Indicator Constraints in Mixed-Integer Programming

Pietro Belotti[1] **Andrea Lodi**[2]
Amaya Nogales-Gómez[3]

[1] FICO, UK

[2] University of Bologna, Italy - `andrea.lodi@unibo.it`

[3] Universidad de Sevilla, Spain

18th CO Workshop @ Aussois, January 7, 2014

Indicator (bigM's) constraints

- We consider the linear inequality

$$a^T x \leq a_0, \tag{1}$$

where $x \in \mathbb{R}^k$ and $(a, a_0) \in \mathbb{R}^{k+1}$ are constant.

Indicator (bigM's) constraints

- We consider the linear inequality

$$a^T x \leq a_0, \tag{1}$$

where $x \in \mathbb{R}^k$ and $(a, a_0) \in \mathbb{R}^{k+1}$ are constant.

- It is a well-known modeling trick in Mixed-Integer Linear Programming (MILP) to use a binary variable y multiplied by a sufficiently big (non-negative) constant M in order to deactivate constraint (1)

$$a^T x \leq a_0 + My. \tag{2}$$

Indicator (bigM's) constraints

- We consider the linear inequality

$$a^T x \leq a_0, \tag{1}$$

where $x \in \mathbb{R}^k$ and $(a, a_0) \in \mathbb{R}^{k+1}$ are constant.

- It is a well-known modeling trick in Mixed-Integer Linear Programming (MILP) to use a binary variable y multiplied by a sufficiently big (non-negative) constant M in order to deactivate constraint (1)

$$a^T x \leq a_0 + My. \tag{2}$$

- It is also well known the risk of such a modeling trick, namely
  - weak Linear Programming (LP) relaxations, and
  - numerical issues.

Complementarity Reformulation

- An alternative for logical implications and general deactivations is given by the *complementary* reformulation

$$(a^T x - a_0)\bar{y} \leq 0, \tag{3}$$

where $\bar{y} = 1 - y$ and has been used for decades in the Mixed-Integer Nonlinear Programming literature (MINLP).

Complementarity Reformulation

- An alternative for logical implications and general deactivations is given by the *complementary* reformulation

$$(a^T x - a_0)\bar{y} \leq 0, \tag{3}$$

where $\bar{y} = 1 - y$ and has been used for decades in the Mixed-Integer Nonlinear Programming literature (MINLP).

- The obvious drawback of the above reformulation is its *nonconvexity*.
- Thus, the complementary reformulation has been used so far if (and only if) the problem at hand was already nonconvex, as it is often the case, for example, in Chemical Engineering applications.

Our goal

- In this talk we argue against this common rule of always pursuing a linear reformulation for logical implications.

Our goal

- In this talk we argue against this common rule of always pursuing a linear reformulation for logical implications.

- We do that by exposing a class of Mixed-Integer Convex Quadratic Programming (MIQP) problems arising in *Supervised Classification* where the Global Optimization (GO) solver `Couenne` using reformulation (3) is consistently faster than virtually any state-of-the-art commercial MIQP solver like `IBM-Cplex`, `Gurobi` and `Xpress`.

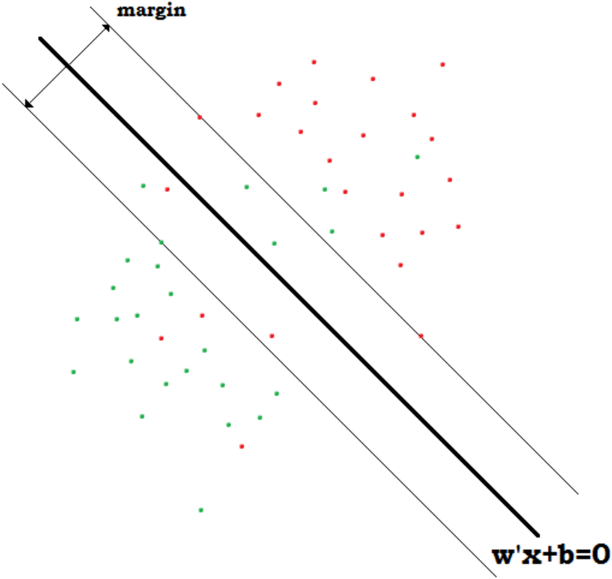
Our goal

- In this talk we argue against this common rule of always pursuing a linear reformulation for logical implications.

