Code Festival 2017 Elimination Tournament Round 2

2017 年 11 月 26 日

For international readers: English editorial starts on page 4.

A: Colorful MST

色 i が塗られた頂点どうしは縮約して 1 つの頂点であるとみなしても問題ありません．これは G においてこれらの頂点たちを含む連結成分が非連結であったとしても G' においてはすでに連結であるため，これらを連結にするような辺を追加する必要がないことから分かります．このようにすると，彩色済みの頂点は K 個以下であり，塗られている色は相異なるようなグラフのみ与えられると考えていいことが分かります．

まずは答えの下限を考えてみましょう．クラスカル法の要領で辺を $K-1$ 本追加して最小全域森を作ったときが答えの下限であることは明らかです．そこで，どのように彩色を行ったとしてもこのような下限を達成不可能な場合があるかどうかを考えてみることにします．結論から言えば，この森に対して適切に彩色したときに G' が連結にすることが必ず可能です．以下では，これを示します．以下のような問題を考えます．

n 個の木 $T_1, \ldots, T_n$ からなる森 F が与えられる．T_i は未彩色の頂点が x_i 個，彩色済みの頂点が y_i 個からなる．F は以下のような制約を満たす．

- $x_i, y_i \geq 0$
- $x_i + y_i > 0$
- $\sum y_i \leq K$
- $\sum x_i + y_i = K + n - 1$

以下のような操作を何度でも自由な順番で行うことが可能である．最終的に $(0, K)$ からなる木 1 つのみを残すことは可能か？

1. $T_i = (x_i, y_i)$ について $x_i > 0$ であるとき，$T_i = (x_i - 1, y_i + 1)$ に変化させる．
2. 2 つの異なる木 T_i, T_j について，$x_i + x_j > 0$ ならば T_i, T_j を G から取り除き，$(x_i + x_j - 1, y_i + y_j)$ を G に追加する．

F に対して後者の操作を行ったあとも，上記の制約を満たしています．

$x_i = 0$ なる木しかない場合，すでに $(0, K)$ であるはずなので考慮する必要はありません．まず，$x_i > 0$ なる木が 1 つしかない場合を考えます．$\sum y_i = m$ とします．このとき，$\sum x_i = K + n - 1 - m$ となります．$m \leq K$ より，$K + n - 1 - m \geq n - 1$ なので，$n - 1$ 個の色が塗られた頂点のみからなる木を操作 2 により $x_i > 0$ なる木に全てまとめることが可能です．このようにして得られた木 $T = (x, y)$ は，$x + y = K$ を満たしているので，x が 0 になるまで操作 1 を行うことで，$(0, K)$ が得られます．$x_i > 0$ なる木が複数ある場合も，はじめにそれらを全て 1 つにまとめてしまえばよいです．

以上より，同じ色の頂点どうしを予め縮約し，クラスカル法の要領で $K - 1$ 本の辺を連結して得られた最小全域森 F に適切に彩色を行うことで，G' を連結にすることが可能なことが示されました．縮約は実装上は長さ 0 の辺で連結すると考えて構いません．以上より，この問題はクラスカル法とほぼ同様のアルゴリズムにより $O((N + M) \log M)$ で解くことができます．

B: Many Swaps Sorting

N の値に応じて各操作がどのような置換で表されるかは変わってしまいます．そこで，どのような N に対しても性質が変化しないような操作があるかどうかを考えてみることにします．操作を観察すると，1 番の操作と $N-1$ 番の 2 種類の操作が以下のような性質を持っていることが分かります．

- 1 番の操作：先頭にある数が末尾へと移動する
- $N-1$ 番の操作：先頭にある数と末尾にある数が入れ替わる

突然ですが長さ N の数列をバブルソートするときの隣接スワップの回数と，スワップされる 2 つの要素を指し示すポインタの移動回数を考えてみます．隣接スワップを行う回数は最大でも $N(N-1)/2$ 回で抑えられます．ポインタの移動回数は最大でも N^2 回で抑えられます．$N=200$ のとき，これらの合計は 59900 です．

