Problema Rafaela – descrierea soluției

Stud. Andrei Ciocan şi Vlad Ionescu
Universitatea Politehnica Bucuresti

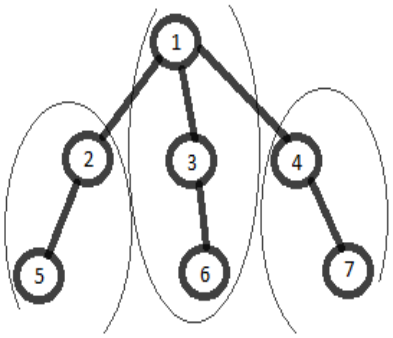

Fig. 1

Solutie 100 puncte: O(logN) pe query, O(log^2N) pe update

Se aplica algoritmul de heavy-path-decomposition, obtinandu-se o impartire a arborelui in lanturi (algoritmul garanteaza ca, pentru a se ajunge de la un nod catre radacina se parcurg maxim O(logN) lanturi).
Prima observatie care conduce catre rezolvarea problemei este ca, un nod (care va face si el parte dintr-un lant), are un singur fiu care continua propriul lant, restul fiilor fiind extremitati ale unor noi lanturi (spre exemplu, in figura alaturata, nodul 1 isi continua propriul lant spre nodul 3, ceilalti fii, nodurile 2 si 4 fiind capetele unor noi lanturi).
O alta observatie este ca numarul maxim de cetatenti care pleaca spre capitala folosesc o muchie incidenta capitalei. Vom incerca sa aflam cati cetateni vin dinspre fiii si, de asemenea, cati vin „din sus” (dinspre radacina spre capitala). Pentru ultima problema raspunsul este simplu: numarul total de cetateni minus numarul de cetateni care se afla in subarborele capitalei. Ne vom concentra de acum incolo sa aflam pentru fiii unei capitale care este maximul de cetateni care vin pe o muchie.
In acest moment putem diferentia fiii unui nod: nodul (unic) care continua lantul si ceilalti:

1) Cum aflam numarul de cetateni care trec pe muchia care uneste capitala de fiul care continua lantul? In momentul in care avem de efectuat un update (se adauga sau se scade o valoare dintr-un nod), toate nodurile pornind din acel nod spre radacina vor fi afectate. Pe fiecare lant se va mentine un arbore de intervale separat, iar un update va modifica arborii de intervale ale tuturor lanturilor care duc spre radacina. Pentru a afla numarul de cetateni care vin dinspre acest nod catre tata, e suficient sa interogam arborele de intervale. Astfel aflam raspunsul pentru nodul care continua lantul.
2) Cum aflam maximul pentru toate celelalte noduri (care nu continua lantul)? Dupa cate am evidentiat mai sus, aceste noduri cu siguranta vor fi extremitati ale unor noi lanturi. Astfel, pentru fiecare nod vom tine un set (o multime ordonata de elemente) cu valorile (numarul de cetateni din subarborele respectiv) fiecarui fiu care nu continua lantul (caci pentru acest fiu am stabilit deja cum putem afla raspunsul). In momentul in care apare un update, pe langa actualizarea arborilor de intervale ale fiecarui lant catre radacina, vom actualiza si set-urile parintilor acestor noduri „extreme” ale lanturilor (si aceste noduri vor fi afectate de update-uri). Actualizarea unui set se face „explicit”, eliminand din set vechea valoare, updatand-o si inserand-o pe cea noua. Cand avem un query pentru o capitala, vom alege (printre celelalti 2 candidati posibili descrisi mai sus) si maximul din acest set.

Obtinem astfel o complexitate de: O(logN) pe query (se alege maximul din set-ul nodului + un query pe arborele de intervale aferent fiului care continua lantul + aflarea numarului de cetateni care trec pe muchia „de sus” – dinspre radacina) si O(log^2N) pe update (updatarea celor O(logN) arbori de intervale si updatarea set-urilor parintilor nodurilor „extreme” de pe lanturi).