```
library("Matrix")
library("igraph")
library("rdist")
library("gurobi")
library("plyr")

I = 10
J = 22
T = 8
D = 5
C = 2
K = 2
M = 1000

H = c(2,    2,  3,  2,  2,  2,  2,  2,  2,  2,  2,  2,  2,  2,  2,  2,  2,  2,  2,  3,  3)
Cohort = matrix(c(1,    0,
                  1,    0,
                  1,    0,
                  1,    0,
                  0,    1,
                  0,    1,
                  1,    0,
                  1,    0,
                  0,    1,
                  0,    1,
                  0,    1,
                  0,    1,
                  1,    0,
                  1,    0,
                  1,    0,
                  1,    0,
                  1,    0,
                  1,    0,
                  1,    0,
                  1,    0,
                  0,    1,
                  1,    0), nrow=J, ncol=C, byrow=T)
P = matrix(c(0,     0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  1,  1,  0,  0,  0,  0,  1,  1,  0,  0,
             0,     0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  1,  1,  0,  0,  0,  0,  1,  1,  0,  0,
             0,     0,  1,  1,  0,  0,  1,  1,  1,  1,  1,  1,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
             0,     0,  1,  1,  0,  0,  1,  1,  1,  1,  1,  1,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
             0,     0,  0,  0,  1,  1,  0,  0,  0,  0,  0,  0,  0,  0,  1,  1,  1,  1,  0,  0,  0,  0,
             1,     1,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  1,  1,
             0,     0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  1,  1,  0,  0,  0,  0,  0,  1,
             0,     0,  0,  0,  0,  0,  0,  0,  0,  0,  1,  1,  0,  0,  1,  1,  1,  1,  0,  0,  0,  0,
             1,     1,  0,  0,  0,  0,  0,  0,  1,  1,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  1,
             1,     1,  0,  0,  1,  1,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  1), nrow=I, ncol=J, byrow=T)

Amat = Matrix(0, nrow=(4 + J + 3*I*J + 2*I + ((J-2)*I)/2 + I*C + I*D + (T - 1)*D*C + (T - 2)*D*C + I*(T-1)*D + I*(T-2)*D + D*(J-2)/2 + J*C
bvec = c()
dir = c()

cvec = c()
for (i in 1:I){
  for (j in 1:J){
    for (t in 1:T){
      for (d in 1:D){
        for (c in 1:C){
          if (t == 1) {
            cvec = c(cvec, 1)
          }
          else if (H[j] == 2) {
            if (t == 3 || t == 4) {
              cvec = c(cvec, 100)
            }
            else if (t == 7 || t == 8) {
              cvec = c(cvec, 1)
            }
            else {
              cvec = c(cvec, 0)
            }
          }
          else if (H[j] == 3) {
            if (t == 2 || t == 3 || t == 4) {
              cvec = c(cvec, 100)
            }
            else if (t == 6 || t == 7 || t == 8) {
              cvec = c(cvec, 1)
            }
            else {
              cvec = c(cvec, 0)
            }
          }
        }
      }
    }
  }
}

for (i in 1:I){
  for (k in 1:K){
```

```
    if (d == 1) {
      cvec = c(cvec, 3)
    } else if(d == 5) {
      cvec = c(cvec, 3)
    } else {
      cvec = c(cvec, 1)
    }
  }
}

for (i in 1:I){
  for (d in 1:D){
    if (d == 1) {
      cvec = c(cvec, 3)
    } else if(d == 5) {
      cvec = c(cvec, 3)
    } else {
      cvec = c(cvec, 1)
    }
  }
}

cvec = c(cvec, rep(0,I*J))

X_Vars = I*J*T*D*C
Z_Vars = I*D + X_Vars
Y_Vars = I*K + Z_Vars
W_Vars = I*J + Y_Vars

#constraint (1) - Match number of courses
Amat[1,(Y_Vars+1):W_Vars] = c(rep(1, I*J))
bvec = c(bvec, 22)
dir = c(dir, '=')

#constraint (1) - Match number of courses
Amat[2,1:X_Vars] = c(rep(1, I*J*T*D*C))
bvec = c(bvec, 22)
dir = c(dir, '=')

