Challenge Novembre – 2020

Calculator with Two Buttons

A solution with OPL CPLEX by Alex Fleischer
afleischer@fr.ibm.com

OPL (Optimization Programming Language) is an abstract modeling language that helps model easily optimization problems that can be solved both with IBM CPLEX linear programming and IBM CPLEX constraint programming CPOptimizer (CPO)

Let us remember that with ILOG (French company bought by IBM in 2009) we had two kind of decision engines:

A) Rule based (JRules, ODM)
B) Optimization based (CPLEX)

Here a small example of a tiny optimization model, in English, OPL and Python

Zoo, bus, kids and optimization

With words	In OPL	In Python / DoCplex

300 kids need to travel to the London zoo
The school may rent 40 seats and 30 seats buses for 500 and 400 £
How many buses of each to minimize cost ?
 
	int nbKids=300;
	float costBus40=500;
	float costBus30=400;
	
	dvar int+ nbBus40;
	dvar int+ nbBus30;
	
	minimize
	costBus40*nbBus40
	+nbBus30*costBus30;
	
	subject to
	{
	40*nbBus40+
	nbBus30*30
	>=nbKids;
	}
		from docplex.mp.model import Model
		
		mdl = Model(name='buses')
		
		nbbus40 =
		mdl.integer_var(name='nbBus40')
		nbbus30 =
		mdl.integer_var(name='nbBus30')
		
		mdl.add_constraint(nbbus40*40 +
		nbbus30*30 >= 300, 'kids')
		
		mdl.minimize(nbbus40*500 +
		nbbus30*400)
		
		mdl.solve()
		
		print(nbbus40.solution_value);
		print(nbbus30.solution_value);

We can call CPLEX from many languages (C,C++,.NET,Java,Python …) but using OPL leads to a clear frontier between the model and the code that will embed the model. (Not far from Decision Model and Notation (DMN) principle: “The notation is designed to be readable by business and IT users alike. This enables various groups to effectively collaborate in defining a decision model”)

Now let’s move to the Novembre 2020 DMC challenge:

A calculator, initially displaying **0**, has only two buttons: “**+**” adds **1** to the number on display; “**x**” multiplies number on display by **10**. What is the least number of button presses needed to show 5034?

In OPL CPLEX no need to be very clever, we need to translate the constraints.

```
//using CP;

int maxOperations=50;
int target=5034;

range r=0..maxOperations;

// The values we will see starting with 0
dvar int+ x[r] in 0..2*target;
// the operations to get the value target
dvar boolean opeisadd1[r]; // true means we add 1 false means *10
// Number of operations
dvar int+ nbOperations in 1..maxOperations-1;

minimize nbOperations ;
subject to
{
  x[0]==0;
  forall(i in r:i!=maxOperations)
  {
     (opeisadd1[i]==1 && i<=nbOperations) => (x[i+1]==x[i]+1);
     (opeisadd1[i]==0 && i<=nbOperations) => (x[i+1]==x[i]*10);
     (x[i]==target) == (nbOperations==i);
  }
}

execute
{
  writeln("Number of operations = ",nbOperations);
  for(var i=0;i<nbOperations;i++)
     if (opeisadd1[i]==1) write("+1 ");
     else write("*10 ");
  writeln();

}
```

Which gives

```
Number of operations = 15
+1 +1 +1 +1 +1 *10 *10 +1 +1 +1 *10 +1 +1 +1 +1
```

PS :

Here we relied on Linear Programming. If we want to use Constraint Programming we simply need to add “using CP;” at the beginning of the model

Within CPLEX we have 2 tools:

And for this specific challenge, both tools alone work fine with the same OPL model. Sometimes for real life problems we need both tools together.

Making Decision Optimization Simple : https://www.linkedin.com/pulse/making-decision-optimization-simple-alex-fleischer/