8

October 1, 2023

```
[ ]: using JuMP
     using CPLEX
```

```
[ ]: model = Model(CPLEX.Optimizer)
```

```
A JuMP Model
Feasibility problem with:
Variables: 0
Model mode: AUTOMATIC
CachingOptimizer state: EMPTY_OPTIMIZER
Solver name: CPLEX
```

```
[ ]: demand = [1500, 2100, 1800, 1950]
```

```
4-element Vector{Int64}:
 1500
 2100
 1800
 1950
```

```
[ ]: regular_hrs = 480
     max_overtime = 80
     production_rate = 160
     max_subcontract = 500
     regular_pay = 50
     overtime_pay = 55
     subcontracting_cost = 9000
     backlog_cost = 1200
     inventory_holding_cost = 50
     hiring_cost = 1000
     firing_cost = 1200
```

```
1200
```

```
[ ]: num_init_workers = 600
     inventory_start = 200
     inventory_end = 300
```

```
300
```

[ ]:

```julia
@variable(model, total_overtime_hrs[1:4], lower_bound = 0, Int)
```

```
4-element Vector{VariableRef}:
 total_overtime_hrs[1]
 total_overtime_hrs[2]
 total_overtime_hrs[3]
 total_overtime_hrs[4]
```

[ ]:

```julia
@variable(model, w[1:4], lower_bound = 0, Int) # Number of workers in each quarter
```

```
4-element Vector{VariableRef}:
 w[1]
 w[2]
 w[3]
 w[4]
```

[ ]:

```julia
@variable(model, w_h[1:4], lower_bound = 0, Int) # Number of workers being hired in each quarter
```

```
4-element Vector{VariableRef}:
 w_h[1]
 w_h[2]
 w_h[3]
 w_h[4]
```

[ ]:

```julia
@variable(model, w_f[1:4], lower_bound = 0, Int) # Number of workers being fired in each quarter
```

```
4-element Vector{VariableRef}:
 w_f[1]
 w_f[2]
 w_f[3]
 w_f[4]
```

[ ]:

```julia
@variable(model, production[1:4], lower_bound = 0, Int)
```

```
4-element Vector{VariableRef}:
 production[1]
 production[2]
 production[3]
 production[4]
```

[ ]:

```julia
@variable(model, subcontract[1:4], lower_bound = 0, Int)
```

```
4-element Vector{VariableRef}:
 subcontract[1]
 subcontract[2]
 subcontract[3]
 subcontract[4]
```

```
[ ]: @variable(model, inventory[1:4], lower_bound = 0, Int)
```

```
4-element Vector{VariableRef}:
 inventory[1]
 inventory[2]
 inventory[3]
 inventory[4]
```

```
[ ]: @variable(model, backlog[1:4], lower_bound = 0, Int)
```

```
4-element Vector{VariableRef}:
 backlog[1]
 backlog[2]
 backlog[3]
 backlog[4]
```

```
[ ]: @constraint(model, w[1]== num_init_workers + w_h[1] - w_f[1]) # # relating number of workers to workers being hired and fired
```

$$w_1 - w\_h_1 + w\_f_1 = 600$$

```
[ ]: @constraint(model, production[1] + inventory_start + subcontract[1] == demand[1] + inventory[1] + backlog[1]) # Relating inventory with production subcontracting and backlog
```

$$\mathit{production}_1 + \mathit{subcontract}_1 - \mathit{inventory}_1 - \mathit{backlog}_1 = 1300$$

```
[ ]: @constraint(model, w[2:4]== w[1:3] + w_h[2:4] - w_f[2:4]) # relating number of workers to workers being hired and fired
```

$$[-w_1+w_2-w\_h_2+w\_f_2, -w_2+w_3-w\_h_3+w\_f_3, -w_3+w_4-w\_h_4+w\_f_4] \in \text{MathOptInterface.Zeros(3)}$$

```
[ ]: @constraint(model,production[2:4] + inventory[1:3] + subcontract[2:4] == demand[2:4] + inventory[2:4] + backlog[1:3] + backlog[2:4]) # Relating inventory with production subcontracting and backlog
```

$$[\mathit{production}_2+\mathit{subcontract}_2+\mathit{inventory}_1-\mathit{inventory}_2-\mathit{backlog}_1-\mathit{backlog}_2-2100, \mathit{production}_3+\mathit{subcontract}_3+\mathit{inve}$$

```
[ ]: @constraint(model, total_overtime_hrs <= max_overtime * w[1:4]) # Max overtime constraint
```

$$[\mathit{total\_overtime\_hrs}_1-80w_1, \mathit{total\_overtime\_hrs}_2-80w_2, \mathit{total\_overtime\_hrs}_3-80w_3, \mathit{total\_overtime\_hrs}_4-8$$

```
[ ]: @constraint(model, production <= w[1:4] * regular_hrs / production_rate +␣
      ↪total_overtime_hrs / production_rate) # Max production constraint
```

$$[-0.00625 total\_overtime\_hrs_1 - 3w_1 + production_1, -0.00625 total\_overtime\_hrs_2 - 3w_2 + production_2, -0.00625t$$

```
[ ]: @constraint(model, subcontract[1:4] .<= max_subcontract) # Max subcontracted␣
      ↪amount constraint
```

```
4-element Vector{ConstraintRef{Model, MathOptInterface.
 ↪ConstraintIndex{MathOptInterface.ScalarAffineFunction{Float64},␣
 ↪MathOptInterface.LessThan{Float64}}, ScalarShape}}:
 subcontract[1] <= 500
 subcontract[2] <= 500
 subcontract[3] <= 500
 subcontract[4] <= 500
```

```
[ ]: @constraint(model, inventory[4] == inventory_end) # final inventory should be␣
      ↪300 vaccum cleaners
```

