OPTIMIZATION IN INDUSTRY - J. DARLAY

Miniconference Report
CHAU Dang Minh

Summary. Dr. J. Darly, the Head of Science at Hexaly Company, gave an introduction to optimization in industry, including how such kind of optimization differs from academic one, and introduced Hexaly's optimization language for a particular problem.

In the heart of academic optimization theory lies popular problems, such as linear regression, support vector machine and decision tree. When applying these problems into industrial problems, a scientist or engineer has to choose and combine based on a large amount of mathematical properties in order to ensure tractability, runtime constraints, business constraints, scalability and maintainability. Moreover, the problem definition can also change. Examples of such problems include Audience forecast (TF1), Telecom user behavior, and Failure prediction.

In the algorithmic approach, the first step is modelling, with an aim to understand the problem, find the objectives, variables and constraints, as well as define an appropriate mathematical model for the problem. The second step is solving. Usually, the priority is not an optimal solution, but to define what a good bound and a good solution are, then attempt to find a satisfying solution. Due to the nature of an industrial optimization problem, such approach is not suitable, since these time-consuming steps have to be completely run for every new problem, despite the fact that some problems may have some similar patterns. Therefore, the business approach is to create an optimization language and solvers.

An optimization language, AMPL, has been studied in the course of Practical Optimization, consisting of data, a model and a solver script. Hexaly's language also serves the same concepts but more user friendly. The syntax is similar to that of Python. A program contains the following main functions. The *input* function is used to transform the data into a desired format. The *model* function is used to define the objective and constraints. Finally, the *output* function is used to transform the solution to a desired format. However, as in the introduction, the solver is built-in and cannot be defined precisely, which may be to preserve authenticity. A simple program has been written in this language in order to solve the Santa's Workshop Tour Problem. After solving the problem, it can be discovered that Hexaly provides good infrastructure, since 14 million iterations were completed in just 35 seconds. A sample of the program is provided in Figure 1. The development of the interactive environment is also worth appreciating, since it lets programmers focus on the optimization deed and gives good feedback in the progress.

