Custom penalty in QUBO problems

Alessandro Giovagnoli

1 Constraints in a QUBO problem

In a QUBO formulation, apart from the objective function that we want to minimize (or maximize) we often need to introduce constraints which the variables will be subject to. Often the constraint can be expressed as simple algebraic equations. One example is requiring that for each variable $x_i, x_j \in 0, 1$ their sum must be less or equal than 1

$$x_i + x_j \geq 1 \tag{1}$$

It is then possible to map this constraint into a penalty that we can add or subtract form our objective function.

Keeping in mind the constraint in Equation 1, then we want our penalty to be 0 if $x_i + x_j \leq 1$, and 1 otherwise. We can then create a truth table of this penalty.

x_i	x_j	Penalty
0	0	0
0	1	0
1	0	0
1	1	1

Table 1: Penalties for a simple AND constraint

We easily recognize the logical *and* operator, and we easily see that it can be reproduced as $x_i \cdot x_j$. So the penalty will be

$$P(x_i + x_j \leq 1) = x_i \cdot x_j \tag{2}$$

Here follow some other examples of simple constraint that may be introduced and their corresponding penalty.

Classical constraint	QUBO penalty
$x + y \leq 1$	xy
$x + y \geq 1$	$1 - x - y + xy$
$x + y = 1$	$1 - x - y + 2xy$
$x \leq y$	$x - xy$

Table 2: Examples of constraints and penalties

Question What if we want to add a penalty for a generic and custom constraint?

So, instead of selecting a geometric constraint as done so far, we want now to find a penalty for a completely custom truth table. As a small example we take the following

x_1	x_2	x_3	Penalty
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

Table 3: Custom penalty

So this is a completely custom penalty where the configurations $(0, 0, 0)$ and $(0, 0, 1)$ cannot be chosen, since it has penalty 1; $(0, 1, 0)$ can instead be chosen, and so on.

2 Fundamental theorem of Boolean algebra

In order to decompose a generic truth table into logic gates, we can use the fundamental theorem stating that

$$f(x_1, \dots, x_n) = \sum_{i_1=0}^{1} \cdots \sum_{i_n=0}^{1} x_0^{i_0} \dots x_n^{i_n} f(x_1, \dots, x_n) \tag{3}$$

where $x^i = x$ if $i = 1$, otherwise $x^i = \bar{x}$. By $\bar{x}$ we mean NOT x.

It can easily be shown that all the basic logical operators can be reproduced with a penalty counterpart. We already showed the *and* operator in Equation 2. Equally we can find a penalty for the *or* operator and for the not operator.

Logical operator	QUBO penalty
x *AND* y	xy
x *OR* y	$x + y - xy$
NOT x	$1 - x$

Table 4: Mapping logical operators into QUBO penalties

This means that, by applying the theorem to the custom constraint we introduced, then we can decompose it into basic logic gates, and for one of them we have a penalty mapping.

3 Penalty of a custom constraint in a QUBO problem

Taking the example of the custom constraints shown in Table 3, we can apply the fundamental theorem of Equation 3. Then we get

$$\begin{aligned} f(x_1,x_2,x_3) = \\ \bar{x}_1\bar{x}_2\bar{x}_3 f(0,0,0) + \bar{x}_1\bar{x}_2 x_3 f(0,0,1) + \bar{x}_1 x_2\bar{x}_3 f(0,1,0) + \bar{x}_1 x_2 x_3 f(0,1,1) \\ + x_1\bar{x}_2\bar{x}_3 f(1,0,0) + x_1\bar{x}_2 x_3 f(1,0,1) + x_1 x_2\bar{x}_3 f(1,1,0) + x_1 x_2 x_3 f(1,1,1) \\ = \bar{x}_1\bar{x}_2\bar{x}_3 \cdot 1 + \bar{x}_1\bar{x}_2 x_3 \cdot 1 + \bar{x}_1 x_2\bar{x}_3 \cdot 0 + \bar{x}_1 x_2 x_3 \cdot 1 \\ + x_1\bar{x}_2\bar{x}_3 \cdot 1 + x_1\bar{x}_2 x_3 \cdot 1 + x_1 x_2\bar{x}_3 \cdot 0 + x_1 x_2 x_3 \cdot 1 \\ = \bar{x}_1\bar{x}_2\bar{x}_3 + \bar{x}_1\bar{x}_2 x_3 + \bar{x}_1 x_2 x_3 + x_1\bar{x}_2\bar{x}_3 + x_1\bar{x}_2 x_3 + x_1 x_2 x_3 \\ = \bar{x}_1\bar{x}_2(\bar{x}_3 + x_3) + \bar{x}_1 x_2 x_3 + x_1\bar{x}_2(\bar{x}_3 + x_3) + x_1 x_2 x_3 \\ = (\bar{x}_1 + x_1)\bar{x}_2 + (\bar{x}_1 + x_1)x_2 x_3 = \bar{x}_2 + x_3 \end{aligned} \tag{4}$$

And thus our Boolean penalty is

$$P(x_1,x_2,x_3) = \bar{x}_2 + x_3 = (\text{NOT } x_2) \text{ OR } x_3 \tag{5}$$

We cannot though directly use this penalty as shown in Equation 5 since the sums actually stand for logic operators. For example, if we evaluate the cases $P(0,0,1)$ or $P(1,0,1)$ we obtain 2 while in the penalty table we see that it should be 1.

Since the penalty function will be added (or subtracted) to the objective function, we now have to convert the Boolean penalty in operations in $\mathbb{N}$. This can be done through Table 4.

We then get

$$P(x_1,x_2,x_3) = \bar{x}_2 + x_3 - \bar{x}_2 x_3 = (1 - x_2) + x_3 - (1 - x_2)x_3 \tag{6}$$

which we can rewrite as

$$P(x_1,x_2,x_3) = 1 - x_2 + x_2 x_3 \tag{7}$$

This is our final penatly that we can sum or subtract to the objective function, and it can be easily verified that respects the penalty table.

This is a small example of a procedure that can be used to compute the penalty of any custom constraint.