ISTITUTO DI ANALISI DEI SISTEMI ED INFORMATICA
CONSIGLIO NAZIONALE DELLE RICERCHE

A. Frangioni, C. Gentile

PERSPECTIVE CUTS FOR 0-1 MIXED INTEGER PROGRAMS

R. 577 Novembre 2002

Antonio Frangioni – Dipartimento di Informatica, Corso Italia 40, 56125 Pisa (Italy). Email: frangio@di.unipi.it.

Claudio Gentile – Istituto di Analisi dei Sistemi ed Informatica del CNR, Viale Manzoni 30 - 00185 Roma, Italy. Email: gentile@iasi.rm.cnr.it.

ISSN: 1128–3378

Collana dei Rapporti dell'Istituto di Analisi dei Sistemi ed Informatica, CNR

viale Manzoni 30, 00185 ROMA, Italy

tel. ++39-06-77161
fax ++39-06-7716461
email: iasi@iasi.rm.cnr.it
URL: http://www.iasi.rm.cnr.it

Abstract

We show that the convex envelope of the objective function of a Mixed-Integer Programming problem is the perspective function of the continuous part of the objective function. Using a characterization of the subdifferential of the perspective function we derive a family of valid inequalities that can be used to substantially improve the performances of an enumerative (Branch & Bound) approach for at least one particular model with the required structure, the Unit Commitment problem in electrical power production.

Key words: Mixed-Integer Programs, Valid Inequalities, Unit Commitment problem

1. Introduction

In many real-world problems both discrete and continuous decisions have to be made about the same entity. One of the most common cases is the one where a continuous variable p is constrained to lie in the (disconnected) set $\{0\} \cup [p_{min}, p_{max}]$ for some $0 \leq p_{min} \leq p_{max}$. This is, e.g., the case of a variable representing the output of a production process that can either be "inactive", and therefore nothing is produced, or "active", and therefore the output of the process must lie between some minimum and maximum amount. Examples can be found in Distribution and Production Planning problems, Financial Trading and Planning problems, the problem of synthesizing a processing system, or the problem of determining the optimal positioning of a new product in a multiattribute space; see, e.g., [9], [1], [9], [2], [5] [13] and the references therein. Indeed, this structure is so widespread that `XPRESS-MP` [4] and `Cplex` [8] solver suites provide built-in special support for these *semi-continuous* variables.

As a mathematical program, this structure is usually expressed through the following Mixed-Integer Program (MIP)

$$\begin{cases} \min f(p) + cu \\ p_{min}u \leq p \leq p_{max}u \quad ; \\ u \in \{0,1\} \end{cases} \tag{1}$$

where p, p_{min} and p_{max} are, in general, n-vectors. In the following, we will assume that $f : \mathbb{R}^n \to \mathbb{R}$ is a closed convex function that is finite everywhere in the hyperrectangle $\mathcal{P} = \{p : p_{min} \leq p \leq p_{max}\}$. The binary variable u models the decision of "activating the process", i.e., decides whether $p = 0$ or $p \in \mathcal{P}$, at the fixed cost c. We assume that $f(0) = 0$, because any constant term in f can be embedded in the constant c.

Of course, (1) is usually only a small fragment of a larger problem where other constraints are imposed on u and/or p. For instance, constraints may exist that confine p in a subset of $\mathcal{P}$; extending our treatment to this case is immediate, and we will not comment further on this issue. More interestingly, many fragments of the form (1) can be simultaneously present in a given model, with different data p_{min}, p_{max}, f and c, to represent, e.g., different products and/or different productive processes and/or different time instants and/or different geographical locations. For instance, in the thermal Unit Commitment problem (cf. Section 3.1) in electrical power production, the production process is that of a thermal generating unit that burns some type of fuel (oil, gas, coal, ...). A set of generating units, with different capacities and generating costs, can be used to satisfy the (forecasted) power demand over a discretized time horizon. Hence, there are as many blocks of the form (1) as the number of generating units multiplied by the number of time instants.

When a (MIP) comprising one or more blocks of the form (1) has to be solved with an enumerative approach, such as a Branch & Bound or Branch & Cut algorithm, the continuous relaxation of (1)

$$\begin{cases} \min f(p) + cu \\ p_{min}u \leq p \leq p_{max}u \\ u \in [0,1] \end{cases} \tag{2}$$

is usually a part of the problem that is solved in order to derive lower bounds on the objective function value of the original (MIP). In a Branch & Cut algorithm, these lower bounds can be improved by adding cuts that provide a better description of the convex hull of the integer solutions; these cuts, however, depend on the structure of the other constraints, e.g., those linking different blocks (1) together, as the feasible region in (2) has only vertices with an integer value

for u. Here we focus on a different way for improving the lower bound that depends only on the structure of a single block.

