using CPLEX;

 int n=...; // number of warehouses

 int m=...; // number of branches

 int k=...; // possible sizes

 int o=...; // possible levels

 range PWAREHOUSES=1..n;

 range BRANCHES=1..m;

 range Size=1..k;

 range Levels=1..o;

 float F[PWAREHOUSES]=...;

 float p[PWAREHOUSES]=...;

 float L[PWAREHOUSES]=...;

 float d[BRANCHES]=...;

 float T[PWAREHOUSES][BRANCHES]=...;

 float A[Size]=...;

 float V[Size]=...;

 float Units[Size]=...;

 //variables

 dvar boolean x[PWAREHOUSES][Size];

 dvar boolean y[PWAREHOUSES][BRANCHES][Levels];

 dvar float+ z[PWAREHOUSES][BRANCHES][Size][Levels];

 minimize sum(w in PWAREHOUSES) ((F[w]+L[w])\*sum(s in Size)(A[s]\*x[w][s]))

 +sum( w in PWAREHOUSES, s in Size) (V[s]\*sum(b in BRANCHES) d[b]\*(z[w][b][s][1]\*(1-p[w])

 +z[w][b][s][2]\*sum(e in PWAREHOUSES: e!=w)p[e]\*y[e][b][1]))

 +sum(b in BRANCHES, w in PWAREHOUSES)(d[b]\*T[w][b]\*(y[w][b][1]\*(1-p[w])

 +y[w][b][2]\*sum(e in PWAREHOUSES: e!=w)p[e]\*y[e][b][1]));

 subject to{

 warehouses\_limit\_constraint:

 sum(w in PWAREHOUSES, s in Size) x[w][s] <=5;

 Eight\_constraint:

 x[8][3]==1;

 Sizteenth\_constraint:

 x[16][3]==1;

 forall(w in PWAREHOUSES)

 one\_size\_constraint:

 sum(s in Size) x[w][s]<=1;

 forall(b in BRANCHES)

 forall(r in Levels)

 one\_warehouse\_constraint:

 sum(w in PWAREHOUSES) y[w][b][r]==1;

 forall(w in PWAREHOUSES)

 forall(b in BRANCHES)

 backup\_warehouse\_constraint:

 sum(r in Levels) y[w][b][r]<=1;

 forall(w in PWAREHOUSES)

 Capacity\_constraint:

 sum(b in BRANCHES) d[b]\*(y[w][b][1]+y[w][b][2]\*sum(e in PWAREHOUSES: e!=w)p[e]\*y[e][b][1])

 <= sum(s in Size) x[w][s]\*Units[s];

 forall(w in PWAREHOUSES, b in BRANCHES, s in Size, r in Levels)

 Linearity\_constraint1:

 z[w][b][s][r] <= x[w][s];

 forall(w in PWAREHOUSES, b in BRANCHES, s in Size, r in Levels)

 Linearity\_constraint2:

 z[w][b][s][r] <= y[w][b][r];

 forall(w in PWAREHOUSES, b in BRANCHES, s in Size, r in Levels)

 Linearity\_constraint3:

 z[w][b][s][r] >= x[w][s]+y[w][b][r]-1;

 }