ここで，1 番の操作をポインタの移動だと考えてみると，2 番の操作が隣接スワップと同様の操作だと考えられます．このことから適切に操作を行うとバブルソートを行うことが可能であり，10^5 回以下の操作で必ずソートすることが可能であることが分かります．各操作は $O(N)$ で実行可能であり，操作回数は $O(N^2)$ で抑えられるので操作を愚直に実行しても計算量は $O(N^3)$ と十分高速です．

Code Festival 2017 Elimination Tournament Round 2

November 26, 2017

A: Colorful MST

We can safely contract the vertices painted in the same color i into a single vertex. This is because, even if the connected components that contain these vertices are disconnected, they will be already connected in G', so we do not need to add edges to connect them. Now, we can assume that there is at most K vertices that are already painted, and those colors are distinct.

First, let us consider the lower limit of the answer. It is obvious that the minimum spanning forest obtained by adding $K-1$ edges in a similar way to Kruskal's algorithm, is the lower limit. Is there a case where this lower limit cannot be achieved regardless of how we paint the remaining vertices? In conclusion, it is always possible to paint the vertices so that G' is connected. Let us show this by considering the following problem:

> You are given a forest F consisting of n trees $T_1, \ldots, T_n$. T_i consists of x_i unpainted vertices and y_i painted vertices. F satisfies the following constraints:
>
> - $x_i, y_i \geq 0$
> - $x_i + y_i > 0$
> - $\sum y_i \leq K$
> - $\sum x_i + y_i = K + n - 1$
>
> You can perform the following operation any number of times in any order. Is it possible to leave only one tree consisting of $(0, K)$?
>
> 1. When $x_i > 0$ for $T_i = (x_i, y_i)$, change it to $T_i = (x_i - 1, y_i + 1)$.
> 2. For two different trees T_i and T_j, if $x_i + x_j > 0$, remove T_i and T_j from F and add $(x_i + x_j - 1, y_i + y_j)$ to F.

Performing the latter operation on F does not break the constraints above.

We do not need to consider the case where there is only trees such that $x_i = 0$, since in such a case it should be already $(0, K)$. First, consider the case where there is only one tree such that $x_i > 0$. Let $\sum y_i = m$. Then, $\sum x_i = K + n - 1 - m$. Since $m \leq K$, $K + n - 1 - m \geq n - 1$ holds, and the trees consisting of $n-1$ painted vertices can be merged to the tree such that $x_i > 0$ with operation 2. The

tree $T = (x, y)$ obtained in this way satisfies $x + y = K$, so we can obtain $(0, K)$ by repeating operation 1 until x becomes 0. In the case where there are multiple trees such that $x_i > 0$, we can first merge them into one and do the same.

Thus, it is shown that we can make G' connected by contracting the vertices in the same color beforehand and properly painting the minimum spanning forest F obtained by adding $K - 1$ edges in a way similar to Kruskal's algorithm. Contraction can be implemented as adding an edge of length 0. Therefore, the problem can be solved by an algorithm which is almost the same as Kruskal's algorithm, in $\mathrm{O}((N + M) \log M)$ time.

B: Many Swaps Sorting

Depending on the value of N, the operations correspond to different permutation, so let us find operations whose property remains the same for different N. Observing the operations, we can see that Operation 1 and $N-1$ have the following properties:

- Operation 1: the element at the beginning goes to the end
- Operation $N-1$: the elements at the beginning and the end are swapped

By the way, let us deal with some subject in bubble sort. When a sequence of length N is sorted by bubble sort, consider the number of swaps and the number of moves of the pointer that points to the two elements being swapped. There are at most $N(N-1)/2$ swaps and at most N^2 moves of the pointer. When $N = 200$, their sum is 59900.

Now, see Operation 1 as moving the pointer. Then, Operation $N-1$ corresponds to swapping elements. Thus, we can perform bubble sort by a proper sequence of operations, and it is always possible to sort the given sequence in 10^5 or less operations. Since each operation can be done in $\mathrm{O}(N)$ time and the number of operation is $\mathrm{O}(N^2)$, the time complexity will be $O(N^3)$ even in naive implementation, which is fast enough.