Integer Programming

Mehmet Hakan Satman (Ph.D.)

December 2, 2025

Contents

1 The Knapsack Problem

Items	1	2	3	4	5	6
Weight	2	3	4	5	9	7
Value	3	4	5	6	10	8

Table 1: Knapsack problem data

The capacity of the knapsack is 15 kg. We want to maximize the total value of the items in the knapsack without exceeding its capacity.

$$\begin{aligned}
&\text{Maximize } Z = 3x_1 + 4x_2 + 5x_3 + 6x_4 + 10x_5 + 8x_6 \\
&\text{Subject to} \\
&\qquad 2x_1 + 3x_2 + 4x_3 + 5x_4 + 9x_5 + 7x_6 \leq 15 \\
&\qquad x_i \in \{0, 1\} \quad \text{for } i = 1, 2, \ldots, 6
\end{aligned}$$

Now, suppose that the item 3 and item 4 are related, meaning that if we include item 3 in the knapsack, we must also include item 4 (If we include item 4, we must also include item 3). This can be modeled with the following constraint:

$$\begin{aligned}
&\text{Maximize } Z = 3x_1 + 4x_2 + 5x_3 + 6x_4 + 10x_5 + 8x_6 \\
&\text{Subject to} \\
&\qquad 2x_1 + 3x_2 + 4x_3 + 5x_4 + 9x_5 + 7x_6 \leq 15 \\
&\qquad x_3 = x_4 \\
&\qquad x_i \in \{0, 1\} \quad \text{for } i = 1, 2, \ldots, 6
\end{aligned}$$

Here is the Julia solution using Operations Research package:

```
using OperationsResearchModels
values = [3, 4, 5, 6, 10, 8]
weights = [2, 3, 4, 5, 9, 7]
capacity = 15
problem = KnapsackProblem(values, weights, capacity)
result = solve(problem)
```

The output:

```
julia> result.objective
18.0

julia> result.selected
6-element Vector{Bool}:
 1
 1
 1
 1
 0
 0
```

2 Project Selection Problem

Project	Year 1	Year 2	Year 3	Return
1	50	20	30	200
2	60	30	20	250
3	40	10	50	150
4	30	40	30	180
5	20	50	40	160
Budget	150	100	120	

Table 2: Project selection data

Project 1 requires 50 units of resource in year 1, 20 units in year 2, and 30 units in year 3. When the project is completed, it returns 200 units. First year bagget is 150 units, second year budget is 100 units, and third year budget is 120 units. We want to select projects to maximize the total return without exceeding the yearly budgets.

$$\text{Maximize } Z = 200x_1 + 250x_2 + 150x_3 + 180x_4 + 160x_5$$

Subject to

$$\begin{aligned}
&50x_1 + 60x_2 + 40x_3 + 30x_4 + 20x_5 \leq 150 \quad \text{(Year 1 Budget)} \\
&20x_1 + 30x_2 + 10x_3 + 40x_4 + 50x_5 \leq 100 \quad \text{(Year 2 Budget)} \\
&30x_1 + 20x_2 + 50x_3 + 30x_4 + 40x_5 \leq 120 \quad \text{(Year 3 Budget)} \\
&x_i \in \{0, 1\} \quad \text{for } i = 1, 2, \ldots, 5
\end{aligned}$$

Here is the Julia solution using Operations Research package:

```
using JuMP, HiGHS
model = Model(HiGHS.Optimizer)
@variable(model, x[1:5], Bin)
@objective(model, Max, 200x[1] + 250x[2] + 150x[3] + 180x[4] + 160x[5])
@constraint(model, 50x[1] + 60x[2] + 40x[3] + 30x[4] + 20x[5] <= 150)
@constraint(model, 20x[1] + 30x[2] + 10x[3] + 40x[4] + 50x[5] <= 100)
@constraint(model, 30x[1] + 20x[2] + 50x[3] + 30x[4] + 40x[5] <= 120)
optimize!(model)
println(value.(x))
```

The output:

```
[1.0, 1.0, 0.0, 1.0, 0.0]
```

3 The Assignment Problem

	Task 1	Task 2	Task 3	Task 4
Worker 1	9	2	7	8
Worker 2	6	4	3	7
Worker 3	5	8	1	8
Worker 4	7	6	9	4

Table 3: Assignment problem cost matrix

We have 4 workers and 4 tasks. The cost of assigning each worker to each task is given in the table above. We want to assign each worker to exactly one task such that the total cost is minimized.

Minimize $Z = 9x_{11} + 2x_{12} + 7x_{13} + 8x_{14} + 6x_{21} + \ldots + 9x_{43} + 4x_{44}$

Subject to

$$
\begin{aligned}
x_{11} + x_{12} + x_{13} + x_{14} &= 1 \quad \text{(Worker 1 assigned to one task)} \\
x_{21} + x_{22} + x_{23} + x_{24} &= 1 \quad \text{(Worker 2 assigned to one task)} \\
x_{31} + x_{32} + x_{33} + x_{34} &= 1 \quad \text{(Worker 3 assigned to one task)} \\
x_{41} + x_{42} + x_{43} + x_{44} &= 1 \quad \text{(Worker 4 assigned to one task)} \\
x_{11} + x_{21} + x_{31} + x_{41} &= 1 \quad \text{(Task 1 assigned to one worker)} \\
x_{12} + x_{22} + x_{32} + x_{42} &= 1 \quad \text{(Task 2 assigned to one worker)} \\
x_{13} + x_{23} + x_{33} + x_{43} &= 1 \quad \text{(Task 3 assigned to one worker)} \\
x_{14} + x_{24} + x_{34} + x_{44} &= 1 \quad \text{(Task 4 assigned to one worker)} \\
x_{ij} &\in \{0, 1\} \quad \text{for } i, j = 1, 2, 3, 4
\end{aligned}
$$

Here is the Julia solution using Operations Research package:

```
using OperationsResearchModels
costs = [9 2 7 8; 6 4 3 7; 5 8 1 8; 7 6 9 4]
problem = AssignmentProblem(costs)
result = solve(problem)
```

The output:

```
julia> result.cost
13.0

julia> result.solution
4×4 Matrix{Float64}:
  0.0  1.0   0.0  0.0
  1.0  0.0  -0.0  0.0
  0.0  0.0   1.0  0.0
 -0.0  0.0   0.0  1.0
```

This means:

- Worker 1 is assigned to Task 2
- Worker 2 is assigned to Task 1
- Worker 3 is assigned to Task 3
- Worker 4 is assigned to Task 4