Graph Theory

Solutions 3

Problem 1(a): For each $j \in [n]$, the star with center j corresponds to the Prüfer code $(j, j, \dots, j)$. Indeed, when creating the Prüfer code we remove all but one of the leaves, and always write down the label j. (if $i \neq n$ then we are left with the vertices i, n, and if $i = n$ then we are left with $n-1, n$). Another way of seeing this is by using the fact that a vertex i of degree d_i in the tree appears exactly $d_i - 1$ times in the Prüfer code (prove this!).

Problem 1(b): Consider a Prüfer code in which a, b are the only labels, $a \neq b$. Recall that a vertex appears in the Prüfer code if and only if it is **not** a leaf. Therefore, all vertices but a, b are leaves, and a, b are not leaves. Since the tree is connected, a, b must be adjacent. So we see that the tree consists of the edge ab and of $n-2$ leaves, each of which is adjacent to a or to b. Since a, b are not leaves, there must be at least one leaf adjacent to a and at least one leaf adjacent to $b4$. Such trees are sometimes called "double stars".

Problem 2: If $r = 1$ then T is a tree and there is exactly $1 = n^{1-2}|T|$ spanning tree containing T. Let us do induction on r and assume we know the statement for $r-1$.

So T has r components, the question is how many ways there are to add the remaining $r-1$ edges to make it a spanning tree. Well, let us count how many such extensions use some particular edge e between T_i and T_j. If we add e to T, then we get a forest with $r-1$ components whose sizes are $|T_1|, \dots, |T_{i-1}|, |T_{i+1}|, \dots, |T_{j-1}|, |T_{j+1}|, \dots |T_r|$ and $|T_i| + |T_j|$. Then induction tells us that the number of trees extending this is exactly $n^{r-3} \frac{|T_i|+|T_j|}{|T_i||T_j|} \prod_{k=1}^{r} |T_k|$. There are $|T_i||T_j|$ choices for e (this is the number of missing edges between T_i and T_j), so the number of trees extending T containing an edge between T_i and T_j is exactly $n^{r-3}(|T_i| + |T_j|) \prod_{k=1}^{r} |T_k|$.

If we sum these up for each pair of components (T_i, T_j), we get

$$\sum_{1 \leq i < j \leq r} (|T_i| + |T_j|) \cdot n^{r-3} \prod_{k=1}^{r} |T_k| = (r-1) \sum_{i=1}^{r} |T_i| \cdot n^{r-3} \prod_{k=1}^{r} |T_k| = (r-1) n^{r-2} \prod_{k=1}^{r} |T_k|$$

trees, because in the sum $\sum_{1 \leq i < j \leq r} (x_i + x_j)$, each term x_i is counted $r-1$ times.

But in the above sum we counted every tree exactly $r-1$ times. Indeed, as we mentioned, a spanning tree extends T by $r-1$ edges, and e can be any one of those. In fact, one particular tree was counted once for each choice of the edge e. So to get the actual number of extensions, we need to divide the above formula by $r-1$. We obtain $n^{r-2}\prod_{k=1}^{r}|T_k|$, exactly what we wanted to show.

Problem 3(a): The Prüfer code is $(1,6,1,4,4,6,1,1)$. Explanation: At the beginning, the leaf with minimal value is 2, its neighbour is 1. The next leaf to be removed is 3, its neighbour is 6. Next, 5 is removed, its neighbour is 1. Following this, $7,8$ are removed, their neighbour is 4. Now 4 is a leaf with neighbour 6. Finally, 6 and then 9 are the minimal-value leaves, and in each case the unique neighbour is 1.

Let us find the map f associated to the tree in Joyal's proof. The unique path between the left end 4 and the right end 5 is $4,6,1,5$. So these are the vertices of the directed cycles in the digraph corresponding to f. In two-line notation, the permutation of $4,6,1,5$ should be $\begin{pmatrix} 1 & 4 & 5 & 6 \\ 4 & 6 & 1 & 5 \end{pmatrix}$. In other words, $f(1)=4$, $f(4)=6$, $f(5)=1$, $f(6)=5$. To finish, we take care of the trees "hanging" on the vertices $4,6,1,5$, setting $f(2)=1$, $f(9)=1$, $f(10)=1$, $f(3)=6$, $f(7)=4$, $f(8)=4$.

Problem 3(b): Denote the tree by T. Since the code has length 6, T has 8 vertices. First, we identify the leaves of T. These are precisely the vertices which do not appear in the Prüfer code (recall that a vertex of degree d appears exactly $d-1$ times). So the leaves are $2,3,4,6,8$. The leaf with the smallest label is 2, so it was removed first, and its unique neighbour is 5. After this operation, the leaves are $3,4,6,8$ (5 did not become a leaf at this point since it appears later in the code). So the next leaf to be removed is 3 and its unique neighbour is 1. After this, the leaves are $4,6,8$ (neither 1 nor 5 are leaves at this point). Now 4 is removed, and its unique neighbour is 1. Since 1 does not appear later in the code, it's now a leaf; namely, the leaves are $1,6,8$. So the next vertex to be removed is 1, its unique neighbour is 7. Now the leaves are $6,8$. So 6 is removed, its unique neighbour is 7. Finally, 7 is removed, its unique neighbour is 5. At the end we are left with the vertices $5,8$, which are adjacent.