Chair Transport

The Chair Transport problem is a classic transport problem of goods from production to customer (depot)

Problem

- Minimize transportation costs.

Sets

- $p \in Plants = \{P1, P2\}$
- $d \in Depots = \{D1, D2, D3, D4\}$

Parameters

- $PlantCapacity_p$: Maximal chair production capacity for plant p
- $DepotRequirement_d$: No of chairs needed at depot d
- F: The cost of transporting one chair one kilometer
- $PDdist_{p,d}$: Distance in kilometers from plant p to depot d

Decision variables

- Amount of chairs transported from plant p to depot d: $x_{p,d} \geq 0$.

Model

Objective:

- Total transport costs minimized:

$$\sum_{p,d} F \cdot PDdist_{p,d} \cdot x_{p,d}$$

Constraints:

- Production capacity limit for plant p:

$$\sum_d x_{p,d} \leq PlantCapacity_p \ \forall p$$

- Chair demand for depot d:

$$\sum_p x_{p,d} \geq DepotRequirement_d \ foralld$$

The above model is a classic transportation model.

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

`ChairTransport.jl`

from the book web-site, is given below:

```
#*****************************************************************************
# Chair Transport, "Mathematical Programming Modelling" (42112)
using JuMP
using HiGHS
#*****************************************************************************

#*****************************************************************************
# PARAMETERS
Plants = ["P1", "P2"]             # set of plants
P = length(Plants) # P is the size of the Plants set
Depots = ["D1", "D2", "D3", "D4"]
D = length(Depots)
PlantCapacity = [7500, 8500] # plant capacity in chairs
DepotRequirement = [3250, 3500, 3500, 3000] # depot capacity in chairs
PDdist = [ 137  92  48 173; # Distance between Plants and Depots
            54  109 111  85]
F=0.0375
#*****************************************************************************


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

@variable(CT, x[1:P, 1:D] >= 0)
```

```
@objective(CT, Min, sum( PDdist[p,d] * F * x[p,d] for p=1:P, d=1:D)) # Min cost

@constraint(CT, [p=1:P],sum(x[p,d] for d=1:D)<=PlantCapacity[p]) # Plant Capacity limit

@constraint(CT, [d=1:D],sum(x[p,d] for p=1:P)>=DepotRequirement[d]) # Depot req

print(CT) # print model to screen (only usable for small models)
#*****************************************************************************

#*****************************************************************************
# solve
optimize!(CT)
println("Termination status: $(termination_status(CT))")
#*****************************************************************************

#*****************************************************************************
if termination_status(CT) == MOI.OPTIMAL
    println("Optimal objective value: $(objective_value(CT))")
    println("Solution:")
    for p = 1:P
      for d in 1:D
        println("  $(Plants[p]) $(Depots[d]) = $(value(x[p,d]))")
      end
    end
else
    println("No optimal solution available")
end
#*****************************************************************************
```