MAT 168 HW 1

Hardy Jones
999397426
Professor Köeppe
Spring 2015

1.1 The first thing we should do is try to understand the domain. The steel company wants to know how many hours to allocate to each job. Since the description does not mention that each job must be a full hour–or some minimum fraction of an hour–we assume that hours can be real valued. Since the hours are what the problem explicitly asks for, these can be our decision variables.

Call these

- x_1 := Band
- x_2 := Coil

Next we look at the other information in the statement.

- Each product has a different rate, but these are constant values not constrained by anything else in the process.

  We have

  - $r_1 = 200 \frac{ton}{hour}$
  - $r_2 = 140 \frac{ton}{hour}$
- Each product has a different profit, but these are constant values not constrained by anything else in the process.

  We have

  - $p_1 = 25 \frac{dollar}{ton}$
  - $p_2 = 30 \frac{dollar}{ton}$
- Each product has a maximum constraint on the weight that can be produced. Since the problem does not state otherwise, we assume this can be a real value.
- The objective is to maximum profit.

  We can find the profit of either product by taking the expression

$$p_i \cdot r_i \cdot x_i, \text{ for } i \in \{1, 2\}$$

With this information, we can now formalize the problem.

$$\begin{aligned}
\text{maximize} \quad & 5000x_1 + 4200x_2 \\
\text{subject to} \quad & 200x_1 \leq 6000 \\
& 140x_2 \leq 4000 \\
& x_1 + x_2 \leq 40 \\
& x_1, x_2 \geq 0 \\
& x_1, x_2 \in \mathbb{R}
\end{aligned}$$

It turns out what we have modeled is an instance of the fractional knapsack problem. We can see it better if we simplify the first two constraints to

$\left.\begin{aligned} x_1 &\leq 30 \\ x_2 &\leq \frac{200}{7} \end{aligned}\right\}$, respectively.

Here, our "knapsack" is the number of hours to schedule. The maximum "weight" of the "knapsack" is the 40 hours available for the next week. The maximum amount of each "item" we want to take is the first two constraints. Finally, we have the amount of profit per "weight" (in this case hours).

Importantly, the decision variables in the problem are positive real values. The variables being real values allows us to formalize the model as a fractional knapsack rather than 0-1 knapsack or some other version.

Since we already know that fractional knapsack has optimal substructure and satisfies the greedy choice property, we can use a greedy approach to solve this.

Let's catalog the steps.

- We have a system of constraints
  $$\begin{aligned}
  200x_1 &\leq 6000 \\
  140x_2 &\leq 4000 \\
  x_1 + x_2 &\leq 40 \\
  x_1 &\geq 0 \\
  x_2 &\geq 0
  \end{aligned}$$
  Bands are worth more from the profit perspective, so we take as many as possible. Using the first constraint, this gives us a direct number for x_1.
  Namely $200x_1 \leq 6000 \implies x_1 \leq 30$.
  This choice does not violate any other constraints, so we choose the maximum possible value x_1 can take which is 30.
- Our new system is
  $$\begin{aligned}
  140x_2 &\leq 4000 \\
  x_2 &\leq 10 \\
  x_2 &\geq 0
  \end{aligned}$$

So we take the maximum value possible for x_2 which is 10.

The profit can now be computed and we end up with the following result:

With a choice of 30 hours making Bands and 10 hours making Coils, the company can make an optimally maximized profit of $\$192,000$.

We can verify this by modeling it with ZIMPL:

```
var band real;
var coil real;

maximize profit: 5000 * band + 4200 * coil;

subto hours: band + coil <= 40;
subto rate_band: 200 * band <= 6000;
subto rate_coil: 140 * coil <= 4000;
```

And then solving with scip:

```
SCIP> read hw1_steel_2.zpl

read problem <hw1_steel_2.zpl>
============


base directory for ZIMPL parsing: </home/joneshf/school/ucd/mat168>

original problem has 2 variables (0 bin, 0 int, 0 impl, 2 cont) and 3 constraints
SCIP> optimize

...

SCIP Status        : problem is solved [optimal solution found]
Solving Time (sec) : 0.00
Solving Nodes      : 1
Primal Bound       : +1.92000000000000e+05 (2 solutions)
Dual Bound         : +1.92000000000000e+05
Gap                : 0.00 %

SCIP> display solution

objective value:                                  192000
band                                                  30   (obj:5000)
coil                                                  10   (obj:4200)

SCIP>
```

1.2 We use as our decision variables the number of tickets of each flight combination, say $x_i, i \in \{1, 2, \ldots, 9\}$.

Now we look at the rest of the information:

- Each flight has a maximum constraint on the number of passengers since the plane can only hold 30 people.
- Each flight has only so many seats available for each fare, say $s_i, i \in \{1, 2, \ldots, 9\}$.
- Each flight fare has a ticket price associated with it, say $p_i, i \in \{1, 2, \ldots, 9\}$.
- Our objective function is to maximize revenue for each flight.