Problem (1) can be equivalently seen as the minimization over all (p, u) of the nonconvex function

$$f(p,u) = \begin{cases} 0 & \text{if } p = u = 0 \\ f(p) + c & \text{if } u = 1 \text{ and } p_{min} \le p \le p_{max} \\ +\infty & \text{otherwise} \end{cases} . \tag{3}$$

The best possible convex relaxation of this problem is obtained by minimizing over all (p, u) the function $\overline{co}f$, i.e., the closed convex function with the smallest (in set–inclusion sense) epigraph containing that of f; this is called the *convex envelope* of f. In general, computing the convex envelope is a nontrivial task (see, e.g., [10], [12]); in this particular case it can be readily done, showing that $\overline{co}f(p, u)$ is nothing but a section of the *perspective function* of $f(p)$, see, e.g., [7] §IV.2.2. Using a result of [3] we can exploit this fact to derive valid inequalities for the problem, related to those of [11], which can then be used within an enumerative approach to improve the lower bound w.r.t. the one obtained by (2); this can be shown to result in a decrease of the number of the nodes in the enumeration tree and in an improvement of the running time for at least a class of relevant applications, that is, Unit Commitment problems.

The structure of the paper is the following: in Section 2 $\overline{co}f$ is characterized, some of its useful properties are discussed and the valid inequalities are described. In Section 3, the use of the valid inequalities within an enumerative approach is discussed, focusing in particular on the case where $f(p)$ is quadratic; some computational results are reported for Unit Commitment problems that show how the proposed technique can considerably improve the performances of an enumerative method. Finally, in Section 4 some conclusions are drawn.

Throughout the paper the following notation is used. Given a set X, $I_X(x) = 0$ if $x \in X$ (and $+\infty$ otherwise) is its *indicator function*, $intX$ is its *interior*, $extX$ the set of its *extreme points*, coX is the *convex hull* of X and $\overline{co}X$ is the *closure* of coX; when X is the *epigraph* of a function f, i.e., $X = epif = \{(v, x) : v \ge f(x)\}$, we denote by $cof = coX$ and $\overline{co}f = \overline{co}X$ the *convex envelope* of f and its closure, respectively. Given a convex function f, $domf = \{x : f(x) < \infty\}$ is its *domain*, $\partial f(x)$ is its *subdifferential* at x and $f'(x; d)$ is its directional derivative at x along direction d; we remind that for any finite convex function $f'(x; d) = \sup\{sd : s \in \partial f(x)\}$. We will often use the shorthand p/u for $(1/u)p$, where p is a vector and u a scalar.

2. Characterization of $\overline{co}f$

To characterize $\overline{co}f$ we just need to compute the convex hull of points in the epigraphical space pertaining to the two disconnected "sides" of $domf$. In other words, given any $\theta \in [0, 1]$ and $\bar{p} \in \mathcal{P}$, we consider the point

$$(1 - \theta)[0, 0, 0] + \theta[f(\bar{p}) + c, \bar{p}, 1] = [\theta(f(\bar{p}) + c), \theta\bar{p}, \theta] .$$

Since for $\theta = 0$ we obtain $[0, 0, 0]$, we can assume $\theta > 0$ and make the following identifications

$$\theta \equiv u \qquad\qquad p \equiv u\bar{p}$$

to obtain

$$[uf(p/u) + uc, p, u]$$

and therefore

$$h(p,u)=\overline{co}f(p,u)=\begin{cases} 0 & \text{if } p=u=0 \\ uf(p/u)+cu & \text{if } p_{min}u\le p\le p_{max}u,\ u\in(0,1] \\ +\infty & \text{otherwise} \end{cases} . \tag{4}$$

In other words, h is just (a section of) the *perspective function* of $f(p)$. $\mathcal{F}=dom\ h=co\ dom f$ is the pyramid having as base $\mathcal{P}\times\{1\}$ and vertex $[0,0]$ (when $n=1$, the triangle of vertices $[0,0]$, $[p_{min},1]$, $[p_{max},1]$). Note that the explicit definition of $h(0,0)$ in (4) is not strictly necessary as the result is obtained by continuity: for every sequence $\{p_k,u_k\}\subset\mathcal{F}$ that converges to $[0,0]$ we have

$$0\le u_k(f(p_k/u_k)+c)\le u_k\left(c+\sup_{p\in\mathcal{P}}f(p)\right)$$

and therefore

$$\lim_{k\to\infty}u_k(f(p_k/u_k)+c)=0$$

since f is finite over the compact set $\mathcal{P}$.

The set *epi* h is a (section of a) cone pointed in the origin and having as "lower shape" that of $f(p)$; in other words, h is linear on the straight lines (segments) of the form $p=\bar{p}u$ with $u\in[0,1]$ for any $\bar{p}\in\mathcal{P}$, in fact

$$h(\bar{p}u,u)=uf(\bar{p})+cu\ .$$

This is confirmed by first-order analysis: if $f(p)$ is differentiable in $\mathcal{P}$ then h is differentiable in *int* $\mathcal{F}$ and

$$\nabla h(p,u)=\begin{bmatrix} \nabla f(p/u) \\ f(p/u)-(p/u)\nabla f(p/u)+c \end{bmatrix} .$$

Since ∇h depends only on p/u it is constant on all points of the form $[\bar{p}u,u]$.

