Knapsack Problem

Jiasui Qin

October 9 2020

Imagine that we have 50 unique objects labeled from 1 to 50. Their values, weights, and volumes are defined as below:

value = floor(100+50cos(i)) thousands of dollars
weight = floor(100+50cos(7i+1)) kg
volume = floor(20+10cos(8i-3)) liters

(Note: i represents its label, floor function gives the greatest integer less than or equal to the given number.)
Suppose that we have a container which has a weight capacity of 1000 kg and a volume capacity of 100 liters.

1. **Find the way to put items into our container to maximize the total value.**

   Mathematical formulation:

   Variables are denoted as $x_i (1 \leq i \leq 50)$ representing whether or not to take an object. If yes, it equals to 1. If not, it equals to 0.

   Since we want to maximize the total value, the objective function would be the sum of every object's value, which can be represented as

   $$\sum_{i=1}^{50} floor(100 + 50cos(i))x_i$$

   Since our container only has a weight capacity of 1000 kg and a volume capacity of 100 liters, the total weights of objects has to be less than or equal to 1000, the total volumes of objects has to be less than or equal to 100. We then have constraints:

   $$\sum_{i=1}^{50} floor(100 + 50cos(7i + 1))x_i \leq 1000$$

$$\sum_{i=1}^{50} floor(20 + 10cos(8i - 3))x_i \leq 100$$

Lastly, all variables are either 1 or 0 because every object is unique:

$$x_1, x_2, ..., x_{50} \in \{0, 1\}$$

Since we will use Lpsolve to solve this LP, I wrote the following Python code to generate the Lpsolve input file:

```
# Jiasui Qin
# This code outputs an lpsolve-formatted LP which
# determines what objects yield the maximum value
# when objects are unique.

import numpy as np
import math

# print the objective function
outstring = "max: "
for i in range(1, 51):
        value = math.floor(50 * np.cos(i) + 100)
        outstring = outstring + "+" + str(value) + "*x_" + str(i)
outstring = outstring+";"
print(outstring)

# print the constraint that weight capacity is 1000 kg
outstring = ""
for i in range(1, 51):
        weight = math.floor(50 * np.cos(7*i+1) + 100)
        outstring = outstring + "+" + str(weight) + "*x_" + str(i)
outstring = outstring+"<=1000;"
print(outstring)

# print the constraint that volume capacity is 100 liters
outstring = ""
for i in range(1, 51):
        volume = math.floor(10 * np.cos(8*i-3) + 20)
        outstring = outstring + "+" + str(volume) + "*x_" + str(i)
outstring = outstring+"<=100;"
print(outstring)
```

```
# define variables to be bianary
outstring = "bin "
for i in range(1, 50):
        outstring = outstring + "x_" + str(i) + ", "
outstring = outstring+"x_50;"
print(outstring)
```

After running the python code above, I got the following Lpsolve input file:

max: $+127*x\_1+79*x\_2+50*x\_3+67*x\_4+114*x\_5+148*x\_6+137*x\_7+92*x\_8+54*x\_9+58*x\_10+100*x\_11+142*x\_12+145*x\_13+106*x\_14+62*x\_15+52*x\_16+86*x\_17+133*x\_18+149*x\_19+120*x\_20+72*x\_21+50*x\_22+73*x\_23+121*x\_24+149*x\_25+132*x\_26+85*x\_27+51*x\_28+62*x\_29+107*x\_30+145*x\_31+141*x\_32+99*x\_33+57*x\_34+54*x\_35+93*x\_36+138*x\_37+147*x\_38+113*x\_39+66*x\_40+50*x\_41+80*x\_42+127*x\_43+149*x\_44+126*x\_45+78*x\_46+50*x\_47+67*x\_48+115*x\_49+148*x\_50;$

