Problema arb (Andrei Ciocan)

Prima oara vom sorta lista fiilor fiecărui nod după valoarea iniţială. În continuare vom împarţi fiecare întrebare in subprobleme : si anume aflându-ne in nodul x, să aflăm numărul de drumuri care trebuie să treacă prin părintele lui x astfel încât x sa fie accesat de y ori.
Soluţia acestor subprobleme se rezolva in O(log(n)) : Avem lista fiilor sortata descrescător după valoare : V1 >= V2 >= V3 ….>= Vx >= ….Vp .
după (v1-v2) operaţii lista va arata : v2, v2, v3, ….vx …vp
după alte (v3-v2) *2 operaţii lista va arata : v3, v3 , v3 ,v4 ,….vx….vp

Se observa ca după aceste operaţii primul element din lista a fost accesat de v1-v3 ori, al doilea de v2-v3 ori etc. Astfel, Vx va fi accesat când lista fiilor va arata : V(x)-1 , V(x)-1, V(x)-1, …..V(x)-1, Vx , $V(x+_{1})$,$V(x+_{2})$ ….Vp. Astfel X este accesat de y ori când valoarea din nodurile pana la x (inclusiv x) vor fi V(x) – y. Astfel va trebui sa căutăm binar ultima valoare din lista care este mai mare sau egala cu V(x)-y .

Având calculata rezultatul pentru nivelul curent, fie ea Y1, o vom trimite părintelui (il notam P) si vom calcula de cate accesări a părintelui lui P sunt necesare pentru ca P sa fie accesat de Y1 ori etc. Soluţia o vom afla când ajungem în rădăcină. Astfel, in fiecare nod facem maxim log (n) operaţii, adâncimea arborelui este log(n), complexitatea fiind query*log (n)*hmax.