#constraint (15)
row_start = 3
row_end = ((J-2)*I)/2 + 2

I_Count = 1
J_Count = 1
for(g in row_start:row_end){
  Wij = array(0,dim=c(I,J))
  Wij[I_Count, J_Count] = 1
  Wij[I_Count, J_Count + 1] = -1

  Wij.vector = integer(I*J)
  vector.position = 1
  for (i in 1:I){
    for (j in 1:J){
      count = count + Wij[i,j]
      Wij.vector[vector.position] = Wij[i,j]
      vector.position = vector.position + 1
    }
  }

  Amat[g,(Y_Vars+1):W_Vars] = Wij.vector
  bvec = c(bvec, 0)
  dir = c(dir, '=')

  if (J_Count == 19){
    J_Count = 1
    I_Count = I_Count + 1
  }
  else {
    J_Count = J_Count + 2
  }
}

#constraint (10)
J_Count = 1
count = 0

row_start = 1 + row_end
row_end = row_end + (J)
for(g in row_start:row_end){
  Wij = array(0,dim=c(I,J))
  for (i in 1:I) {
    Wij[i, J_Count] = 1
  }

  Wij.vector = integer(I*J)
  vector.position = 1
  for (i in 1:I){
    for (j in 1:J){
      count = count + Wij[i,j]
      Wij.vector[vector.position] = Wij[i,j]
```

```
      vector.position = vector.position + 1
    }
  }

  Amat[g,(Y_Vars+1):W_Vars] = Wij.vector
  bvec = c(bvec, 1)
  dir = c(dir, '=')

  J_Count = J_Count + 1
}

#constraint (11)
I_Count = 1
count = 0

row_start = 1 + row_end
row_end = row_end + (I)
for(g in row_start:row_end){
  Wij = array(0,dim=c(I,J))
  for (j in seq(1, J, 2)) {
    Wij[I_Count, j] = 1
    if (j == 21) {
      Wij[I_Count, j + 1] = 1
    }
  }

  Wij.vector = integer(I*J)
  vector.position = 1
  for (i in 1:I){
    for (j in 1:J){
      count = count + Wij[i,j]
      Wij.vector[vector.position] = Wij[i,j]
      vector.position = vector.position + 1
    }
  }

  Yik = array(0,dim=c(I,K))
  Yik[I_Count,1] = -1
  Yik[I_Count,2] = -2

  Yik.vector = integer(I*K)
  vector.position = 1
  for (i in 1:I){
    for (k in 1:K){
      Yik.vector[vector.position] = Yik[i,k]
      vector.position = vector.position + 1
    }
  }

  Amat[g,(Y_Vars+1):W_Vars] = Wij.vector
  Amat[g,(Z_Vars+1):Y_Vars] = Yik.vector
  bvec = c(bvec, 0)
  dir = c(dir, '=')

  if (I_Count == 10){
    I_Count = 1
  }
  else {
    I_Count = I_Count + 1
  }
}

#constraint (12)
I_Count = 1
D_Count = 1
count = 0

row_start = 1 + row_end
row_end = row_end + (I*D)
for(g in row_start:row_end){
  Xijtdc = array(0,dim=c(I,J,T,D,C))
  for (j in 1:J) {
    for (t in 1:T) {
      for (c in 1:C) {
        Xijtdc[I_Count, j, t, D_Count, c] = 1
      }
    }
  }

  count = 0
  Xijtdc.vector = integer(I*J*T*D*C)
  vector.position = 1
  for (i in 1:I){
    for (j in 1:J){
      for (t in 1:T){
        for (d in 1:D){
          for (c in 1:C){
            count = count + Xijtdc[i,j,t,d,c]
            Xijtdc.vector[vector.position] = Xijtdc[i,j,t,d,c]
            vector.position = vector.position + 1
          }
```

```
        }
      }
    }
  }

  Zid = array(0,dim=c(I,D))
  Zid[I_Count, D_Count] = -M

  Zid.vector = integer(I*D)
  vector.position = 1
  for (i in 1:I){
    for (d in 1:D){
      count = count - Zid[i,d]/M
      Zid.vector[vector.position] = Zid[i,d]
      vector.position = vector.position + 1
    }
  }

  Amat[g,1:X_Vars] = Xijtdc.vector
  Amat[g,(X_Vars+1):Z_Vars] = Zid.vector
  bvec = c(bvec, 0)
  dir = c(dir, '<=')

  if (D_Count == 5){
    D_Count = 1
    I_Count = I_Count + 1
  }
  else {
    D_Count = D_Count + 1
  }
}

#constraint (13)
I_Count = 1
J_Count = 1
count = 0

row_start = 1 + row_end
row_end = row_end + (I*J)
for(g in row_start:row_end){
  Xijtdc = array(0,dim=c(I,J,T,D,C))
  for (t in 1:T) {
    for (d in 1:D) {
      for (c in 1:C) {
        Xijtdc[I_Count, J_Count, t, d, c] = 1
      }
    }
  }

  count = 0
