DM d'algorithmique

October 13, 2018

1 Question 1.2

1.1 Pseudo code

```
function optimum_tree (alpha, beta):

    let R # the roots
    let P # the weighted paths
    let W # the summed weights
    let n = size(alpha)

    # Initialise values
    for i in 0..n:
        W[i, i] = P[i, i] = alpha[i] # summed paths and weights initialisation
        R[i, i] = i

    # Pre compute W
    for i in 0..n:
        for j in i+1..n:
            W[i, j] = W[i, j-1] + beta[j] + alpha[j]

    # Build R and P
    for subtreeLen in 2..n-1:
        # We build all the possible optimal trees of length subtreeLen
        for i in 0..n-subtreeLen:
            j = i + subtreeLen

            # find the best root for the subtree i..j
            # according to the article, we don’t have to check all
            # the values from i to j but only a subset:
            for k in R[i, j-1]..R[i+1, j]:
                currentWeight = P[i, k+1] + P[k+1, j] + W[i, j]
                # Then update the current values of R and P if we have a better value
                update_min_weight_and_min_root(R, k, P, currentWeight)
```

```
return R
```

1.2 Preuve d'optimalité

L'algorithme consiste en la construction de l'arbre en respectant la contrainte :

$$P_{ij} = \min_{R_{i,j-1} < k \leq R_{i+1,j}} (P_{i,k-1} + P_{k,j}) + W_{ij} \tag{1}$$

Où P, R et W représentent :

- P_{mn} le poids du sous-arbre optimal consitut des noeuds $A_m...A_n$
- R_{mn} l'indice de sa racine
- W_{mn} la somme $\alpha_m, \beta_{m+1}, \alpha_{m+1}, ..., \alpha_n$

Montrons la construction d'un arbre respectant pour tout intervalle $A_i, A_{i+1}...Aj$ la contrainte (1) produit un arbre optimal.

Tout d'abord : Si un arbre de noeuds $A_i...A_j$ de racine k avec $i < k \leq j$ est optimal, alors des sous-arbres droit et gauche de cet arbre, constitués respectivement des noeuds $A_i...A_{k-1}$ et $A_{k+1}...A_j$ sont optimaux. En effet dans le cas contraire, si un des sous-arbres est sous-optimal, en choisissant à sa place un sous-arbre optimal, on améliore le poids de l'arbre $A_i...A_j$ qui ne serait donc pas optimal.

On en déduit que pour construire un arbre optimal, il suffit de choisir sa racine telle que le poids de l'arbre complet constitué du sous-arbre optimal à gauche de la racine, et du sous-arbre optimal à droite, soit minimal, autrement dit, trouver R_{ij} tel que :

$$P_{i,R_{ij}-1} + P_{R_{ij},j} = \min_{i < k \leq j} (P_{i,k-1} + P_{k,j}) \tag{2}$$

D'autre part, d'après le corollaire de l'article, on sait qu'il existe une telle racine R_{ij} respectant la condition $R_{i,j-1} \leq R_{ij} \leq R_{i+1,j}$. On en déduit qu'il suffit de chercher la racine seulement dans cet intervalle, à savoir trouver R_{ij} tel que :

$$P_{i,R_{ij}-1} + P_{R_{ij},j} = \min_{R_{i,j-1} \leq k \leq R_{i+1,j}} (P_{i,k-1} + P_{k,j}) \tag{3}$$

On en déduit que la procédure induite par (1) permet bien de construire un arbre optimal.