АНАЛИЗ НА РЕШЕНИЕТО НА ЗАДАЧА МРАВКИ

Според условието, можем да си представим обектите в задачата като кореново дърво (множеството от градове), в което всеки град преставлява връх в дървото (номериран с число от 1 до N) и за всеки връх – подредено множество от числа, номерирани от 1 до M (мравуняците във всеки град). Коренът на дървото ще бъде първия връх, а родителят на град X в дървото ще бъде градът, от който мравките са се преселили в град X.

За целта на анализа първо ще улесним задачата, като въведем правилното, че родителят на връх X > 1 ще бъде връх (X – 1). Тогава можем да си сведем задачата до следната: За даден масив от M числа, прилагам N – 1 пъти двойката задания от : "увеличаване числата в интервала [L:R] със V" и "Каква е сумата на числата от i до j". За да се реши задачата със сложност, подходяща за времевия лимит на задачата, трябва да се използва бърза структура от данни. В авторското решение се изполозва интервално дърво. Това е дърво с ~2M върха, което позволява всяко от заданията (увеличаване и търсене на сума) да се изпълни със сложност O(lgM).

Дървото се нарича интервално, защото всеки връх в него пази отговорът на задачата за даден интервал от поредни елементи на масива (мравуняците). То е двоично дърво, като двата наследника на всеки връх пазят отговора за под интервали на родителя си. По конкретно, ако даден връх съдържа отговора за интервалът [L:R], то единият наследник съдържа отговор за [L, (L+R)/2], а другият за ((L + R)/2:R] (целочислено деление).

В конкретната задача, всеки връх на интервалното дърво (отговарящ за интервала [L:R]) ще съдържа две числа: sum и incr. Sum ще бъде равно на сумата на числата в този интервал, като се игнорират всички задания за увеличаване на елементи, който покриват изцяло интервала. Т.е. ако някое задание увеличава елементите в интервал L1, R1, ще бъде игнорирано от този връх, ако L1<L<=R<R1. Incr ще е сумата всички V, на заявки за увеличаване на интвервал [L, R], които изпълняват условието: L1<=L<=R<=R1 и това условие не е изпълнено за никой от родителите на R. Това е условие (I).

Тези стойности ни позволяват да дефинираме следните рекурсивни зависимости: ако връх P има два наследника Q и R, то P.sum = Q.sum + R.sum + (R-L+1) * P.incr. Когато правим заявка от за увеличаване на елементи дефинирана от L1, R1, V1, трябва да увеличим X.incr със текущото V1, за всяко X, което изпълнява гореспоменатото условие (I), и после да приложим рекурсивната зависимост в/у всички върхове по пътя от X до корена на дървото.

За да можем с даденото дърво да извлечем отговора за даден интервал i, j ще дефинираме следната фунцкия f(i, j, node), където node е връх от интервалното дърво с деца node(Left) и node(Right):

Ако i <= node.L <= node.R<= J => f(i, j, node) = node.sum * allIncr, където allIncr е равно на сумата на всички X.incr, за всяко X по пътя от корена до node.

Ако i > node.R или node.L > j (т.е. върха не се засива с исканият интервал) => f(i, j, node) = 0,

в противен случай => f(i,j, node) = f(i, j, node.Left) + f(i, j, node.Right).

Така дефинирахме двете искани операции и може да се докаже, че имплементирани правилно и двете имат сложност O(lgM) където M е броят елементи на масива.

След като решихме тази подзадача, можем да я използваме в решението на авторската задача за произволно дърво. За да успеем да го направим ще трябва да направим нашето дърво персистентно. Това означава, че във всеки момент трябва да имаме достъп до дървото, изглеждащо по начина, по който е изглеждало преди всяка операция до сега. Т.е. да можем да върнем "времето назад" и да работим с интервалното дърво, отговарящо масива от числа (мравуняци) сякаш даден брой последни операции не са се случили. Нека означим всяко едно такова дърво с t(x), където x е даден връх (град) от първоначалното дърво и t(x) е дървото след като сме приложили двете операции за дадения връх. T(1) е началният момент, който показва големината на мравуняците в корена на нашето дърво, t(2) е моментът в интервалното дърво, след като сме направили операциите за връх 2 от град 2 и т.н.. Тоест ако имаме два върха X и Y, за които връх X е баща на връх Y в първоначалното дърво на градовете, то интервалното дърво T(Y) ще бъде дървото T(X), с приложени двете операции за връх Y. Тъй като сложността за всяка от операциите за промяна на масив от числа (мравуняци) чрез интервалното дърво ще бъде O(lgM), то броят на върховете в интервалното дърво T(X), различни от върховете в интервалното дърво T(Y) ще бъде пропорционален на lgM. Също, благодарение на начина, по който дефинирахме рекурсивните функции, това множество ще бъде свързано (ако си представим, че връзката м/у върховете в интервалното дърво е двупосочна).

Това означава, че можем да построим новото дърво T(Y) като вместно да променяме T(X) за дадените върхове, създадем нови свързани помежду си върхове и върховете по "фронта" (листата и върховете с по едно дете) на това множество "закачим" за съответните им деца в T(X).

Същата стратегия, например, ще приложим и за град/връх Z, насленик на връх Y. Новите върхове в интервалното му дърво ще можем да "закачим" за върхове в интервалното дърво T(Y) и/или T(X).

Така, на всяка заявка ще добавяме по lgM нови върха, което ни оставя с решение, което използва O(NlgM) памет и има времева сложност O(NlgM).

Автори: Георги Георгиев и Йордан Чапъров