  Xijtdc.vector = integer(I*J*T*D*C)
  vector.position = 1
  for (i in 1:I){
    for (j in 1:J){
      for (t in 1:T){
        for (d in 1:D){
          for (c in 1:C){
            count = count + Xijtdc[i,j,t,d,c]
            Xijtdc.vector[vector.position] = Xijtdc[i,j,t,d,c]
            vector.position = vector.position + 1
          }
        }
      }
    }
  }

  Wij = array(0,dim=c(I,J))
  Wij[I_Count, J_Count] = -M

  Wij.vector = integer(I*J)
  vector.position = 1
  for (i in 1:I){
    for (j in 1:J){
      count = count - Wij[i,j]/M
      Wij.vector[vector.position] = Wij[i,j]
      vector.position = vector.position + 1
    }
  }

  Amat[g,1:X_Vars] = Xijtdc.vector
  Amat[g,(Y_Vars+1):W_Vars] = Wij.vector
  bvec = c(bvec, 0)
  dir = c(dir, '<=')

  if (J_Count == 22){
    J_Count = 1
    I_Count = I_Count + 1
  }
  else {
    J_Count = J_Count + 1
  }
```

```
}

#constraint (16)
I_Count = 1
J_Count = 1
count = 0

row_start = 1 + row_end
row_end = row_end + (I*J)
for(g in row_start:row_end){
  Xijtdc = array(0,dim=c(I,J,T,D,C))
  for (t in 1:T) {
    for (d in 1:D) {
      for (c in 1:C) {
        Xijtdc[I_Count, J_Count, t, d, c] = 1
      }
    }
  }

  count = 0
  Xijtdc.vector = integer(I*J*T*D*C)
  vector.position = 1
  for (i in 1:I){
    for (j in 1:J){
      for (t in 1:T){
        for (d in 1:D){
          for (c in 1:C){
            count = count + Xijtdc[i,j,t,d,c]
            Xijtdc.vector[vector.position] = Xijtdc[i,j,t,d,c]
            vector.position = vector.position + 1
          }
        }
      }
    }
  }

  Amat[g,1:X_Vars] = Xijtdc.vector
  bvec = c(bvec, 1)
  dir = c(dir, '<=')

  if (J_Count == 22){
    J_Count = 1
    I_Count = I_Count + 1
  }
  else {
    J_Count = J_Count + 1
  }
}

#constraint (2)
I_Count = 1
count = 0

row_start = 1 + row_end
row_end = row_end + (I)
for(g in row_start:row_end){
  Wij = array(0,dim=c(I,J))
  for (j in 1:J) {
    Wij[I_Count, j] = 1
  }

  Wij.vector = integer(I*J)
  vector.position = 1
  for (i in 1:I){
    for (j in 1:J){
      count = count + Wij[i,j]
      Wij.vector[vector.position] = Wij[i,j]
      vector.position = vector.position + 1
    }
  }

  Amat[g,(Y_Vars+1):W_Vars] = Wij.vector
  bvec = c(bvec, 4)
  dir = c(dir, '<=')

  if (I_Count == 10){
    I_Count = 1
  }
  else {
    I_Count = I_Count + 1
  }
}

#constraint (2)
I_Count = 1
C_Count = 1
count = 0

row_start = 1 + row_end
row_end = row_end + (I*C)
for(g in row_start:row_end){
```

```
  Wij = array(0,dim=c(I,J))
  for (j in seq(1, J, 2)) {
    Wij[I_Count, j] = Cohort[j, C_Count]
    if (j == 21) {
      Wij[I_Count, j + 1] = Cohort[j + 1, C_Count]
    }
  }

  Wij.vector = integer(I*J)
  vector.position = 1
  for (i in 1:I){
    for (j in 1:J){
      count = count + Wij[i,j]
      Wij.vector[vector.position] = Wij[i,j]
      vector.position = vector.position + 1
    }
  }

  Amat[g,(Y_Vars+1):W_Vars] = Wij.vector
  bvec = c(bvec, 1)
  dir = c(dir, '<=')

  if (I_Count == 10){
    I_Count = 1
    C_Count = C_Count + 1
  }
  else {
    I_Count = I_Count + 1
  }
}

#constraint (14)
count = 0
row_start = 1 + row_end
row_end = row_end + (I*J)
vector.position = Y_Vars + 1
for(g in row_start:row_end){
  count = count + 1
  Amat[g,vector.position] = 1
  vector.position = vector.position + 1
}
for (i in 1:I){
  for (j in 1:J){
    bvec = c(bvec, P[i,j])
    dir = c(dir, '<=')
  }
}

#constraint (4)
T_Count = 1
D_Count = 1
C_Count = 1
count = 0

row_start = 1 + row_end
row_end = row_end + ((T-1)*D*C)
for(g in row_start:row_end){
  Xijtdc = array(0,dim=c(I,J,T,D,C))

  for (i in 1:I) {
    for (j in 1:J) {
