Advanced Forest Planning

Dr. Bettinger

Chaz Merritt

09/06/2024

Assignment 2 - Linear Programming

1. Objective Function

# Define binary decision variables for MIP: fully harvested (1) or not harvested (0)  
x = pulp.LpVariable.dicts("x", ((i, j) for i in range(n\_stands) for j in range(n\_periods)), cat='Binary')  
  
# Objective function: Minimize the deviation from the target harvest volume  
deviation = pulp.LpVariable.dicts("deviation", (j for j in range(n\_periods)), lowBound=0)  
  
# Objective: Minimize the sum of deviations from the target harvest volume  
prob += pulp.lpSum(deviation[j] for j in range(n\_periods))

1. Resource Constraints

# 1. Each stand can be harvested at most once  
for i in range(n\_stands):  
 prob += pulp.lpSum([x[(i, j)] for j in range(n\_periods)]) <= 1  
  
# 2. Minimum harvest age constraint

for i in range(n\_stands):  
 for j in range(n\_periods):  
 current\_age = initial\_ages[i] + j \* 5 # Adjust the age for the current period  
 if current\_age < 35:  
 prob += x[(i, j)] == 0 # Prevent harvesting if age is less than 35  
  
# 3. Adjacency constraint: No two adjacent stands can be harvested in the same period  
for pair in adjacency\_pairs:  
 i, k = pair   
 for j in range(n\_periods):  
 prob += x[(i, j)] + x[(k, j)] <= 1

1. Accounting Rows

# 4. Harvest volume in each period and deviation constraint  
for j in range(n\_periods):  
 harvest\_volume = pulp.lpSum([acres\_per\_stand \* volume\_per\_acre[i][j] \* x[(i, j)] for i in range(n\_stands)])  
 prob += harvest\_volume - deviation[j] <= T  
 prob += harvest\_volume + deviation[j] >= T

1. Solution

Objective value: 36.0000000

Lower bound: 0.000

Gap: inf

Enumerated nodes: 42586749

I didn’t get a completely optimal minimization, but I also didn’t get to let the program run for more than an hour. Also I modified the program to output the MBF and harvest schedule to a csv file once the optimal solution is reached, which I didn’t let it do due to time.