# DS 775

# Use of Optimization Programming Language for P.3.4-15, Week 3

Rick W. Lentz

21 September, 2017

## Data File

```
 Workers = {KC DH HB SC KS NK};// set of workers
 Days = {MON TUE WED THU FRI};// set of days
 wageRates = #[
 KC:25
DH:26
HB:24
SC:23
KS:28
NK:30
]#;
 minHours  = #[
 KC:8
DH:8
HB:8
SC:8
KS:7
NK:7
]#;
 hoursOpen = #[
MON:14
TUE:14
WED:14
THU:14
FRI:14
]#;

 WorkAssigns = {
 <KC,MON,6>
<KC,WED,6>
<KC,FRI,6>
<DH,TUE,6>
<DH,THU,6>
<HB,MON,4>
<HB,TUE,8>
<HB,WED,4>
<HB,FRI,4>
<SC,MON,5>
<SC,TUE,5>
<SC,WED,5>
<SC,FRI,5>
<KS,MON,3>
<KS,WED,3>
<KS,THU,8>
<NK,THU,6>
<NK,FRI,2>
};
```

## Model File

```
// Initialize data from file
 {string} Workers = ...;
 {string} Days = ...;

 float wageRates[Workers] = ...;
 float minHours[Workers] = ...;
 float hoursOpen[Days] = ...;

 tuple workerAssignType {
  string w;
  string d;
  float availHrs;
}

{workerAssignType} WorkAssigns = ...;

tuple connection {
	string w;
	string d;
}
```

```
// every possible assignment of a worker to a day is a "connection"
{connection} Connections = { <w,d> | <w,d,availHrs> in WorkAssigns};
float Availability[Connections] = [ <t.w,t.d>:t.availHrs | t in WorkAssigns];


dvar float+ Hours[Connections];


constraint ctSupply[Workers];
constraint ctDemand[Days];
constraint ctAvail[Connections];


minimize
    sum( c in Connections ) wageRates[c.w] * Hours[c];
  // replace with correct sum over Connections to get the total labor cost
// note if c is a connection, then the hourly wage for the worker w in that connection
// is wageRates[c.w]


subject to {
  forall( w in Workers  )
    ctSupply[w]:  // each worker gets at least minimum Hours
    sum(  c in Connections : c.w == w)
      Hours[c]  >= minHours[w];
  forall( d in Days)
    ctDemand[d]: // total hours supplied by workers each day equals demand
                         sum( con in Connections : con.d == d)
```

```
            Hours[con] == hoursOpen[d];

  forall (c in Connections)

    ctAvail[c]:  // each worker gets assigned no more than their available hours

       sum( w in Workers : w == c.w)

            Hours[c] <= Availability[<c.w,c.d>];

}


execute{

   for( var c in Connections ) {

                         writeln(c, Hours[c]);

   }

}
```

## Results

```
// solution (optimal) with objective 1755
 <"KC" "MON">3
 <"KC" "WED">2
 <"KC" "FRI">4
 <"DH" "TUE">2
 <"DH" "THU">6
 <"HB" "MON">4
 <"HB" "TUE">7
 <"HB" "WED">4
 <"HB" "FRI">4
 <"SC" "MON">5
 <"SC" "TUE">5
 <"SC" "WED">5
 <"SC" "FRI">5
 <"KS" "MON">2
 <"KS" "WED">3
 <"KS" "THU">2
 <"NK" "THU">6
 <"NK" "FRI">1
```