$+92*x\_1+62*x\_2+50*x\_3+62*x\_4+93*x\_5+127*x\_6+148*x\_7+144*x\_8+119*x\_9+84*x\_10+57*x\_11+50*x\_12+68*x\_13+101*x\_14+134*x\_15+149*x\_16+140*x\_17+111*x\_18+76*x\_19+53*x\_20+52*x\_21+75*x\_22+110*x\_23+139*x\_24+149*x\_25+135*x\_26+103*x\_27+69*x\_28+51*x\_29+56*x\_30+83*x\_31+118*x\_32+144*x\_33+148*x\_34+128*x\_35+94*x\_36+63*x\_37+50*x\_38+61*x\_39+91*x\_40+125*x\_41+147*x\_42+145*x\_43+121*x\_44+86*x\_45+58*x\_46+50*x\_47+66*x\_48+99*x\_49+132*x\_50 <= 1000;$

$+22*x\_1+29*x\_2+14*x\_3+12*x\_4+27*x\_5+25*x\_6+10*x\_7+17*x\_8+29*x_9+19*x\_10+10*x\_11+23*x\_12+28*x\_13+14*x\_14+12*x\_15+27*x\_16+24*x\_17+10*x\_18+17*x\_19+29*x\_20+19*x\_21+10*x\_22+23*x\_23+28*x\_24+13*x\_25+13*x\_26+28*x\_27+24*x\_28+10*x\_29+18*x\_30+29*x\_31+18*x\_32+10*x\_33+23*x\_34+28*x\_35+13*x\_36+13*x\_37+28*x\_38+24*x\_39+10*x\_40+18*x\_41+29*x\_42+18*x\_43+10*x\_44+24*x\_45+28*x\_46+13*x_47+13*x\_48+28*x\_49+23*x\_50 <= 100;$

(ensure all variables are binary)

bin x_1, x_2, x_3, x_4, x_5,..., x_47, x_48, x_49, x_50;

I then ran Lpsolve with the input file above and got the following result:

Value of objective function: 1099.00000000

Actual values of the variables:

x_7	1
x_11	1
x_18	1
x_25	1
x_26	1
x_29	1
x_33	1
x_37	1
x_44	1

All other variables are zero.

Conclusion:

As shown in the result, while all variables except for the ones listed above are zero, we can conclude that without violating any constraints, the best way to take objects to maximize total values is taking object 7, 11, 18, 25, 26, 29, 33, 37, and 44. And the maximum value we can take is 1099 thousands of dollars.

2. **Now suppose that we are picky about the obejects we take and want items labeled with a prime number to make up at least 70 percent of the total value. How can we take items to maximize the total value now?**

Mathematical formulation:

Similar to the previous question, variables are denoted as $x_i (1 \leq i \leq 50)$ representing whether or not to take an object. If yes, it equals to 1. If not, it equals to 0.

The objective function would still be the same, which is the sum of every object's value:

$$\sum_{i=1}^{50} floor(100 + 50cos(i))x_i$$

We also keep the constraints concerning the weight capacity and the vol-

ume capacity:

$$\sum_{i=1}^{50} floor(100 + 50cos(7i + 1))x_i \leq 1000$$

$$\sum_{i=1}^{50} floor(20 + 10cos(8i - 3))x_i \leq 100$$

And we would add a new constraint here to make sure that the value of prime indexed objects take up 70 percent of the total value:

$$\frac{\text{value of prime indexed items}}{\text{total value}} \geq 0.7$$

$$\frac{\sum_{i=1}^{50} floor(100 + 50cos(i))x_i, i \text{ is prime}}{\sum_{i=1}^{50} floor(100 + 50cos(i))x_i} \geq 0.7$$

Lastly, all variables are still binary because every object is unique:

