Анализ на задача `connect`

Тагове: графи, минимално покриващо дърво, най-кратък път, динамично програмиране, битови маски

Можем накратко да опишем търсеното в задачата като намирането на *частично минимално покриващо* дърво, което да покрие K фиксирани върха на претеглен граф. Тази задача е добре изучавана и това, което търсим се нарича *минимално дърво на Щайнер*. Стандартно за минимално покриващо дърво, включващо всички върхове в графа, има добре известни алгоритми, но в случая за *частичното минимално покриващо дърво* задачата става **NP**-пълна и затова ограничението за K е толкова малко - няма намерени полиномиални алгоритми.

Решение на втора подзадача - 22 точки

Ограниченията са много малки и затова предвиденото решение е с пълно изчерпване. Най-лесното, което можем да направим е да разгледаме за всяко ребро дали участва или не в *частичното минимално покриващо дърво*, което са твърде много възможности за разглеждане - 2^M (достигащо 2^{40}). Но върховете са достатъчно малко, така че можем вместо за ребрата, да разгледаме кои върхове ще се включат в оптималното дърво. Това са 2^N възможности или най-много 2^{20}. За всяка възможности сме фиксирали само от кои върхове се състои *частичното минимално покриващо дърво* и за да намерим самото дърво, то всъщност е нормално *минимално покриващо дърво* за подграфа, който се състои само от фиксираните върхове и ребрата между тях. Така след като фиксираме върховете, които ще гледаме, трябва да пуснем някой от стандартните алгоритми за намиране на минимално покриващо дърво (алгоритъм на Прим, алгоритъм на Крускал) в получения подграф и да видим кои ребра е най-оптимално да участват в минималното покриващо дърво. Отново се наблюдава, че много участници не са се справили с тази подзадача, която в основата си използва най-базово пълно изчерпване, а така се губят доста ценни точки (макар и описаното да е една идея по-сложна подзадача с пълно изчерпване от стандартното).

Сложност: $O(2^N M \log_2 M)$. (ограничението по време е достатъчно голямо, така че описаният подход с тази сложност спокойно да мине подзадачата)

Решение на трета подзадача - 14 точки

Важното допълнително ограничение в подзадачата е $K \leq 3$. Ще опишем само как се справяме със случая $K = 3$, а по-лесният случай с $K = 2$ очевидно може да се реши с намиране на най-краткия път между двата върха, но и решението, което ще опишем за $K = 3$ ще работи и за $K = 2$.

Нека разгледаме по-внимателно какво точно представлява *частичното минимално покриващо дърво* при $K = 3$. Листата на това дърво трябва да са главни върхове, защото ако това не е така, то можем да започнем да премахваме листа, които не са главни върхове, докато всички листа са само от главните върхове, при което ще получим по-добро покриващо дърво и няма да загубим свързаността между главните върхове. Нека си представим, че тръгваме от главен връх, който е листо, и се движим по ребра, докато не се появи разклонение (ако има такова, но да допуснем засега, че има). Очевидно това разклонение няма как да има три или повече части, защото във всяка част трябва да стигнем до главен връх, а има останали 2 главни върха, като нямаме и цикли, защото сме в дърво. Това означава, че имаме точно 2 разклонения, които са пътища към другите 2 главни върха. Всъщност този връх, до който стигнахме и има разклонения към другите главни върхове, е много важен. Можем да си го мислим като централен връх, който има непресичащи се пътища до 3-те главни върха.

Получихме, че имаме централен връх, който има непресичащи се пътища до главните вър-

хове. Лесно се вижда, че тези пътища трябва да са най-кратките възможни, защото ако има по-кратки, то ще получим по-добро *частично минимално покриващо дърво*. Остава да видим и случая, в който не намерим такъв централен връх, защото като тръгнем от листо на дървото нямаме разклонения (това означава, че дървото е пръчка). Това означава, че някъде "по средата" ще минем през втория главен връх и накрая на пътя ще стигнем третия главен връх. Тогава просто можем да мислим този междинен главен връх за централния връх на дървото, т.е. да позволим централният връх да е и някой от главните върхове.

Сега вече имаме хубав начин да намерим *частичното минимално покриващо дърво* при $K = 3$. Фиксираме всеки връх за централен и сумираме най-краткият път от централния връх до главните върхове. Накрая търсеното *частично минимално покриващо дърво* ще е за този централен връх, за който сме получили най-малка сума. За да можем да реализираме това бързо трябва да използваме стандартната модификация на алгоритъма на Дийкстра, при която намираме най-краткият път от всеки връх до някой друг (в случая някой от главните), което става като просто гледаме задачата наобратно - тръгваме от някой от главните върхове и намираме най-краткият път от него до всички останали. Така в крайна сметка цялото решение за тази подзадача е следното. За всеки от главните върхове намираме най-краткият път от него до всички останали. След това с тази информация можем да намерим цената на *частичното минимално покриващо дърво*, ако разгледаме всеки връх за централен на това дърво. Може да се забележи, че това решение ще работи и при $K = 2$.

