual PSP), we obtain the following Auxiliary Dual Problem (ADP):

$$ADP: \quad \begin{cases} \Gamma = \text{Min}\, u^T(b - B\overline{y}) \\ \text{subject to:} \\ A^T v \geq 0 \\ v^T e = -1 \\ -\left(v^T B\right)_j \geq UB_j \\ \left(v^T B\right)_j \geq LB_j \\ v \leq 0 \end{cases}$$

As will be shown below, this problem will be solved for different values of LB_j and UB_j in order to cover the decision variable y_j of RMP. Introducing two new sets of variables ϑ_j and μ_j, for the additional inequalities, the corresponding auxiliary primal problem (APP) which defines an extreme ray covering y_j takes the following form:

$$APP: \quad \begin{cases} \Gamma' = \text{Max}\, -\xi - \sum_j UB_j \theta_j + \sum_j LB_j \mu_j \\ \text{subject to:} \\ Ax - \sum_j B^j \theta_j + \sum_j B^j \mu_j - \xi e \leq b - B\overline{y} \\ x, \theta, \mu, \xi \geq 0 \end{cases}$$

In order to generate a cut where a decision variable y_{j_0} of RMP is considered α-covered in the cut, we setup the current APP updating the right hand side of APP using the current optimal solution $\overline{y}$ of the RMP and we fix for $j = j_0$ the coefficient of θ_{j_0} and μ_{j_0} in the objective function to be: $LB_{j_0} = UB_{j_0} = +\eta$ (or $-\eta$). After we solve the APP we generate a cut which has exactly the same form as the feasibility Benders cut using the optimal value of dual decision variables. The parameter η is the average of the coefficients of α-covered decision variables in the classical Benders cut and takes in each iteration the following value:

$$\eta = (1/k) \sum_j \left|\left(v^T B\right)_j\right|, \quad j \in \left\{ j : \left|\left(v^T B\right)_j\right| \geq \alpha \operatorname*{Max}_{\forall j} \left\{ \left|\left(v^T B\right)_j\right| \right\} \right\}$$

Notice that k represents the total number of decision variables which are α-covered in the classical Benders cut: $k = \sum_j j,\ j \in \{j : |(v^T B)_j| \geq \alpha\, \text{Max}_{\forall j}\{|(v^T B)_j|\}\}$. Fixing the coefficients LB_{j_0}, UB_{j_0} of APP's objective function, to a nonzero value equals to $+\eta$ (or $-\eta$), we are making sure that at least the variable decision y_{j_0} will be α-covered in the next cut generated. Note that this procedure does not constraint the values of the other coefficients.

In the general form of the CCB algorithm the values of LB_j and UB_j in APP are:

$$LB_j = -\eta/\alpha \quad \text{and} \quad UB_j = \eta/\alpha, \quad \forall j \neq j_0$$
$$LB_{j_0} = UB_{j_0} = +\eta \text{ or } -\eta$$

The sign of the parameter η depends on the value of the dual variable of DSP($\overline{y}$) and the corresponding column of matrix B (e.g. B^j):

- If $v_{j_0} B^{j_0}$ (or $u_{j_0} B^{j_0}$) ≥ 0 then $LB_{j_0} = UB_{j_0} = +\eta$ and if $v_{j_0} B^{j_0}$ (or $u_{j_0} B^{j_0}$) ≤ 0 then $LB_{j_0} = UB_{j_0} = -\eta$
- If $v_j B^j$ (or $u_j B^j$) ≥ 0 then $LB_j = 0$ and $UB_j = +\eta/\alpha$ and if $v_j B^j$ (or $u_j B^j$) ≤ 0 then $LB_j = -\eta/\alpha$ and $UB_j = 0, \forall j \neq j_0$

The choice of the y_{j_0} decision variable to be covered in the next cut is based on the corresponded column of matrix B. If the elements of B^{j_0} column are not equal to zero then the corresponded y_{j_0} could be the next candidate decision variable.

Our idea is to produce not only one extra cut by APP but a number of cuts (α-covering cut bundle) where we guarantee that other variables are α-covered. In CCB strategy not only the ASP (or DSP) is solved but we have also the successive solution of APP using the same optimal solution of the current RMP. In each solution of APP, the parameters LB_j, UB_j are changed and fixed to a certain value for the generation of a new cut. Before resolving the APP the cut produced is added to RMP and for another not yet α-covered variable $y_{j_0'}$, the parameters $LB_{j_0'}$ and $UB_{j_0'}$ are fixed equal to η (or $-\eta$). At the same time the bounds LB_{j_0}, UB_{j_0} of the coefficient of the variable y_{j_0} are re-initialized. A second cut is produced and the iterations continue. Before the production of this α-covering cut bundle, the classical Benders procedure is applied to produce a classical cut from the optimal solution of DSP. After the generation of this cut a test is performed in order to determined which variable has been α-covered in order to produce the α-covering cut bundle which covers other decision variables.

The CCB procedure stops when predetermined maximum number of cuts has been attainted or when all possible decision variables of the RMP have been α-covered. In real case problems the developed model are large scale models and it is better to stop CCB generation when a predetermined maximum number of cuts has been attainted than to stop when all possible decision variables have been covered (Saharidis et al. 2010). Saharidis et al. (2010) found that the choice of the number of cuts produced in each iteration of CCB procedure influences the behavior of the algorithm. Increasing the total number of cuts the total number of iterations decreases, but the total CPU solution time corresponding to the solution of APP models is higher compared to the total CPU time when less cut (in the cost of higher total number of iterations) are produced for the bundle. There is a critical point of total number of cuts after which if it increases more the time spent to generate them becomes computational expensive for CCB. Our motivation to develop the MDC strategy is based on the idea that it is computational expensive to cover all decision variables in the bundle of cuts and for this reason we suggest the generation of a cuts bundle where a certain number of decision variables are covered and the rest are covered by MDC strategy. MDC generation can be also used as a standalone procedure especially in the case where the solution of ASP (or DSP) is computational expensive. In the following section the MDC generation strategy is presented.

