**Problem 1**

See Excel file

Benefit is 105

Person 1 does jobs 1, 8 and 9

Person 2 does jobs 4, 7 and 9

Person 3 does jobs 5 and 6

Person 4 does jobs 1, 2 and 3

Person 5 does jobs 2

**Problem 2**

See Draftkings Excel file for model.

PG Damian Lillard

SG Devin Booker

SF Ben Simmons

PF Robert Covington

C John Collins

G Jermey Lamb

F Taurean Prince

Util Paul George

The expected fantasy points was 264.779.

Entering in this team in the contest resulted in the team beating about 80% of contestants. So the total payout was $1.50. Given the type, $1 was entered and $.50 was won. If these players had combined for one less assist or two less points, then the payout would have been $0.

**Problem 3 a**

range machine = 1..4;

range job = 1..15;

range slotsfromlast = 1..12;

float processtime[job] = [23,14,7,62,21,17,19,55,16,13,27,31,35,18,9];

dvar int+ jobsched[job, machine,slotsfromlast] in (0..1);

dvar float+ contribution[job] ;

minimize sum(i in job) contribution[i];

subject to {

forall (i in job) sum(j in machine)sum(k in slotsfromlast) jobsched[i,j,k]==1;

forall (j in machine) forall (i in job) forall (k in slotsfromlast) processtime[i]\*

k\*jobsched[i,j,k]<=contribution[i];

forall (j in machine)forall (k in slotsfromlast) sum (i in job) jobsched[i,j,k]<=1;

};

main {

if (!thisOplModel.generate()){ // generates this model as a solution.

if (cplex.solve()) {// If there exists a solution enter

writeln( "Objective Value ", cplex.getObjValue() );

for ( var j in thisOplModel.machine) {

for ( var k in thisOplModel.slotsfromlast) {

for ( var i in thisOplModel.job) {

if (thisOplModel.jobsched[i][j][k]>.99){

writeln("The ",k," to last job on machine ",j," is job ", i);

}

}

}

}

} else {

writeln("Either Unbounded or Infeasible");

}

}else{

writeln("OPL could not read problem")

}

}

**The Output**

Objective Value 674

The 1 to last job on machine 1 is job 12

The 2 to last job on machine 1 is job 11

The 3 to last job on machine 1 is job 2

The 1 to last job on machine 2 is job 13

The 2 to last job on machine 2 is job 1

The 3 to last job on machine 2 is job 14

The 4 to last job on machine 2 is job 15

The 1 to last job on machine 3 is job 8

The 2 to last job on machine 3 is job 5

The 3 to last job on machine 3 is job 6

The 4 to last job on machine 3 is job 10

The 1 to last job on machine 4 is job 4

The 2 to last job on machine 4 is job 7

The 3 to last job on machine 4 is job 9

The 4 to last job on machine 4 is job 3

The solution has a value of 674

Machine 1 does jobs 2, 11, 12

Machine 2 does jobs 15, 14, 1, 13

Machine 3 does jobs 10, 6, 5, 8

Machine 4 does jobs 3, 9, 7, 4

**Problem 3 b.**

range machine = 1..4;

range job = 1..15;

range slots = 1..12;

float processtime[job] = [23,14,7,62,21,17,19,55,16,13,27,31,35,18,9];

dvar int+ jobsched[job, machine,slots] in (0..1);

dvar float+ maxtime ;

minimize maxtime;

subject to {

forall (i in job) sum(j in machine)sum(k in slots) jobsched[i,j,k]==1;

forall (j in machine) forall (k in slots) sum(i in job)sum(l in slots:l<=k)

processtime[i]\*jobsched[i,j,l]<=maxtime;

forall (j in machine)forall (k in slots) sum (i in job) jobsched[i,j,k]<=1;

forall (j in machine) forall(k in slots:k<=11)sum(i in job)jobsched[i,j,k+1]-

sum(i in job)jobsched[i,j,k]<=0;

//This last constraint forces consecutive jobs scheduled.