      if (H[j] == 2) {
        Xijtdc[i,j,T_Count,D_Count,C_Count] = 1
      }
      Xijtdc[i,j,T_Count + 1,D_Count,C_Count] = 1
      if(C_Count==2){
        if(j == 13 || j == 14 || j == 19 || j == 20 || j == 22){
          Xijtdc[i,j,T_Count + 1,D_Count,1] = 1
        }
      }
    }
  }

  count = 0
  Xijtdc.vector = integer(I*J*T*D*C)
  vector.position = 1
  for (i in 1:I){
    for (j in 1:J){
      for (t in 1:T){
        for (d in 1:D){
          for (c in 1:C){
            count = count + Xijtdc[i,j,t,d,c]
            Xijtdc.vector[vector.position] = Xijtdc[i,j,t,d,c]
            vector.position = vector.position + 1
          }
        }
      }
    }
  }

  Amat[g,1:X_Vars] = Xijtdc.vector
```

```
  bvec = c(bvec, 1)
  dir = c(dir, '<=')

  if (C_Count == 2){
    C_Count = 1
    D_Count = D_Count + 1
  }
  else {
    C_Count = C_Count + 1
  }
  if (D_Count == 6) {
    D_Count = 1
    T_Count = T_Count + 1
  }
}

#constraint (4)
T_Count = 1
D_Count = 1
C_Count = 1
count = 0

row_start = 1 + row_end
row_end = row_end + ((T-2)*D*C)
for(g in row_start:row_end){
  Xijtdc = array(0,dim=c(I,J,T,D,C))

  for (i in 1:I) {
    for (j in 1:J) {
      if (H[j] == 3) {
        Xijtdc[i,j,T_Count,D_Count,C_Count] = 1
      }
      Xijtdc[i,j,T_Count + 1,D_Count,C_Count] = 1
      Xijtdc[i,j,T_Count + 2,D_Count,C_Count] = 1
      if(C_Count==2){
        if(j == 13 || j == 14 || j == 19 || j == 20 || j == 22){
          Xijtdc[i,j,T_Count + 1,D_Count,1] = 1
          Xijtdc[i,j,T_Count + 2,D_Count,1] = 1
        }
      }
    }
  }

  count = 0
  Xijtdc.vector = integer(I*J*T*D*C)
  vector.position = 1
  for (i in 1:I){
    for (j in 1:J){
      for (t in 1:T){
        for (d in 1:D){
          for (c in 1:C){
            count = count + Xijtdc[i,j,t,d,c]
            Xijtdc.vector[vector.position] = Xijtdc[i,j,t,d,c]
            vector.position = vector.position + 1
          }
        }
      }
    }
  }

  Amat[g,1:X_Vars] = Xijtdc.vector
  bvec = c(bvec, 1)
  dir = c(dir, '<=')

  if (C_Count == 2){
    C_Count = 1
    D_Count = D_Count + 1
  }
  else {
    C_Count = C_Count + 1
  }
  if (D_Count == 6) {
    D_Count = 1
    T_Count = T_Count + 1
  }
}

#constraint (3)
I_Count = 1
T_Count = 1
D_Count = 1
count = 0

row_start = 1 + row_end
row_end = row_end + ((T-1)*D*I)
for(g in row_start:row_end){
  Xijtdc = array(0,dim=c(I,J,T,D,C))

  for (j in 1:J) {
    for (c in 1:C) {
      if (H[j] == 2) {
```

```
        Xijtdc[I_Count,j,T_Count,D_Count,c] = 1
      }
      Xijtdc[I_Count,j,T_Count + 1,D_Count,c] = 1
    }
  }

  count = 0
  Xijtdc.vector = integer(I*J*T*D*C)
  vector.position = 1
  for (i in 1:I){
    for (j in 1:J){
      for (t in 1:T){
        for (d in 1:D){
          for (c in 1:C){
            count = count + Xijtdc[i,j,t,d,c]
            Xijtdc.vector[vector.position] = Xijtdc[i,j,t,d,c]
            vector.position = vector.position + 1
          }
        }
      }
    }
  }

  Amat[g,1:X_Vars] = Xijtdc.vector
  bvec = c(bvec, 1)
  dir = c(dir, '<=')

  if (D_Count == 5){
    D_Count = 1
    T_Count = T_Count + 1
  }
  else {
    D_Count = D_Count + 1
  }
  if (T_Count == 8) {
    T_Count = 1
    I_Count = I_Count + 1
  }
}

#constraint (3)
I_Count = 1
T_Count = 1
D_Count = 1
count = 0

row_start = 1 + row_end
row_end = row_end + ((T-2)*D*I)
for(g in row_start:row_end){
  Xijtdc = array(0,dim=c(I,J,T,D,C))

  for (j in 1:J) {
    for (c in 1:C) {
      if (H[j] == 3) {
        Xijtdc[I_Count,j,T_Count,D_Count,c] = 1
      }
      Xijtdc[I_Count,j,T_Count + 1,D_Count,c] = 1
      Xijtdc[I_Count,j,T_Count + 2,D_Count,c] = 1
    }