3 Maximum density cut (MDC) generation

The basic idea explored in this paper is the generation of cuts where the maximum number of variables of RMP is considered. From the complimentary slackness theorem, we know that for the solution of a problem where the kth constraint is not active constraint, the corresponding kth dual decision variable is equal to zero. If at the optimal solution, a big number of PSP's constraints are not active (e.g. a big number of dual decision variables are equal to zero) then the generated Benders cut (feasibility or optimality) tends to be a low density cut. In order to generate a cut which restricts significantly the solution space of RMP (e.g. high density cut), a feasible solution should be found where the maximum number of PSP's constraints are active (primal maximum density cut generation) or where the maximum number of DSP's decision variables is not equal to zero (dual maximum density cut generation).

3.1 Primal maximum density cut generation (maximizing active constraints), primal-MDC

Transforming the maximum possible number of PSP constraints into active constraints and then solving the resulting PSP to optimality leads to the production of a cut that has the highest possible density. We can measure the extent of constraint satisfaction by introducing slack variables to the PSP. Through the complimentary slackness theorem we know that for the optimal solution of the primal systems, whenever the slack variable of the kth constraint is positive, the kth variable of its dual counterpart is zero. In order to produce a high density optimality (or feasibility) cut we should be able to find an extreme point (or extreme ray) where we get the same optimal solution of PSP, (under the condition of multi-optimal solution) but the number of active constraints is maximum.

The primal-MDC using an Auxiliary Primal Slave Problem (APSP) leads to a reduction in the number of non-active constraints. Without loss of generality, we consider a feasible PSP with $y = \overline{y}$. In order to find the extreme point of the solution space of PSP, where the number of active constraints is maximum and where we get the same optimal solution, we introduce slack variables to PSP, constraints in order to measure the extent of them and an additional constraint in order to get the same optimal solution. Then the following APSP is obtained:

Auxiliary Primal Slave Problem (APSP)

$$f_4 = \text{Min}\, e^T h$$

st.

$$Ax + Is = b - B\overline{y'}$$

$$s \le Mh$$

$$c^T x = f\left(x^*\right)$$

$$x \in \Re_+^n \ge 0,\; s \in \Re_+^m,\; h \in \{0, 1\}^m$$

where M is a large positive number, I is the $m \times m$ identity matrix and s, h are vectors of dimension m, $f(x^*)$ is the optimal solution of PSP at the current iteration of Benders algorithm, and e^T is the unit vector. Solving the APSP, the obtained solution corresponds to the same optimal solution where the number of active constraints is maximum. Considering without loss of generality that the optimal solution of APSP gives the first m_1 constraints as active constraints and the rest m_2 $(m_1 + m_2 = m)$ as inequalities then using this information the following New Primal Slave Problem (NPSP) is constructed:

New Primal Slave Problem (NPSP)

$$f_5 = \text{Min}\, c^T x$$

st.

$$A_1 x = b_1 - B_1\overline{y}$$

$$A_2 x \le b_2 - B_2\overline{y}$$

$$c^T x = f\left(x^*\right)$$

$$x \in \Re_+^n$$

where A_1, A_2 are $m_1 \times n$ and $m_2 \times n$ matrices ($A = \left[\begin{smallmatrix} A_1 \\ A_2 \end{smallmatrix}\right]$), B_1, B_2 are $m_1 \times q$ and $m_2 \times q$ matrices ($B = \left[\begin{smallmatrix} B_1 \\ B_2 \end{smallmatrix}\right]$) and b_1, b_2 are vectors of dimension m_1 and m_2 ($b = \left[\begin{smallmatrix} b_1 \\ b_2 \end{smallmatrix}\right]$). We solve the NPSP to optimality and we produce a cut in the same way as in the classic Benders algorithm. The cut produced in this way has the maximum number of non-zero coefficients. It should be noticed that the form of APSP does not change the structure that Benders subproblems may have (ex. block decomposable), because the introduced auxiliary integer variables do not link the problem constraints.

We should pay attention to the fact that the above procedure of maximizing the number of active constraints involves the solution of a mixed integer linear problem (MILP) formulation in order to produce the linear program NPSP. Generally the solution time of APSP is not long even if it is a MILP because if PSP is a block decomposable problem, the addition of auxiliary variables do not change its structure. However, if we need to develop a simpler version of the APSP, we can use approximation techniques relaxing the binary variables to continuous 0–1 or using the method presented in Freund et al. (1985), Hadigheh et al. (2007), Schittkowski (2009) in order to define the always-active constraints in combination with approximation techniques. In the context of this paper and for the numerical examples presented in Sect. 4, we use the exact method presented above in order to get the exact MDC and fully examine its impact.

3.2 Dual maximum density cut generation (maximizing dual values), dual-MDC

When Benders algorithm produces a low density cut, a large number of dual variables of the slave problem have the value of zero. An alternative method in order to generate a high density cut could be based on the solution of the dual slave problem. Maximizing the nonzero value of the dual variables we guarantee that the resulting cut has a maximum number of non-zero coefficient covering with this way a maximum number RMP decision variables.