So we can formalize this model as:

$$
\begin{aligned}
\text{maximize} \quad & \sum_{i=1}^{9} p_i x_i \\
\text{subject to} \quad & \forall i \in \{1,2,\ldots,9\}, && x_i \leq s_i \\
& \forall j \in \{0,1,2\}, && \sum_{i=3j+1}^{3j+3} x_i \leq 30 \\
& \forall i \in \{1,2,\ldots,9\}, && x_i \geq 0 \\
& \forall i \in \{1,2,\ldots,9\}, && x_i \in \mathbb{Z}
\end{aligned}
$$

We model this in ZIMPL in order to arrive at a solution.

```
set Flight := {"Ithaca to Newark", "Newark to Boston", "Ithaca to Boston"};
set Fare := {"Y", "B", "M"};

param prices[Fare * Flight] :=
    |"Ithaca to Newark", "Newark to Boston", "Ithaca to Boston"|
|"Y"|                300,                160,                360|
|"B"|                220,                130,                280|
|"M"|                100,                 80,                140|;

param seating[Fare * Flight] :=
    |"Ithaca to Newark", "Newark to Boston", "Ithaca to Boston"|
|"Y"|                  4,                  8,                  3|
|"B"|                  8,                 13,                 10|
|"M"|                 22,                 20,                 18|;

var passenger[Fare * Flight] integer >= 0;

maximize revenue: sum <fare, flight> in Fare * Flight:
    prices[fare, flight] * passenger[fare, flight];

subto seating_limit: forall <fare, flight> in Fare * Flight:
    passenger[fare, flight] <= seating[fare, flight];
subto flight_limit: forall <flight> in Flight:
    (sum <fare> in Fare: passenger[fare, flight]) <= 30;
```

Now we can use scip to find the solution:

```
SCIP> read hw1_airline.zpl

read problem <hw1_airline.zpl>
============


base directory for ZIMPL parsing: </home/joneshf/school/ucd/mat168>

original problem has 9 variables (0 bin, 9 int, 0 impl, 0 cont) and 12 constraints
SCIP> opt

...

SCIP Status        : problem is solved [optimal solution found]
Solving Time (sec) : 0.00
Solving Nodes      : 1
Primal Bound       : +1.47100000000000e+04 (2 solutions)
Dual Bound         : +1.47100000000000e+04
Gap                : 0.00 %

SCIP> disp solution

objective value:                                  14710
passenger$Y$Ithaca to Newark                          4   (obj:300)
passenger$Y$Newark to Boston                          8   (obj:160)
passenger$Y$Ithaca to Boston                          3   (obj:360)
passenger$B$Ithaca to Newark                          8   (obj:220)
passenger$B$Newark to Boston                         13   (obj:130)
passenger$B$Ithaca to Boston                         10   (obj:280)
passenger$M$Ithaca to Newark                         18   (obj:100)
passenger$M$Newark to Boston                          9   (obj:80)
passenger$M$Ithaca to Boston                         17   (obj:140)
```

So we see that the optimal revenue is \$14,710, assuming the appropriate number of tickets are sold.

2-2 (a) As stated in the problem, we use the decision variables

- x_1 := dollars invested in domestic stocks (in millions)
- x_2 := dollars invested in foreign stocks (in millions)

And we can read the five constraints as follows:

i. The sum of x_1 and x_2 can be at most 12.

ii. x_1 can be at most 10.

iii. x_2 can be at most 7.

iv. x_1 must be at least twice as much as x_2.

v. x_2 must be at least twice as much as x_1.

N.B. It might seem like we need an additional constraint that $x_1, x_2 \geq 0$, however, the last two constraints together ensure this fact.

We want to maximize the returns for domestic and foreign stocks, which are 0.11 and 0.17, respectively.

Now we can create a model:

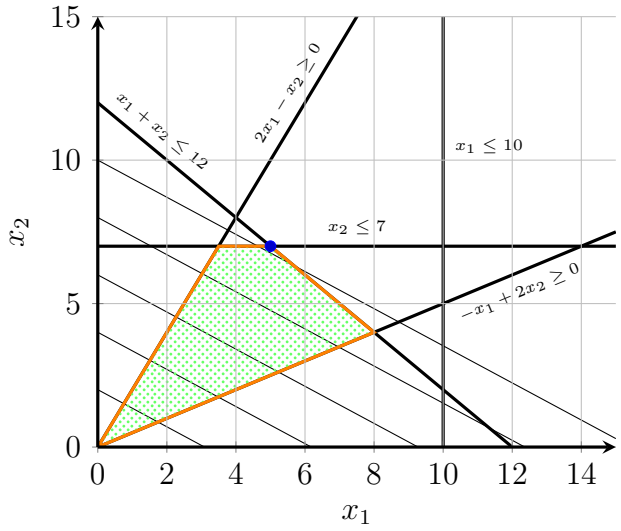
$$
\begin{aligned}
\text{maximize} \quad & 0.11x_1 + 0.17x_2 \\
\text{subject to} \quad & x_1 + x_2 \leq 12 \\
& x_1 \leq 10 \\
& x_2 \leq 7 \\
& 2\,x_1 - x_2 \geq 0 \\
& -x_1 + 2x_2 \geq 0 \\
& x_1, x_2 \in \mathbb{R}
\end{aligned}
$$