};

main {

if (!thisOplModel.generate()){ // generates this model as a solution.

if (cplex.solve()) {// If there exists a solution enter

writeln( "Objective Value ", cplex.getObjValue() ); //writes out an objective value

for ( var j in thisOplModel.machine) {

for ( var k in thisOplModel.slots) {

for ( var i in thisOplModel.job) {

if (thisOplModel.jobsched[i][j][k]>.99){

writeln("The ", k," job on machine ", j, " is job ", i);

}

}

}

}

} else {

writeln("Either Unbounded or Infeasible");

}

}else{

writeln("OPL could not read problem")

}

}

OUTPUT from Script File

Objective Value 92

The 1 job on machine 1 is job 14

The 2 job on machine 1 is job 12

The 3 job on machine 1 is job 9

The 4 job on machine 1 is job 2

The 5 job on machine 1 is job 10

The 1 job on machine 2 is job 13

The 2 job on machine 2 is job 11

The 3 job on machine 2 is job 3

The 4 job on machine 2 is job 1

The 1 job on machine 3 is job 15

The 2 job on machine 3 is job 4

The 3 job on machine 3 is job 5

The 1 job on machine 4 is job 6

The 2 job on machine 4 is job 7

The 3 job on machine 4 is job 8

The solution is 92 with

Machine 1 doing jobs 14, 12, 9, 2, 10

Machine 2 doing jobs 13, 11, 3, 1

Machine 3 doing jobs 15, 4, 5

Machine 4 doing jobs 6, 7, 8.

**Problem 4**

If I didn’t have to make minimum payments. I could have only spent 65,650 or so to pay off my loans and been done in month 83. (Model with first 6 constraints and pay highest interest first). With minimum monthly payments requirement, then it costs 66,470 and takes another month to pay it off. (The next two constraints, but it is not quite right. It doesn’t let you make a payment of less than the minimum payment.)

My model still isn’t quite right, but it took over 10 minutes to solve. It meets most of the requirements, but still has the problem that it can’t pay less than the minimum to payoff a loan entirely, so it spreads those last few payments out over the minimum number of months necessary to satisfy this problem.

range loan = 1..3;

range month = 1..120;

float interest[loan] = [.0075,.005, .01];

float starting[loan] = [30000,8000,12000];

float minimumpayment[loan] = [300,50,350];

dvar float+ amountpaid[loan, month];

dvar float+ amountowed[loan, month];

dvar int madepayment[loan,month]in 0..1;

dvar int payover[loan,month]in 0..1;

minimize sum(i in loan) sum (j in month) amountpaid[i,j];

subject to {

forall (j in month)sum(i in loan)amountpaid[i,j]<=800;

forall (j in month)forall(i in loan)amountpaid[i,j]<=800\*madepayment[i,j];

forall (i in loan) amountowed[i,1]==starting[i];

forall (j in month:j<=119)forall(i in loan)amountowed[i,j+1] == (1+interest[i])\*amountowed[i,j]-

amountpaid[i,j];

forall (i in loan) amountowed[i,120]==0;

forall (j in month)forall(i in loan)amountowed[i,j]<=35000\*madepayment[i,j];

forall (j in month)forall(i in loan)amountpaid[i,j]>=minimumpayment[i]\*madepayment[i,j];

forall (j in month)forall(i in loan)amountpaid[i,j]>=(minimumpayment[i]+50)\*payover[i,j];

forall (j in month:j<=84)sum(i in loan)payover[i,j]>=2;

};

main {

if (!thisOplModel.generate()){ // generates this model as a solution.

if (cplex.solve()) {// If there exists a solution enter

writeln( "Objective Value ", cplex.getObjValue() ); //writes out an objective value

for ( var i in thisOplModel.loan) {

for (var j in thisOplModel.month) {

if (thisOplModel.amountpaid[i][j]>.99){

writeln("Loan ",i," in month ",j," is paid ",thisOplModel.amountpaid[i][j],

" remaining balance ", thisOplModel.amountowed[i][j]);

}

}

}

for ( var i in thisOplModel.loan) {

for (var j in thisOplModel.month) {

if (thisOplModel.payover[i][j]>.99){

writeln("Paid extra on loan ", i," in month ", j);

