Challenge October 2025
Decision with two objectives

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

CPLEX is available within IBM AI platform watsonx.AI
And OPL is a high level modeling language for CPLEX

If we could assume one of the objectives is more important than the other one we could use a staticLex objective within CPLEX.

We would write

```
tuple feature
{
  key int id;
  int cost;
  int value;
}

{feature} features=
{
  <1,5000,7>,
  <2,4000,6>,
  <3,3000,5>,
  <4,2000,4>,
  <5,1000,3>,
  <6,1000,2>,
  <7,1000,1>,
  <8,1000,1>,
  <9,1000,1>,
  <10,1000,1>
};

float budget=10000;

dvar boolean x[features];
```

```
dvar int obj1;
dvar int obj2;

// plain staticLex
maximize staticLex(obj1,obj2);
subject to
{
  // cost should be less than budget
  sum(f in features) x[f]*f.cost<=budget;
  // obj1 is the number of features
  obj1==sum(f in features) x[f];
  // obj2 is the total value of the features
  obj2==sum(f in features) x[f]*f.value;
}

execute display
{
  writeln("obj1=",obj1);
  writeln("obj2=",obj2);
  writeln("features : ");
  for(var f in features) if (x[f]==1) write(f.id," ");
  writeln();

}
```

Which gives

```
obj1=7
obj2=17
features :
3 4 5 6 7 8 9
```

But if we can't assume one of the objective is more important than the other one, then we can rely on solutions pools and then filter the solutions in order to get not dominated solutions. A human user would then decide what to do …

In OPL CPLEX we can write

```
tuple feature
```

```
{
  key int id;
  int cost;
  int value;
}

{feature} features=
{
  <1,5000,7>,
  <2,4000,6>,
  <3,3000,5>,
  <4,2000,4>,
  <5,1000,3>,
  <6,1000,2>,
  <7,1000,1>,
  <8,1000,1>,
  <9,1000,1>,
  <10,1000,1>
};

float budget=10000;

dvar boolean x[features];

dvar int obj1;
dvar int obj2;

// not dominated
maximize obj1+obj2;
subject to
{
  // cost should be less than budget
  sum(f in features) x[f]*f.cost<=budget;
  // obj1 is the number of features
  obj1==sum(f in features) x[f];
  // obj2 is the total value of the features
  obj2==sum(f in features) x[f]*f.value;
}

execute display
```

```
{
  writeln("obj1=",obj1);
  writeln("obj2=",obj2);
  writeln("features : ");
  for(var f in features) if (x[f]==1) write(f.id," ");
  writeln();

}


main

{

   //writeln("solution pools")

   thisOplModel.generate();
   cplex.solnpoolintensity=4;
   cplex.solnpoolcapacity=1000;
   cplex.solnpoolreplace=2;
   cplex.populatelim=2000;


   cplex.populate();
   var nsolns = cplex.solnPoolNsolns;
   //writeln("n sol ",nsolns);

   var obj1=new Array(nsolns);
   var obj2=new Array(nsolns);;

   for(var s = 0; s < nsolns; s++)
   {
      thisOplModel.setPoolSolution(s);
      //thisOplModel.postProcess();
      obj1[s]=thisOplModel.obj1.solutionValue;
      obj2[s]=thisOplModel.obj2.solutionValue;

   }

   writeln("not dominated solutions");
```

```
   writeln();

   for(var s = 0; s < nsolns; s++)
   for(var s2 = s+1; s2 < nsolns; s2++) if ((obj1[s]==obj1[s2]) &&
(obj2[s]==obj2[s2]))
   {
      obj1[s2]=0;
      obj2[s2]=0;
   }

   for(var s = 0; s < nsolns; s++)
   {
      var dominated=0;
      for(var s2 = 0; s2 < nsolns; s2++) if (s!=s2) if
((obj1[s2]>=obj1[s]) && (obj2[s2]>=obj2[s]))
            dominated=1;
      if (dominated==0)
      {
          thisOplModel.setPoolSolution(s);
              thisOplModel.postProcess();
          writeln();
      }

   }
   cplex.clearModel();
}
```

Which gives

```
not dominated solutions

obj1=7
obj2=17
features :
3 4 5 6 7 8 9

obj1=4
obj2=18
features :
2 3 4 5
```