$$x_1, x_2, ..., x_{50} \in \{0, 1\}$$

Since we will use Lpsolve to solve this LP, I wrote the following Python code to generate the Lpsolve input file:

```
# Jiasui Qin
# This code outputs an lpsolve-formatted LP which
# determines what objects yield the maximum value,
# while the value of prime indexed items take up 70%
# of the total value.

import numpy as np
import math

# this function determines if the index is prime,
# we assume that the index is a positive integer
def isprime(index):
        prime=False
        if(index>1):
                prime = True
                for i in range(2, index//2+1):
                        if ((index % i)==0):
                                prime=False
                                break
```

```
        return prime

# print the objective function
outstring = "max: "
for i in range(1, 51):
        value = math.floor(50 * np.cos(i) + 100)
        outstring = outstring + "+" + str(value) + "*x_" + str(i)
outstring = outstring+";"
print(outstring)

# print the constraint that weight capacity is 1000 kg
outstring = ""
for i in range(1, 51):
        weight = math.floor(50 * np.cos(7*i+1) + 100)
        outstring = outstring + "+" + str(weight) + "*x_" + str(i)
outstring = outstring+"<=1000;"
print(outstring)

# print the constraint that volume capacity is 100 liters
outstring = ""
for i in range(1, 51):
        volume = math.floor(10 * np.cos(8*i-3) + 20)
        outstring = outstring + "+" + str(volume) + "*x_" + str(i)
outstring = outstring+"<=100;"
print(outstring)

# print the constraint that the value of prime indexed items take up 70%
# of the total value
outstring = ""
for i in range(1, 51):
        if(isprime(i)):
                value = math.floor(50 * np.cos(i) + 100)/0.7
                outstring = outstring + "+" + str(value) + "*x_" + str(i)
outstring = outstring+">="
for i in range(1, 51):
        value = math.floor(50 * np.cos(i) + 100)
        outstring = outstring + "+" + str(value) + "*x_" + str(i)
outstring = outstring+";"
print(outstring)

# define variables to be bianary
outstring = "bin "
for i in range(1, 50):
        outstring = outstring + "x_" + str(i) + ", "
```

outstring = outstring+"x_50;"
print(outstring)

After running the python code above, I got the following Lpsolve input file:

max: $+127*x\_1+79*x\_2+50*x\_3+67*x\_4+114*x\_5+148*x\_6+137*x\_7+92*x\_8+54*x\_9+58*x\_10+100*x\_11+142*x\_12+145*x\_13+106*x\_14+62*x\_15+52*x\_16+86*x\_17+133*x\_18+149*x\_19+120*x\_20+72*x\_21+50*x\_22+73*x\_23+121*x\_24+149*x\_25+132*x\_26+85*x\_27+51*x\_28+62*x\_29+107*x\_30+145*x\_31+141*x\_32+99*x\_33+57*x\_34+54*x\_35+93*x\_36+138*x\_37+147*x\_38+113*x\_39+66*x\_40+50*x\_41+80*x\_42+127*x\_43+149*x\_44+126*x\_45+78*x\_46+50*x\_47+67*x\_48+115*x\_49+148*x\_50;$
$+92*x\_1+62*x\_2+50*x\_3+62*x\_4+93*x\_5+127*x\_6+148*x\_7+144*x\_8+119*x\_9+84*x\_10+57*x\_11+50*x\_12+68*x\_13+101*x\_14+134*x\_15+149*x\_16+140*x\_17+111*x\_18+76*x\_19+53*x\_20+52*x\_21+75*x\_22+110*x\_23+139*x\_24+149*x\_25+135*x\_26+103*x\_27+69*x\_28+51*x\_29+56*x\_30+83*x\_31+118*x\_32+144*x\_33+148*x\_34+128*x\_35+94*x\_36+63*x\_37+50*x\_38+61*x\_39+91*x\_40+125*x\_41+147*x\_42+145*x\_43+121*x\_44+86*x\_45+58*x\_46+50*x\_47+66*x\_48+99*x\_49+132*x\_50 <= 1000;$
$+22*x\_1+29*x\_2+14*x\_3+12*x\_4+27*x\_5+25*x\_6+10*x\_7+17*x\_8+29*x_9+19*x\_10+10*x\_11+23*x\_12+28*x\_13+14*x\_14+12*x\_15+27*x\_16+24*x\_17+10*x\_18+17*x\_19+29*x\_20+19*x\_21+10*x\_22+23*x\_23+28*x\_24+13*x\_25+13*x\_26+28*x\_27+24*x\_28+10*x\_29+18*x\_30+29*x\_31+18*x\_32+10*x\_33+23*x\_34+28*x\_35+13*x\_36+13*x\_37+28*x\_38+24*x\_39+10*x\_40+18*x\_41+29*x\_42+18*x\_43+10*x\_44+24*x\_45+28*x\_46+13*x_47+13*x\_48+28*x\_49+23*x\_50 <= 100;$
$+112.85714285714286*x\_2+71.42857142857143*x\_3+162.85714285714286*x\_5+195.71428571428572*x\_7+142.85714285714286*x\_11+207.14285714285717*x\_13+122.85714285714286*x\_17+212.85714285714286*x\_19+104.28571428571429*x\_23+88.57142857142858*x\_29+207.14285714285717*x\_31+197.14285714285717*x\_37+71.42857142857143*x\_41+181.42857142857144*x\_43+71.42857142857143*x\_47 >= +127*x\_1+79*x\_2+50*x\_3+67*x\_4+114*x\_5+148*x\_6+137*x\_7+92*x\_8+54*x\_9+58*x\_10+100*x\_11+142*x\_12+145*x\_13+106*x\_14+62*x\_15+52*x\_16+86*x\_17+133*x\_18+149*x\_19+120*x\_20+72*x\_21+50*x\_22+73*x\_23+121*x\_24+149*x\_25+132*x\_26+85*x\_27+51*x\_28+62*x\_29+107*x\_30+145*x\_31+141*x\_32+99*x\_33+57*x\_34+54*x\_35+93*x\_36+138*x\_37+147*x\_38+113*x\_39+66*x\_40+50*x\_41+80*x\_42+127*x\_43+149*x\_44+126*x\_45+78*x\_46+50*x\_47+67*x\_48+115*x\_49+148*x\_50;$
(ensure all variables are binary)
bin x_1, x_2, x_3, x_4, x_5,..., x_47, x_48, x_49, x_50;