}

}

}

} else {

writeln("Either Unbounded or Infeasible");

}

}else{

writeln("OPL could not read problem")

}

}

From the engine log of OPL. (12 million nodes)

Elapsed time = 634.64 sec. (310046.54 ticks, tree = 1365.51 MB, solutions = 18)

12204076 3638253 66836.8246 2 66840.9366 66834.1900 31604884 0.01%

12204078 3638251 infeasible 66840.9366 66834.1900 31604893 0.01%

12204084 3638245 infeasible 66840.9366 66834.1900 31604930 0.01%

12204086 3638243 infeasible 66840.9366 66834.1900 31604930 0.01%

12204088 3638243 66835.0696 2 66840.9366 66834.1900 31604935 0.01%

12204089 3638244 66837.6515 2 66840.9366 66834.1900 31604944 0.01%

12204092 3638243 infeasible 66840.9366 66834.1900 31604945 0.01%

12204094 3638241 infeasible 66840.9366 66834.1900 31604958 0.01%

12204096 3638241 infeasible 66840.9366 66834.1900 31604976 0.01%

12204097 3638240 infeasible 66840.9366 66834.1900 31604985 0.01%

Elapsed time = 642.89 sec. (313861.82 ticks, tree = 1349.29 MB, solutions = 18)

Root node processing (before b&c):

Real time = 0.08 sec. (58.90 ticks)

Parallel b&c, 8 threads:

Real time = 645.81 sec. (315201.43 ticks)

Sync time (average) = 189.64 sec.

Wait time (average) = 236.19 sec.

------------

Total (root+branch&cut) = 645.89 sec. (315260.33 ticks)