  }


  count = 0
  Xijtdc.vector = integer(I*J*T*D*C)
  vector.position = 1
  for (i in 1:I){
    for (j in 1:J){
      for (t in 1:T){
        for (d in 1:D){
          for (c in 1:C){
            count = count + Xijtdc[i,j,t,d,c]
            Xijtdc.vector[vector.position] = Xijtdc[i,j,t,d,c]
            vector.position = vector.position + 1
          }
        }
      }
    }
  }

  Amat[g,1:X_Vars] = Xijtdc.vector
  bvec = c(bvec, 1)
  dir = c(dir, '<=')

  if (D_Count == 5){
    D_Count = 1
    T_Count = T_Count + 1
  }
  else {
    D_Count = D_Count + 1
  }
  if (T_Count == 7) {
```

```
    T_Count = 1
    I_Count = I_Count + 1
  }
}

#constraint (5)
T_Count = 1
D_Count = 1
C_Count = 1
count = 0

row_start = 1 + row_end
row_end = row_end + (T*D*C)
for(g in row_start:row_end){
  Xijtdc = array(0,dim=c(I,J,T,D,C))

  for (i in 1:I) {
    for (j in 1:J) {
      Xijtdc[i,j,T_Count,D_Count,C_Count] = 1
    }
  }

  Xijtdc.vector = integer(I*J*T*D*C)
  vector.position = 1
  for (i in 1:I){
    for (j in 1:J){
      for (t in 1:T){
        for (d in 1:D){
          for (c in 1:C){
            Xijtdc.vector[vector.position] = Xijtdc[i,j,t,d,c]
            vector.position = vector.position + 1
          }
        }
      }
    }
  }

  Amat[g,1:X_Vars] = Xijtdc.vector
  bvec = c(bvec, 1)
  dir = c(dir, '<=')

  if (C_Count == 2){
    C_Count = 1
    D_Count = D_Count + 1
  }
  else {
    C_Count = C_Count + 1
  }
  if (D_Count == 6) {
    D_Count = 1
    T_Count = T_Count + 1
  }
}

#constraint (9)
I_Count = 1
T_Count = 1
D_Count = 1
count = 0

row_start = 1 + row_end
row_end = row_end + (I*T*D)
for(g in row_start:row_end){
  Xijtdc = array(0,dim=c(I,J,T,D,C))

  for (j in 1:J) {
    for (c in 1:C) {
      Xijtdc[I_Count,j,T_Count,D_Count,c] = 1
    }
  }

  Xijtdc.vector = integer(I*J*T*D*C)
  vector.position = 1
  for (i in 1:I){
    for (j in 1:J){
      for (t in 1:T){
        for (d in 1:D){
          for (c in 1:C){
            Xijtdc.vector[vector.position] = Xijtdc[i,j,t,d,c]
            vector.position = vector.position + 1
          }
        }
      }
    }
  }

  Amat[g,1:X_Vars] = Xijtdc.vector
  bvec = c(bvec, 1)
  dir = c(dir, '<=')

  if (D_Count == 5){
```

```
    D_Count = 1
    T_Count = T_Count + 1
  }
  else {
    D_Count = D_Count + 1
  }
  if (T_Count == 9) {
    T_Count = 1
    I_Count = I_Count + 1
  }
}

#constraint (17)
row_start = 1 + row_end
row_end = row_end + (1)
for(g in row_start:row_end){
  Xijtdc = array(0,dim=c(I,J,T,D,C))

  for (i in 1:I) {
    for (j in 1:J) {
      for(d in 1:D){
        for(c in 1:C){
          if (H[j] == 2) {
            Xijtdc[i,j,8,d,c] = 1
          }
        }
      }
    }
  }

  count = 0
  Xijtdc.vector = integer(I*J*T*D*C)
  vector.position = 1
  for (i in 1:I){
    for (j in 1:J){
      for (t in 1:T){
        for (d in 1:D){
          for (c in 1:C){
            count = count + Xijtdc[i,j,t,d,c]
            Xijtdc.vector[vector.position] = Xijtdc[i,j,t,d,c]
            vector.position = vector.position + 1
          }
        }
      }
    }
  }

  Amat[g,1:X_Vars] = Xijtdc.vector
  bvec = c(bvec, 0)
  dir = c(dir, '=')
}

#constraint (18)
row_start = 1 + row_end
row_end = row_end + (1)
for(g in row_start:row_end){
  Xijtdc = array(0,dim=c(I,J,T,D,C))

  for (i in 1:I) {
    for (j in 1:J) {
      for(d in 1:D){
        for(c in 1:C){
          if (H[j] == 3) {
            Xijtdc[i,j,7,d,c] = 1
            Xijtdc[i,j,8,d,c] = 1
          }
        }
      }
    }
  }

  count = 0
  Xijtdc.vector = integer(I*J*T*D*C)
  vector.position = 1
  for (i in 1:I){
    for (j in 1:J){
