2023학년도 1학기

Computer Algorithms

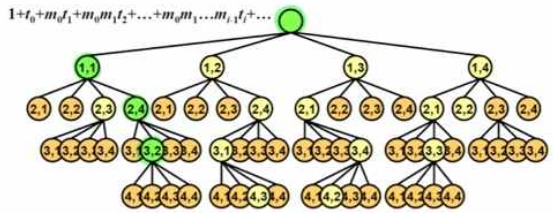
Hands-On Assignment 2 Report

과 목 명	ComputerAlgorithms(01)
담 당 교 수	Se Eun Oh
학 과	컴퓨터공학과
학 번	2071035
이 름	이소민 (LeeSomin)
제 출 일	2023.05.03

Computer Algorithms 2071035
HA#3-1 : Monte Carlo Simulation for n-Queens Problem Lee Somin

m_i : # of promising child nodes at level i
t_i : total # of child nodes at level i

① # of nodes expanded when selecting (1,1) and (2,4)?

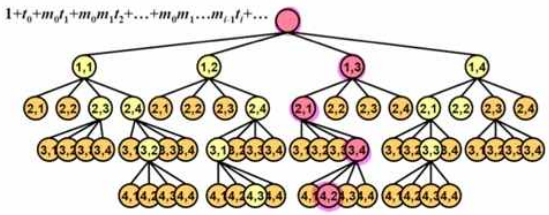


level 0 : $m_0 = 4$ $t_0 = 4$
level 1 : $m_1 = 2$ $t_1 = 4$
level 2 : $m_2 = 1$ $t_2 = 4$
level 3 : $m_3 = 0$ $t_3 = 4$

$1 + t_0 + m_0t_1 + m_0m_1t_2 + m_0m_1m_2t_3$
$= 1 + 4 + 4 \times 4 + 4 \times 2 \times 4 + 4 \times 2 \times 1 \times 4$
$= 1 + 4 + 16 + 32 + 32 = 85$

A. estimation : 85

② # of nodes expanded when selecting (1,3)?

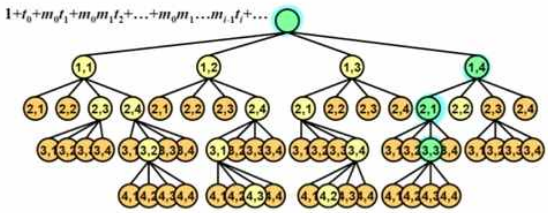


level 0 : $m_0 = 4$ $t_0 = 4$
level 1 : $m_1 = 1$ $t_1 = 4$
level 2 : $m_2 = 1$ $t_2 = 4$
level 3 : $m_3 = 1$ $t_3 = 4$

$1 + t_0 + m_0t_1 + m_0m_1t_2 + m_0m_1m_2t_3$
$= 1 + 4 + 4 \times 4 + 4 \times 1 \times 4 + 4 \times 1 \times 1 \times 4$
$= 1 + 4 + 16 + 16 + 16 = 53$

A. estimation : 53

③ # of nodes expanded when selecting (1,4) and (2,1)?



level 0 : $m_0 = 4$ $t_0 = 4$
level 1 : $m_1 = 2$ $t_1 = 4$
level 2 : $m_2 = 1$ $t_2 = 4$
level 3 : $m_3 = 0$ $t_3 = 4$

$1 + t_0 + m_0t_1 + m_0m_1t_2 + m_0m_1m_2t_3$
$= 1 + 4 + 4 \times 4 + 4 \times 2 \times 4 + 4 \times 2 \times 1 \times 4$
$= 1 + 4 + 16 + 32 + 32 = 85$

A. estimation : 85

ha3_2.py

1. Code:

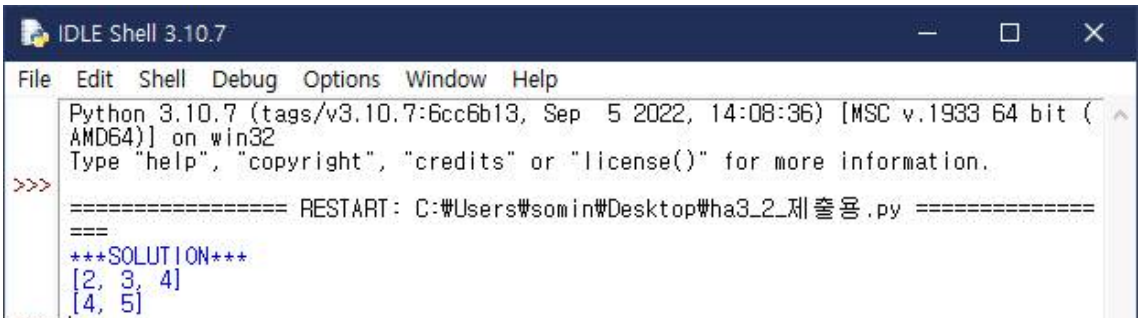
ha3_2_제출용.py - C:₩Users₩somin₩Desktop₩ha3_2_제출용.py (3.10.7)