Script output

Total paid 66840.9

Month 84

Loan 1 in month 1 is paid 350 remaining balance 30000

Loan 1 in month 2 is paid 350 remaining balance 29875

Loan 1 in month 3 is paid 350 remaining balance 29749.1

Loan 1 in month 4 is paid 350 remaining balance 29622.2

Loan 1 in month 5 is paid 350 remaining balance 29494.3

Loan 1 in month 6 is paid 350 remaining balance 29365.6

Loan 1 in month 7 is paid 350 remaining balance 29235.8

Loan 1 in month 8 is paid 350 remaining balance 29105.1

Loan 1 in month 9 is paid 350 remaining balance 28973.4

Loan 1 in month 10 is paid 350 remaining balance 28840.7

Loan 1 in month 11 is paid 350 remaining balance 28707

Loan 1 in month 12 is paid 350 remaining balance 28572.3

Loan 1 in month 13 is paid 350 remaining balance 28436.6

Loan 1 in month 14 is paid 350 remaining balance 28299.8

Loan 1 in month 15 is paid 350 remaining balance 28162.1

Loan 1 in month 16 is paid 350 remaining balance 28023.3

Loan 1 in month 17 is paid 350 remaining balance 27883.5

Loan 1 in month 18 is paid 350 remaining balance 27742.6

Loan 1 in month 19 is paid 350 remaining balance 27600.7

Loan 1 in month 20 is paid 350 remaining balance 27457.7

Loan 1 in month 21 is paid 350 remaining balance 27313.6

Loan 1 in month 22 is paid 350 remaining balance 27168.4

Loan 1 in month 23 is paid 350 remaining balance 27022.2

Loan 1 in month 24 is paid 350 remaining balance 26874.9

Loan 1 in month 25 is paid 350 remaining balance 26726.4

Loan 1 in month 26 is paid 350 remaining balance 26576.9

Loan 1 in month 27 is paid 350 remaining balance 26426.2

Loan 1 in month 28 is paid 350 remaining balance 26274.4

Loan 1 in month 29 is paid 350 remaining balance 26121.5

Loan 1 in month 30 is paid 350 remaining balance 25967.4

Loan 1 in month 31 is paid 350 remaining balance 25812.1

Loan 1 in month 32 is paid 350 remaining balance 25655.7

Loan 1 in month 33 is paid 350 remaining balance 25498.1

Loan 1 in month 34 is paid 350 remaining balance 25339.4

Loan 1 in month 35 is paid 350 remaining balance 25179.4

Loan 1 in month 36 is paid 350 remaining balance 25018.3

Loan 1 in month 37 is paid 350 remaining balance 24855.9

Loan 1 in month 38 is paid 700 remaining balance 24692.3

Loan 1 in month 39 is paid 700 remaining balance 24177.5

Loan 1 in month 40 is paid 700 remaining balance 23658.9

Loan 1 in month 41 is paid 700 remaining balance 23136.3

Loan 1 in month 42 is paid 700 remaining balance 22609.8

Loan 1 in month 43 is paid 700 remaining balance 22079.4

Loan 1 in month 44 is paid 700 remaining balance 21545

Loan 1 in month 45 is paid 700 remaining balance 21006.6

Loan 1 in month 46 is paid 700 remaining balance 20464.1

Loan 1 in month 47 is paid 700 remaining balance 19917.6

Loan 1 in month 48 is paid 700 remaining balance 19367

Loan 1 in month 49 is paid 700 remaining balance 18812.2

Loan 1 in month 50 is paid 700 remaining balance 18253.3

Loan 1 in month 51 is paid 700 remaining balance 17690.2

Loan 1 in month 52 is paid 700 remaining balance 17122.9

Loan 1 in month 53 is paid 700 remaining balance 16551.3

Loan 1 in month 54 is paid 700 remaining balance 15975.5

Loan 1 in month 55 is paid 700 remaining balance 15395.3

Loan 1 in month 56 is paid 700 remaining balance 14810.7

Loan 1 in month 57 is paid 700 remaining balance 14221.8

Loan 1 in month 58 is paid 700 remaining balance 13628.5

Loan 1 in month 59 is paid 700 remaining balance 13030.7

Loan 1 in month 60 is paid 700 remaining balance 12428.4

Loan 1 in month 61 is paid 700 remaining balance 11821.6

Loan 1 in month 62 is paid 700 remaining balance 11210.3

Loan 1 in month 63 is paid 700 remaining balance 10594.4

Loan 1 in month 64 is paid 700 remaining balance 9973.84

Loan 1 in month 65 is paid 700 remaining balance 9348.65

Loan 1 in month 66 is paid 700 remaining balance 8718.76

Loan 1 in month 67 is paid 700 remaining balance 8084.15

Loan 1 in month 68 is paid 700 remaining balance 7444.78