      for (t in 1:T){
        for (d in 1:D){
          for (c in 1:C){
            count = count + Xijtdc[i,j,t,d,c]
            Xijtdc.vector[vector.position] = Xijtdc[i,j,t,d,c]
            vector.position = vector.position + 1
          }
        }
      }
    }
  }

  Amat[g,1:X_Vars] = Xijtdc.vector
  bvec = c(bvec, 0)
  dir = c(dir, '=')
```

```
}

#constraint (19)
D_Count = 1
J_Count = 1
count = 0

row_start = 1 + row_end
row_end = row_end + D*(J-2)/2
for(g in row_start:row_end){
  Xijtdc = array(0,dim=c(I,J,T,D,C))
  for (i in 1:I) {
    for(t in 1:T){
      for(c in 1:C){
        Xijtdc[i, J_Count, t, D_Count, c] = 1
        Xijtdc[i, J_Count + 1, t, D_Count, c] = 1
      }
    }
  }

  Xijtdc.vector = integer(I*J*T*D*C)
  vector.position = 1
  for (i in 1:I){
    for (j in 1:J){
      for (t in 1:T){
        for (d in 1:D){
          for (c in 1:C){
            count = count + Xijtdc[i,j,t,d,c]
            Xijtdc.vector[vector.position] = Xijtdc[i,j,t,d,c]
            vector.position = vector.position + 1
          }
        }
      }
    }
  }

  Amat[g,1:X_Vars] = Xijtdc.vector
  bvec = c(bvec, 1)
  dir = c(dir, '<=')

  if (D_Count == 5){
    D_Count = 1
    J_Count = J_Count + 2
  }
  else {
    D_Count = D_Count + 1
  }
}

#constraint (20)
J_Count = 1
C_Count = 1
count = 0

row_start = 1 + row_end
row_end = row_end + (J*C)
for(g in row_start:row_end){
  Xijtdc = array(0,dim=c(I,J,T,D,C))
  for (i in 1:I) {
    for (t in 1:T) {
      for (d in 1:D) {
        Xijtdc[i, J_Count, t, d, C_Count] = 1
      }
    }
  }

  Xijtdc.vector = integer(I*J*T*D*C)
  vector.position = 1
  for (i in 1:I){
    for (j in 1:J){
      for (t in 1:T){
        for (d in 1:D){
          for (c in 1:C){
            Xijtdc.vector[vector.position] = Xijtdc[i,j,t,d,c]
            vector.position = vector.position + 1
          }
        }
      }
    }
  }

  Amat[g,1:X_Vars] = Xijtdc.vector
  bvec = c(bvec, Cohort[J_Count, C_Count])
  dir = c(dir, '<=')

  if (C_Count == 2){
    C_Count = 1
    J_Count = J_Count + 1
  }
  else {
    C_Count = C_Count + 1
```

```
  }
}


#constraint (6)
T_Count = 1
D_Count = 1
count = 0

row_start = 1 + row_end
row_end = row_end + ((T-1)*D)
for(g in row_start:row_end){
  Xijtdc = array(0,dim=c(I,J,T,D,C))

  for (i in 1:I) {
    for (j in 1:J) {
      for (c in 1:C){
        if (j == 13 || j == 14 || j == 19 || j == 20) {
          Xijtdc[i,j,T_Count,D_Count,c] = 1
        }
        Xijtdc[i,j,T_Count + 1,D_Count,c] = 1
      }
    }
  }

  count = 0
  Xijtdc.vector = integer(I*J*T*D*C)
  vector.position = 1
  for (i in 1:I){
    for (j in 1:J){
      for (t in 1:T){
        for (d in 1:D){
          for (c in 1:C){
            count = count + Xijtdc[i,j,t,d,c]
            Xijtdc.vector[vector.position] = Xijtdc[i,j,t,d,c]
            vector.position = vector.position + 1
          }
        }
      }
    }
  }

  Amat[g,1:X_Vars] = Xijtdc.vector
  bvec = c(bvec, 1)
  dir = c(dir, '<=')

  if (D_Count == 5){
    D_Count = 1
    T_Count = T_Count + 1
  }
  else {
    D_Count = D_Count + 1
  }
}

#constraint (7)
T_Count = 1
D_Count = 1
count = 0

row_start = 1 + row_end
row_end = row_end + ((T-2)*D)
for(g in row_start:row_end){
  Xijtdc = array(0,dim=c(I,J,T,D,C))

  for (i in 1:I) {
    for (j in 1:J) {
      if (j == 22) {
        Xijtdc[i,j,T_Count,D_Count,1] = 1
      }
      Xijtdc[i,j,T_Count + 1,D_Count,2] = 1
      Xijtdc[i,j,T_Count + 2,D_Count,2] = 1
    }
  }

  Xijtdc.vector = integer(I*J*T*D*C)
  vector.position = 1
  for (i in 1:I){
    for (j in 1:J){
      for (t in 1:T){
        for (d in 1:D){
