HW2

Haolin Liu

1. 1. The maxpool operation is very similar to ReLU in the sense that they both use a max function.

Similar to ReLU encoding, we design auxiliary integer variable such that

Different from ReLU encoding, where we can directly use the lower and upper bound of the target variable, in this situation we need to consider the bounds of both variables. To that, we define the following

We can have the following MILP encoding:

Based on suggestions found here (<https://or.stackexchange.com/questions/711/how-to-formulate-linearize-a-maximum-function-in-a-constraint>), we can alternatively define variable called M that represents a value larger than or equal to the maximum possible difference between the two variables.

And reformulate the encoding as:

The two formulation are effectively the same.

* 1. With the provided transformation, we first compute the box bounds of all variables.

Clearly the box bounds along is unable to prove the property.

We now rewrite the maxpools as their MILP encoding.

Define integer variables such that

For :

For :

Now we explore the different values can take.

It is clear that cannot be 0, so no need to explore this branch.

Set to 1, and we explore the two branches.

Here, we rewrite :

First we explore .

The MILP encoding for becomes:

Now we simplify as

This can be further backpropagated to be:

The box bound of this calculation is then [0,2]+[0,1]+0.5 = [0.5,3.5]

Minimum of this bound is greater than 0, therefore the property does hold for .

Then we explore

The MILP encoding for becomes:

Similarly, becomes

We perform further backpropagation:

The box The box bound of this calculation is then [0,1]+[0,3]+0.5 = [0.5, 4.5]

Minimum of this bound is greater than 0, therefore the property does hold for as well.

Following the method discussed in class, MILP with backpropagation has indeed proved the desired property.