- We do that by exposing a class of Mixed-Integer Convex Quadratic Programming (MIQP) problems arising in *Supervised Classification* where the Global Optimization (GO) solver `Couenne` using reformulation (3) is consistently faster than virtually any state-of-the-art commercial MIQP solver like `IBM-Cplex`, `Gurobi` and `Xpress`.

- This is quite counter-intuitive because, in general, convex MIQPs admit more efficient solution techniques both in theory and in practice, especially by benefiting of virtually all machinery of MILP solvers.

Support Vector Machine (SVM)



The input data

- Ω: the population.
- Population is partitioned into two classes, $\{-1, +1\}$.
- For each object in Ω, we have
  - $x = (x^1, \ldots, x^d) \in X \subset \mathbb{R}^d$: predictor variables.
  - $y \in \{-1, +1\}$: class membership.

- The goal is to find a hyperplane $\omega^\top x + b = 0$ that aims at separating, if possible, the two classes.
- Future objects will be classified as

$$\begin{aligned} y = +1 \quad &\text{if} \quad \omega^\top x + b > 0 \\ y = -1 \quad &\text{if} \quad \omega^\top x + b < 0 \end{aligned} \tag{4}$$

Soft-margin approach

$$\min \frac{\omega^\top \omega}{2} + \sum_{i=1}^{n} g(\xi_i)$$

subject to

$$\begin{array}{rcll} y_i(\omega^\top x_i + b) & \geq & 1 - \xi_i & i = 1, \ldots, n \\ \xi_i & \geq & 0 & i = 1, \ldots, n \\ \omega \in \mathbb{R}^d, b \in \mathbb{R} & & & \end{array}$$

where n is the size of the sample and $g(\xi_i) = \frac{C}{n}\xi_i$ the most popular choice for the loss function.

Ramp Loss Model (Brooks, *OR*, 2011)

- Ramp Loss Function $g(t) = (\min\{t, 2\})^+$ yielding the Ψ-learning approach, with $(a)^+ = \max\{a, 0\}$.

Ramp Loss Model (Brooks, *OR*, 2011)

- Ramp Loss Function $g(t) = (\min\{t, 2\})^+$ yielding the Ψ-learning approach, with $(a)^+ = \max\{a, 0\}$.

$$\min \frac{\omega^\top \omega}{2} + \frac{C}{n}(\sum_{i=1}^{n} \xi_i + 2\sum_{i=1}^{n} z_i)$$

s.t. (RLM)

$$\begin{aligned} y_i(\omega^\top x_i + b) &\geq 1 - \xi_i - M z_i & \forall i = 1, \ldots, n \\ 0 \leq \xi_i &\leq 2 & \forall i = 1, \ldots, n \\ z &\in \{0,1\}^n & \\ \omega \in \mathbb{R}^d, b &\in \mathbb{R} & \end{aligned}$$

with $M > 0$ big enough constant.

Expectations (and Troubles)

- In principle, RLM is a tractable Mixed-Integer Convex Quadratic Problem that nowadays commercial (and even some noncommercial) solvers should be able to solve:
    - convex objective function,
    - linear constraints, and
    - binary variables,

  not much more difficult than a standard Mixed-Integer *Linear* Problem.

Expectations (and Troubles)

- In principle, RLM is a tractable Mixed-Integer Convex Quadratic Problem that nowadays commercial (and even some noncommercial) solvers should be able to solve:
  - convex objective function,
  - linear constraints, and
  - binary variables,

  not much more difficult than a standard Mixed-Integer *Linear* Problem.
- However, the bigM constraints in the above model destroy the chances of the solver to consistently succeed for $n > 50$.