          for (c in 1:C){
            count = count + Xijtdc[i,j,t,d,c]
            Xijtdc.vector[vector.position] = Xijtdc[i,j,t,d,c]
            vector.position = vector.position + 1
          }
        }
      }
    }
  }
```

```
  Amat[g,1:X_Vars] = Xijtdc.vector
  bvec = c(bvec, 1)
  dir = c(dir, '<=')

  if (D_Count == 5){
    D_Count = 1
    T_Count = T_Count + 1
  }
  else {
    D_Count = D_Count + 1
  }
}

#constraint (8)
T_Count = 1
D_Count = 1
count = 0

row_start = 1 + row_end
row_end = row_end + (T*D)
for(g in row_start:row_end){
  Xijtdc = array(0,dim=c(I,J,T,D,C))

  for (i in 1:I) {
    for (j in 1:J) {
      Xijtdc[i,j,T_Count,D_Count,2] = 1
      if(j == 13 || j == 14 || j == 19 || j == 20 || j == 22){
        Xijtdc[i,j,T_Count,D_Count,1] = 1
      }
    }
  }

  Xijtdc.vector = integer(I*J*T*D*C)
  vector.position = 1
  for (i in 1:I){
    for (j in 1:J){
      for (t in 1:T){
        for (d in 1:D){
          for (c in 1:C){
            Xijtdc.vector[vector.position] = Xijtdc[i,j,t,d,c]
            vector.position = vector.position + 1
          }
        }
      }
    }
  }

  Amat[g,1:X_Vars] = Xijtdc.vector
  bvec = c(bvec, 1)
  dir = c(dir, '<=')

  if (D_Count == 5){
    D_Count = 1
    T_Count = T_Count + 1
  }
  else {
    D_Count = D_Count + 1
  }
}

myLP = list()
myLP$obj = cvec
myLP$A = Amat
myLP$sense = dir
myLP$rhs = bvec
myLP$vtypes = "B"

mysol = gurobi(myLP)
mysol$objval
mysol$x

### Lagrangian Relaxation ###

# Create objective function of subproblems
u = 100
uvec = c(rep(u,I + I*D + I*J))
cvec_LR = cvec
relaxed_Cons = Amat[125:404,]

for (l in 1:length(uvec)) {
  cvec_LR = cvec_LR + uvec[l]*relaxed_Cons[l,]
}

# Create A matrix of subproblems
Amat_LR = Amat[-125:-404,]
bvec_LR = bvec[-125:-404]
dir_LR = dir[-125:-404]

bvec_LR_Relaxed = (-1)*bvec[125:404]

# Subproblem Function
```

```
SP = function(obj_vec, Amat, dir, bvec, ub=NULL, lb=NULL){
  mySP1 = list()
  mySP1$modelsense = "min"
  mySP1$obj = obj_vec
  mySP1$A = Amat
  mySP1$sense = dir
  mySP1$rhs = bvec
  mySP1$vtypes = "B"
  mySP1$ub = ub
  mySP1$lb = lb

  mysol = gurobi(mySP1)
  x.h = mysol$x
  z.SP = mysol$objval
  list(x.h = x.h, z.SP = z.SP)
}

# Master Problem Function
MP = function(myMP, Amat, rhs){
  myMP$A = rBind(myMP$A, Amat)
  myMP$sense = c(myMP$sense, c(">=", ">=", ">=", ">="))
  myMP$rhs = c(myMP$rhs, rhs)

  mysol = gurobi(myMP)

  theta1 = mysol$x[1]
  theta2 = mysol$x[2]
  theta3 = mysol$x[3]
  theta4 = mysol$x[4]
  u = mysol$x[5:284]
  alpha = mysol$pi

  list(u=u, alpha=alpha, UB = mysol$objval, myMP = myMP)
}

### Iteration 0: Initialize ###

UB = Inf
LB = -Inf
Incumbent = -Inf
X = NULL

cvec_LR_1 = cvec_LR[1:X_Vars]
cvec_LR_2 = cvec_LR[(X_Vars+1):Z_Vars]
cvec_LR_3 = cvec_LR[(Z_Vars+1):Y_Vars]
cvec_LR_4 = cvec_LR[(Y_Vars+1):W_Vars]

remove_EmptyRows = function(Amat, dir, bvec) {
  zeroes = which(apply(Amat==0,1,all))
  new_Amat = Amat[-zeroes,]
  new_bvec = bvec[-zeroes]
  new_dir = dir[-zeroes]

  list(new_Amat = new_Amat, new_bvec = new_bvec, new_dir = new_dir)
}

zeroes = which(apply(Amat_LR[,1:X_Vars]==0,1,all))
new_Amat = Amat[-zeroes,]
new_bvec = bvec_LR[-zeroes]
new_dir = dir_LR[-zeroes]

SP_1 = remove_EmptyRows(Amat_LR[,1:X_Vars], dir_LR, bvec_LR)
SP_2 = remove_EmptyRows(Amat_LR[,(X_Vars+1):Z_Vars], dir_LR, bvec_LR)
SP_3 = remove_EmptyRows(Amat_LR[,(Z_Vars+1):Y_Vars], dir_LR, bvec_LR)
SP_4 = remove_EmptyRows(Amat_LR[,(Y_Vars+1):W_Vars], dir_LR, bvec_LR)

Amat_LR_1 = SP_1$new_Amat
Amat_LR_2 = SP_2$new_Amat
Amat_LR_3 = SP_3$new_Amat
Amat_LR_4 = SP_4$new_Amat

dir_LR_1 = SP_1$new_dir
dir_LR_2 = SP_2$new_dir