The first-order analysis can be extended to the case where $f(p)$ is nondifferentiable by characterizing ∂h; this has already been done in [3], but we include a short proof because it helps in providing geometric insight on the shape of *epi* h. Since we know that h is linear on the lines $p=\bar{p}u$ for $\bar{p}\in\mathcal{P}$, its directional derivative along the direction $[p,u]$ must be identical for all the points of the line. Hence, the scalar product between any subgradient $[s_1,s_2]\in\partial h(p,u)$ and $[p,u]$ must be identical; in particular, we have

$$[s_1,s_2][p,u]=h(p,u)=uf(p/u)+cu \qquad \forall[s_1,s_2]\in\partial h(p,u)$$

whence all the subgradients of h in (p,u) must have the form

$$[s_1,c+f(p/u)-s_1(p/u)]\ .$$

But s_1 has to be a subgradient of $h_u(p)=h(p,u)$; using Theorem VI.4.2.1 in [7] it is easy to prove that $\partial h_u(p)=\partial f(p/u)$, hence

$$\partial h(p,u)\subseteq\{g(s)=[s,c+f(p/u)-s(p/u)]:s\in\partial f(p/u)\}\ . \tag{5}$$

It is easy to verify that every vector $g(s)$ in (5) satisfies the subgradient inequality

$$h(p,u)\ge h(\bar{p},\bar{u})+[s_1,s_2]([p,u]-[\bar{p},\bar{u}]) \qquad \forall[s_1,s_2]\in\partial h(\bar{p},\bar{u})$$

therefore it is a subgradient of h at (p,u), and "$\subseteq$" can be replaced by "$=$" in (5). Note once again that $\partial h(p,u)$ depends only on p/u, and thus it is constant on the lines $p = \bar{p}u$.

It is now possible to characterize the epigraph of the function h. All points $[v,p,u] \in epi\ h$ must satisfy the subgradient inequality

$$v \geq h(\bar{p},\bar{u}) + [s_1,s_2]([p,u] - [\bar{p},\bar{u}]) \qquad \forall [s_1,s_2] \in \partial h(\bar{p},\bar{u}) \tag{6}$$

for all $[\bar{p},\bar{u}] \in \mathcal{F}$; since $\partial h(p,u)$ is constant on the lines $p = \bar{p}u$, it is sufficient to consider only points of the form $[\bar{p},1]$.

Theorem 2.1. *The $n+2$-dimensional set epi h is bounded by the following linear inequalities:*

$$p_{min}u \leq p, \quad p \leq p_{max}u, \quad u \leq 1\ ; \tag{7}$$

$$v \geq f(\bar{p}) + c + [s, c + f(\bar{p}) - s\bar{p}]([p,u] - [\bar{p},1]) \qquad s \in \partial f(p), p \in \mathcal{P}\ . \tag{8}$$

In particular, the $2n+1$ inequalities (7) define maximal faces of epi h of dimension $n+1$, while the – possibly infinitely many – inequalities (8) define maximal faces of epi h of dimension at least one.

Note that, since f is finite everywhere, $\partial f(p)$ is compact for all $p \in \mathcal{P}$; hence, from (9) we have that all the linear constraints in (6) for one $[\bar{p},1]$ corresponding to subgradients $s \notin ext\partial f(\bar{p})$ can be obtained as a convex combination of (at most $n+1$) constraints corresponding to subgradients in $ext\partial f(\bar{p})$. Hence, $\partial f(p)$ in (8) can be replaced with $ext\partial f(p)$ (of course the two coincide if f is actually differentiable in p, as $s = \nabla f(p)$ is the only possible choice in (8)). In case $n = 1$, for instance,

$$\partial f(p) = [f'_-(p), f'_+(p)]$$

where f'_- and f'_+ are respectively the left and right derivative of f, and

$$\partial h(p,u) = co\{g(f'_-(p/u)), g(f'_+(p/u))\} \tag{9}$$

is a segment in $\mathbb{R}^2$ whose extremes correspond to $f'_-(p/u)$ and $f'_+(p/u)$. Hence, (8) becomes

$$\begin{array}{l} v \geq f(\bar{p}) + c + [f'_-(\bar{p}), c + f(\bar{p}) - f'_-(\bar{p})\bar{p}]([p,u] - [\bar{p},1]) \\ v \geq f(\bar{p}) + c + [f'_+(\bar{p}), c + f(\bar{p}) - f'_+(\bar{p})\bar{p}]([p,u] - [\bar{p},1]) \end{array} \tag{10}$$

We refere to each inequality of the form (8) as a *perspective cut*.

It is interesting to contrast h for two special cases of f, namely the linear case $f(p) = bp$ and the (convex) quadratic case $f(p) = ap^2 + bp$ (with $a > 0$, and $n = 1$ for simplicity). In the former case we get

$$h(p,u) = uf(p/u) + cu + I_{\mathcal{F}} = bp + cu + I_{\mathcal{F}} = f(p) + cu + I_{\mathcal{F}}\ .$$

Hence, in the linear case the convex envelope of f is characterized, other than by the inequalities (7), by the unique perspective cut $v \geq bp + cu$ that defines a maximal face of *epi* h of dimension $n+1$. Thus, in the linear – and therefore piecewise-linear – case h coincides with the objective function of (2), which is therefore the best possible convex relaxation.