Solving the MIQP by `IBM-Cplex`

- 23 instances from Brooks, `Type B`, $n = 100$, time limit of 3,600 CPU seconds.

Solving the MIQP by IBM-Cplex

- 23 instances from Brooks, Type B, $n = 100$, time limit of 3,600 CPU seconds.

IBM-Cplex			
		% gap	
time (sec.)	nodes	ub	lb
3,438.49	16,142,440	–	–
tl	12,841,549	–	23.61
tl	20,070,294	–	37.82
tl	20,809,936	–	9.37
tl	17,105,372	–	26.17
tl	13,865,833	–	22.67
tl	14,619,065	–	21.40
tl	13,347,313	–	14.59
tl	12,257,994	–	22.22
tl	13,054,400	–	23.13
tl	14,805,943	–	12.37
tl	12,777,936	–	21.97
tl	14,075,300	–	23.32
tl	13,994,099	–	12.48
tl	10,671,225	–	23.08
tl	12,984,857	–	22.72
tl	12,564,000	–	14.11
tl	11,217,844	–	23.45
tl	12,854,704	–	22.72
tl	14,018,831	–	12.43
tl	11,727,308	–	23.55
tl	15,482,162	–	18.67
tl	12,258,164	–	14.88

Reformulating by Complementarity

$$\min \frac{\omega^\top \omega}{2} + \frac{C}{n}\left(\sum_{i=1}^{n} \xi_i + 2\sum_{i=1}^{n}(1-\bar{z}_i)\right)$$

$$\begin{aligned}
(y_i(\omega^\top x_i + b) - 1 + \xi_i)\cdot\bar{z}_i \geq 0 & \qquad \forall i = 1,\dots,n \\
0 \leq \xi_i \leq 2 & \qquad \forall i = 1,\dots,n \\
\bar{z} \in & \; \{0,1\}^n \\
\omega \in & \; \mathbb{R}^d \\
b \in & \; \mathbb{R},
\end{aligned}$$

- where $\bar{z}_i = 1 - z_i$, and
- the resulting model is a Mixed-Integer Nonconvex Quadratically Constrained Problem (MIQCP) that `IBM-Cplex`, like all other commercial solvers initially developed for MILP, cannot solve (yet).

Solving the MIQCP by Couenne

- Despite the nonconvexity of the above MIQCP, there are several options to run the new model as it is and one of them is the open-source solver Couenne belonging to the Coin-OR arsenal.

Solving the MIQCP by Couenne

- Despite the nonconvexity of the above MIQCP, there are several options to run the new model as it is and one of them is the open-source solver Couenne belonging to the Coin-OR arsenal.

Couenne			
		% gap	
time (sec.)	nodes	ub	lb
163.61	17,131	–	–
1,475.68	181,200	–	–
tl	610,069	14.96	15.38
160.85	25,946	–	–
717.20	131,878	–	–
1,855.16	221,618	–	–
482.19	56,710	–	–
491.26	55,292	–	–
1,819.42	216,831	–	–
807.95	89,894	–	–
536.40	62,291	–	–
1,618.79	196,711	–	–
630.18	83,676	–	–
533.77	65,219	–	–
2,007.62	211,157	–	–
641.05	72,617	–	–
728.93	73,142	–	–
1,784.93	193,286	–	–
752.50	84,538	–	–
412.16	48,847	–	–
2,012.62	223,702	–	–
768.73	104,773	–	–
706.39	70,941	–	–

What does Couenne do?

- Although,
  - Convex MIQP should be much easier than nonconvex MIQCP, and
  - `IBM-Cplex` is by far more sophisticated than `Couenne`

one can still argue that a comparison in performance between two different solution methods and computer codes is anyway hard to perform.

What does Couenne do?

- Although,
  - Convex MIQP should be much easier than nonconvex MIQCP, and
  - IBM-Cplex is by far more sophisticated than Couenne

  one can still argue that a comparison in performance between two different solution methods and computer codes is anyway hard to perform.