Without loss of generality we consider a bounded DSP with $y = \overline{y}'$. In order to find the extreme point of DSP's solution space, where the maximum of decision variables take a non-zero value, two groups of decision variables and constraints are added. Revising also the objective function of DSP, we obtain the following Auxiliary DSP (ADSP):

Auxiliary Dual Slave Problem (ADSP)

$$f_6 = \text{Min} \sum_m \left(k_m^1 + k_m^2\right)$$

st.

$$A^T u \geq c$$

$$u_m \sum_q B_{m,q} \leq -n\left(1 - k_m^1\right) + M k_m^1$$

$$u_m \sum_q B_{m,q} \geq n\left(1 - k_m^2\right) - M k_m^2$$

$$k_m^1 + k_m^2 \leq 1$$

$$u \in \Re_-^m,\ k_m^1, k_m^2 \in Z^m = \{0, 1\}^m$$

where A is $m \times n$ matrix, B is $m \times q$ matrix, u is an m-vector, k_m^1, k_m^2 are m-vectors and η is a positive number defined by the last generated Benders cut as in the CCB strategy (for more details see Sect. 2.2). Notice that $u_m \sum B_{m,q}$ could be positive or negative. For this reason we could add the valid inequality $k_m^1 + k_m^2 \leq 1$ in order to restrict further the solution space of ADSP. We solve the ADSP to optimality and we produce a cut in the same way as in the classic Benders algorithm. This cut has the maximum number of non-zero coefficients giving rise to a cut with the highest density. It should be noticed that the form of ADSP does not change the structure that Benders sub-problems may have (i.e. block decomposable), because the introduced auxiliary integer variables do not link the problem constraints. Finally, a modified version of ADSP could be used in order to speed-up the generation of MDC. The integrality constraint of k_m^1, k_m^2 can be relaxed resulting to a continuous linear problem:

Relaxed-Auxiliary Dual Slave Problem (ADSP)

$$f_6 = \text{Min} \sum_m \left(k_m^1 + k_m^2\right)$$

st.

$$A^T u \geq c$$

$$u_m \sum_q B_{m,q} \leq -n\left(1 - k_m^1\right) + M k_m^1$$

$$u_m \sum_q B_{m,q} \geq n\left(1 - k_m^2\right) - M k_m^2$$

$$k_m^1 + k_m^2 \leq 1$$

$$u \in \Re_-^m,\ k_m^1, k_m^2 \in (0, 1)^m$$

In the context of this paper and for the numerical examples presented in Sect. 4, we use the relaxed ADSP. Solving the relaxed-ADSP instead of ADSP results in a solution, which is feasible to the DSP since ADSP solution space is a restriction of the DSP solution space even after removing the integrality constraint. This is the reason that allows us to use the solution resulting by the relaxed ADSP. This solution based on our numerical results produces a high quality cut approximating the best cut that can be obtained by ADSP.

The role of MDC generation when is implemented in the CCB framework is to cover the rest of the decision variables which are not covered by the classical Benders cut and the bundle of cut produced by CCB generation. The difference between the standalone implementation of MDC strategy and its implementation in the CCB framework is that after the addition of Benders cut to RMP, three new algorithmic steps are added if the classical Benders cut is a low density cut. The first step is to run CCB generation and add the bundle of cut to RMP. The second step is to check which decision variables are not covered by the classical Benders cut and CCB generation. Finally, the third step is to build an NPSP when primal MDC is applied or to build an NDSP when dual MDC is applied. The critical difference between the standalone implementation of MDC and the implementation of MDC in the framework of CCB is that in the later case slack variables are added to PSP only for the constraints that are not active at the optimal solution of the current PSP. Moreover, the series of APP when the primal MDC is used and when the dual MDC is used, the constraints bounding the coefficients of RMP's decision variables are added to the DSP only for the coefficient that are equal to zero at the optimal solution of the current DSP and/or the series of ADP. After these three steps the algorithm continues as in the classical implementation of CCB strategy and the produced MDC covers the rest of the decision variables which were not covered by Benders cut and CCB.

In the standalone implementation of MDC strategy, in each iteration of the algorithm after the solution of the PSP, the generated Benders cut is examined. If this cut is a high density cut then the algorithm does not produce an MDC cut, proceeds to the solution of RMP and stops if optimality criterion is satisfied. If not it sends the information to update the PSP for the generation of another cut. If the Benders cut is a low density cut then the RMP and the corresponded APSP are solved. When APSP is solved the corresponded NPSP is constructed based on the optimal solution of APSP and the MDC is generated (using primal-MDC or dual-MDC strategy). After the solution of RMP the optimality criterion is examine. If it is not satisfied then the PSP is updated by the optimal solution of RMP and the algorithm continues.

4 Numerical examples

4.1 Description of case studies

In general, the classic Benders algorithm does not always provide cuts with the maximum number of active constraints, and this is the main reason that a low density cut is obtained. We have applied Benders decomposition, CCB and (primal and dual) MDC generation techniques to problem instances arising from two different applications where the convergence of classical Benders algorithm is slow. We have chosen to apply CCB and MDC strategies in these particular case studies because we observed that applying the classical Benders algorithm produces a low density cuts most of the time. On average 70 % of the coefficients of RMP decision variables are equal to zero, which confirms that the cuts tend to have low density. The addition of a single low density cut does not significantly restrict the solution space of the RMP, thus leading to an increased number of iterations and CPU solution time. As such, CCB and MDC cut generation are appropriate strategies to use in order to solve this problem. An additional reason to apply MDC strategy is that PSP is easy to be solved requiring lower CPU solution time for APP, APSP, NPSP and ADSP than the CPU time required for RMP which could be an advantage as we mention in the conclusions if parallel optimization is applied in order to solve APP, APSP, NPSP and ADSP during the solution of RMP. For a majority of instances a sufficient reduction regarding both the number of iterations and the solution time of the algorithm is observed. Notice that, for both case studies, the decomposition is done between the continuous and the binary variables of the IP. The slave problems result by fixing all the binary variables of IP and RMP contains all the integer constraints of IP.