dir_LR_3 = SP_3$new_dir
dir_LR_4 = SP_4$new_dir

bvec_LR_1 = SP_1$new_bvec
bvec_LR_2 = SP_2$new_bvec
bvec_LR_3 = SP_3$new_bvec
bvec_LR_4 = SP_4$new_bvec

### Subproblems ###
mySP_1 = SP(cvec_LR_1, Amat_LR_1, dir_LR_1, bvec_LR_1)
mySP_2 = SP(cvec_LR_2, Amat_LR_2, dir_LR_2, bvec_LR_2)
mySP_3 = SP(cvec_LR_3, Amat_LR_3, dir_LR_3, bvec_LR_3)
mySP_4 = SP(cvec_LR_4, Amat_LR_4, dir_LR_4, bvec_LR_4)

x.h = c(mySP_1$x.h, mySP_2$x.h, mySP_3$x.h, mySP_4$x.h)
X = cBind(X, x.h)
LB = max(LB, mySP_1$z.SP + mySP_2$z.SP + mySP_3$z.SP + mySP_4$z.SP)
```

```
### Master Problem ###
uvec_MP_1 = c()
uvec_MP_2 = c()
uvec_MP_3 = c()
uvec_MP_4 = c()
for (g in 1:nrow(relaxed_Cons)){
  uvec_MP_1 = c(uvec_MP_1, (-1)*sum(relaxed_Cons[g,1:X_Vars]*mySP_1$x.h))
  uvec_MP_2 = c(uvec_MP_2, (-1)*sum(relaxed_Cons[g,(X_Vars+1):Z_Vars]*mySP_2$x.h))
  uvec_MP_3 = c(uvec_MP_3, (-1)*sum(relaxed_Cons[g,(Z_Vars+1):Y_Vars]*mySP_3$x.h))
  uvec_MP_4 = c(uvec_MP_4, (-1)*sum(relaxed_Cons[g,(Y_Vars+1):W_Vars]*mySP_4$x.h))
}

myMP = list()
myMP$modelsense = "max"
myMP$obj = c(1, 1, 1, 1, bvec_LR_Relaxed)
myMP$A = Matrix(c(1, 0, 0, 0, uvec_MP_1,
                  0, 1, 0, 0, uvec_MP_2,
                  0, 0, 1, 0, uvec_MP_3,
                  0, 0, 0, 1, uvec_MP_4),nrow=4, ncol=(length(uvec_MP_1)+4), byrow=T, sparse=T)
myMP$sense = c("<=","<=","<=","<=")
myMP$rhs = c(sum(mySP_1$x.h*cvec[1:X_Vars]),
             sum(mySP_2$x.h*cvec[(X_Vars+1):Z_Vars]),
             sum(mySP_3$x.h*cvec[(Z_Vars+1):Y_Vars]),
             sum(mySP_4$x.h*cvec[(Y_Vars+1):W_Vars]))

myMP$vtypes = "C"
myMP$lb = c(-1000000, -1000000, -1000000, -1000000, rep(-1000000, 10), rep(0, 270))
myMP$ub = c(1000000, 1000000, 1000000, 1000000, rep(1000000, length(bvec_LR_Relaxed)))

mysol = gurobi(myMP)

theta1 = mysol$x[1]
theta2 = mysol$x[2]
theta3 = mysol$x[3]
theta4 = mysol$x[4]
uvec = mysol$x[5:284]

UB = mysol$objval

check = (LB==UB)
LB
UB

count = 0
while(!check){
  cvec_LR = cvec

  for (l in 1:length(uvec)) {
    cvec_LR = cvec_LR + uvec[l]*relaxed_Cons[l,]
  }

  cvec_LR_1 = cvec_LR[1:X_Vars]
  cvec_LR_2 = cvec_LR[(X_Vars+1):Z_Vars]
  cvec_LR_3 = cvec_LR[(Z_Vars+1):Y_Vars]
  cvec_LR_4 = cvec_LR[(Y_Vars+1):W_Vars]

  mySP_1 = SP(cvec_LR_1, Amat_LR_1, dir_LR_1, bvec_LR_1)
  mySP_2 = SP(cvec_LR_2, Amat_LR_2, dir_LR_2, bvec_LR_2)
  mySP_3 = SP(cvec_LR_3, Amat_LR_3, dir_LR_3, bvec_LR_3)
  mySP_4 = SP(cvec_LR_4, Amat_LR_4, dir_LR_4, bvec_LR_4)

  x.h = c(mySP_1$x.h, mySP_2$x.h, mySP_3$x.h, mySP_4$x.h)
  X = cBind(X, x.h)
  LB = max(LB, mySP_1$z.SP + mySP_2$z.SP + mySP_3$z.SP + mySP_4$z.SP)

  uvec_MP_1 = c()
  uvec_MP_2 = c()
  uvec_MP_3 = c()
  uvec_MP_4 = c()
  for (g in 1:nrow(relaxed_Cons)){
    uvec_MP_1 = c(uvec_MP_1, (-1)*sum(relaxed_Cons[g,1:X_Vars]*mySP_1$x.h))
    uvec_MP_2 = c(uvec_MP_2, (-1)*sum(relaxed_Cons[g,(X_Vars+1):Z_Vars]*mySP_2$x.h))
    uvec_MP_3 = c(uvec_MP_3, (-1)*sum(relaxed_Cons[g,(Z_Vars+1):Y_Vars]*mySP_3$x.h))
    uvec_MP_4 = c(uvec_MP_4, (-1)*sum(relaxed_Cons[g,(Y_Vars+1):W_Vars]*mySP_4$x.h))
  }

  Amat_MP = Matrix(c(1, 0, 0, 0, uvec_MP_1,
                     0, 1, 0, 0, uvec_MP_2,
                     0, 0, 1, 0, uvec_MP_3,
                     0, 0, 0, 1, uvec_MP_4),nrow=4, ncol=(length(uvec_MP_1)+4), byrow=T, sparse=T)

  rhs_MP = c(sum(mySP_1$x.h*cvec[1:X_Vars]),
             sum(mySP_2$x.h*cvec[(X_Vars+1):Z_Vars]),
             sum(mySP_3$x.h*cvec[(Z_Vars+1):Y_Vars]),
             sum(mySP_4$x.h*cvec[(Y_Vars+1):W_Vars]))

  MP.out = MP(myMP, Amat_MP, rhs_MP)
  myMP = MP.out$myMP
  uvec = MP.out$u
  UB = MP.out$UB
```

```
    check = (LB==UB)
    count
    count = count + 1
  }
```