Week VIII

Overview

- Enumeration;
- Dynamic Programming.

Enumeration

Enumeration is the process that **list** of all the solutions and **test** each one individually.

Dynamic Programming

Dynamic programming is a solution method by breaking them into a collection of simpler **sub-problems**, solving them once and storing their solutions.

It requires two properties:

- Recursion;
- Suboptimal Structure.

Knapsack Problem

```
m_{0,weight} ← 0
for each cell m_{item,weight} do
    m_{item,weight} ← m_{item−1,weight} if weight_{item} > weight
    m_{item,weight} ← max(m_{item−1,weight}, m_{item−1,w−weight_{item}} + v_i) if weight_{item} ≤ weight
```

Tower of Hanoi

Algorithm 1: Hanoi(BFS)

Input: Disk, Source, Destination, Auxiliary Rod

```
if Disk == 1 then
    move Disk from Source to Destination;
else
    Hanoi(Disk-1,Source,Auxiliary Rod, Destination)
    move Disk from Source to Destination;
    Hanoi(Disk-1, Auxiliary Rod, Destination, Source)
```

Bellman-Ford Algorithm

Algorithm 2: Bellman-Ford's Algorithm

Input: undirected, connected graph G, weights $c: E(G) \rightarrow \mathbb{R}$, nodes V, source s

```
d_v : distance to reach node v
p_v : node predecessor to node v
Q ← ∅ : set of "unknown distance" nodes.
for each node v in V do
    d_v ← ∞
    p_v ← FALSE
d_s ← 0
for each node v in V do
    for each edge (u, v) ∈ E do
        temp-dist ← d_u + c_uv
        if temp-dist < d_v then
            d_v ← temp-dist
            p_v ← u
    for each edge (u, v) ∈ G do
        if d_u + c_uv < d_v then
            return Error: Negative Cycle Exist
return d_v,p_v
```

Fernando Dias
fernando.dias@aalto.fi

Aalto University