In the first case study the problem addressed concerns the development of a general model for scheduling of crude oil in a refinery (Saharidis 2006; Saharidis et al. 2009; Saharidis and Ierapetritou 2009b). 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 units (CDU) and/or another system of pipelines. The units which compose the system under consideration consist of ports, storage tanks, CDUs and pipelines. The crude oil is brought into the refinery by boats or by pipelines and is loaded in the storage tanks. After loading, the crude oil must stay for fixed period into the storage tanks in order to separate the water from the crude oil. The storage tanks have a capacity possibly exceeding thousands of cube meters. 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 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. Three different mixing strategies are possible: (a) the mix is produced in the storage tanks, (b) just before the CDU within the pipeline and (c) a combination of these two strategies. Concerning the distillation, two different options are possible: (a) the mix required by CDUs must have an exact concentration and (b) concentrations of components required by the CDU mix must only satisfy some upper and lower bounds.

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, the number of ports, storage tanks and CDUs and also storage capacity and pumping capacity are given. Concerning the problem constraints the model includes material balances to account for transfer between the port tanks and the CDUs, operational constraints expressing the system's specificities as for example the requirement that a tank cannot be loaded and unloaded in the same time period. The number of decision variables for the examples considered varies between 768 and 2112 and the number of constraints is between 744 and 1860.

In order to further assess the efficiency of MDC strategy, we also considered a second case study. This also concerns a scheduling problem for multi-product, multi-purpose batch plants. A detail description of the underlying problem can be found in Ierapetritou and Floudas (1998). In brief, the problem considers the optimization of the plant capacity to improve a given economic objective. Usually profit, cost, or production makespan is considered. The decision variables involve 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 maximum capacities of production units, minimum requirements for operation and the processing time for the specified tasks. The constraints involve material balance throughout the plan, assignment constraints to ensure that each task is assigned to one unit, capacity constraints to ensure that the capacity limits of the units are not exceeded 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 490 to 882 and number of constraints between 5549 and 9981.

Finally, before the presentation in the following section of the numerical results obtained by the problems arising from these two case studies in this paragraph we have to give a brief discussion for parameters n and α of CCB and MDC strategies. In practice, for n (the percentage of α-covered decision variable in the cut) and α (the percentage of coefficient's value with the maximum absolute value) parameter, tuning will be necessary in order to obtain the maximum reduction in the number of iterations or in the overall computing time. On the one hand increasing n and α will increase the number of cuts considered to be low density cuts and perhaps only a small number of high density cuts could be produced. On the other hand, decreasing n and α will increase the produced cuts considered as high density cuts which means that only a small number of cuts are going to be produced. Notice that the number of possible cuts depends on the values of parameters n and α. For example, in MDC strategy, when the extreme point obtained by the solution of the current APSP produces a cut where less than n % of the decision variables of RMP are involved as α-covered, the MDC is not added to RMP. A good strategy in order to converge to optimality as fast as possible is to maintain a balance between the parameters n and α. This balance should be based on the idea that only the MDC with the highest density significantly restrict the RMP and MDC of medium or low density may result in more complicated RMP which is hard to solve to optimality without significantly restricting its solution space. Notice that for the numerical examples presented in Tables 1–3 n is equal to 30 % and α is equal to 10 %. We have chosen these values for these parameters because for the presented case studies we found that these values give the best results.

4.2 Numerical results

In this subsection we present several numerical results which are obtained after the application of classical Benders algorithm, CCB and (primal and dual) MDC strategy. The objective of this section is not only to show the benefit applying MDC strategy in the context of CCB but also to demonstrate the effect of standalone implementation of MDC strategy, generating and adding MDC to RMP in addition to classical Bender cuts.

In order to accelerate Benders decomposition algorithm the priority is to reduce the total number of RMP problems that need to be solved. Decreasing the number of main iterations usually results in reduction of CPU solution time. CPU solution time is the time that the algorithm spends in order to solve the RMP but also DSP, APP problems in CCB strategy and APSP, NPSP, ADSP problems in MDC strategy. Reducing the number of iterations, results in a significant reduction of CPU solution time due to smaller number of RMP's that have to be solved. Applying the classical Benders algorithm in case study 1 and case study 2, the Benders cuts produced in each iteration are low density cut as shown in the 2$^{\text{nd}}$ columns of Tables 1 and 2 (average density no more than 27 %). For both Tables 1 and 2, we present the relative difference between the classical implementation of Benders and the standalone implementation of MDC strategy (dual or primal). In the 4$^{\text{th}}$ and 6$^{\text{th}}$ columns of these tables we find the reduction of CPU solution applying MDC strategy and in the 5$^{\text{th}}$ and 7$^{\text{th}}$ columns the reduction of the total number of iterations. In addition for both Table 1 and 2, we give in columns 8$^{\text{th}}$, 9$^{\text{th}}$, 10$^{\text{th}}$ and 11$^{\text{th}}$ the improvement of CCB strategy when MDC (primal or dual) is applied. Finally, in Appendix we present the analytic numerical results for each example where the total number of iterations and the CPU solution time needed until optimality is reached for original Benders and Benders with CCB and/or with MDC strategy. We note that the results for CCB and classical Benders are based on our previous work presented in Saharidis et al. (2010).