$$inventory_4 = 300$$

```
[ ]: total_cost = sum(w .* regular_hrs .* regular_pay) + sum(hiring_cost .* w_h) +␣
      ↪sum(firing_cost .* w_f) + sum(inventory_holding_cost .* inventory) +␣
      ↪sum(backlog_cost .* backlog) + sum(subcontracting_cost .* subcontract) +␣
      ↪sum(overtime_pay .* total_overtime_hrs)
```

$$24000w_1 + 24000w_2 + 24000w_3 + 24000w_4 + 1000w\_h_1 + 1000w\_h_2 + 1000w\_h_3 + 1000w\_h_4 + 1200w\_f_1 + 1200w\_f_2 + 1$$

```
[ ]: @objective(model, Min, total_cost)
```

$$24000w_1 + 24000w_2 + 24000w_3 + 24000w_4 + 1000w\_h_1 + 1000w\_h_2 + 1000w\_h_3 + 1000w\_h_4 + 1200w\_f_1 + 1200w\_f_2 + 1$$

```
[ ]: optimize!(model)
```

```
Lift and project cuts applied:  1

Root node processing (before b&c):
  Real time             =    0.05 sec. (0.45 ticks)
Parallel b&c, 8 threads:
  Real time             =    0.00 sec. (0.00 ticks)
  Sync time (average)   =    0.00 sec.
  Wait time (average)   =    0.00 sec.
                        ------------
Total (root+branch&cut) =    0.05 sec. (0.45 ticks)
Version identifier: 22.1.1.0 | 2022-11-26 | 9160aff4d
Tried aggregator 2 times.
MIP Presolve eliminated 9 rows and 2 columns.
MIP Presolve added 5 rows and 5 columns.
Aggregator did 1 substitutions.
Reduced MIP has 19 rows, 34 columns, and 66 nonzeros.
Reduced MIP has 0 binaries, 34 generals, 0 SOSs, and 0 indicators.
Presolve time = 0.00 sec. (0.05 ticks)
Found incumbent of value 7.3939200e+07 after 0.00 sec. (0.12 ticks)
Tried aggregator 1 time.
Detecting symmetries…
MIP Presolve eliminated 5 rows and 5 columns.
MIP Presolve added 5 rows and 5 columns.
Reduced MIP has 19 rows, 34 columns, and 66 nonzeros.
Reduced MIP has 0 binaries, 34 generals, 0 SOSs, and 0 indicators.
Presolve time = 0.00 sec. (0.04 ticks)
MIP emphasis: balance optimality and feasibility.
MIP search method: dynamic search.
Parallel mode: deterministic, using up to 8 threads.
Root relaxation solution time = 0.00 sec. (0.05 ticks)

        Nodes                                         Cuts/
   Node  Left     Objective  IInf  Best Integer    Best Bound    ItCnt     Gap

*     0+    0                      7.39392e+07   15000.0000            99.98%
*     0+    0                      7.38201e+07   15000.0000            99.98%
      0     0   5.96996e+07    10   7.38201e+07   5.96996e+07       10   19.13%
*     0+    0                      5.97157e+07   5.96996e+07             0.03%
*     0+    0                      5.97156e+07   5.96996e+07             0.03%
      0     0   5.96996e+07    10   5.97156e+07      Fract: 1       11    0.03%
*     0+    0                      5.97005e+07   5.96996e+07             0.00%
```

[ ]:
```julia
@show value.(w) # Number of workers in each quarter
```

```
value.(w) = [620.0000000001232, 621.0, 621.0, 621.0]

4-element Vector{Float64}:
 620.0000000001232
 621.0
 621.0
 621.0
```

[ ]:
```julia
@show value.(inventory)
```

```
value.(inventory) = [560.0000000003697, 323.00000000036965, 387.0, 300.0]

4-element Vector{Float64}:
 560.0000000003697
 323.00000000036965
 387.0
 300.0
```

```
[ ]: @show value.(backlog)
```

```
value.(backlog) = [-0.0, -0.0, -0.0, 0.0]

4-element Vector{Float64}:
 -0.0
 -0.0
 -0.0
  0.0
```

```
[ ]: @show value.(w_h)
```

```
value.(w_h) = [20.000000000123226, 0.9999999998767743, -0.0, -0.0]

4-element Vector{Float64}:
 20.000000000123226
  0.9999999998767743
 -0.0
 -0.0
```

```
[ ]: @show value.(w_f)
```

```
value.(w_f) = [0.0, 0.0, 0.0, 0.0]

4-element Vector{Float64}:
 0.0
 0.0
 0.0
 0.0
```

```
[ ]: @show value.(subcontract)
```

```
value.(subcontract) = [0.0, -0.0, 0.9999999996303232, 0.0]

4-element Vector{Float64}:
  0.0
 -0.0
  0.9999999996303232
  0.0
```

```
[ ]: @show value.(production)
```

```
value.(production) = [1860.0000000003697, 1863.0, 1863.0, 1863.0]

4-element Vector{Float64}:
 1860.0000000003697
 1863.0
 1863.0
 1863.0
```

[ ]:

```
@show value.(total_overtime_hrs)
```

```
value.(total_overtime_hrs) = [0.0, 0.0, 0.0, 0.0]
```

```
4-element Vector{Float64}:
 0.0
 0.0
 0.0
 0.0
```

[ ]:

```
@show objective_value(model)
```

```
objective_value(model) = 5.970049999999967e7
```

```
5.970049999999967e7
```