- However, the reported results are rather surprising, especially if one digs into the way in which Couenne solves the problem, namely considering three aspects:
  1. McCormick Linearization,
  2. Branching, and
  3. alternative L_1 norm.

McCormick Linearization

- The most crucial observation is that the complementarity constraints are internally reformulated by `Couenne` through the classical McCormick linearization
  1. $\vartheta_i = y_i(\omega^\top x_i + b) - 1 + \xi_i$, with $\vartheta_i^L \leq \vartheta_i \leq \vartheta_i^U$, and
  2. $u_i = \vartheta_i \bar{z}_i$

  for $i = 1, \dots, n$.

McCormick Linearization

- The most crucial observation is that the complementarity constraints are internally reformulated by `Couenne` through the classical McCormick linearization
  1. $\vartheta_i = y_i(\omega^\top x_i + b) - 1 + \xi_i$, with $\vartheta_i^L \leq \vartheta_i \leq \vartheta_i^U$, and
  2. $u_i = \vartheta_i \bar{z}_i$

  for $i = 1, \ldots, n$. Then, the product corresponding to each new variable u_i is linearized as

$$u_i \geq 0 \tag{5}$$

$$u_i \geq \vartheta_i^L \, \bar{z}_i \tag{6}$$

$$u_i \geq \vartheta_i + \vartheta_i^U \, \bar{z}_i - \vartheta_i^U \tag{7}$$

$$u_i \leq \vartheta_i + \vartheta_i^L \, \bar{z}_i - \vartheta_i^L \tag{8}$$

$$u_i \leq \vartheta_i^U \, \bar{z}_i \tag{9}$$

  again for $i = 1, \ldots, n$, where (5) are precisely the complementarity constraints and ϑ_i^L plays the role of the bigM.

Branching

- It is well known that a major component of GO solvers is the iterative tightening of the convex (most of the time linear) relaxation of the nonconvex feasible region by branching on continuous variables.

Branching

- It is well known that a major component of GO solvers is the iterative tightening of the convex (most of the time linear) relaxation of the nonconvex feasible region by branching on continuous variables.
- However, the default version of Couenne does not take advantage of this possibility and branches (first) on binary variables z's.
- Thus, again it is surprising that such a branching strategy leads to an improvement over the sophisticated branching framework of IBM-Cplex.

Alternative L_1 norm

- A natural question is if the reported results are due to the somehow less sophisticated evolution of MILP solvers in their MIQP extensions with respect to the MILP one.

Alternative L_1 norm

- A natural question is if the reported results are due to the somehow less sophisticated evolution of MILP solvers in their MIQP extensions with respect to the MILP one.

- In order to answer this question we performed an experiment in which the quadratic part of the objective function has been replaced by its L_1 norm making the entire bigM model linear. Ultimately, the absolute value of ω is minimized.

Alternative L_1 norm

- A natural question is if the reported results are due to the somehow less sophisticated evolution of MILP solvers in their MIQP extensions with respect to the MILP one.
- In order to answer this question we performed an experiment in which the quadratic part of the objective function has been replaced by its L_1 norm making the entire bigM model linear. Ultimately, the absolute value of ω is minimized.
- Computationally, this has no effect and Couenne continues to consistently outperform MILP solvers on this very special (modified) class of problems.

Tightening ω's based on the objective function

- Bound reduction is a crucial tool in MINLP: it allows one to eliminate portions of the feasible set while guaranteeing that at least one optimal solution is retained.

Tightening ω's based on the objective function

- Bound reduction is a crucial tool in MINLP: it allows one to eliminate portions of the feasible set while guaranteeing that at least one optimal solution is retained.
- Among those reductions, we observed that Couenne does a very simple bound tightening (at the root node) based on the computation of an upper bound, i.e., a feasible solution, of value, say U,

$$\omega_i \in \left[-\sqrt{2U}, \sqrt{2U}\right] \qquad \forall i = 1, \dots, d.$$

Tightening ω's based on the objective function

- Bound reduction is a crucial tool in MINLP: it allows one to eliminate portions of the feasible set while guaranteeing that at least one optimal solution is retained.
- Among those reductions, we observed that Couenne does a very simple bound tightening (at the root node) based on the computation of an upper bound, i.e., a feasible solution, of value, say U,

$$\omega_i \in \left[-\sqrt{2U}, \sqrt{2U}\right] \qquad \forall i = 1, \ldots, d.$$

- We did implement this simple bound tightening in IBM-Cplex and it is already very effective by triggering further propagation on binary variables (i.e., fixings) but only if the initial bigM values are tight enough.