Applying MDC cut generation, the produced cut involves the maximum number of decisions variables of RMP and thus we expect that it improves the convergence of Benders algorithm. In Table 1, we present some numerical results which correspond to case study 1 (Saharidis 2006; Saharidis et al. 2009). With MDC strategy, for the 15 problems tested an optimal solution could be obtained requiring a significantly reduced CPU time and number of iterations compared with the standard implementation of Benders method. A reduction up to 97 % for some examples is observed concerning the solution time and up to 79 % concerning the total number of iterations. This important reductions justify that for Benders decomposition the generation of low density cuts is an important drawback and whatever is the applied method (CCB or MDC or CCB with MDC) to cover a maximum number of RMP decision variable the improvement is significant. It should be noticed that the decrease in CPU time and number of iterations is really high (more than 50 %) when the density of the Benders cuts is low. For Examples 7 and 14 where low average density appears, we get on average of 54 % reduction in CPU time and 45 % in number of iterations. However, as shown in Examples 6, 10, 11 and 15, we still get a significant decrease when the density of cuts is higher (but still low).

In Table 2, we present another series of numerical examples that correspond to case study 2 (Ierapetritou and Floudas 1998) in order to further assess the efficiency of MDC

Table 1 Comparison between classic Benders and MDC cut generation

Ex.	Average density of Benders cuts	α-covered cuts generated per iter. of CCB	Relative difference classic Benders and primal-MDC		Relative difference classic Benders and dual-MDC		Improvement of CCB adding primal-MDC		Improvement of CCB adding dual-MDC	
			CPU (s)	# of iter	CPU (s)	# of iter	CPU (s)	# of iter	CPU (s)	# of iter
Ex. 1	10 %	100	48 %	60 %	57 %	60 %	9 %	9 %	11 %	9 %
Ex. 2	12 %	90	6 %	55 %	19 %	55 %	16 %	8 %	24 %	8 %
Ex. 3	11 %	70	59 %	67 %	70 %	68 %	6 %	10 %	18 %	5 %
Ex. 4	11 %	100	76 %	78 %	76 %	78 %	0 %	10 %	0 %	10 %
Ex. 5	15 %	100	97 %	79 %	97 %	79 %	5 %	10 %	8 %	10 %
Ex. 6	20 %	100	45 %	64 %	56 %	63 %	7 %	9 %	11 %	9 %
Ex. 7	9 %	30	58 %	55 %	58 %	54 %	9 %	16 %	20 %	9 %
Ex. 8	10 %	30	31 %	34 %	34 %	33 %	15 %	13 %	15 %	13 %
Ex. 9	12 %	30	56 %	50 %	63 %	49 %	11 %	17 %	22 %	11 %
Ex. 10	20 %	20	62 %	61 %	63 %	61 %	6 %	14 %	12 %	11 %
Ex. 11	20 %	100	29 %	34 %	36 %	32 %	7 %	5 %	9 %	1 %
Ex. 12	9 %	100	55 %	50 %	63 %	52 %	18 %	9 %	28 %	8 %
Ex. 13	11 %	30	19 %	45 %	32 %	42 %	24 %	11 %	34 %	6 %
Ex. 14	9 %	30	51 %	35 %	53 %	39 %	15 %	22 %	17 %	20 %
Ex. 15	18 %	30	50 %	55 %	51 %	52 %	18 %	17 %	20 %	15 %

Table 2 Comparison between classic Benders and MDC cut generation

Ex.	Average density of Benders cuts	α-covered cuts generated per iter. of CCB	Relative difference classic Benders and primal-MDC		Relative difference classic Benders and dual-MDC		Improvement of CCB adding primal-MDC		Improvement of CCB adding dual-MDC	
			CPU (s)	# of iter	CPU (s)	# of iter	CPU (s)	# of iter	CPU (s)	# of iter
Ex. 16	18 %	10	12 %	12 %	16 %	6 %	6 %	33 %	12 %	29 %
Ex. 17	18 %	10	49 %	17 %	49 %	16 %	8 %	26 %	11 %	25 %
Ex. 18	21 %	10	38 %	36 %	39 %	32 %	20 %	16 %	16 %	15 %
Ex. 19	27 %	10	13 %	25 %	15 %	23 %	2 %	22 %	7 %	19 %
Ex. 20	13 %	15	52 %	38 %	41 %	36 %	19 %	17 %	20 %	15 %
Ex. 21	25 %	10	37 %	25 %	37 %	25 %	28 %	35 %	29 %	39 %
Ex. 22	27 %	10	34 %	18 %	35 %	16 %	19 %	26 %	23 %	24 %
Ex. 23	18 %	15	14 %	61 %	18 %	57 %	10 %	56 %	20 %	53 %
Ex. 24	7 %	15	56 %	31 %	58 %	25 %	−7 %	45 %	−4 %	36 %
Ex. 25	19 %	15	38 %	31 %	39 %	28 %	30 %	45 %	30 %	43 %
Ex. 26	18 %	15	42 %	37 %	42 %	37 %	24 %	53 %	24 %	52 %
Ex. 27	6 %	10	67 %	55 %	67 %	54 %	5 %	25 %	13 %	20 %
Ex. 28	9 %	10	20 %	35 %	23 %	34 %	−10 %	35 %	−3 %	31 %
Ex. 29	11 %	15	58 %	48 %	58 %	48 %	24 %	37 %	25 %	37 %
Ex. 30	10 %	15	44 %	62 %	44 %	62 %	13 %	26 %	15 %	25 %

