Анализ на задача `transport`

Тагове: дърво, броене на възможности, динамично програмиране

Спирките и пътната мрежа в Олимпово образуват ориентирано дърво с корен 1. В повечето задачи дървета са неориентирани или има корен и насочеността става имплицитно от връх "надолу". Тук дървото е ориентирано главно за да може естествено пътищата, които разглеждаме от връх до наследник, да са единствените възможни. В анализа вместо спирки, ще използваме върхове. Важно е да обърнем внимание, че правим покритие на върховете на дървото, а не на ребрата (в общия случай има разлика в броя), което е видно и от примера. Конкретно за подходите разглеждани в решенията няма голяма значение.

Решение на първа подзадача – 7 точки

Подзадачата е предвидена за най-директното решение с пълно изчерпване. Разглеждаме всяко възможно подмножество на дадените линии. За дадено подмножество маркираме всички върхове от съответните линии и проверяваме дали се покриват всички върхове.

Сложност: $O(2^K KN)$.

Решение на втора подзадача – 13 точки

Можем малко да оптимизираме пълното изчерпване с по-бърза проверка. Нека предварително за всеки връх намерим множеството от линии, които го покриват. Ако запазим тази информация в битова маска, то проверката дали подмножество от линии покрива всички върхове може да стане с побитово "и" между всяка маска на връх и текущата маска на разглежданите линии. Интересен е въпросът дали можем да се справим и по-добре – да правим проверката с по-добра сложност от $O(N)$, но на пръв поглед няма добър начин.

Сложност: $O(2^K N)$.

Решение на трета подзадача – 11 точки (=0+0+11)

Това е първата подзадача от поредицата, където дървото е пръчка и то е поредицата $1 \to 2 \to \ldots \to N$, т.е. може да си мислим, че имаме масив, който искаме да покриваме с интервали. Ще опишем подход, който не може да се обобщи за дърво (разбира се пълното решение работи и тук, но без особено подобрение). Нека смятаме стандартното динамично $dp[i]$ – брой начини да покрием точно префикса на подмасива от 1 до i (без да излизаме надясно). Лесно се вижда, че трябва да използваме поне един интервал, който завършва в i. Удобният начин за всички решения на тази задача е като сме фиксирали даден интервал (линия), да разгледаме два варианта – той е необходим за покритието (т.е. без него не е пълно) или просто се добавя към вече валидно покритие. Оказва се сравнително неприятно извеждането на рекурентна зависимост. Нека сме фиксирали текущ интервал $[l, i]$ и искаме да преброим в колко покрития той е необходим. Тогава можем да считаме, че преди да го добавим, сме имали предишно покритие на префикса от 1 до p (където p е между $l-1$ и $i-1$) и като го добавим вече покриваме всичко до i. Трябва да отчетем и още възможности, а именно може да има допълнителни интервали след p, които не допринасят с нищо, но трябва да ги броим. За да не почнем да броим някои възможности по няколко пъти, то трябва тези интервали да започват от $p+2$ нататък, за да няма по-голяма покрита част от префикса до p преди добавянето на текущия интервал.

В тази подзадача е предвидено най-директната имплементация на подобно динамично. Нека обобщим смятането на $dp[i]$. За всяко i разглеждаме интервалите, които завършват в i, като ги фиксираме един по един. В началото отчитаме възможностите съответния интервал $[l, i]$ да не е съществен за покритието до i, като просто удвояваме досегашната стойност на $dp[i]$ (към всяка стара възможност можем да добавим текущия интервал, като така получаваме още толкова покрития). След това обхождаме всички възможни стойности на p от $l-1$ до $i-1$ за старото покритие преди новия интервал. За всяко p преброяваме (с обхождане) всички по-малки интервали, които можем да ползваме допълнително – тези с краища от $p+2$ до i (за тези с десни краища i трябва да ползваме само предишните срещнати в обхождането на интервалите). Ако те са cnt на брой, то възможности за фиксираното p са $dp[p] \times 2^{cnt}$, които трябва да добавим към $dp[i]$.

Сложност: $O(KNK)$.

Решение на четвърта подзадача – 23 точки (=0+0+11+12)

Правим директна оптимизация на предния подход, като с прекъмпют предварително преброяваме всички интервали с краища между две стойности. Така за смятането на $dp[i]$ и фиксирано p единствено ни трябва броя интервали от прекъмпюта с краища между $p+2$ и $i-1$. Няма нужда до i, защото можем да сортираме текущите интервали по нарастващ ляв край и така преди текущия интервал в обхождането има само по-широки или със същата дължина интервали.

Сложност: $O(KN)$ с предварителен *count sort* на левите краища на интервалите или $O(KN + K\log K)$.

Решение на пета подзадача – 36 точки (=0+0+11+12+13)

Можем да забележим, че бройките интервали, които използваме при смятането на фиксирано $dp[i]$, са само за десни краища до $i-1$. Затова преди да смятаме самото динамично, ще направим още един прекъмпют за съответните суми при всяка възможна граница за ляв край. Така оптимизираме изцяло вътрешния цикъл до константа. Възможно е допълнително да оптимизираме и второто предпроцесване и така да паднем под квадратична сложност, но това няма смисъл в тази задача. Важно е да се отбележи, че в тази подзадача трябва да правим първото предпроцесване със сложност $O(N^2)$, докато преди можеше и със сложност $O(NK)$.

Сложност: $O(N^2 + K)$ с предварителен *count sort* на левите краища на интервалите или $O(N^2 + K\log K)$.

Решение на шеста подзадача – 11 точки (=0+0+0+0+0+11)

TBD...

Сложност: $O(N+K)$.

Решение на седма подзадача – 54 точки (=0+7+6+11+0+11+19)

TBD...

Сложност: $O(N^3 + K)$ с предварителен *count sort* на началните краища на пътищата или $O(N^3 + K\log K)$.

Пълно решение – 100 точки

TBD...

Сложност: $O(N^2 + K)$ с предварителен *count sort* на началните краища на пътищата или $O(N^2 + K\log K)$.

Началната версия на задачата нямаше това изкуствено ограничение за пътищата да са само от връх до наследник. Но се оказва, че тя е значително по-трудна и няма намерено решение засега. Също е интересен въпросът дали има решение с по-добра сложност на задачата. Поради трудностите да направим по-добро от квадратично решение за пръчка, вероятно няма по-добро в общия случай на дърво.

Автор: Илиян Йорданов