Loan 1 in month 69 is paid 700 remaining balance 6800.62

Loan 1 in month 70 is paid 700 remaining balance 6151.62

Loan 1 in month 71 is paid 700 remaining balance 5497.76

Loan 1 in month 72 is paid 700 remaining balance 4838.99

Loan 1 in month 73 is paid 524.366 remaining balance 4175.29

Loan 1 in month 74 is paid 350 remaining balance 3682.24

Loan 1 in month 75 is paid 350 remaining balance 3359.85

Loan 1 in month 76 is paid 350 remaining balance 3035.05

Loan 1 in month 77 is paid 350 remaining balance 2707.81

Loan 1 in month 78 is paid 350 remaining balance 2378.12

Loan 1 in month 79 is paid 350 remaining balance 2045.96

Loan 1 in month 80 is paid 350 remaining balance 1711.3

Loan 1 in month 81 is paid 350 remaining balance 1374.14

Loan 1 in month 82 is paid 350 remaining balance 1034.44

Loan 1 in month 83 is paid 350 remaining balance 692.203

Loan 1 in month 84 is paid 350 remaining balance 347.395

Loan 2 in month 1 is paid 50 remaining balance 8000

Loan 2 in month 2 is paid 50 remaining balance 7990

Loan 2 in month 3 is paid 50 remaining balance 7979.95

Loan 2 in month 4 is paid 50 remaining balance 7969.85

Loan 2 in month 5 is paid 50 remaining balance 7959.7

Loan 2 in month 6 is paid 50 remaining balance 7949.5

Loan 2 in month 7 is paid 50 remaining balance 7939.24

Loan 2 in month 8 is paid 50 remaining balance 7928.94

Loan 2 in month 9 is paid 50 remaining balance 7918.59

Loan 2 in month 10 is paid 50 remaining balance 7908.18

Loan 2 in month 11 is paid 50 remaining balance 7897.72

Loan 2 in month 12 is paid 50 remaining balance 7887.21

Loan 2 in month 13 is paid 100 remaining balance 7876.64

Loan 2 in month 14 is paid 50 remaining balance 7816.03

Loan 2 in month 15 is paid 100 remaining balance 7805.11

Loan 2 in month 16 is paid 50 remaining balance 7744.13

Loan 2 in month 17 is paid 50 remaining balance 7732.85

Loan 2 in month 18 is paid 50 remaining balance 7721.52

Loan 2 in month 19 is paid 50 remaining balance 7710.13

Loan 2 in month 20 is paid 50 remaining balance 7698.68

Loan 2 in month 21 is paid 50 remaining balance 7687.17

Loan 2 in month 22 is paid 50 remaining balance 7675.61

Loan 2 in month 23 is paid 100 remaining balance 7663.98

Loan 2 in month 24 is paid 100 remaining balance 7602.3

Loan 2 in month 25 is paid 100 remaining balance 7540.32

Loan 2 in month 26 is paid 100 remaining balance 7478.02

Loan 2 in month 27 is paid 100 remaining balance 7415.41

Loan 2 in month 28 is paid 100 remaining balance 7352.48

Loan 2 in month 29 is paid 50 remaining balance 7289.25

Loan 2 in month 30 is paid 50 remaining balance 7275.69

Loan 2 in month 31 is paid 50 remaining balance 7262.07

Loan 2 in month 32 is paid 50 remaining balance 7248.38

Loan 2 in month 33 is paid 50 remaining balance 7234.62

Loan 2 in month 34 is paid 50 remaining balance 7220.8

Loan 2 in month 35 is paid 50 remaining balance 7206.9

Loan 2 in month 36 is paid 50 remaining balance 7192.93

Loan 2 in month 37 is paid 50.0003 remaining balance 7178.9

Loan 2 in month 38 is paid 100 remaining balance 7164.79

Loan 2 in month 39 is paid 100 remaining balance 7100.62

Loan 2 in month 40 is paid 100 remaining balance 7036.12

Loan 2 in month 41 is paid 100 remaining balance 6971.3

Loan 2 in month 42 is paid 100 remaining balance 6906.16

Loan 2 in month 43 is paid 100 remaining balance 6840.69

Loan 2 in month 44 is paid 100 remaining balance 6774.89

Loan 2 in month 45 is paid 100 remaining balance 6708.77

Loan 2 in month 46 is paid 100 remaining balance 6642.31

Loan 2 in month 47 is paid 100 remaining balance 6575.52

Loan 2 in month 48 is paid 100 remaining balance 6508.4

Loan 2 in month 49 is paid 100 remaining balance 6440.94

Loan 2 in month 50 is paid 100 remaining balance 6373.15

Loan 2 in month 51 is paid 100 remaining balance 6305.01

Loan 2 in month 52 is paid 100 remaining balance 6236.54

Loan 2 in month 53 is paid 100 remaining balance 6167.72