File Edit Format Run Options Window Help

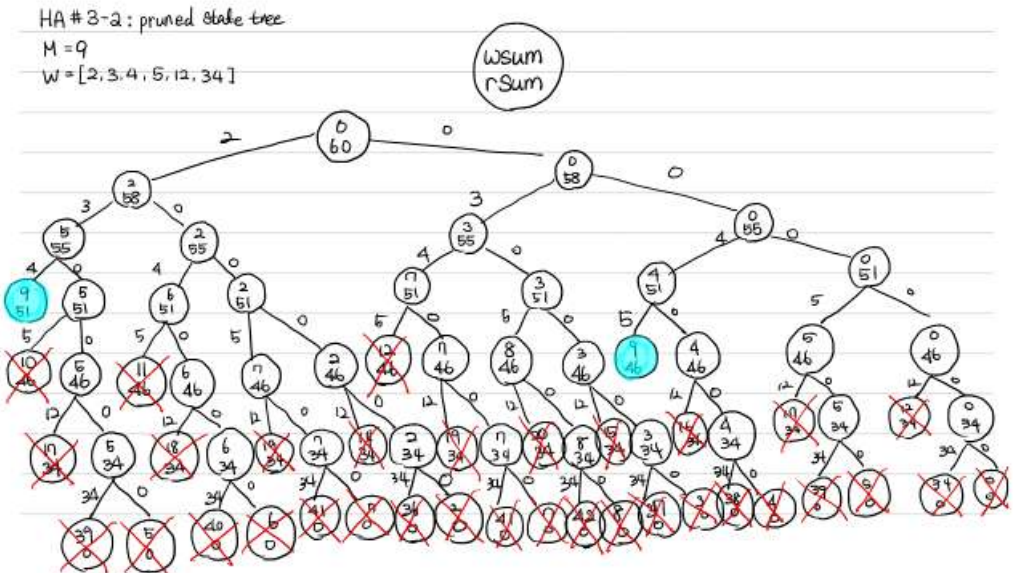
```
# ComputerAlgorithms 2071035 Lee Somin
# HA 3-2: Subset Sum

M = 9 # Value of subset sum wanted
w = [3, 34, 4, 12, 5, 2] # array which to find the subset sum from

def sum_of_subsets(i,wsum,rsum,nodes,W):
    #check whether to keep expanding
    if((wsum+rsum>=M) and ((wsum == M) or (wsum+W[i+1]<=M))):
        # when subset sum equals to M is found,
        # print the solution
        if(wsum == M):
            print(nodes)
        else:
            # select i+1th int and check if
            nodes.append(W[i+1])
            # update wsum, rsum and call sum_of_subset for child nodes
            # to check if child nodes meet the subset sum condition
            sum_of_subsets(i+1,wsum+W[i+1],rsum-W[i+1],nodes,W)
            # deselect(i+1th int)
            nodes.pop()
            # update wsum, rsum and call sum_of_subset for child nodes
            # to check if child nodes meet the subset sum condition
            sum_of_subsets(i+1,wsum,rsum-W[i+1],nodes,W)


if __name__=='__main__':
    nodes=[] # array of elements added for subset sum
    rsum=0 # sum of remaining values in original array
    # sort w in to W which is in ascending order
    W = sorted(w)
    # add all elements of original array to initialize rsum
    for i in w:
        rsum+=i
    # find subset that satisfies the condition
    print("***SOLUTION***")
    # start from node 0, and the function will call sum_of_subsets(0 ~ )
    sum_of_subsets(-1,0,rsum,nodes,W)

```

Ln: 1 Col: 0

2. Result:



3. Pruned State Tree



rsum이 M보다 작을 때, wsum이 M보다 클 때 expand를 하지 않으며 backtracking algorithm을 적용시키면 [2, 3, 4]와 [4, 5]의 답이 나오는데, 이는 코드에서 나온 솔루션의 결과와 같다.

이 backtracking에서는 36개의 노드에서 Expand를 하고 다음 결과를 탐색한다.