(b) We can formulate this model in ZIMPL:

```
var domestic real;
var foreign real;

maximize returns: 0.11 * domestic + 0.16 * foreign;

subto total_both: domestic + foreign <= 12;
subto total_domestic: domestic <= 10;
subto total_foreign: foreign <= 7;
subto twice_foreign: 2 * domestic >= foreign;
subto twice_domestic: 2 * foreign >= domestic;
```

And solve it with SCIP:

```
SCIP> read hw1_invest.zpl

read problem <hw1_invest.zpl>
============


base directory for ZIMPL parsing: </home/joneshf/school/ucd/mat168>

original problem has 2 variables (0 bin, 0 int, 0 impl, 2 cont) and 5 constraints
SCIP> opt

...

SCIP Status        : problem is solved [optimal solution found]
Solving Time (sec) : 0.00
Solving Nodes      : 1
Primal Bound       : +1.67000000000000e+00 (2 solutions)
Dual Bound         : +1.67000000000000e+00
Gap                : 0.00 %

SCIP> disp solution

objective value:                                  1.67
domestic                                             5   (obj:0.11)
foreign                                              7   (obj:0.16)

SCIP>
```



(c)

(d)

4-2 (a) We use one decision variable for each combination of designer and project. The variable represents how many hours the designer spends on the said project.

Say $x_{ij}, i \in \{1, 2, 3\}, j \in \{1, 2, 3, 4\}$

Now lets look at the rest of the information

- We have that each designer can spend a maximum of 80 hours working.
- We have the productivity rating for each designer on each project.
  Say $p_{ij}, i \in \{1, 2, 3\}, j \in \{1, 2, 3, 4\}$
- We have some estimates–that we'll use as upper bounds–for how long each project takes.
  - $e_1 = 70$
  - $e_2 = 50$
  - $e_3 = 85$
  - $e_4 = 35$

  So we know that for each project i, the sum of all the hours each designer spends on project i must equal e_i.

Our objective function is to maximize productivity. We can calculate productivity as the sum of all $p_{ij} \cdot x_{ij}$.

Now we can create a model:

$$
\begin{aligned}
\text{maximize} \quad & \sum_{i=1}^{3}\sum_{j=1}^{4} p_{ij}x_{ij} \\
\text{subject to} \quad & \forall i \in \{1,2,3\} \sum_{j=1}^{4} x_{ij} \leq 80 \\
& \sum_{i=1}^{3} x_{i1} \leq 70 \\
& \sum_{i=1}^{3} x_{i1} \geq 70 \\
& \sum_{i=1}^{3} x_{i2} \leq 50 \\
& \sum_{i=1}^{3} x_{i2} \geq 50 \\
& \sum_{i=1}^{3} x_{i3} \leq 85 \\
& \sum_{i=1}^{3} x_{i3} \geq 85 \\
& \sum_{i=1}^{3} x_{i4} \leq 35 \\
& \sum_{i=1}^{3} x_{i4} \geq 35 \\
& x_{ij} \in \mathbb{R}
\end{aligned}
$$

(b) We can formulate this model in ZIMPL:

```
set Designer := { "Designer 1", "Designer 2", "Designer 3" };
set Project := { "Project 1", "Project 2", "Project 3", "Project 4" };

param productivity[Designer * Project] :=
                |"Project 1", "Project 2", "Project 3", "Project 4"|
|"Designer 1"|          90,           80,           10,           50|
|"Designer 2"|          60,           70,           50,           65|
|"Designer 3"|          70,           40,           80,           85|;

param estimate[Project] :=
    <"Project 1"> 70, <"Project 2"> 50, <"Project 3"> 85, <"Project 4"> 35;

var hours[Designer * Project] real;

maximize productivity: sum <designer, project> in Designer * Project:
    productivity[designer, project] * hours[designer, project];
```

```
subto max_hours: forall <designer> in Designer:
    (sum <project> in Project: hours[designer, project]) <= 80;

subto requirement: forall <project> in Project:
    (sum <designer> in Designer: hours[designer, project]) == estimate[project];
```

And solve it with SCIP:

```
SCIP> read hw1_productivity.zpl

read problem <hw1_productivity.zpl>
============


base directory for ZIMPL parsing: </home/joneshf/school/ucd/mat168>

Block memory allocation count 97
original problem has 12 variables (0 bin, 0 int, 0 impl, 12 cont) and
7 constraints
SCIP> opt

...

SCIP Status        : problem is solved [optimal solution found]
Solving Time (sec) : 0.00
Solving Nodes      : 1
Primal Bound       : +1.88250000000000e+04 (1 solutions)
Dual Bound         : +1.88250000000000e+04
Gap                : 0.00 %

SCIP> disp solution

objective value:                                18825
hours$Designer 1$Project 1                         70   (obj:90)
hours$Designer 1$Project 2                         10   (obj:80)
hours$Designer 2$Project 2                         40   (obj:70)
hours$Designer 2$Project 3                          5   (obj:50)
hours$Designer 2$Project 4                         35   (obj:65)
hours$Designer 3$Project 3                         80   (obj:80)
```