In the quadratic case, instead, we get

$$h(p,u) = uf(p/u) + cu + I_{\mathcal{F}} = (1/u)ap^2 + bp + cu + I_{\mathcal{F}}\ . \tag{11}$$

Hence, something can be gained by using h as the objective function: since $u \leq 1$, $h(p,u) \geq ap^2 + bp + cu$. In particular, elementary calculus shows that the maximum of $h(p,u) - (ap^2 + bp + cu)$ over $\mathcal{F}$ is $ap_{max}^2/4$, attained at $[p^*, u^*] = [p_{max}/2, 1/2]$. Thus, in the quadratic case ($a > 0$) h is a better objective function, for a continuous relaxation, than $f(p) + cu$. It is interesting to remark that the largest difference between the two is obtained for $u = 1/2$, i.e., that h "penalizes" precisely the "most nonintegral" points in $\mathcal{F}$, which is, at least intuitively, a positive fact.

In the quadratic case, however, using h as the objective function has a serious potential drawback: $h(p,u)$ is a much "more nonlinear" function than $ap^2 + bp + cu$ and it is even nondifferentiable at $[0,0]$. In other words, while (2) is a convex quadratic problem, and therefore it is not substantially more costly to solve than a LP, a relaxation of a MIP with many blocks of type (1) having h as a part of the objective function could be considerably more difficult to solve. The interior-point method of [3] could be used here, but efficient implementations of that approach are not widely available, and have not already been fully shown to be competitive with the sophisticated LP and QP solvers available; furthermore, interior-point methods may be less well-suited than simplex-like methods in the context of enumerative approaches. Alternatively, Theorem 2.1 suggests using a *partial characterization* of *epi* h as the objective function by iteratively collecting a finite subset of perspective cuts. This is analogous to what is done for the feasible region in Branch & Cut approaches like that of [11], and – possibly more to the point in this case – what is done for the objective function in several NonDifferentiable Optimization algorithms (see, e.g., [6]).

In other words, given a (fragment of a) solution $[v^*, p^*, u^*]$ with a fractional value for u corresponding to an approximation of h by a finite number of perspective cuts, it is very easy to compute the true value of $h(p^*, u^*)$ via (4) to check whether $v^* \approx h(p^*, u^*)$. If not, cuts (8) corresponding to $[p^*/u^*, 1]$ are (strongly) violated by $[v^*, p^*, u^*]$, and therefore can be added to the current set of inequalities, improving the approximation of h and, possibly, the lower bound. This only requires to be able to compute at least one (preferably extreme) element of $\partial f(p)$.

This approach is strongly related to that of [11], which is more general; however, a number of differences exist. First and foremost, because in our case the "convex combinator" u is a variable of the original formulation rather than being added for algorithmic purposes, separation of perspective cuts is very easy, while in the approach of [11] a potentially large-scale nonlinear problem – with nonlinear constraints – have to be solved. We separate inequalities for each block (1) individually rather than only one inequality for the entire problem, and therefore our inequalities are global by nature and do not require lifting. Even if the approach of [11] is applied to a single block (1), a smaller scale but potentially nontrivial nonlinear program still has to be solved to compute the projection of $[v^*, p^*, u^*]$ over *epi* h under some norm; because of this, the perspective cuts obtained by [11] and the ones we generate, for the same $[v^*, p^*, u^*]$, would be different. Therefore, our approach is clearly simpler (even though not entirely immediate, as shown in the next section) to implement using widely available and efficient tools with respect to the approach proposed in [11].

3. Using perspective cuts in a B&B approach

In this section we will show that, despite of the low dimensionality of the faces of *epi* $\overline{co} f$ that they represent, perspective cuts can significantly improve the performances of an enumerative approach for the solution of a (MIP) containing fragments of the form (1). Let us remark that these results are only meant to show that the proposed convexification procedure can significantly

improve the lower bound and the running time of a B&B algorithm with respect to a standard continuous relaxation.

3.1. The Unit Commitment problem

The thermal Unit Commitment problem in electrical power production is as follows. A set I of thermal units is given, where each unit $i \in I$ is characterized by a maximum and minimum power output, p^i_{min} and p^i_{max}, respectively, and by a convex quadratic power cost function $f^i(p) = a^i p^2 + b^i p + c^i$, when the unit is committed (actively generating power). Over a discretized set T of time instants covering some time horizon (e.g., hours or half-hours in a day or a week), an estimate d_t for $t \in T$ of the total power demand is available. The Unit Commitment problem requires to generate, for each time period, enough power to meet the forecasted demand at minimal total cost. The operation of thermal units must satisfy a number of technical constraints: the most standard requirement is that whenever unit i is turned on it must remain committed for at least τ^i_u consecutive time instants, and, analogously, whenever unit i is turned off it must remain decommitted for at least τ^i_d consecutive time instants.

