# АНАЛИЗ НА РЕШЕНИЕТО НА ЗАДАЧА DISCO

Първата стъпка е леко да преобразуваме задачата и отсега нататък да не търсим максималния брой върхове, до които може да стигнем, а минималния брой върхове, до които не може да стигнем след като отрежем най-много К ребра. Отговорът за оригиналната задача ще бъде просто броят на върховете в дървото минус броя на маркираните (върхове с готин пич, чието ребро към техния родител е отрязано) и минималния брой на върховете, до които не може да стигнем.

Следващата стъпка е да съпоставим на всеки връх цена, която ще съответства на броя на немаркираните върхове в неговото поддърво. Ако броят на немаркираните върхове в поддървото му е 0, от тук нататък може да игнорираме този връх. Сега лесно може да забележим, че всички листа са ни маркирани. Може да си улесним живота, ако "компресираме" дървото, като направим всички маркирани върхове да са ни листа, като просто игнорираме всички върхове под всеки маркиран връх.

След няколко доста прости преобразования искаме да решим следната задача: Имаме дърво с O(N) върха и всеки връх има цена. Искаме да изберем подмножество от най-много К върха, така че да няма път от корена до някое листо, който да не преминава през поне един връх от подмножеството. Искаме да изберем такова подмножество, така че сумата от цените на върховете в него да е минимална.

**Решение O(N*K^2):**

От тук повечето опитни състезатели биха видели, че тази задача лесно може да се реши с динамично O(N*K^2), където стейта е [връх от дървото][колко най-много върхове ще изберем в текущото поддърво]. Ако дървото ни беше двоично то тогава просто щяхме да имаме за всеки стейт, 2 случая или ще вземем текущия връх в подмножеството или ще разделим текущото К на лявото и дясното ни поддърво. Но за жалост в задачата дървото е произволно, така че може да направим малък трик, за да го направим двоично. Вижте края на анализа за детайли към този трик.

**Решение O(N*K):**

Решението ни O(N*K) е по-лесно и доста по-хитро (надявам се) от другото решение. Отново ще направим динамично със същия стейт, но вече транзициите ни ще са малко по-различни. Може да си представим дървото като нарисувано на лист – всеки връх си има точно определено място и лесно може да кажем кои поддървета са наляво от даден връх и кои надясно. Когато сме в даден връх i, знаем, че не сме покрили никой връх от корена до i. Също така ще приемем, че сме покрили по-някакъв начин всички върхове наляво от текущия връх. Сега остава да покрием поддървото на i и всичко върхове надясно. Единият ни вариант е да покрием i и тогава сме готови с поддървото му и с всички в ляво от него, така че от тук трябва да отидем до следващия най-близък непокрит син по пътя от i към корена. Когато отидем в този най-близък връх отново попадаме в ситуация, където сме покрили всички върхове наляво, но не сме покрили никой връх от текущия до корена и никой връх в дясното поддърво. Ако решим да не покриваме i, то тогава трябва да покрием всяко поддърво, което “излиза” от i. За да спазим нашата нотация с ляво и дясно, трябва да отидем в най-левия син на i и отново всички неща, които трябва да са изпълнени за нашия стейт са валидни! По този начин отново ще имаме O(N*K) стейта, но за всеки стейт ще имаме O(1) транзиции.

Как да си направим дървото двоично:

Ако имаме връх (A), който има 4 деца - (1), (2), (3) и (4), то тогава може да направим нов връх (X), който има цена 0. Сега връх (A) ще има три сина (1) - (2) и (X), а (X) ще има 2 сина - (3) и (4). Цените на всички върхове, както и децата на върхове (освен (A)) се запазват. Ако повторим тази стъпка още веднъж като направим още един нов връх (Y), който има деца (2) и (X), а (A) има синове (1) и (Y), то тогава ще имаме двоично дърво. Лесно може да се докаже, че когато върховете имат цена 0, те не променят по никакъв начин задачата. Аналогично алгоритъма може да се приложи и за повече от 4 деца. Също така е много важно да се забележи, че броят на върховете в поддървото в най-лошия случай ще са от порядъка на 2N.

Въпроси/забележки/комен тари/награди: rhristov@mit.edu

*Автор: Румен Христов*