Tightening ω's based on the objective function

- Bound reduction is a crucial tool in MINLP: it allows one to eliminate portions of the feasible set while guaranteeing that at least one optimal solution is retained.
- Among those reductions, we observed that `Couenne` does a very simple bound tightening (at the root node) based on the computation of an upper bound, i.e., a feasible solution, of value, say U,

$$\omega_i \in \left[-\sqrt{2U}, \sqrt{2U}\right] \qquad \forall i = 1, \ldots, d.$$

- We did implement this simple bound tightening in `IBM-Cplex` and it is already very effective by triggering further propagation on binary variables (i.e., fixings) but only if the initial bigM values are tight enough.
- In other words, when the bigM values are large it is very hard to solve the problem without changing them during search.

Much more sophisticated propagation

- It has to be noted that Couenne internal bigM values (namely ϑ_i^L) are much more conservative (and safe) than those used in the SVM literature.

Much more sophisticated propagation

- It has to be noted that `Couenne` internal bigM values (namely ϑ_i^L) are much more conservative (and safe) than those used in the SVM literature.
- Nevertheless, the sophisticated bound reduction loop implemented by GO solvers does the job.

Much more sophisticated propagation

- It has to be noted that Couenne internal bigM values (namely ϑ_i^L) are much more conservative (and safe) than those used in the SVM literature.
- Nevertheless, the sophisticated bound reduction loop implemented by GO solvers does the job. Iteratively,
  - new feasible solutions propagate on the ω variables,
  - that leads to strengthen MC constraints (by changing the ϑ_i^L bounds),
  - that in turn propagates on binary variables.

Much more sophisticated propagation

- It has to be noted that Couenne internal bigM values (namely ϑ_i^L) are much more conservative (and safe) than those used in the SVM literature.
- Nevertheless, the sophisticated bound reduction loop implemented by GO solvers does the job. Iteratively,
  - new feasible solutions propagate on the ω variables,
  - that leads to strengthen MC constraints (by changing the ϑ_i^L bounds),
  - that in turn propagates on binary variables.
  - Conversely, branching on the $\bar{z}_i$
  - either $(\bar{z}_i = 0)$ increases the lower bound,
    thus triggering additional ω tightening,
  - or $(\bar{z}_i = 1)$ tightens the lower bound ϑ_i^L, thus
    propagating again on ω.

Much more sophisticated propagation

- It has to be noted that Couenne internal bigM values (namely ϑ_i^L) are much more conservative (and safe) than those used in the SVM literature.
- Nevertheless, the sophisticated bound reduction loop implemented by GO solvers does the job. Iteratively,
    - new feasible solutions propagate on the ω variables,
    - that leads to strengthen MC constraints (by changing the ϑ_i^L bounds),
    - that in turn propagates on binary variables.
    - Conversely, branching on the $\bar{z}_i$
    - either $(\bar{z}_i = 0)$ increases the lower bound, thus triggering additional ω tightening,
    - or $(\bar{z}_i = 1)$ tightens the lower bound ϑ_i^L, thus propagating again on ω.
- Switching off in Couenne any of these components leads to a dramatic degradation in the results.

Conclusions

- In a broad sense, we are using the SVM with the ramp loss to investigate the possibility of exploiting tools from (noncomvex) MINLP in MILP or (convex) MIQP, essentially, the reverse of the common path.

- More precisely, we have argued that sophisticated (nonconvex) MINLP tools might be very effective to face one of the most structural issues of MILP, which is dealing with the weak continuous relaxations associated with bigM constraints.

- A lot to be done . . .