Introducing binary variables u_{it} indicating the commitment of unit i at time instant t and p_{it} indicating the corresponding power output, a basic formulation of the Unit Commitment problem is

$$\min \sum_{i \in I} \sum_{t \in T} a^i p^2_{it} + b^i p_{it} + c^i u_{it} \tag{12}$$

$$\sum_{i \in I} p_{it} = d_t \qquad t \in T \tag{13}$$

$$u_{i,t+r} \geq u_{i,t} - u_{i,t-1} \qquad i \in I, t \in T, r = 1, \ldots, \min\{\tau^i_u, |T| - t\} \tag{14}$$

$$u_{i,t+r} \leq 1 - u_{i,t-1} + u_{i,t} \qquad i \in I, t \in T, r = 1, \ldots, \min\{\tau^i_d, |T| - t\} \tag{15}$$

$$p^i_{min} u_{it} \leq p_{it} \leq p^i_{max} u_{it} \qquad i \in I, t \in T \tag{16}$$

$$u_{it} \in \{0, 1\} \qquad i \in I, t \in T \tag{17}$$

This basic formulation may have to be complicated in order to take into account other characteristics of a real energy production problem. For instance, other technical constraints are often imposed by either global security requirements, such as *spinning reserve* constraints or *network* constraints, or by operating restrictions of the thermal units, such as *ramp rate* constraints. Often, thermal units incur in a considerable start-up cost when they initiate production after a period of inactivity, and the cost may depend on the duration of the inactive period. Furthermore, other types of generating units, such as hydro units or nuclear units, with very different operating constraints can be used to satisfy part of the power demand in (13). It is out of the scope of this paper to provide a detailed description of all variants of the Unit Commitment problem; the interested reader is referred, e.g., to [2] and the references therein. In the following, the formulation (12) – (17) will be assumed; it already contains $|I| \times |T|$ blocks of the form (1), which would therefore be contained in any more complex formulation.

3.2. Implementation details

In order to test the effectiveness of adding perspective cuts, we implemented a cutting plane scheme on the objective function within a Branch & Bound algorithm for solving Unit Commitment problems. The implementation was not entirely straightforward: in fact, although

`Cplex 8.0` [8] provides a Branch & Cut algorithm for solving Mixed-Integer (convex) Quadratic Programs, some of the – rather nonstandard – operations required for implementing our scheme are not supported by the – otherwise very flexible – B&C algorithm contained in the `Cplex 8.0` library. In particular, changing the quadratic part of the objective function during the execution of the B&C is not allowed by the API of the Cplex callable libraries; furthermore, when an integer solution is found its objective function value for the "linearized" relaxation does not provide a valid upper bound, that can be obtained only by evaluating the original quadratic objective function. All this prevented us from relying directly on the efficient and sophisticated B&C implementation in Cplex, using the many *callback functions* provided for tailoring it to our specific application, and forced us to implement a standard B&B algorithm from scratch, using Cplex only to solve the Quadratic Programs at each node of the B&B tree. The most relevant details of our implementations are briefly described in the following.

The first Quadratic Program solved at the root node is just the continuous relaxation of (12) – (17). At each node of the B&B tree, a standard cut separation phase can be executed: given the current solution (p^*, u^*) of the relaxation, consider any pair (i,t) such that u^*_{it} is not integer, compute the (unique) perspective cut

$$v \geq \left(2a\frac{p^*}{u^*} + b\right) p + \left(c - a\left(\frac{p^*}{u^*}\right)^2\right) u \tag{18}$$

associated with the point $(\bar{p}_{it} = p^*_{it}/u^*_{it}, 1)$ and add it to the formulation if it is violated (beyond a certain threshold).

The first time that a cut is generated for a given pair (i,t), the corresponding fragment of the quadratic function must be removed from the formulation, and the extra variable v_{it} must be added to the model to represent the maximum between all the linear functions associated with the pair (i,t) (cf. (18)); in this case we also add to the formulation the cuts associated with the two special points $(p^i_{min}, 1)$ and $(p^i_{max}, 1)$.

The cut separation routine have to distinguish whether or not the objective function corresponding to the pair (i,t) has already been linearized; in the latter case we set $v^*_{it} = a^i(p^*_{it})^2 + b^i p^*_{it} + c^i u^*_{it}$, while in the former case we read the value v^*_{it} from the solution of the continuous relaxation. Then, the cut is violated if the following condition holds:

$$v^*_{it} < \left(2a^i\bar{p}_{it} + b^i\right) p^*_{it} + \left(c^i - a^i\left(\bar{p}_{it}\right)^2\right) u^*_{it} \ - \epsilon = a^i\frac{\left(p^*_{it}\right)^2}{u^*_{it}} + b^i p^*_{it} + c^i u^*_{it} \ - \epsilon,$$

where ϵ is a fixed threshold (10^{-4} in our experiments).

The above separation procedure can be used in a number of different ways within the Branch & Bound algorithm; in particular, the following three binary choices give rise to $2^3 = 8$ alternative strategies:

- the separation procedure is applied only at the root node (CUT = R) or throughout the Branch & Bound tree (CUT = T);
- the original fragment of the quadratic function is restored when for a "linearized" pair (p_{it}, u_{it}) the variable u_{it} is fixed to 1 in a branching (DELIN = D), or the current piecewise-linear approximation of h is kept even if u_{it} is fixed to 1(DELIN = L);
- the separation procedure is applied only once for each node of the B&B tree (SINGLE = S), or the procedure is repeatedly applied, after reoptimization of the QP, until no new cuts are found (SINGLE = M, for "multiple iterations").

