Task assignment using CPLEX

Roberto Meroni* Hugo Sanz González†

October 2023

Consider the problem of distributing the work needed to perform a set of tasks T among a set of computers C. Each task $t \in T$ requires $r_t \geq 0$ compute units to complete, while each computer $c \in C$ has $r_c \geq 0$ compute units available. The goal is to find a task assignment that minimizes the load (defined as the amount of processing resources assigned to that computer divided by its processing capacity) of the highest loaded computer by modeling this problem as a linear program, expressing it in the ILOG OPL language, and solving it via the CPLEX programming environment.

1. *Explain the LP model in Eqs. (2)–(5).*

Let $0 \leq x_{tc} \leq 1$ be the fraction of task $t \in T$ completed by computer $c \in C$. Each task must be completed entirely, hence the first set of constraints is

$$\sum_{c \in C} x_{tc} = 1, \quad \text{for } t \in T.$$

The amount of compute units requested by task $t \in T$ that are handled by computer $c \in C$ equals $r_t x_{tc}$. Therefore the number of compute units handled by computer c is $\sum_{t \in T} r_t x_{tc}$, which must be at most its compute capacity r_c. This must be true for each computer, hence the second set of constraints is

$$\sum_{t \in T} r_t x_{tc} \leq r_c, \quad \text{for } c \in C$$

We will discuss the role of the last constraint and the objective function in the next section.

2. *Explain how equation (1) is implemented in the model.*

The original problem statement is not directly expressed as a linear program, since its objective is not a linear function in the decision variables. One way to linearize an objective function of the form $\max_{y_i} \alpha_i y_i$ to be minimized, for $1 \leq i \leq n$, $\alpha_i \in \mathbb{R}$, is to introduce an additional decision variable z, add one constraint for each argument $\alpha_i y_i$ of the maximum function that bounds z from below by $\alpha_i y_i$, and let the objective function to minimize be z.

In this case z represents the load of the highest loaded computer. For each computer $c \in C$, it must be the case that its load is at most z. The number

*roberto.meroni@estudiantat.upc.edu
†hugo.sanz@estudiantat.upc.edu

of compute units handled by c is $\sum_{t\in T} r_t x_{tc}$, so its load is $(1/r_c)\sum_{t\in T} r_t x_{tc}$. We therefore have

$$z \geq \frac{1}{r_c}\sum_{t\in T} r_t x_{tc}, \quad \text{for } c \in C.$$

The objective z is bounded from below by the load of each computer, and the LP minimizes z, so $\min z$ subject to the previous constraints equals the load of the highest loaded computer.

3. *Implement the model in OPL and solve it using CPLEX with the provided input data.*

We want to solve the task assignment problem for four tasks requiring

$$r_{t_1} = 261.27,\ r_{t_2} = 560.89,\ r_{t_3} = 310.51,\ r_{t_4} = 105.80$$

compute units using three computers with respective capacities

$$r_{c_1} = 505.67,\ r_{c_2} = 503.68,\ r_{c_3} = 701.78.$$

The CPLEX solver output window shows the text

```
Total load 1238.47
// solution (optimal) with objective 0.723773179127243
CPU 1 loaded at 72.377317913%
CPU 2 loaded at 72.377317913%
CPU 3 loaded at 72.377317913%
```

(Here "total load" is expressed in absolute compute units.) The solution window provides the assignment

$$A = \begin{bmatrix} 0.996 & 0 & 0.004 \\ 0 & 0.096 & 0.904 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix},$$

where a_{ij} is the percentage of task t_i completed by computer c_j for $1 \leq i \leq 4$ and $1 \leq j \leq 3$.

4. *Write a pre-processing block to check whether computers have enough total capacity to serve the resources requested by all the tasks.*

We compute the total load needed by the tasks and the capacity of all computers altogether. Then we check whether the former quantity exceeds the latter, in which case we print a warning message.

```
execute {
  // total load of the tasks
  var totalLoad=0;
  for (var t=1;t<=nTasks;t++)
    totalLoad += rt[t];
  writeln("Total load " + totalLoad);

  //total capacity of the computers
  var totalCapacity=0;
  for (var c=1;c<=nCPUs;c++)
```

```
    totalCapacity += rc[c];
  writeln("Total capacity is " + totalCapacity);

  if (totalCapacity >= totalLoad)
    writeln("Total capacity is enough for tasks")
  else
    writeln("The load required by tasks exceeds the total capacity of the CPUs")
}
```

For example, if we replace the load needed by task 1 from $x_{t_1} = 261.27$ to 1261, then the script log shows the message "`The load required by tasks exceeds the total capacity of the CPUs`".

5. *Implement the model in OPL using a main file and solve it using CPLEX with the provided input data.*

   The main file looks as follows:

```
1  main {
2    var src = new IloOplModelSource("Lab1.mod");
3    var def = new IloOplModelDefinition(src);
4    var cplex = new IloCplex();
5    var model = new IloOplModel(def,cplex);
6    var data = new IloOplDataSource("Lab1.dat");
7    model.addDataSource(data);
8    model.generate();
9    cplex.epgap=0.01;
10   if (cplex.solve()) {
11     writeln("Max load " + 100 * cplex.getObjValue() + "%");
12     for (var c=1;c<=model.nCPUs;c++) {
13       var load=0;
14       for (var t=1;t<=model.nTasks;t++)
15         load+=(model.rt[t]* model.x_tc[t][c]);
16       load = (1/model.rc[c])*load;
17       writeln("CPU " + c + " loaded at " + 100 * load + "%");
18     }
19   } else {
20     writeln("No solution found");
21   }
22   model.end();
23   data.end();
24   def.end();
25   cplex.end();
26   src.end();
27 };
```

(Here we replaced the expression `load` by `100 * load` in lines 11 and 17 to print the load as a percentage.) The output remains the same as before:

```
Total load 1238.47
Total capacity is 1711.13
Total capacity is enough for tasks
Max load 72.3773179%
CPU 1 loaded at 72.3773179%
CPU 2 loaded at 72.3773179%
CPU 3 loaded at 72.3773179%
```

6. *Is it possible to reduce the capacity of all computers uniformly by the same percentage?*

Note that 27.623% of the capacity of each computer is not being used, so we can uniformly reduce their capacity by this percentage to achieve maximum utilization. We multiply the original capacity of each computer by the ratio total compute needed/total capacity $\approx 1238.47/1711.13$ to obtain the new computer capacities

$$r_{c_1} = 365.9904,\ r_{c_2} = 364.5501,\ r_{c_3} = 507.9295.$$

This gives the solver output

```
Total load 1238.47
Total capacity is 1238.47
Total capacity is enough for tasks
// solution (optimal) with objective 1
CPU 1 loaded at 100%
CPU 2 loaded at 100%
CPU 3 loaded at 100%
```

as desired.