strategy when it is applied alone or in combination with CCB. As shown in Table 2, the average density of cuts produced in the second case study is on the same order than in the previous case. This leads to an equivalent decrease in the CPU time and total number of iterations as compared with case study 1. The addition of MDC cut in the context of CCB results in a reduction between 6 % and 30 % of the CPU solution time and between 15 % and 56 % of the total number of iterations. The only exception is observed for Examples 24 and 28 where the standalone implementation of CCB is more beneficial for Benders than its combination with MDC (primal or dual). The reason is, that in both cases, the solution CPU time is low (on average 30 seconds) and the cumulative CPU time needed for the generation of MDC compared to this time, is high.

After the incorporation of MDC to CCB, we have observed that for case study 1 where the DSP is more computational expensive than in case study 2, we get larger reduction of CPU and total number of iterations. This is an indication that for problems where the classical implementation of Benders decomposition gives low density cuts and the resulted PSP (or DSP) is difficult problem (Zakeri et al. 1998) it is better to produce a bundle of cuts with a small number of cuts and cover the rest of the decision variable with a MDC cut. If, for example we take Ex. 9 and instead of 30 cuts we produce 60, the CPU solution time and total number of iteration for CCB is better. This is not true when MDC is incorporated to CCB. As shown in Table 3 increasing the number of cuts from 30 to 60 the total CPU time decreases when CCB is implemented without MDC (from 609 seconds to 628 seconds). The results are different when MDC is applied in combination with CCB where Benders algorithm converges faster with a bundle of 30 cuts than with a bundle of 60 cuts. (Best CPU solution time is equal to 530 seconds when dual-MDC is applied in combination with CCB with 30 cuts.)

It should be noticed that when the CCB needs a lot of iterations to converge, the implementation of MDC cut in the context of the classical Benders algorithm gives similar results. For Examples 5, 11, 12, 13 and 14, CCB strategy needs a high number of iterations in order to converge to optimality (see Appendix). For the same examples the implementation of MDC gives similar results. This phenomenon is explained by the fact that CCB needs some extra time to produce the CCB and as the number of algorithm's iterations increases the cumulative effect from the production of these cuts becomes more important. We have observed also that if the CCB CPU solution time is low then the positive effect of the addition of MDC is less important compared to the cases where the CPU solution time of CCB is higher. However, we have to notice that the effect of MDC cut is always positive even if in some case it is not as important. As we mentioned in our previous work (Saharidis et al. 2010) CCB strategy produce a bundle of cuts which have similar density as the classical Benders cuts. Based on this observation, Tables 1 and 2 show also that the disaggregated version of a high density cut (i.e. CCB) which correspond to a bundle of cuts has more positive effect than its aggregated version (i.e. MDC). Our numerical results show that in general it is better to cover the decision variable of RMP producing a bundle of low density cuts (as CCB strategy does) than one only high density cut (as MDC strategy does) under the condition that the time spent for the generation of the bundle of low density cuts is less than the time spent for the generation of the high density cut.

In all examples presented in Tables 1 and 2, the addition of MDC cut (primal or dual) improves CCB strategy, decreasing significant the total number of iterations and the CPU solution time (see columns 8^{th}, 9^{th}, 10^{th} and 11^{th} of Tables 1 and 2). We have to notice that this reduction (up to 30 % concerning the CPU solution time and up to 56 % concerning the number of iterations) corresponds to a supplementary reduction after the implementation of CCB strategy, making it very important. We have observed also that the total number of

Table 3 Example 9 using 30 and 60 cuts

CCB				CCB + primal-MDC				CCB + dual-MDC			
with 30 cuts		with 60 cuts		with 30 cuts		with 60 cuts		with 30 cuts		with 60 cuts	
CPU (s)	# of iter	CPU (s)	# of iter	CPU (s)	# of iter	CPU (s)	# of iter	CPU (s)	# of iter	CPU (s)	# of iter
682	36	661	34	609	30	628	28	530	32	630	30

iterations decreases significant more than the CPU solution time which is something that we expected to see (e.g. Benders algorithm could converge in one iteration if we generate and add all Benders cuts to RMP). Finally, we observed that Benders with CCB and dual-MDC requires in most of the times slightly more iteration to converge than CCB with primal-MDC but the total CPU solution time is lower for dual-MDC than for primal-MDC. The main reason is that even if in dual-MDC, we solve slightly more times the RMP the generation of dual-MDC is less complicated than the generation of primal-MDC because APSP is more complicate problem than ADSP which is a continuous problem. We have to notice that this observation does not prove that always dual-MDC is better than primal-MDC. Our opinion is that if the PSP (or DSP) has high number of multi-optimal solutions maybe the first approach to test would be the primal-MDC and if not then the dual-MDC we expect to have the most impact to Benders algorithm (with or without CCB).

5 Summary

A novel acceleration strategy for Benders decomposition algorithm has been presented in this paper, namely the Maximum Density Cut (MDC) generation strategy. The motivation to develop MDC strategy was that it was found that the choice of the number of cuts produced in each iteration of CCB strategy influences the behavior of the algorithm. Incorporating MDC strategy to CCB algorithm was the solution to this problem. The presented examples illustrated the applicability and efficiency of this strategy. We have provided empirical evidence to amply demonstrate that MDC strategy implemented in the framework of CCB strategy (or standalone) significant restrict the CPU solution time and the total number of iterations. Adopting MDC cut generation as a standalone procedure or in combination with CCB strategy, in the case where the standard implementation of Benders algorithm gives low density cuts, the number of iterations always decreases, generally resulting in a better solution time since the RMP has to be solved a fewer number of times.