I then ran Lpsolve with the input file above and got the following result:

Value of objective function: 995.00000000

Actual values of the variables:
x_7 1
x_11 1
x_18 1
x_19 1
x_29 1
x_37 1
x_43 1
x_44 1
All other variables are zero.

Conclusion:

As shown in the result, we can conclude that the best way to take objects in this case is taking object 7, 11, 18, 19, 29, 37, 43, and 44. Compared to the previous question, we now only take 8 items instead of 9. The maximum value we can take is 995 thousands of dollars, which is 104 thousands of dollars lower. However, we do take more items that are labeled with prime numbers in this case.

3. **Now suppose that these objects are no longer unique, we can take as many as we want for each labeled item. Find the way to maximize the total value in this case with the same constraints on the weight capacity and the volume capacity.**

Mathematical formulation:

Variables here are also denoted as $x_i (1 \leq i \leq 50)$. Because objects now are not unique, variables represent the number of each type of objects to take.

Since we still want to maximize the total value, the objective function is the same:

$$\sum_{i=1}^{50} floor(100 + 50cos(i))x_i$$

Constraints on the weight capacity and the volume capacit are also the same:

$$\sum_{i=1}^{50} floor(100 + 50cos(7i + 1))x_i \leq 1000$$

$$\sum_{i=1}^{50} floor(20 + 10cos(8i - 3))x_i \leq 100$$

We don't have to set the variables to be binary in this case because they are no longer unique. However, we can only take the whole object and we can't take a negative amount. Therefore, all variables must be non-negative integer values:

$$x_1, x_2, ..., x_{50} \in \mathbb{Z}_{\geq 0}$$

Since we will use Lpsolve to solve this LP, I wrote the following Python code to generate the Lpsolve input file:

```
# Jiasui Qin
# This code outputs an lpsolve-formatted LP which
# determines what objects yield the maximum value
# when objects are not unique.

import numpy as np
import math

# print the objective function
outstring = "max: "
for i in range(1, 51):
        value = math.floor(50 * np.cos(i) + 100)
        outstring = outstring + "+" + str(value) + "*x_" + str(i)
outstring = outstring+";"
print(outstring)

# print the constraint that weight capacity is 1000 kg
outstring = ""
for i in range(1, 51):
        weight = math.floor(50 * np.cos(7*i+1) + 100)
        outstring = outstring + "+" + str(weight) + "*x_" + str(i)
outstring = outstring+"<=1000;"
print(outstring)

# print the constraint that volume capacity is 100 liters
outstring = ""
for i in range(1, 51):
        volume = math.floor(10 * np.cos(8*i-3) + 20)
        outstring = outstring + "+" + str(volume) + "*x_" + str(i)
```

```
outstring = outstring+"<=100;"
print(outstring)

# define variables to be integer values
outstring = "int "
for i in range(1, 50):
        outstring = outstring + "x_" + str(i) + ", "
outstring = outstring+"x_50;"
print(outstring)
```

After running the python code above, I got the following Lpsolve input file:

max: $+127*x\_1+79*x\_2+50*x\_3+67*x\_4+114*x\_5+148*x\_6+137*x\_7+92*x\_8+54*x\_9+58*x\_10+100*x\_11+142*x\_12+145*x\_13+106*x\_14+62*x\_15+52*x\_16+86*x\_17+133*x\_18+149*x\_19+120*x\_20+72*x\_21+50*x\_22+73*x\_23+121*x\_24+149*x\_25+132*x\_26+85*x\_27+51*x\_28+62*x\_29+107*x\_30+145*x\_31+141*x\_32+99*x\_33+57*x\_34+54*x\_35+93*x\_36+138*x\_37+147*x\_38+113*x\_39+66*x\_40+50*x\_41+80*x\_42+127*x\_43+149*x\_44+126*x\_45+78*x\_46+50*x\_47+67*x\_48+115*x\_49+148*x\_50;$

$+92*x\_1+62*x\_2+50*x\_3+62*x\_4+93*x\_5+127*x\_6+148*x\_7+144*x\_8+119*x\_9+84*x\_10+57*x\_11+50*x\_12+68*x\_13+101*x\_14+134*x\_15+149*x\_16+140*x\_17+111*x\_18+76*x\_19+53*x\_20+52*x\_21+75*x\_22+110*x\_23+139*x\_24+149*x\_25+135*x\_26+103*x\_27+69*x\_28+51*x\_29+56*x\_30+83*x\_31+118*x\_32+144*x\_33+148*x\_34+128*x\_35+94*x\_36+63*x\_37+50*x\_38+61*x\_39+91*x\_40+125*x\_41+147*x\_42+145*x\_43+121*x\_44+86*x\_45+58*x\_46+50*x\_47+66*x\_48+99*x\_49+132*x\_50 <= 1000;$

$+22*x\_1+29*x\_2+14*x\_3+12*x\_4+27*x\_5+25*x\_6+10*x\_7+17*x\_8+29*x_9+19*x\_10+10*x\_11+23*x\_12+28*x\_13+14*x\_14+12*x\_15+27*x\_16+24*x\_17+10*x\_18+17*x\_19+29*x\_20+19*x\_21+10*x\_22+23*x\_23+28*x\_24+13*x\_25+13*x\_26+28*x\_27+24*x\_28+10*x\_29+18*x\_30+29*x\_31+18*x\_32+10*x\_33+23*x\_34+28*x\_35+13*x\_36+13*x\_37+28*x\_38+24*x\_39+10*x\_40+18*x\_41+29*x\_42+18*x\_43+10*x\_44+24*x\_45+28*x\_46+13*x_47+13*x\_48+28*x\_49+23*x\_50 <= 100;$

(ensure all variables are integer values)

int x_1, x_2, x_3, x_4, x_5,..., x_47, x_48, x_49, x_50;

I then ran Lpsolve with the input file above and got the following result:

Value of objective function: 1330.00000000

Actual values of the variables:

x_19	1
x_37	1
x_44	7

Conclusion:

In this case when the number of each item is unlimited, we get the result of taking one x_19, one x_37, and seven x_44. Compared to previous questions, we take less type of items, but we take more of x_44 and achieve a larger total value, which is 1330 thousands of dollars.