No matter if a master student pursues a career in academic or industry, and if they want to be specialized in the field of optimization, optimization problems are always a part of their work. Except for few number of publications on theoretical foundations, comparing to the number of publications on the improvement of approaches for new real-life problems, every publication on such improvements can be considered as a solution to an optimization problem. Even in a foundational field such as logic, since undecidability has been discovered, researchers have been aiming to provide tractable partial solution. Therefore, Dr. J. Darly's lecture on industrial optimization is valuable for us master students. It reminds us the role of a general progress, rather than solving each problem individually as well as arising problems in real business. Then come optimization languages as an effort to implement them.
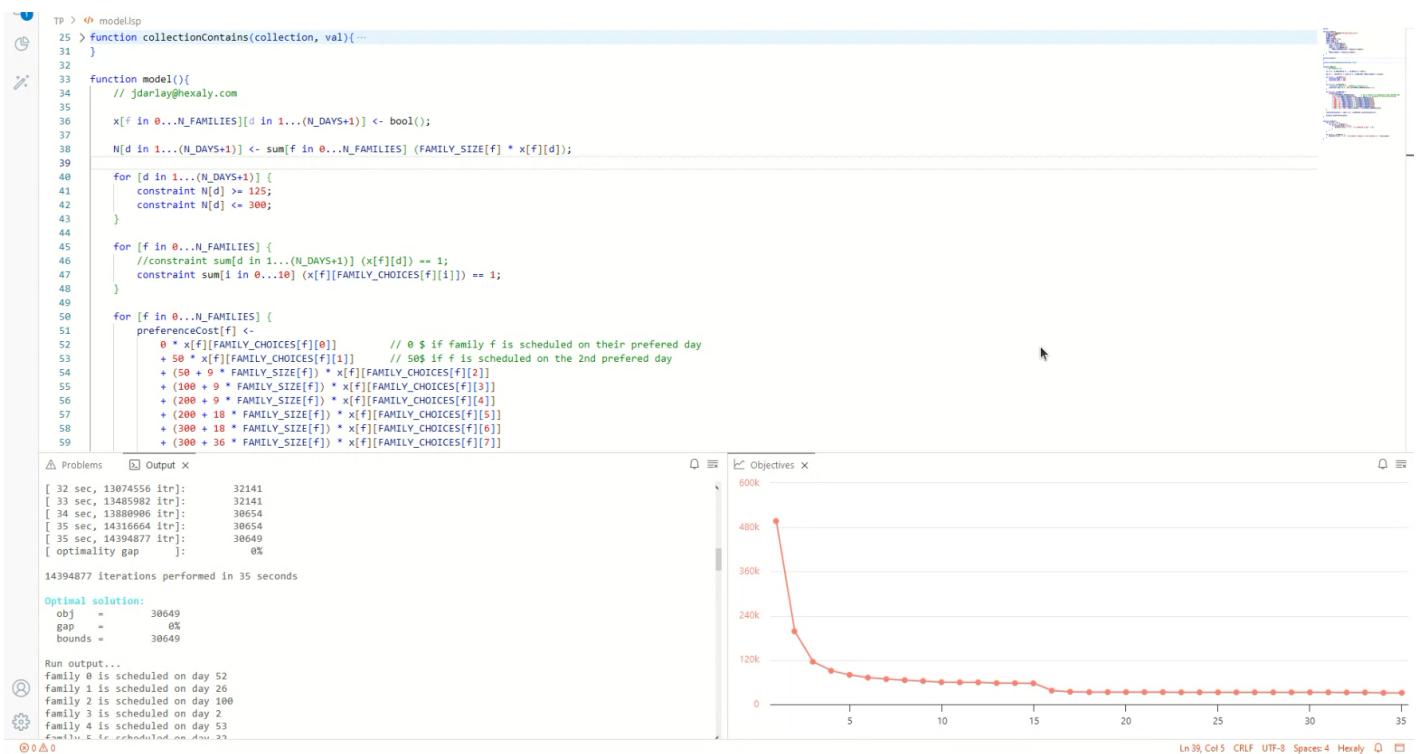
```
TP > </> model.lsp
25 > function collectionContains(collection, val){...
31 }
32
33   function model(){
34       // jdarlay@hexaly.com
35
36       x[f in 0...N_FAMILIES][d in 1...(N_DAYS+1)] <- bool();
37
38       N[d in 1...(N_DAYS+1)] <- sum[f in 0...N_FAMILIES] (FAMILY_SIZE[f] * x[f][d]);
39
40       for [d in 1...(N_DAYS+1)] {
41           constraint N[d] >= 125;
42           constraint N[d] <= 300;
43       }
44
45       for [f in 0...N_FAMILIES] {
46           //constraint sum[d in 1...(N_DAYS+1)] (x[f][d]) == 1;
47           constraint sum[i in 0...10] (x[f][FAMILY_CHOICES[f][i]]) == 1;
48       }
49
50       for [f in 0...N_FAMILIES] {
51           preferenceCost[f] <-
52               0 * x[f][FAMILY_CHOICES[f][0]]         // 0 $ if family f is scheduled on their prefered day
53               + 50 * x[f][FAMILY_CHOICES[f][1]]      // 50$ if f is scheduled on the 2nd prefered day
54               + (50 + 9 * FAMILY_SIZE[f]) * x[f][FAMILY_CHOICES[f][2]]
55               + (100 + 9 * FAMILY_SIZE[f]) * x[f][FAMILY_CHOICES[f][3]]
56               + (200 + 9 * FAMILY_SIZE[f]) * x[f][FAMILY_CHOICES[f][4]]
57               + (200 + 18 * FAMILY_SIZE[f]) * x[f][FAMILY_CHOICES[f][5]]
58               + (300 + 18 * FAMILY_SIZE[f]) * x[f][FAMILY_CHOICES[f][6]]
59               + (300 + 36 * FAMILY_SIZE[f]) * x[f][FAMILY_CHOICES[f][7]]
```

```
⚠ Problems    ⊠ Output ×                                                          ⌂ ≡        📈 Objectives ×
[ 32 sec, 13074556 itr]:        32141
[ 33 sec, 13485982 itr]:        32141
[ 34 sec, 13880906 itr]:        30654
[ 35 sec, 14316664 itr]:        30654
[ 35 sec, 14394877 itr]:        30649
[ optimality gap      ]:        0%

14394877 iterations performed in 35 seconds

Optimal solution:
  obj    =        30649
  gap    =           0%
  bounds =        30649

Run output...
family 0 is scheduled on day 52
family 1 is scheduled on day 26
family 2 is scheduled on day 100
family 3 is scheduled on day 2
family 4 is scheduled on day 53
```

Ln 39, Col 5   CRLF   UTF-8   Spaces: 4   Hexaly

Figure 1: Sample code