Our perspective for the future is the use of parallel optimization in order to exchange sub-optimal information between the DSP and RMP problems. A new strategy can thus be developed where the two problems communicate with each other and exchange information about the feasible solution that they find in each iteration of the simplex algorithm, for the linear problems, and the branch and bound algorithm, for the mixed integer linear RMP problem. This intermediate exchange of sub-optimal information may give cuts with higher density than the Benders cuts and eliminate in some iteration the need to solve the APSP and NPSP. An overall comment is that the gain in the number of iterations is more significant than the gain in the CPU solution time. This observation may justify the use of parallel optimization in order to solve APP, APSP, NPSP and ADSP during the solution of RMP. In general, the solution of RMP is the most computational expensive step in Benders decomposition and the time spent for its solution could be useful for the parallel solution of the other problems for the generation of cuts.

Acknowledgements G.K.D. Saharidis would like to gratefully acknowledge financial support from Kathikas Institute of Research & Technology under the 8801019-2010 grant, the European Commission under the grant FP7-PEOPLE-2011-CIG, GreenRoute, 293753 and the action "Supporting Postdoctoral Researchers" of the Operational Program "Education and Lifelong Learning" (Action's Beneficiary: General Secretariat for Research and Technology), co-financed by the European Social Fund (ESF) and the Greek State.

M. Ierapetritou would like to gratefully 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 R832721-010 grant.

Appendix

Ex.	# of variables int./cont.	# of con.	Classic Benders			CCB			Primal-MDC		Dual-MDC		CCB + primal-MDC		CCB + dual-MDC	
			CPU (s)	# of iter.	Average density of Benders cuts	# cuts per iter.	CPU (s)	# of iter.	CPU (s)	# of iter.	CPU (s)	# of iter.	CPU (s)	# of iter.	CPU (s)	# of iter.
Ex. 1	720/1200	1860	54010	2297	10 %	100	767	35	28089	918	22991	921	699	32	681	32
Ex. 2	288/480	744	1508	814	12 %	90	1193	40	1413	369	1221	365	1005	37	912	37
Ex. 3	360/600	810	86417	3500	11 %	70	1202	41	35662	1145	26110	1109	1129	37	981	39
Ex. 4	480/840	1040	1683	817	11 %	100	60	61	401	179	399	180	60	55	60	55
Ex. 5	504/864	1110	43217	3317	15 %	100	311	99	1433	698	1389	697	297	89	287	89
Ex. 6	420/720	925	1629	228	20 %	100	301	23	897	82	712	84	281	21	269	21
Ex. 7	480/840	1040	1922	271	9 %	30	642	32	812	123	801	124	587	27	512	29
Ex. 8	420/720	952	2222	112	10 %	30	117	8	1523	74	1472	75	99	7	99	7
Ex. 9	1344/768	1664	4262	215	12 %	30	682	36	1879	107	1562	109	609	30	530	32
Ex. 10	504/864	1110	2712	109	20 %	20	295	28	1032	43	1009	43	277	24	259	25
Ex. 11	588/1008	1295	2525	151	20 %	100	701	87	1789	99	1612	102	649	83	641	86
Ex. 12	672/1152	1480	2944	213	9 %	100	991	92	1333	107	1102	102	809	84	713	85
Ex. 13	576/960	1296	2347	207	11 %	30	1701	99	1897	113	1599	121	1301	88	1119	93
Ex. 14	504/864	972	1628	191	9 %	30	691	111	798	124	765	116	588	87	571	89
Ex. 15	540/960	1155	1628	161	18 %	30	692	41	819	73	805	77	569	34	556	35

Ex.	# of variables int./cont.	# of con.	Classic Benders			CCB			Primal-MDC		Dual-MDC		CCB + primal-MDC		CCB + dual-MDC	
			CPU (s)	# of iter.	Average density of Benders cuts	# cuts per iter.	CPU (s)	# of iter.	CPU (s)	# of iter.	CPU (s)	# of iter.	CPU (s)	# of iter.	CPU (s)	# of iter.
Ex. 16	100/390	5549	69	34	18 %	10	49	24	61	30	58	32	46	16	43	17
Ex. 17	120/468	6657	7022	1014	18 %	10	3272	799	3589	840	3575	852	2999	595	2899	598
Ex. 18	120/468	6657	960	566	21 %	10	529	342	591	365	586	385	425	287	444	291
Ex. 19	120/468	6657	72	216	27 %	10	60	152	63	161	61	167	59	119	56	123
Ex. 20	140/546	7765	26111	1012	13 %	15	11401	601	12535	632	15412	645	9245	498	9112	512
Ex. 21	140/546	7765	36663	2727	25 %	10	22513	2014	23101	2044	23087	2056	16199	1312	16002	1231
Ex. 22	140/546	7765	2285	624	27 %	10	1391	489	1518	512	1495	525	1129	360	1075	371
Ex. 23	140/546	7765	137	260	18 %	15	111	96	118	102	112	111	100	42	89	45
Ex. 24	140/546	7765	72	16	7 %	15	27	11	32	11	30	12	29	6	28	7
Ex. 25	160/624	8873	44042	2915	19 %	15	26989	2002	27396	2008	26852	2100	18915	1101	18813	1151
Ex. 26	160/624	8873	63068	3612	18 %	15	36514	2292	36625	2265	36479	2285	27851	1071	27742	1108
Ex. 27	160/624	8873	3785	615	6 %	10	167	51	1258	279	1241	285	158	38	145	41
Ex. 28	160/624	8873	69	85	9 %	10	31	26	55	55	53	56	34	17	32	18
Ex. 29	180/702	9981	29642	1915	11 %	15	11714	978	12498	995	12387	999	8954	612	8801	615
Ex. 30	180/702	9981	12782	819	10 %	15	6624	242	7162	309	7145	315	5734	178	5656	181

