Silesian University of Technology

FACULTY OF AUTOMATIC CONTROL, ELECTRONICS AND COMPUTER SCIENCE

Advanced Optimization Methods

Integer and binary integer linear programming

Łukasz Kania, Szymon Zosgórnik
Year 1, semester 1, Macro DataScience

May 3, 2020

1 Task 1

Find $min \leftarrow x_1 + x_2 + 2x_3$ under constraints:

$$2x_1 + 3x_2 \geq 5$$
$$x_2 + 2x_3 \geq 4$$
$$x_i \geq 0, int, i = 1, 2, 3$$

2 Task 2

Find the solution to the knapsack problem for $w_{max} = 6$ and the following parameters:

i	v_i	w_u
1	5	4
2	3	3
3	4	2
4	3	2

3 Solution

All solutions were obtained using Branch and Bound method described in the instruction to the exercise. Matlab was used to implement the proper algorithm. At the listings 1 and 2 there are presented functions which were used to solve given problem. Script containing described solutions is shown at the listing 3. Output of the program is attached below.

```
Optimal solution found.
Optimal solution found.
Optimal solution found.
The optimal solution is: 0  2  1
Cost value is: 4

Optimal solution found.
Optimal solution found.
Optimal solution found.
The optimal solution is: 1  0  1  0
Cost value is: -9
```

```
function [Xopt, fval, LB, UB, allIntegers] = LinProg(f, A, b, LB, UB)
    allIntegers = 0;
    [Xopt, fval] = linprog(f, A, b, [], [], LB, UB);
    if isempty(Xopt)
        return
    end
    allIntegers = 1;
    for i = 1:length(Xopt)
        if abs(round(Xopt(i)) - Xopt(i)) > 0.00001
            LB(i) = ceil(Xopt(i));
            UB(i) = floor(Xopt(i));
            allIntegers = 0;
            break
        end
    end
    Xopt = Xopt';
end
```

Listing 1: Wrapper function for linear programming solver

```
function BranchAndBound(f, A, b, LB, UB)
    global results;
    [Xopt, fval, newLB, newUB, allIntegers] = LinProg(f, A, b, LB, UB);
    if isempty(Xopt)
        return;
    end
    if allIntegers
        results(end + 1, :) = {Xopt, fval};
        return;
    end
    if isempty(results) || fval < min([results{:,2}])
        BranchAndBound(f, A, b, newLB, UB);
        BranchAndBound(f, A, b, LB, newUB);
    end
end
```

Listing 2: Branch and bound - recursive function

```matlab
% Task 1
f = [1 1 2];
A = -[2 3 0; 0 1 2];
b = -[5; 4];
LB = [0, 0, 0];
UB = [Inf, Inf, Inf];

global results;
results = {};
BranchAndBound(f, A, b, LB, UB);
[~, solutionIndex] = min([results{:,2}]);
disp("The optimal solution is: " + num2str(results{solutionIndex, 1}))
disp("Cost value is: " + results{solutionIndex, 2})


% Task 2
f = -[5 3 4 3];
A = [4 3 2 2];
b = 6;
LB = [0 0 0 0];
UB = [1 1 1 1];

results = {};
BranchAndBound(f, A, b, LB, UB);
[~, solutionIndex] = min([results{:,2}]);
disp("The optimal solution is: " + num2str(results{solutionIndex, 1}))
disp("Cost value is: " + results{solutionIndex, 2})
```

Listing 3: Main script

4 Results

4.1 Task 1

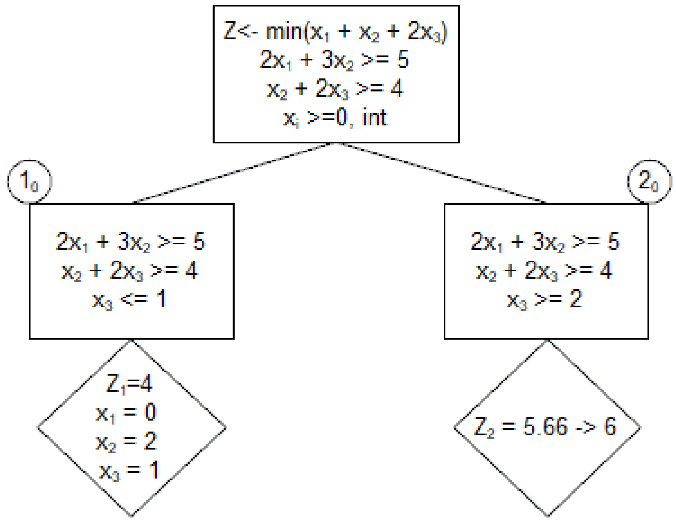

Figure 1: Task 1 results

The Proper solution is shown at the left side of the *Figure 1*. There were only 3 steps needed to obtain it.
The first was to calculate constraints basing on information given in task content. After that there were 2 possible steps. To calculate 1_0 and 2_0 part of the tree. $X's$ results of the first branch were integers with minimum Z_1 equal to 4. The second one was $Z_2 = 5.6(6)$ which gives two information. The first one is that, the $x's$ aren't integers, the second more important is the new Z_2 is greater than the 'parallel' Z_1, which automatically means, this branch should be discarded, because the best solution obtained from that part of the tree would be ≥ 6.
To sum up the *cost function value is 6* with *decision variables* equal to $x_1 = 0, x_2 = 2, x_3 = 1$

4.2 Task 2

The solution is shown at the right site of the *Figure 2* below. The results are kind of similar to the first task. The cost function value obtained in branch 1_0 was (almost - because of the cost value being approximated) the same, as the one obtained in branch 2_0. Searching further into branch 1_0 would only show results < 9 so the algorithm was stopped, and the value obtained in branch 2_0 was accepted as solution, as every decision variable was from desired set - $0, 1$.
To sum up the *cost function value is 9* with *decision variables* equal to $x_1 = 1, x_2 = 0, x_3 = 1, x_4 = 0$

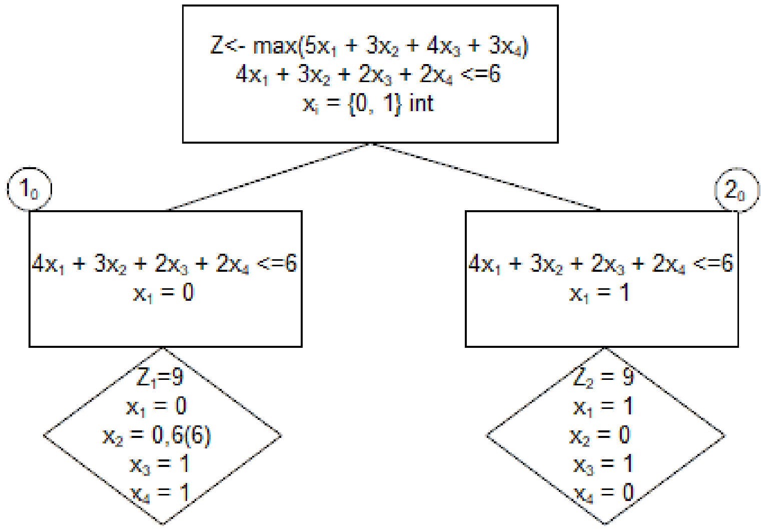

Figure 2: Task 2 results