Team 2 Resource Allocation

Riad Alharithi, Karthik Manjunath, Jennifer Lynn Zeitouni, Sara Ferdousi, Vladislav Ruchin

12/1/2021

Resource Allocation Model

Variables

i is set of Employees {E1, E2, ..} varies from 1 to n = total number of employees

j is set of projects { P1, P2, ..} varies one to p = total number of projects

k is set of classifications {class 1, class 2, ..} varies from 1 to c = number of classifications at the company

$X_{i,j,k}$ Continuous: Hours that an employee i work in functional class k in project j

$Y_{i,j,k} = 0, 1$ Binary variable, If employee i is assigned to classification k for project j then $y_{i,j,k}$is 1 otherwise it is 0.

Data

$S_{i,k}$ =1 Binary, if employee is qualified to do task k

$P_{i,k}$: Continuous, Profit = Hourly rate for employee i classification k in project j * 10% *

$R_{i,k}$: Continuous, billing rate for i when performing task classification k

$T_{j,k}$ Continuous, Total hours per project for task classification k

Objective Function:

$$\text{Maximize } \sum_{i=1}^{1=n} \sum_{j=1}^{j=p} \sum_{k=1}^{k=c} X_{i,j,k} \cdot P_{i,k}$$

Constraints

- Utilization: Total hours assigned to a employee who classified as engineer/ designer/ surveyor should not exceed 1664 hours per year.

$\sum_{j=1}^{j=p} \sum_{k=1}^{k=c} X_{i,j,k} \leq 1664$ hours $\forall i$

- Big M: Linking the constraints, in this scenario each employee can not be assigned more than the total hours per task

$X_{i,j,k} \leq T[j,k] * Y_{i,j,k} \, \forall i, j, k$

- Skills set: Every employee i assigned on any project j should have the appropriate skill. for example, Junior Engineer can not do Senior Engineer task but the opposite is correct.

$Y_{i,j,k} \leq S_{i,k} \ \forall \ i, j, k$

- Total Hours constraints: the total hours for each employee i on every project should be less than or equal than the negotiated hours in the project budget.

$\sum_{i=1}^{i=n} X_{i,j,k} = T_{j,k} \forall j, k$

Code

Libraries Setup

```
library(ompr, quietly = TRUE)
library(magrittr, quietly = TRUE)
library(pander, quietly = TRUE)
library(ROI, quietly = TRUE)
```

```
## ROI: R Optimization Infrastructure

## Registered solver plugins: nlminb, glpk, lpsolve, neos, symphony.

## Default solver: auto.
```

```
library(ROI.plugin.glpk, quietly = TRUE)
library(ompr.roi, quietly = TRUE)
library(pander, quietly = TRUE)
library (devtools)
```

```
## Loading required package: usethis
```

```
devtools::install_github("prof-anderson/TRA")
```

```
## Skipping install of 'TRA' from a github remote, the SHA1 (1feb2d44) has not changed since last instal
##   Use 'force = TRUE' to force installation
```

```
library(TRA)
library(Benchmarking, quietly=TRUE)
library(ROI.plugin.glpk)
library(ROI.plugin.lpsolve)
```

```
##
## Attaching package: 'ROI.plugin.lpsolve'

## The following objects are masked from 'package:lpSolveAPI':
##
##     read.lp, write.lp
```

```
library(ROI.plugin.neos)
library(ROI.plugin.symphony)
library(readr)
```

Reading Data

```
XIJ<-readr::read_csv("/Users/riad/!Riad/PhD/Classes/ETM 640/Project/XIJ.csv")
```

```
## Rows: 7 Columns: 6

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (6): P1, P2, P3, P4, P5, P6

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
TS<- readr::read_csv("/Users/riad/!Riad/PhD/Classes/ETM 640/Project/StaffSKills.csv")
```

```
## Rows: 7 Columns: 10

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (10): E1, E2, E3, E4, E5, E6, E7, E8, E9, E10

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
S<-t(TS)
TP<-readr::read_csv("/Users/riad/!Riad/PhD/Classes/ETM 640/Project/PIK.csv")
```

```
## Rows: 7 Columns: 10

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (10): E1, E2, E3, E4, E5, E6, E7, E8, E9, E10

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
P<-t(TP)
Td <- t(XIJ)
##Assignement_Table<-matrix(0, nrow=10, ncol=9)
Billingrate<-readr::read_csv("/Users/riad/!Riad/PhD/Classes/ETM 640/Project/BillingRates.csv")
```

```
## Rows: 10 Columns: 1

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): Billing Rate

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
Classifications<-readr::read_csv("/Users/riad/!Riad/PhD/Classes/ETM 640/Project/Classifications.csv")
```

```
## Rows: 10 Columns: 1

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): Classification (K)

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
UtilizationRate<-readr::read_csv("/Users/riad/!Riad/PhD/Classes/ETM 640/Project/UtilizationRate.csv")
```

```
## Rows: 10 Columns: 1

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (1): Max Utilization per year

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
n<-10
## Number of employee
p<-6
##Number of projects
c<-7
## Number of classifications
```

Modeling

```
model <- MIPModel()
model<-   add_variable(model, X[i,j,k], i=1:n, j=1:p, k=1:c, lb=0, type= "continuous")
## This variable is to determine the number of hours assigned to an employee on each project with speci
model<- add_variable(model, Y[i,j,k], i=1:n, j=1:p, k=1:c, type = "binary")
## a binary variable to determine if employee is assigned on a project with a specific classification
model<- set_objective(model, sum_expr((X[i,j,k]*P[i,k]), i=1:n, j=1:p, k=1:c), "max")
```

```
## our objective is maximize the profit by multiplying the profit per hour per class (P[i,k]) by the as
model<- add_constraint (model, Y[i,j,k]<=S[i,k], i=1:n, j=1:p, k=1:c)
##Constraints of utilization
model<- add_constraint(model,sum_expr(X[i,j,k], j=1:p, k=1:c) <=1664, i=1:n)
## no employee can be assigned more hours than the hours that are set in the project for that classific
model<- add_constraint(model,sum_expr(X[i,j,k],i=1:n) == Td[j,k], j=1:p, k=1:c)
##constraints of Big M : linking X[i,j,l] to Y[i,j,k]
model<- add_constraint (model, X[i,j,k]<=Td[j,k]*Y[i,j,k], i=1:n, j=1:p, k=1:c)
##Solve the model
result <- solve_model(model, with_ROI(solver = "glpk"))
result
```

```
## Status: optimal
## Objective value: 220465.4
```

```
t <- get_solution(result,X[i,j,k])
```

Output

```
for (i in 1:n) {
for (j in 1:p) {
  for (k in 1:c) {


Assigned_T<- get_solution(result,X[i,j,k]) %>%

    dplyr::filter(value > .9)

  }}}

for (z in 1:n) {

assign(paste0("E",z),Assigned_T %>%
dplyr::filter(i==z))

}
```