References

Andreas, A. K., & Smith, J. C. (2009). Decomposition algorithms for the design of a nonsimultaneous capacitated evacuation tree network. *Networks*, *53*(2), 91–103.

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

Cordeau, J. F., Soumis, F., & Desrosiers, J. (2000). A Benders decomposition approach for the locomotive and car assignment problem. *Transportation Science*, *34*(2), 133–149.

Cordeau, J. F., Pasin, F., & Solomon, M. M. (2006). An integrated model for logistics network design. *Annals of Operations Research*, *144*(1), 59–82.

Cote, G., & Laughton, M. (1984). Large-scale mixed integer programming: Benders-type heuristics. *European Journal of Operational Research*, *16*, 327–333.

Freund, R. M., Roundy, R., & Todd, M. J. (1985). *Identifying the set of always-active constraints in a system of linear inequalities by a single linear program* (Working papers 1674–1985). Massachusetts Institute of Technology (MIT), Sloan School of Management.

Gabrel, V., Knippel, A., & Minoux, M. (1999). Exact solution of multicommodity network optimization problems with general step cost functions. *Operations Research Letters*, *25*, 15–23.

Hadigheh, A. G., Mirnia, K., & Terlaky, T. (2007). Active constraint set invariancy sensitivity analysis in linear optimization. *Journal of Optimization Theory and Applications*, *133*(3), 303–315.

Holmberg, K. (1994). On using approximation of the Benders master problem. *European Journal of Operational Research*, *77*, 11–125.

Ierapetritou, M. G., & Floudas, C. (1998). Effective continuous-time formulation for short-term scheduling: I multipurpose batch processes. *Industrial & Engineering Chemistry Research 37*(11), 4341–4359.

Magnanti, T., & Wong, R. (1981). Accelerating Benders decomposition algorithmic enhancement and model selection criteria. *Operational Research*, *29*, 464–484.

McDaniel, D., & Devine, M. (1977). A modified Benders partitioning algorithm for mixed integer programming. *Management Science*, *24*, 312–319.

Minoux, M. (1986). *Series in discrete mathematics and optimization. Mathematical programming theory and algorithms*. New York: Wiley-Interscience.

Minoux, M. (2001). Discrete cost multicommodity network optimization problems and exact solution methods. *Annals of Operations Research*, *106*, 19–46.

Rei, W., Gendreau, M., Cordeau, J.-F., & Soriano, P. (2006). Accelerating Benders decomposition by local branching. In *Hybrid methods and branching rules in combinatorial optimization*, Montreal.

Saharidis, G. K. (2006). *Pilotage de production a moyen terme et a court terme: contribution aux problématique d'optimisation globale vc locale et a l'ordonnancement dans les raffineries*. Genie Industriel. Vol. PhD. Paris: Ecole Centrale Paris. 150.

Saharidis, G. K. D., & Ierapetritou, M. G. (2009a). Resolution method for mixed integer bi-level linear problems based on decomposition technique. *Journal of Global Optimization*, *44*(1), 29–51.

Saharidis, G. K. D., & Ierapetritou, M. G. (2009b). Scheduling of loading and unloading of crude oil in a refinery with optimal mixture preparation. *Industrial & Engineering Chemistry Research*, *48*(5), 2624–2633.

Saharidis, G. K. D., & Ierapetritou, M. G. (2010). Improving Benders decomposition using maximum feasible subsystem (MFS) cut generation strategy. *Computers & Chemical Engineering*, *34*(8), 1237–1245.

Saharidis, G. K. D., Boile, M., & Theofanis, S. (2011). Initialization of the Benders master problem using valid inequalities applied to fixed-charge network problems. *Expert Systems with Applications*, *38*(6), 6627–6636.

Saharidis, G. K. D., Minoux, M., & Dallery, Y. (2009). Scheduling of loading and unloading of crude oil in a refinery using event-based discrete time formulation. *Computers & Chemical Engineering*, *33*(8), 1413–1426.

Saharidis, G. K., Minoux, M., & Ierapetritou, M. G. (2010). Accelerating Benders method using covering cut bundle generation. *International Transactions in Operational Research*, *17*(2), 221–237.

Schittkowski, K. (2009). An active set strategy for solving optimization problems with up to 200,000,000 nonlinear constraints. *Applied Numerical Mathematics 59*(12), 2999–3007.

Van Roy, T. J. (1983). Cross decomposition for mixed integer programming. *Mathematical Programming*, *25*, 46–63.

Zakeri, G., Philpott, A. B., & Ryan, D. M. (1998). Inexact cuts in Benders decomposition. *SIAM Journal on Optimization*, *10*(3), 643–657.

Zhu, Y., & Kuno, T. (2003). Global optimization of nonconvex MINLP by a hybrid branch-and-bound and revised general benders decomposition approach. *Industrial & Engineering Chemistry Research*, *42*(3), 528–539.