Loan 2 in month 54 is paid 100 remaining balance 6098.56

Loan 2 in month 55 is paid 100 remaining balance 6029.05

Loan 2 in month 56 is paid 100 remaining balance 5959.2

Loan 2 in month 57 is paid 100 remaining balance 5888.99

Loan 2 in month 58 is paid 100 remaining balance 5818.44

Loan 2 in month 59 is paid 100 remaining balance 5747.53

Loan 2 in month 60 is paid 100 remaining balance 5676.27

Loan 2 in month 61 is paid 100 remaining balance 5604.65

Loan 2 in month 62 is paid 100 remaining balance 5532.67

Loan 2 in month 63 is paid 100 remaining balance 5460.34

Loan 2 in month 64 is paid 100 remaining balance 5387.64

Loan 2 in month 65 is paid 100 remaining balance 5314.57

Loan 2 in month 66 is paid 100 remaining balance 5241.15

Loan 2 in month 67 is paid 100 remaining balance 5167.35

Loan 2 in month 68 is paid 100 remaining balance 5093.19

Loan 2 in month 69 is paid 100 remaining balance 5018.66

Loan 2 in month 70 is paid 100 remaining balance 4943.75

Loan 2 in month 71 is paid 100 remaining balance 4868.47

Loan 2 in month 72 is paid 100 remaining balance 4792.81

Loan 2 in month 73 is paid 275.634 remaining balance 4716.77

Loan 2 in month 74 is paid 450 remaining balance 4464.72

Loan 2 in month 75 is paid 450 remaining balance 4037.05

Loan 2 in month 76 is paid 450 remaining balance 3607.23

Loan 2 in month 77 is paid 450 remaining balance 3175.27

Loan 2 in month 78 is paid 450 remaining balance 2741.15

Loan 2 in month 79 is paid 450 remaining balance 2304.85

Loan 2 in month 80 is paid 450 remaining balance 1866.38

Loan 2 in month 81 is paid 450 remaining balance 1425.71

Loan 2 in month 82 is paid 450 remaining balance 982.836

Loan 2 in month 83 is paid 440.937 remaining balance 537.75

Loan 2 in month 84 is paid 100 remaining balance 99.5025

Loan 3 in month 1 is paid 400 remaining balance 12000

Loan 3 in month 2 is paid 400 remaining balance 11720

Loan 3 in month 3 is paid 400 remaining balance 11437.2

Loan 3 in month 4 is paid 400 remaining balance 11151.6

Loan 3 in month 5 is paid 400 remaining balance 10863.1

Loan 3 in month 6 is paid 400 remaining balance 10571.7

Loan 3 in month 7 is paid 400 remaining balance 10277.4

Loan 3 in month 8 is paid 400 remaining balance 9980.21

Loan 3 in month 9 is paid 400 remaining balance 9680.01

Loan 3 in month 10 is paid 400 remaining balance 9376.81

Loan 3 in month 11 is paid 400 remaining balance 9070.58

Loan 3 in month 12 is paid 400 remaining balance 8761.29

Loan 3 in month 13 is paid 350 remaining balance 8448.9

Loan 3 in month 14 is paid 400 remaining balance 8183.39

Loan 3 in month 15 is paid 350 remaining balance 7865.22

Loan 3 in month 16 is paid 400 remaining balance 7593.87

Loan 3 in month 17 is paid 400 remaining balance 7269.81

Loan 3 in month 18 is paid 400 remaining balance 6942.51

Loan 3 in month 19 is paid 400 remaining balance 6611.94

Loan 3 in month 20 is paid 400 remaining balance 6278.06

Loan 3 in month 21 is paid 400 remaining balance 5940.84

Loan 3 in month 22 is paid 400 remaining balance 5600.24

Loan 3 in month 23 is paid 350 remaining balance 5256.25

Loan 3 in month 24 is paid 350 remaining balance 4958.81

Loan 3 in month 25 is paid 350 remaining balance 4658.4

Loan 3 in month 26 is paid 350 remaining balance 4354.98

Loan 3 in month 27 is paid 350 remaining balance 4048.53

Loan 3 in month 28 is paid 350 remaining balance 3739.02

Loan 3 in month 29 is paid 400 remaining balance 3426.41

Loan 3 in month 30 is paid 400 remaining balance 3060.67

Loan 3 in month 31 is paid 400 remaining balance 2691.28

Loan 3 in month 32 is paid 400 remaining balance 2318.19

Loan 3 in month 33 is paid 400 remaining balance 1941.37

Loan 3 in month 34 is paid 400 remaining balance 1560.79

Loan 3 in month 35 is paid 400 remaining balance 1176.39

Loan 3 in month 36 is paid 400 remaining balance 788.158

Loan 3 in month 37 is paid 400 remaining balance 396.039

It didn’t seem worth printing the paid extra.