In the tables, we will indicate each strategy with the corresponding triple CUT DELIN SINGLE; so, for instance, the code TDM will indicate the B&B variant where new cuts are generated at each node (T), the quadratic part of the objective function is restored for all u_{it} variables that are fixed to 1 (D), and more than one round of cut generation may be performed at each node (M).

We also tested our B&B algorithm without adding any perspective cut (CUT = 0, denoted by code 000), and, for comparison, the efficient B&C implementation of Cplex 8.0, again without any other cut apart the standard ones provided by the Cplex library. All the implementations therefore use the quadratic solver in CPLEX 8.0; actually, since two quadratic solvers are available we experimented with both.

Since we were mainly interested on the effect of perspective cuts on the lower bound computation we did not include any heuristic in the B&B algorithm; instead, we provided each variant an initial upper bound associated with a good feasible solution obtained by the Lagrangean heuristic in [2].

3.3. Computational results

We tested all the above variants of B&B algorithms on a set of Unit Commitment instances obtained as in [2]; the instances have 24 time periods and either 10 or 20 units (5 instances for each dimension). The stopping criterion for all variants was either a relative gap lower than 0.1% or the maximum number of nodes (10000) being reached.

All variants were run on a PC with a 2.5 Ghz Pentium-4 processor and 512Mb RAM, running the Linux Debian 3.0 operating system (kernel 2.4.18). The codes were compiled with `gcc 2.95.4` using aggressive optimizations `-O3`.

In Table 1 and Table 2 we report the results for each of the nine strategies obtained while using the barrier algorithm and the dual algorithm, respectively, for solving the QP at each node of the B&B tree. For each instance we report the number of nodes in the B&B tree (n) and the time in seconds (s) needed to solve the problem; if the required 0.1% accuracy is not reached within 10000 nodes (this often occurs for CUT = 0, and sometimes even with CUT = R), we also report the gap of the current best integer solution with the current best bound.

All the eight strategies using perspective cuts clearly outperform the pure B&B algorithm. This is also explained by comparing the previous results with those in Table 3, where the time required to obtain the lower bound at the root node (possibly solving more than one QP for the variants with SINGLE = M) and the corresponding gaps are reported for the barrier and dual simplex algorithm, respectively. Even one single pass of the cut separation routine reduces the gap by almost a factor of 10 at a relatively low computational cost; comparing these results with Table 1 and Table 2 shows that in most cases the gap reached by strategy 000 after 10000 nodes is still larger than the one reached by the other strategies at the root node.

As far as the comparison among the eight strategies is concerned, the following trends seem to emerge from the results:

- CUT = T clearly outperforms CUT = R, i.e., "Branch & Cut" is better than "Cut & Branch" in this case;
- the strategies TLx are usually better than the corresponding strategies TDx, i.e., delinearization seems to hurt performances; this is particularly true when the dual simplex algorithm is used, and it is probably due to a less efficient reoptimization when quadratic terms are created in the objective function;

instance	000			TDM			TDS			TLM			TLS		
	n	t	g%	n	t	g%	n	t	g%	n	t	g%	n	t	g%
P10_24_a	2183	537	-	6	10	-	4	2	-	6	11	-	4	2	-
P10_24_b	4214	1202	-	13	36	-	24	21	-	16	36	-	24	16	-
P10_24_c	10000	2925	0.55	20	68	-	68	58	-	24	79	-	70	62	-
P10_24_d	10000	2390	0.49	25	17	-	16	5	-	4	4	-	6	2	-
P10_24_e	10000	2287	0.43	0	1	-	1	0	-	0	1	-	1	0	-
average	-	-		13	26		23	17		10	26		21	16	
P20_24_a	10000	6145	1.69	160	886	-	193	530	-	56	349	-	201	586	-
P20_24_b	1891	1260	-	52	110	-	68	95	-	36	77	-	75	109	-
P20_24_c	1852	1213	-	157	617	-	165	319	-	147	160	-	160	320	-
P20_24_d	10000	6040	1.93	157	950	-	335	1541	-	164	1007	-	390	1925	-
P20_24_e	10000	6551	1.13	35	67	-	67	167	-	35	247	-	53	123	-
average	-	-		112	526		166	530		88	368		176	613	

instance		RDM			RDS			RLM			RLS		
		n	t	g%	n	t	g%	n	t	g%	n	t	g%
P10_24_a		6	3	-	7	2	-	6	3	-	6	2	-
P10_24_b		29	12	-	52	19	-	38	15	-	35	13	-
P10_24_c		266	105	-	1335	456	-	68	32	-	64	25	-
P10_24_d		25	8	-	385	108	-	2	2	-	35	11	-
P10_24_e		0	1	-	5	1	-	0	1	-	1	0	-
average		65	26		357	117		23	11		28	10	
P20_24_a		10000	10133	0.13	10000	9190	0.19	10000	10464	0.16	10000	9290	0.22
P20_24_b		63	57	-	115	95	-	80	70	-	141	120	-
P20_24_c		519	384	-	505	369	-	488	376	-	476	351	-
P20_24_d		10000	10631	0.11	10000	10631	0.16	10000	11227	0.14	10000	10137	0.16
P20_24_e		698	709	-	698	709	-	1032	1076	-	5354	5052	-
average		-	-		-	-		-	-		-	-	