Сложност: $O(KM \log_2 M)$.

Решение на четвърта подзадача - 29 точки

Тази подзадача е за малко точки, защото решението е много подобно на описаното в предната подзадача. Отново ще разгледаме само най-сложния случай - $K = 4$. Единствено трябва да съобразим по какъв по-добър начин да гледаме *частичното минимално покриващо дърво* за $K = 4$. Както преди, възможно е да имаме един централен връх, от който да имаме непресичащи се пътища до главните върхове. Но това не е единствената възможност. Ако използваме аналогични съображения, то може да видим, че е възможно да имаме два централни върха v_1 и v_2, които са свързани с път. По-точно, от връх v_1 ще имаме непресичащи се пътища до два от главните върхове, а от връх v_2 ще имаме непресичащи се пътища до останалите два главни върха. Сега можем да се възползваме от по-ниските ограничения за N и M. Те ни позволяват да намерим най-краткия път между всеки два върха, като пуснем алгоритъма на Дийкстра от всеки връх. Като имаме тази информация можем да намерим за всеки два централни върха най-доброто *частично минимално покриващо дърво*, в което те са централни върхове, просто като разгледаме всички възможности за това кои два главни върха ще са директно достижими от единия централен връх (останалите два главни върха ще са директно достижимия от другия централен връх) и сумираме съответните най-кратки пътища. Ако позволим централните върхове да съвпадат, то така ще разгледаме и по-лесния случай, в който имахме един централен връх. Също с още малко модификации бихме могли да обобщим това решение, така че да решава и случаите $K = 3$ и $K = 2$, за да не ги гледаме отделно.

Сложност: $O(NM \log_2 M)$.

Решение на пета подзадача - 23 точки

Подзадачата е предвидена по-скоро за по-бавни или нечисти имплементации на пълното решение. Все пак тук ще опишем и една алтернативна идея, която е подходяща за тук, където $K = 4$. Предното решение няма потенциал за развитие, затова трябва да измислим нещо по-различно. Проблемът идва от това, че имаме два централни върха и затова ще се опитаме да имаме само един. Този връх ще наричаме важен връх и ще съставим *частичното минимално покриващо дърво* около него. За тази цел може да използваме всеки връх на пътя между двата

централни върха, така че нека разгледаме единият от тях за важния. Можем да разгледаме всяка възможност за това кои два главни върха са директно достижими от важния връх. Трудната част е как да намерим най-добрата останала част, която свързва този връх с другите два главни върха през централен връх. Ако се замислим това всъщност е нашата задача за $K = 3$, където трите върха са другите два главни върха и важният връх, но не можем да приложим стария подход, защото важният връх може да е всеки от N-те върха, т.е. нямаме една фиксирана тройка върхове, а множество тройки.

Оказва се, че има по-добър подход за $K = 3$. Нека тръгнем от два главни върха и се опитаме да намерим за всеки връх, *частичното минимално покриващо дърво* на този връх и двата главни върха. Това можем да направим, като първо намерим най-краткия път от двата главни върха до всички върхове и след това приемем, че всеки връх е централен, като това означава че просто считаме, че *частичното минимално покриващо дърво* е с тегло равно на сумата от най-кратките пътища от съответния връх до двата главни. Понеже това е само случая, в който покриващото дърво е пръчка, то трябва да разгледаме и възможността не всеки връх да е централен, а до съответния връх да се стига от друг централен. Както вече бяхме казали, пътищата от централния връх до главните са най-кратки пътища. Затова можем да пуснем алгоритъм на Дийкстра, при който в началото сме инициализирали разстоянията на всички върхове да са сумите от най-кратките пътища до главните върхове (началното приемане, че са централни) и се пробваме да релаксираме тези стойности, като пробваме да удължаваме с най-кратки пътища централните върхове, така че да стигнат други върхове. Реализацията на този подход може да стане и само с едно пускане на алгоритъма на Дийкстра, като едновременно намираме най-кратките пътища от главните върхове до останалите и също обединяваме двата най-кратки пътя от главните върхове до другите (приемайки, че те ще са централни) и се пробваме да продължим тези обединения с най-кратки пътища до други върхове.

Когато имаме горната информация, вече е много просто да намерим *частичното минимално покриващо дърво* на 4 върха. Фиксираме всеки връх като важен и разглеждаме всяко възможно разделяне на 4-те главни върха на две двойки и сумираме съответните две части на дървото (едната част е важният връх, свързан с едната двойка главни върхове, а другата част е важният връх, свързан през централен връх с другата двойка върхове). Може да се забележи, че дори не е нужно да считаме, че важният връх е някой от централните, както приехме в началото.

