Solutie kinder, autor Andrei Grigorean

Pentru a implementa eficient cele 3 operatii descrise avem nevoie de un arbore de intervale. In fiecare nod al arborelui de intervale ne vom mentine un trie (radix tree). In acest trie vom insera siruri de lungime logM formate din 0 si 1 reprezentand descompunerile in baza 2 (incepand cu cel mai semnificativ bit) ale tipurilor de oua pe care le detin copiii asociati nodului din arborele de intervale. In mod evident, fiecare frunza a trie-ului va fi ascotiata unui tip de oua. In fiecare frunza vom mentine 2 valori reprezentand numarul de oua rosii, respectiv albastre, din tipul asociat frunzei. In nodurile interne ale trie-ului vom mentine deasemenea 2 valori, reprezentand suma valorilor corespondente din frunzele subarborelui asociat nodului intern. Sa studiem detaliat cum vom proceda in cazul fiecarei operatii:

1. Update – c t p nr: Va trebui sa descompunem numarul t in baza 2 si sa inseram sirul obtinut (sirul incepe cu cel mai semnificiativ bit al numarului t) in cele logN trie-uri din nodurile arborelui de intervale care contin informatie despre copilul c.
2. Update c t: Mai intai aflam cate oua rosii, resepectiv albastre, de tipul t detine copilul c. Acest lucru se face uitandu-ne la trie-ul din frunza arborelui de intervale asociata copilului c. Apoi trebuie sa reactualizam informatiile trie-urilor din nodurile arborelui de intervale care contin informatii despre copilul c.
3. Query a b p x – Pentru orice interval [a, b], ne intereseaza maxim logN noduri ale arborelui de intervale, respectiv trie-urile asociate lor. Pentru a raspunde la query, vom afla pe rand bitii rezultatului, incepand cu cel mai semnificativ bit. Pentru a afla primul bit, ne intereseaza suma valorilor din fiii stangi ai radacinilor trie-urilor. Daca aceasta suma este mai mare sau egala cu x, primul bit este 0, altfel 1. Urmatorii biti ai rezultatului se afla intr-o maniera similara.