Table 1: Results with the barrier algorithm

instance	000			TDM			TDS			TLM			TLS		
	n	t	g%	n	t	g%	n	t	g%	n	t	g%	n	t	g%
P10_24_a	3001	559	-	5	7	-	11	11	-	5	6	-	7	4	-
P10_24_b	4172	1353	-	79	158	-	39	50	-	79	59	-	68	43	-
P10_24_c	10000	2846	0.55	39	156	-	73	207	-	38	31	-	45	19	-
P10_24_d	10000	1781	0.43	3	3	-	4	2	-	2	1	-	3	1	-
P10_24_e	10000	1258	0.44	0	1	-	1	2	-	0	1	-	1	2	-
average	-	-		25	65		26	54		25	20		25	14	
P20_24_a	10000	7137	1.68	116	1393	-	195	1804	-	160	413	-	162	216	-
P20_24_b	2123	1292	-	22	128	-	43	256	-	37	27	-	25	23	-
P20_24_c	2988	4007	-	144	358	-	147	356	-	140	220	-	146	193	-
P20_24_d	10000	6129	1.94	179	3867	-	440	9576	-	191	353	-	336	520	-
P20_24_e	10000	7877	1.14	37	680	-	29	423	-	17	73	-	39	85	-
average	-	-		100	1285		171	2483		109	217		142	207	

instance		RDM			RDS			RLM			RLS		
		n	t	g%	n	t	g%	n	t	g%	n	t	g%
P10_24_a		20	18	-	12	8	-	7	1	-	12	2	-
P10_24_b		84	75	-	49	47	-	84	21	-	82	22	-
P10_24_c		150	285	-	667	1042	-	60	9	-	75	7	-
P10_24_d		3	2	-	374	277	-	2	1	-	31	3	-
P10_24_e		0	1	-	3	2	-	0	1	-	3	0	-
average		51	76		221	275		31	7		41	7	
P20_24_a		10000	52411	0.13	10000	51277	0.20	10000	10000	0.16	10000	7661	0.22
P20_24_b		39	210	-	47	246	-	45	16	-	64	18	-
P20_24_c		481	858	-	503	824	-	492	829	-	538	830	-
P20_24_d		10000	122579	0.12	10000	124517	0.16	10000	8640	0.15	10000	7671	0.18
P20_24_e		923	8320	-	3213	26640	-	1248	1277	-	7372	7613	-
average		-	-		-	-		-	-		-	-	

Table 2: Results with the dual simplex algorithm

instance	with Barrier Algorithm						with Dual Simplex Algorithm					
	000		xxM		xxS		000		xxM		xxS	
	g%	t	g%	t	g%	t	g%	t	g%	t	g%	t
P10_24_a	1.5202	0.10	0.3993	1.40	0.4599	0.22	1.5202	0.29	0.3993	1.22	0.4599	0.66
P10_24_b	1.2159	0.10	0.2969	1.86	0.3157	0.25	1.2159	0.32	0.2888	1.65	0.3066	0.96
P10_24_c	1.4372	0.11	0.2650	1.96	0.3109	0.24	1.4372	0.47	0.2645	4.13	0.3105	1.50
P10_24_d	1.2374	0.08	0.2410	2.45	0.3769	0.20	1.2374	0.26	0.2544	1.61	0.3769	0.82
P10_24_e	0.9459	0.08	0.0871	1.28	0.1312	0.22	0.9459	0.28	0.0866	1.51	0.1273	0.79
P20_24_a	2.3640	0.22	0.2338	3.55	0.2812	0.51	2.3640	1.45	0.2338	11.16	0.2812	4.27
P20_24_b	0.4851	0.22	0.1321	2.80	0.1343	0.50	0.4851	1.48	0.1320	4.62	0.1343	4.44
P20_24_c	1.2421	0.21	0.6927	0.98	0.6933	0.48	1.2421	0.88	0.6927	3.51	0.6933	2.11
P20_24_d	2.3984	0.20	0.2489	4.97	0.2843	0.49	2.3984	1.68	0.2594	23.65	0.2843	8.61
P20_24_e	1.6305	0.23	0.1831	3.38	0.1979	0.55	1.6305	1.70	0.1831	20.60	0.1979	8.45

Table 3: Root node: gap and solution times

- strategy TLS is usually faster than strategy TLM (although some exceptions exist), probably because successive separations at the same node add only cuts that can be also easily obtained, if necessary, in the children of the node.
- on strategy TLS, the dual simplex algorithm has, on average, better performances than the barrier algorithm; this shows that reoptimization is crucial in a B&B framework, suggesting that an interior-point method such as that of [3] may not be the most efficient in this setting.