Сложност: $O(K^2 M \log_2 M + NK^2)$.

Решение на шеста подзадача - 100 точки

Предната идея е много добра и тя може да се продължи и до пълно решение дори и когато $K = 5$. Може да се забележи, че тогава в най-лошия случай имаме три централни върха, като имаме път между два от тях, а третият е някъде "по средата". Двата крайни централни върха са директно свързани по най-кратък път с две двойки главни върхове, а средният централен връх е свързан директно с петият главен връх. Това означава, че ако разгледаме средният централен връх за важен, то трябва да направим същото като при $K = 4$ и единствено да добавим най-краткия път от него до петия главен връх.

Последната описана идея не е много чиста и няма как да работи с $K > 5$ без да се добавят нови усложнения и затова ще представим предвиденото авторово решение, като в последствие може да се забележи, че това което правихме досега всъщност е част от авторовото решение. В предната подзадача забелязахме, че ако разгледаме *частичното минимално покриващо дърво* относно важния връх, то се разпада в общия случай на по-малки *частични минимални покриващи дървета*, което ни навежда на мисълта, че можем да пробваме подхода динамично програмиране. Нека в $dp[v][mask]$ намираме големината на *частичното минимално покриващо дърво* на главните върхове, зададени с битовата маска $mask$ и върха v. Искаме да си мислим

за връх v точно като важния връх, относно който можем да разделим *частичното минимално покриващо дърво* на по-малки *частични минимални покриващи дървета*. Можем за по-просто да разглеждаме всеки път, че дървото се разделя на две части, като по този начин получаваме началната рекурсивна зависимост: $dp[v][mask] = \min_{m \subset mask} dp[v][m] + dp[v][mask \wedge m]$, като с $m \subset mask$ означаваме, че m е подмаска на маската $mask$, т.е. с m сме избрали само част от фиксираните върхове на маската $mask$ (m трябва да е различно от 0), а $mask \wedge m$ е подмаската с останалите върхове на маската.

Тази рекурентна зависимост намира вярно търсената стойност само ако върхът v наистина може да играе ролята на важен връх (в някакъв смисъл е между главните върхове). Ако той не играе ролята на важен връх, то трябва да имаме друг връх, който ще е важен за *частичното минимално покриващо дърво* на v и върховете от маската. Това всъщност означава, че v ще се явява листо на това дърво (иначе щеше да може да направим разделение на дървото около v и той да влезе в ролята на важен). Понеже v е листо, то този връх ще е свързан по най-кратък път до някой важен връх u. Така получаваме пълната рекурентна зависимост: $dp[v][mask] = min(\min_{m \subset mask} dp[v][m] + dp[v][mask \wedge m], \min_u dp[u][mask] + dist(u, v))$, където $dist(u, v)$ е най-краткият път между върховете u и v. Единственият проблем на тази рекурентна зависимост е, че вече имаме цикличност заради втората част. Това е класически случай, в който можем да сметнем динамичното с алчния подход, който използваме при алгоритъма на Дийкстра - първо смятаме най-малките стейтове, защото те няма как да бъдат подобрени, след това смятаме следващите най-малки и т.н. Ако се замислим това е точно, каквото правихме при по-добрата идея за $K = 4$ и по-точно за намирането на *частични минимални покриващи дървета* на всички тройки върхове, при които два от върховете са фиксирани главни върхове (там $mask$ се състои от двата фиксирани главни върха, а v е в ролята на третия връх).

Базовите стойности на динамичното са $dp[v_i][2^i] = 0$ за всеки главен връх v_i (маската я съставяме спрямо индексите на главните върхове). За смятането на всички стойности най-логично е да се движим първо по маските с 2 включени бита, после с 3 включени и т.н., но всъщност ако просто разглеждаме маските в нарастващ ред на стойността, то винаги ще сме разгледали подмаските на дадена маска, защото те имат по-малка стойност. Може да разгледате в имплементацията един побитов трик, с който директно обхождаме подмаските на дадена маска. Когато сме фиксирали дадена маска $mask$, то първо трябва да попълним $dp[v][mask]$ с началните стойности, които са при считането на тези върхове за важни и всевъзможните разцепвания на дървото на две части около v. След като сме намерили тези стойности, то пускаме "алгоритъма на Дийкстра", за да довършим смятането на $dp[v][mask]$, като в началото считаме, че имаме "път" до всеки връх на стойност $dp[v][mask]$ и след това пробваме да релаксираме стойностите $dp[v][mask]$ като ползваме най-кратките пътища до други $dp[u][mask]$.

Сложност: $O(3^K N + 2^K (N + M) \log_2 M)$. (може да се докаже, че броят итерации за обхождането на подмаските на всички маски с K бита е 3^K)

Автор: Илиян Йорданов (zaeta)