Startup Fund

The investment manager has a classical decision: What to invest in. Since it is not possible to divide the investment in smaller sizes, making an investment is modelled with a binary variable. Then the expected profit should be maximized constrainted by the budget. This problem is the classical OR problem, the knapsack problem.

Problem

- Maximize the profit

Sets

- $i \in Investments = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$: The set of different investment possibilities.

Parameters

- $EstProfitFactor_i$: Investment profit factor for investment i
- $CapitalRequirements_i$: Capital requirement for project i
- $Budget = 100$: The available capital

Decision variables

- Invest or not in project i,: $x_i \in \{0, 1\}$. $x_i = 1$ models that an investment **is** made and $x_i = 0$ models that an investment is **not** made.

Model

Objective:

- Maximal profit factor:

$$\sum_i EstProfitFactor_i \cdot x_i$$

Constraints:

- The budget has to be obeyed:

$$\sum_i CapitalRequirements_i \leq Budget \; \forall \; i$$

The full model in Julia/JuMP, available with the name

`StartupFund.jl`

from the book web-site, is given below:

```
#*****************************************************************
# Startup Fund assignment, "Mathematical Programming Modelling" (42112)
using JuMP
using HiGHS
#*****************************************************************

#*****************************************************************
# PARAMETERS
Investments = [1 2 3 4 5 6 7 8 9]
I=length(Investments)

# Data
EstProfitFactor =     [1.7 1.4 1.3 2.1 1.9 1.8 1.5 2.2 1.6]
CapitalRequirements = [ 17  25  19  25  28  23  29  31  18]
Budget = 100
#*****************************************************************


#*****************************************************************
# Model
Investment = Model(HiGHS.Optimizer)

@variable(Investment, x[1:I], Bin) # 1 if investment is made in i

# Maximize est-profit
@objective(Investment, Max,
           sum(EstProfitFactor[i]*x[i] for i=1:I) )
```

```
# budget constraint
@constraint(Investment,
            sum(CapitalRequirements[i]*x[i] for i=1:I) <= Budget)

print(Investment)
#*****************************************************************************


#*****************************************************************************
# solve
optimize!(Investment)
#*****************************************************************************


#*****************************************************************************
# Report results
println("-------------------------------------");
if termination_status(Investment) == MOI.OPTIMAL
    println("RESULTS:")
    println("objective = $(objective_value(Investment))")
    for i in 1:I
        println("Investment $(Investments[i]) $(round(Int8,value(x[i])))")
    end
else
  println("  No solution")
end
println("-------------------------------------");
#*****************************************************************************
```

Now a number of extra requirements are made. This illustrates the possibilities of logic being implemented in constraints. These constraints are the only changes to the model.

New constraints:

- Investment 1 and 4 cannot both be made:

$$x_1 + x_4 \leq 1$$

- Investment 6 and 8 cannot both be made:

$$x_6 + x_8 \leq 1$$

- Investment 6 and 9 can only be made if either investment 2 or 3, or both, are made.

$$x_6 + x_9 \leq 2 \cdot (x_2 + x_3)$$

The full model in Julia/JuMP, available with the name

`StartupFund2.jl`

from the book web-site, is given below:

```
#*************************************************************************
# Startup Fund 2 assignment, "Mathematical Programming Modelling" (42112)
using JuMP
using HiGHS
#*************************************************************************

#*************************************************************************
# PARAMETERS
Investments = [1 2 3 4 5 6 7 8 9]
I=length(Investments)

# Data
EstProfitFactor =     [1.7 1.4 1.3 2.1 1.9 1.8 1.5 2.2 1.6]
CapitalRequirements = [ 17  25  19  25  28  23  29  31  18]
Budget = 100
#*************************************************************************


#*************************************************************************
# Model
Investment = Model(HiGHS.Optimizer)

@variable(Investment, x[1:I], Bin) # 1 if investment is made in i

# Maximize est-profit
@objective(Investment, Max,
           sum(EstProfitFactor[i]*x[i] for i=1:I) )

# budget constraint
@constraint(Investment,
            sum(CapitalRequirements[i]*x[i] for i=1:I) <= Budget)

@constraint(Investment, x[1] + x[4] <= 1)
@constraint(Investment, x[6] + x[8] <= 1)
@constraint(Investment, x[6] + x[9] <= 2*(x[2] + x[3]))

print(Investment)
#*************************************************************************
```

```
#*********************************************************************
# solve
optimize!(Investment)
#*********************************************************************


#*********************************************************************
# Report results
println("-------------------------------------");
if termination_status(Investment) == MOI.OPTIMAL
    println("RESULTS:")
    println("objective = $(objective_value(Investment))")
    for i in 1:I
        println("Investment $(Investments[i]) $(round(Int8,value(x[i])))")
    end
else
  println("  No solution")
end
println("-------------------------------------");
#*********************************************************************
```