27. In trying to understand the combinatorial structure of spanning trees, we can consider the space of *all* possible spanning trees of a given graph and study the properties of this space. This is a strategy that has been applied to many similar problems as well.

Here is one way to do this. Let G be a connected graph, and T and T' two different spanning trees of G. We say that T and T' are *neighbors* if T contains exactly one edge that is not in T', and T' contains exactly one edge that is not in T.

Now, from any graph G, we can build a (large) graph $\mathcal{H}$ as follows. The nodes of $\mathcal{H}$ are the spanning trees of G, and there is an edge between two nodes of $\mathcal{H}$ if the corresponding spanning trees are neighbors.

Is it true that, for any connected graph G, the resulting graph $\mathcal{H}$ is connected? Give a proof that $\mathcal{H}$ is always connected, or provide an example (with explanation) of a connected graph G for which $\mathcal{H}$ is not connected.

Yes, $\mathcal{H}$ will always be connected. To show this, we prove the following fact.

(1) *Let $T = (V, F)$ and $T' = (V, F')$ be two spanning trees of G so that $|F - F'| = |F' - F| = k$. Then there is a path in $\mathcal{H}$ from T to T' of length k.*

Proof. We prove this by induction on k, the case $k = 1$ constituting the definition of edges in $\mathcal{H}$. Now, if $|F - F'| = k > 1$, we choose an edge $f' \in F' - F$. The tree $T \cup \{f'\}$ contains a cycle C, and this cycle must contain an edge $f \notin F'$. The tree $T \cup \{f'\} - \{f\} = T'' = (V, F'')$ has the property that $|F'' - F'| = |F' - F''| = k - 1$. Thus, by induction, there is a path of length $k - 1$ from T'' to T'; since T and T'' are neighbors, it follows that there is a path of length k from T to T'. ∎

28. Suppose you're a consultant for the networking company CluNet, and they have the following problem. The network that they're currently working on is modeled by a connected graph $G = (V, E)$ with n nodes. Each edge e is a fiber-optic cable that is owned by one of two companies—creatively named X and Y—and leased to CluNet.

Their plan is to choose a spanning tree T of G and upgrade the links corresponding to the edges of T. Their business relations people have already concluded an agreement with companies X and Y stipulating a number k so that in the tree T that is chosen, k of the edges will be owned by X and $n - k - 1$ of the edges will be owned by Y.

CluNet management now faces the following problem. It is not at all clear to them whether there even *exists* a spanning tree T meeting these conditions, or how to find one if it exists. So this is the problem they put to you: Give a polynomial-time algorithm that takes G, with each edge labeled X or Y, and either (i) returns a spanning tree with exactly k edges labeled X, or (ii) reports correctly that no such tree exists.

We begin by noticing two facts related to the graph $\mathcal{H}$ defined in the previous problem. First, if T and T' are neighbors in $\mathcal{H}$, then the number of X-edges in T can differ from the number of X-edges in T' by at most one. Second, the solution given above in fact provides a polynomial-time algorithm to construct a T-T' path in H.

We call a tree *feasible* if it has exactly k X-edges. Our algorithm to search for a feasible tree is as follows. Using a minimum-spanning tree algorithm, we compute a spanning tree T with the minimum possible number a of X-edges. We then compute a spanning tree T with the maximum possible number b of X-edges. If $k < a$ or $k > b$, then there clearly there is no feasible tree. If $k = a$ or $k = b$, then one of T or T' is a feasible tree. Now, if $a < k < b$, we construct a sequence of trees corresponding to a T-T' path in $\mathcal{H}$. Since the number of X-edges changes by at most one on each step of this path, and overall it increases from a to b, one of the trees on this path is a feasible tree, and we return it as our solution.