**Problem 5 a**

range city = 1..13; //\*\*\*

range trip = 1..3; //assume no trip back to home

range truck = 1..6;

float dist[city, city] = [[0,2457,712,1433,66,2141,1616,635,2407,1104,644,1167,1057],

[2457,0,1752,1374,2409,365,851,1853,958,2339,1817,1688,1775],

[712,1752,0,954,672,1452,906,275,1737,1195,167,838,778],

[1433,1374,954,0,1368,1010,871,829,1891,967,878,336,445],

[66,2409,672,1368,0,2090,1572,577,2383,1047,593,1101,991],

[2141,365,1452,1010,2090,0,593,1522,1111,1974,1498,1324,1412],

[1616,851,906,871,1572,593,0,1039,1033,1710,987,1078,1124],

[635,1853,275,829,577,1522,1039,0,1956,920,108,633,550],

[2407,958,1737,1891,2383,1111,1033,1956,0,2732,1874,2110,2151],

[1104,2339,1195,967,1047,1974,1710,920,2732,0,1028,654,587],

[644,1817,167,878,593,1498,987,108,1874,1028,0,713,640],

[1167,1688,838,336,1101,1324,1078,633,2110,654,713,0,117],

[1057,1775,778,445,991,1412,1124,550,2151,587,640,117,0]];

dvar int+ X[city, city, trip, truck] in (0..1);

minimize sum(i in city)(sum(j in city)(sum(k in trip)(sum(l in truck)dist[i,j]\*X[i,j,k,l])));

subject to {

forall(k in trip)forall(l in truck)sum(i in city)(sum(j in city)X[i,j,k,l])==1;

forall(i in city:i>=2)sum(l in truck)sum(j in city)(sum(k in trip)X[i,j,k,l])==1;

forall(j in city:j>=2)sum(l in truck)sum(i in city)(sum(k in trip)X[i,j,k,l])==1;

forall(l in truck)sum(j in city)X[1,j,1,l]==1;

forall(l in truck)sum(i in city)X[i,1,3,l]==1;

forall(j in city)forall(l in truck)forall(k in trip:k<=2)sum(i in city)X[j,i,k+1,l]==sum(i in city)X[i,j,k,l];

}

main {

if (!thisOplModel.generate()){ // generates this model as a solution.

if (cplex.solve()) {// If there exists a solution enter

writeln( "Objective Value ", cplex.getObjValue() ); //writes out an objective value

for ( var l in thisOplModel.truck) {

for ( var k in thisOplModel.trip) { //goes through a loop of warehouse

for ( var i in thisOplModel.city) { //goes through a loop of cities out from

for ( var j in thisOplModel.city) { //goes through a loop of cities in to

if (thisOplModel.X[i][j][k][l]>.99){

writeln ("Trip ", k , " of truck ", l, " goes from ", i," to ", j);

}

}

}

}

}

} else {

writeln("Either Unbounded or Infeasible");

}

}else{

writeln("OPL could not read problem")

}

}

/\* Output from script file is

Objective Value 18504

Trip 1 of truck 1 goes from 1 to 5

Trip 2 of truck 1 goes from 5 to 8

Trip 3 of truck 1 goes from 8 to 1

Trip 1 of truck 2 goes from 1 to 10

Trip 2 of truck 2 goes from 10 to 13

Trip 3 of truck 2 goes from 13 to 1

Trip 1 of truck 3 goes from 1 to 7

Trip 2 of truck 3 goes from 7 to 9

Trip 3 of truck 3 goes from 9 to 1

Trip 1 of truck 4 goes from 1 to 4

Trip 2 of truck 4 goes from 4 to 12

Trip 3 of truck 4 goes from 12 to 1

Trip 1 of truck 5 goes from 1 to 11

Trip 2 of truck 5 goes from 11 to 3

Trip 3 of truck 5 goes from 3 to 1

Trip 1 of truck 6 goes from 1 to 6

Trip 2 of truck 6 goes from 6 to 2

Trip 3 of truck 6 goes from 2 to 1

\*/

**Problem 5 b**

range city = 1..13; //\*\*\*

range trip = 1..9; //assume no trip back to home

range truck = 1..3;

float dist[city, city] = [[0,2457,712,1433,66,2141,1616,635,2407,1104,644,1167,1057],

