CS 524 HW8 Sparsh Agarwal 9075905142

Q1.

In [56]:
```
# parameters for our problem
w = [2.500, 5.000, 2.268, 5.670]  # weights
v = [1, 5, 10, 25]  # values
V = 99               # value limit
n = length(w);       # number of items
using JuMP, Cbc

m = Model(solver=CbcSolver())
@variable(m, z[1:n]>=0, Int )
@constraint(m, sum( v[i]*z[i] for i=1:n) == V )
@objective(m, Min, sum( w[i]*z[i] for i=1:n) )

status = solve(m)

println(m)
println(status)
println()
println("z = ", getvalue(z) )
println("objective = ", getobjectivevalue(m) )
```

```
Min 2.5 z[1] + 5 z[2] + 2.268 z[3] + 5.67 z[4]
Subject to
 z[1] + 5 z[2] + 10 z[3] + 25 z[4] = 99
 z[i] ≥ 0, integer, ∀ i ∈ {1,2,3,4}

Optimal

z = [4.0, 0.0, 7.0, 1.0]
objective = 31.546
```

Q2.

In [57]:
```
w = [5, 6, 7, 6, 4, 6, 7, 3, 8, 5]  # weights
v = [2, 4, 5, 3, 3, 2, 3, 1, 2, 4]  # volume
W = 30                # weight limit
V = 15                # volume limit
n = length(w);        # number of items
using JuMP, Cbc

m = Model(solver=CbcSolver())
@variable(m, z[1:n], Bin )
@constraint(m, sum( w[i]*z[i] for i=1:n) <= W )
@constraint(m, sum( v[i]*z[i] for i=1:n) <= V )
@objective(m, Max, sum(z[i] for i=1:n) )

status = solve(m)

println(m)
println(status)
println()
println("z = ", getvalue(z) )
println("objective = ", getobjectivevalue(m) )
```

```
Max z[1] + z[2] + z[3] + z[4] + z[5] + z[6] + z[7] + z[8] + z[9] + z[1
0]
Subject to
 5 z[1] + 6 z[2] + 7 z[3] + 6 z[4] + 4 z[5] + 6 z[6] + 7 z[7] + 3 z[8]
+ 8 z[9] + 5 z[10] ≤ 30
 2 z[1] + 4 z[2] + 5 z[3] + 3 z[4] + 3 z[5] + 2 z[6] + 3 z[7] + z[8] +
2 z[9] + 4 z[10] ≤ 15
 z[i] ∈ {0,1} ∀ i ∈ {1,2,…,9,10}

Optimal

z = [1.0, 0.0, 0.0, 1.0, 1.0, 1.0, 0.0, 1.0, 0.0, 1.0]
objective = 6.0
```

Q3.

In [58]:
```
pl = [10000, 8000, 9000, 6000] #production limit
fc = [9000000, 5000000, 3000000, 1000000]
pc = [1000, 1700, 2300, 2900] #production cost per computer

n = length(fc);

using JuMP, Cbc

m = Model(solver = CbcSolver())

@variable(m, x[1:4] >= 0)
@variable(m, z[1:4], Bin)

@constraint(m, sum(x) <= 20000)
@constraint(m, x .<= 10000*z)   # if x>0 then z=1
# @constraint(m, x .>= 5*(1-z))
@constraint(m, x .<= pl)

@objective(m, Max, sum(x.*3500-((z.*fc)+(x.*pc))))

solve(m)

xopt = getvalue(x)
println(xopt[1], " Plant 1 production")
println(xopt[2], " Plant 2 production")
println(xopt[3], " Plant 3 production")
println(xopt[4], " Plant 4 production")
println()
println("\$", getobjectivevalue(m), " of net profit")
```

```
10000.0 Plant 1 production
8000.0 Plant 2 production
0.0 Plant 3 production
2000.0 Plant 4 production

$2.56e7 of net profit
```

Q4.

In [59]:
```
mini = [3, 2, 9, 5, 12, 4] #minimum investment
maxi = [27, 12, 35, 15, 46, 18] #maximum investment
er = [13, 9, 17, 10, 22, 12] #expected return

n = length(mini);

using JuMP, Gurobi

m = Model(solver = GurobiSolver())

@variable(m, x[1:n] >= 0) #investment
# @variable(m, epsl >= 0)
@variable(m, z[1:n], Bin)

@constraint(m, sum(x) <= 80)
@constraint(m, x .<= 100*z)   # if x>0 then z=1
@constraint(m, x[5]<=x[2]+x[4]+x[6])
@constraint(m, (x[3]-mini[3])<=((maxi[3]-mini[3])*z[6] - (1-z[6])))
@constraint(m, (z.*mini).<= x )
@constraint(m, x .<= (z.*maxi))

@objective(m, Max, sum(x*0.01.*er))

solve(m)

xopt = getvalue(x)
println(xopt[1], " Option 1 investment")
println(xopt[2], " Option 2 investment")
println(xopt[3], " Option 3 investment")
println(xopt[4], " Option 4 investment")
println(xopt[5], " Option 5 investment")
println(xopt[6], " Option 6 investment")
println()
println("\$", getobjectivevalue(m), " Investment return in million")
println("\$", sum(xopt), " Total investment in million")
```

```
Academic license - for non-commercial use only
Optimize a model with 21 rows, 12 columns and 48 nonzeros
Variable types: 6 continuous, 6 integer (6 binary)
Coefficient statistics:
  Matrix range     [1e+00, 1e+02]
  Objective range  [9e-02, 2e-01]
  Bounds range     [1e+00, 1e+00]
  RHS range        [8e+00, 8e+01]
Found heuristic solution: objective -0.0000000
Presolve removed 6 rows and 0 columns
Presolve time: 0.00s
Presolved: 15 rows, 12 columns, 36 nonzeros
Variable types: 6 continuous, 6 integer (6 binary)

Root relaxation: objective 1.351000e+01, 5 iterations, 0.00 seconds

    Nodes    |    Current Node    |     Objective Bounds      |     Wor
k
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node
Time

     0     0   13.51000    0    1   -0.00000   13.51000      -     -
0s
H    0     0                       13.5000000   13.51000  0.07%     -
0s
     0     0     cutoff    0         13.50000   13.50000  0.00%     -
0s

Explored 1 nodes (12 simplex iterations) in 0.00 seconds
Thread count was 4 (of 4 available processors)

Solution count 2: 13.5 -0

Optimal solution found (tolerance 1.00e-04)
Best objective 1.350000000000e+01, best bound 1.350000000000e+01, gap
0.0000%
0.0 Option 1 investment
0.0 Option 2 investment
35.0 Option 3 investment
5.0 Option 4 investment
22.5 Option 5 investment
17.5 Option 6 investment

$13.5 Investment return in million
$80.0 Total investment in million
```