Production planning

Production planning

How does a company decide what proportion of its products to produce inside the company and what to buy from outside the company?

To meet the demands of its customers, a company manufactures products in its own factories (inside production) or buys them from other companies (outside production). The problem is to determine how much of each product should be produced inside the company and how much outside, while minimizing the overall production cost, meeting the demand, and satisfying the resource constraints.

The OPL (Optimization Programming Language) model `production_cloud.mod` minimizes the production cost for a number of products while satisfying customer demand. Each product can be produced either inside the company, or outside at a higher cost. The inside production is constrained by the company's resources, while outside production is considered unlimited.

The model first declares the products and the resources. The data consists of a description of the products, that is, the demand, the inside and outside costs, the resource consumption, and the capacity of the various resources. The variables for this problem are the inside and outside production for each product.

A production planning problem exists because there are limited production resources that cannot be stored from period to period. Choices must be made as to which resources to include and how to model their capacity, their consumption, and their costs.

OPL allows you to write a mathematical representation of your business problem that is separate from your data.

The model (`production_cloud.mod`)

Modeling the variables

In the model, `production_cloud.mod`, two arrays of decision variables are specified.

```
dvar float+ Inside[Products];
dvar float+ Outside[Products];
```

The two decision variables (`dvar`) are the inside production of each product and the outside production of each product, and they are specified as nonnegative real numbers, so this is a linear program.

Modeling the data

The data is defined in the file `production_data.mod` and this is explained in the section entitled "The data".

Modeling the objective function

The specification of a model consists of the objective function and the constraints. The objective function in this example minimizes the total cost of meeting the demand.

```
minimize
 totalInsideCost + totalOutsideCost;
```

These costs are defined as follows:

```
dexpr float totalInsideCost  = sum(p in Products)  p.insideCost * Inside[p];
dexpr float totalOutsideCost = sum(p in Products)  p.outsideCost * Outside[p];
```

This is the sum over each product in the product set of the inside cost of producing that product, multiplied by the inside production, plus the outside cost of producing that product, or acquiring that product, multiplied by the outside production.

Modeling the constraints

The objective function is subject to two constraints.

```
subject to {
  forall( r in Resources )
    ctCapacity:
      sum( k in Consumptions, p in Products
           : k.resourceId == r.name && p.name == k.productId )
        k.consumption* Inside[p] <= r.capacity;

  forall(p in Products)
    ctDemand:
      Inside[p] + Outside[p] >= p.demand;
```

The first constraint (`ctCapacity`) is a capacity constraint on the resources. It ensures that, for each resource in the set of resources, the sum over all the products of the consumption of the resource for that product, multiplied by the inside production, has to be less than or equal to the capacity available for that resource.

The second constraint (`ctDemand`) is the demand constraint. It ensures that, for each product in the set of products, the inside production plus the outside production has to be at least as great as the demand.

The data

For DropSolve, all data must be defined in the form of tables (which in OPL implies using sets of tuples to define these tables). The `production_data.mod` file defines the tuples `TProduct`, `TResource` and `TConsumption` which form the rows of the tables `Products`, `Resources` and `Consumptions` (which defines the amounts of each resource needed for each product). An additional tuple `TPlannedProduction` is defined to form the rows of the output table `plan` which is used to store the final optimal solution.

```
tuple TProduct {
   key string name;
   float demand;
   float insideCost;
   float outsideCost;
 };

 tuple TResource {
   key string name;
   float capacity;
 };

 tuple TConsumption {
   key string productId;
   key string resourceId;
   float consumption;
```

```
}

{TProduct}     Products = ...;
{TResource}    Resources = ...;
{TConsumption} Consumptions = ...;

/// solution
tuple TPlannedProduction {
  key string productId;
  float insideProduction;
  float outsideProduction;
}

{TPlannedProduction} plan;
```

The `production_cloud.dat` file provides data values for the input tables that are defined in the `production_data.mod` file.

```
Products=
{
 <"kluski"    , 100 , 0.6, 0.8 >
 <"capellini" , 200 , 0.8, 0.9 >
 <"fettucine" , 300 , 0.3, 0.4 >
};

Resources =
{
 <"flour" 20>
 <"eggs"  40>
};

Consumptions =
{
  <"kluski"    "flour" 0.5>
  <"kluski"     "eggs" 0.2>

  <"capellini" "flour" 0.4>
  <"capellini"  "eggs" 0.4>

  <"fettucine"  "flour" 0.3>
  <"fettucine"  "eggs"  0.6>
};
```

In this example, there are three types of pasta products - kluski, capellini and fettuccine - and there are two resources, flour and eggs. The consumption represents the amount of a resource used by a product - for example: one unit of kluski requires 0.5 units of flour and 0.2 units of eggs. There is a capacity of 20 units of flour and 40 units of eggs. There is a demand of 100, 200, 300 for each of the three products -- kluski, capellini and fettuccine. And there is an inside and an outside cost to produce each product.

The result

For the data provided, the model returns the optimal solution:

```
Inside = [40 0 0];
Outside = [60 200 300];
```

- The optimal proportions of pasta produced inside the factory are 40 units of kluski, 0 units of capellini, and 0 units of fettucine.
- The optimal proportions of pasta produced outside the factory are 60 units of kluski, 200 units of capellini, and 300 units of fettucine.

Alternative data

The product in this model could be anything. For example, it could be jewelry. So when you produce jewelry, you could have a data file that looks like this:

```
Products=
{
 <"rings"    , 100 , 250, 260 >
 <"earrings" , 150 , 200, 270 >
 };

Resources =
{
 <"gold" 130>
 <"diamonds" 180>
};

Consumptions =
{
  <"rings"    "gold" 3.0>
  <"rings"     "diamonds" 1.0>

  <"earrings" "gold" 2.0>
  <"earrings"  "diamonds" 2.0>
  };
```

In this case, there are two products, rings and earrings, and two resources, gold and diamonds. The recipe calls for three units of gold and one diamond to produce each ring, or two units of gold and two diamonds for each earring. The resources available are 130 units of gold and 180 diamonds. The demand is for 100 rings and 150 pairs of earrings. The inside cost to produce the rings is $250 and to produce the earrings it is $200. The outside cost is $260 for rings and $270 for earrings.

An important point to note here is that the model is the same for pasta, for jewelry, or any other product. The model does not depend on the data.

You can replace the existing data in the file `production_cloud.dat` and run the model again, to obtain an optimal solution for jewelry.