[2457,0,1752,1374,2409,365,851,1853,958,2339,1817,1688,1775],

[712,1752,0,954,672,1452,906,275,1737,1195,167,838,778],

[1433,1374,954,0,1368,1010,871,829,1891,967,878,336,445],

[66,2409,672,1368,0,2090,1572,577,2383,1047,593,1101,991],

[2141,365,1452,1010,2090,0,593,1522,1111,1974,1498,1324,1412],

[1616,851,906,871,1572,593,0,1039,1033,1710,987,1078,1124],

[635,1853,275,829,577,1522,1039,0,1956,920,108,633,550],

[2407,958,1737,1891,2383,1111,1033,1956,0,2732,1874,2110,2151],

[1104,2339,1195,967,1047,1974,1710,920,2732,0,1028,654,587],

[644,1817,167,878,593,1498,987,108,1874,1028,0,713,640],

[1167,1688,838,336,1101,1324,1078,633,2110,654,713,0,117],

[1057,1775,778,445,991,1412,1124,550,2151,587,640,117,0]];

dvar int+ X[city, city, trip, truck] in (0..1);

minimize sum(i in city)(sum(j in city)(sum(k in trip)(sum(l in truck)dist[i,j]\*X[i,j,k,l])));

subject to {

forall(k in trip)forall(l in truck)sum(i in city)(sum(j in city)X[i,j,k,l])<=1;

forall(l in truck)sum(i in city)(sum(j in city)X[i,j,3,l])==1;

forall(i in city:i>=2)sum(l in truck)sum(j in city)(sum(k in trip)X[i,j,k,l])==1;

forall(j in city:j>=2)sum(l in truck)sum(i in city)(sum(k in trip)X[i,j,k,l])==1;

forall(l in truck)sum(j in city)X[1,j,1,l]==1;

forall(l in truck)sum(i in city)sum(k in trip)X[i,1,k,l]==1;

forall(j in city:j>=2)forall(l in truck)forall(k in trip:k<=8)sum(i in city)X[j,i,k+1,l]==sum(i in city)X[i,j,k,l];

forall(i in city)forall(k in trip) forall(l in truck) X[i,i,k,l]==0;

}

main {

if (!thisOplModel.generate()){ // generates this model as a solution.

if (cplex.solve()) {// If there exists a solution enter

writeln( "Objective Value ", cplex.getObjValue() ); //writes out an objective value

for ( var l in thisOplModel.truck) {

for ( var k in thisOplModel.trip) { //goes through a loop of warehouse

for ( var i in thisOplModel.city) { //goes through a loop of cities out from

for ( var j in thisOplModel.city) { //goes through a loop of cities in to

if (thisOplModel.X[i][j][k][l]>.99){

writeln ("Trip ", k , " of truck ", l, " goes from ", i," to ", j);

}

}

}

}

}

} else {

writeln("Either Unbounded or Infeasible");

}

}else{

writeln("OPL could not read problem")

}

}

/\* The output is

Objective Value 9927

Trip 1 of truck 1 goes from 1 to 5

Trip 2 of truck 1 goes from 5 to 8

Trip 3 of truck 1 goes from 8 to 1

Trip 1 of truck 2 goes from 1 to 7

Trip 2 of truck 2 goes from 7 to 9

Trip 3 of truck 2 goes from 9 to 2

Trip 4 of truck 2 goes from 2 to 6

Trip 5 of truck 2 goes from 6 to 4

Trip 6 of truck 2 goes from 4 to 12

Trip 7 of truck 2 goes from 12 to 13

Trip 8 of truck 2 goes from 13 to 10

Trip 9 of truck 2 goes from 10 to 1

Trip 1 of truck 3 goes from 1 to 3

Trip 2 of truck 3 goes from 3 to 11

Trip 3 of truck 3 goes from 11 to 1

\*/