We remark that these results are far from being conclusive, but they were not meant to; we were interested only in proving that a B&C approach can significantly improve when perspective cuts are used.

A more extensive comparison of the eight strategies for the Unit Commitment problem would require a more detailed study within a well structured Branch & Cut algorithm including also cuts for tightening the constraints of the formulation in the standard sense, and possibly even a comparison with different approaches; this will be the subject of a future work.

Finally, in Table 4 we report the results obtained by B&C algorithm of CPLEX 8.0 on our instances, with a time limit of 10000 seconds. While the MIQP solver in Cplex clearly outperforms our run-of-the-mill B&B implementation with CUT = 0, it is the latter which outperforms the former with CUT = T. This seems to indicate that a more sophisticated implementation of a B&C algorithm using perspective cuts could be remarkably efficient for solving Unit Commitment problems, and probably also some of the many other problems which exhibit (1) as substructure.

instance	n	t	g%
P10_24_a	809	18	-
P10_24_b	1024	27	-
P10_24_c	73349	1048	-
P10_24_d	25075	240	-
P10_24_e	85934	666	-
P20_24_a	264179	10000	1.3391
P20_24_b	1205	73	-
P20_24_c	4083	258	-
P20_24_d	331732	10000	1.5010
P20_24_e	245582	10001	0.8741

Table 4: Results obtained with the CPLEX 8.0 MIQP solver

4. Conclusion

Starting by the observation that the convex envelope of a family of nonconvex functions that appear as a fragment of the objective function of many important Mixed-Integer programs is the well-known perspective function, we have proposed a family of perspective cuts for those problems. We have also shown that, despite of the low dimensionality of the faces that they represent, these cuts can be used, at least in one relevant case, to significanlty improve the efficiency of enumerative approaches to the corresponding Mixed Integer Programs.

The proposed cuts are very easy to separate; in particular, they do not require the solution of a nonlinear program – with nonlinear constraints – as in [11]. Yet, integrating them in a

B&B approach requires a partial linearization of the objective function that is not completely straightforward to implement using widely available optimization tools such as Cplex 8.0. Still, the results seem to suggest that this technique may be valuable for a large class of optimization problems; should this be confirmed by further experiments, some better support for this type of approach could be expected in general tools in the future. Even at the current state of technology, however, the improvements in the lower bounds provided by these cuts may largely overbalance the extra computational burden required for handling them.

One interesting issue that still has to be explored is whether other widespread structures in Mixed Integer Programs are amenable of analogous treatments, i.e., whether closed-form convexification formulae can be used for other interesting fragments as well, avoiding the general but potentially costly approach of [11]; we plan to investigate this issue in the future.

References

[1] S. Ahn, L. Escudero, and M. Guignard-Spielberg, "On Modeling Robust Policies for Financial Trading," in *Optimization in Industry 2* (T. Ciriani and R. Leachman, eds.), pp. 163–184, Wiley, Chichester, 1994.

[2] A. Borghetti, A. Frangioni, F. Lacalandra, and C. Nucci, "Lagrangian Heuristics Based on Disaggregated Bundle Methods for Hydrothermal Unit Commitment," *IEEE Trans. on Power Systems*, vol. 18(1), pp. 1–10, 2003.

[3] S. Ceria and J. Soares, "Convex programming for disjunctive convex optimization," *Mathematical Programming*, vol. 86, pp. 595–614, 1999.

[4] Dash Associates Limited, *XPRESS-MP Reference Manual*, 2003.

[5] M. Duran and I. Grossmann, "An Outer-Approximation Algorithm for a Class of Mixed-Integer Nonlinear Programs," *Mathematical Programming*, vol. 36, pp. 307–339, 1986.

[6] A. Frangioni, "Generalized Bundle Methods," *SIAM Journal on Optimization*, vol. 13(1), pp. 117–156, 2002.

[7] J.-B. Hiriart-Urruty and C. Lemaréchal, *Convex Analysis and Minimization Algorithms* I—*Fundamentals*. Grundlehren Math. Wiss. 305, Springer-Verlag, New York, 1993.

[8] ILOG, *ILOG CPLEX 8.0 User Manual*, 2002.

[9] J. Kallrath and J. Wilson, *Business Optimization*. Macmillan Press Ltd., Houndmills, 1997.

[10] A. Rikun, "A Convex Envelope Formula for Multilinear Functions," *J. of Global Optim.*, vol. 10, pp. 425–437, 1997.

[11] R. Stubbs and S. Mehrotra, "A branch-and-cut method for 0-1 mixed convex programming," *Mathematical Programming*, vol. 86, pp. 515–532, 1999.

[12] M. Tawarmalani and N. Sahinidis, "Convex extensions and envelopes of lower semi-continuous functions," *Mathematical Programming*, vol. 93, pp. 515–532, 2002.

[13] J. Zamora and I. Grossmann, "A Global MINLP Optimization Algorithm for the Synthesis of Heat Exchanger Networks with no Stream Splits," *Comput & Chem. Engin.*, vol. 22, pp. 367–384, 1998.