The Ties That Guide Us (incursion)

by Lukas Michel

Throughout, we will refer to the problem in graph theoretic terms: the floor plan describes a tree with N nodes, such that any node has degree ≤ 3.

Subtask 1. No degree 3 node

In this subtask, the tree is actually a line. Assume first that the number N of nodes is odd. While you and your assistant might receive this line in different numberings, there is one special node that can be identified without reference to any numbering, namely the node in the middle. This suggests a two step procedure to get to the secret node: first, we walk to the midpoint, and then we have our assistant (who would not know our starting point) guide us from there—for this, he can simply mark the path from the midpoint to the secret node (by placing exactly one tie in the respective rooms and no ties anywhere else).

Note that if our starting position lies on the opposite side of the midpoint than the secret node, this will need at most two steps over the shortest path between them (namely, when we 'overshoot' and have to walk back to the secret node). However, when we start on the same side as the secret node, actually walking to the midpoint might result in a huge detour. This issue can be solved as follows:

- If our starting node is marked, we simply walk *away* from the midpoint (again, overshooting by at most 1, resulting in a maximum detour of 2).
- If not, we can simply stop at the first marked node we encounter (resulting in no detour at all).

The case where N is even is only slightly more complicated: instead of a unique midpoint, we now have two (neighbouring) nodes in the middle. In the first step, our assistant will simply mark all nodes on the path from the secret node to the node in the middle that is *closer*. Almost the same strategy as above will then work for us to find the secret node where we now try to walk to the marked point in the middle:

- If our current node is marked we walk towards the end that is closer to us; the last marked node we encounter is the secret node.
- If not, we walk to the midpoint that is *further* from our starting location. If we encounter a marked node on the way, the first such node has to be the secret node. Otherwise we follow the markings left by our assistant; the last marked node will be the secret one.

Subtask 2. Precisely one degree 2 node

In this subtask, the tree no longer needs to be a line (and in fact, it will almost never be). However, there is still one special node that both we and our assistant can identify: the unique node with degree 2. Using this observation we can now use a very similar strategy to the line case:

- Our assistant will mark the path from the unique degree 2 node to the secret node.
- We first walk from our current node to the unique degree 2 node until we visit a marked node for the first time.
- At that point, we can follow the markings (moving away from the unique degree 2 node) to the secret node.

However, the third step is actually more subtle this time: as we are no longer just dealing with a line, there might be several possibilites for the next node, and we have no way to tell which one of these is marked without actually walking there. However, whenever we walk to the wrong node, this incurs a cost of 2, so we can't allow ourselves to have this happen too often.

To solve this, root the tree at the unique degree 2 node. We will then always walk into the *larger* of the two subtrees first: if this is correct, everything is fine, and if not, then the marked path has to continue to the *smaller subtree*. Thus, whenever we make a mistake, this halves the size of the remaining subtree, which can only happen $O(\log N)$ times. Let us analyze this more closely to see that it really fits into the task constraints, for which it will be useful to argue in a bottom-up fashion instead:

- Let k denote the number of children of our secret node (note that k is at most 2 except for the easy case that the secret node agrees with our chosen root). Then we might make a detour of up to $2k$ at this node because of overshooting.
- Consider now the marked nodes $v_1, \dots, v_\ell$ apart from the secret node where we we went to the wrong subtree first (numbered from bottom to top), and let s_i denote the corresponding subtree sizes; we moreover write s_0 for the size of the subtree rooted at the secret node. As we always go to the larger subtree first, we have $s_{i+1} \geq 2s_i + 1$. Combining this with $s_0 \geq k + 1$, we see inductively that $s_\ell \geq 2^\ell(k+2) - 1$.
- Our total detour is $2(k + \ell)$. Trying $k = 0, 1, 2$, we see that for a total detour of 32 we would need to have $N \geq s_\ell \geq 2^{16} - 1 = 65\,535$ (achieved for $k = 2, \ell = 14$), which is larger than the maximum number of rooms.

Subtask 3. No further constraints

In the final subtask, we are given a completely general tree. In order to adapt the solution to the previous subtask to this case, we somehow need to find a special node again that can be identified without referring to the numbering of our nodes and edges (or, to use fancy graph theoretic terminology, a node fixed by any automorphism of our tree). While already the line case shows that this need not be possible, we can again solve this by also allowing a set of two neighbouring nodes instead. Possible such choices are then the *centers* of the tree (nodes minimizing the maximum distance to any other node) or the *centroids* (nodes minimizing the maximum component size when we remove them). We can now again simply have our assistant mark the path from the secret node to the special node (or the closer one of the two special nodes, if there is more than one) and then use the same strategy as before to locate the secret node.

In our analysis, we need to consider one extra case: the root might have degree 3. In this case, if the subtree containing the marked path is the smallest, we might actually make a detour of 4 right at the root. However, in order to make a detour of 28 in the subtree of the marked path, this would then need to contain at least $2^{14} - 1 = 16\,383$ nodes (recycling the analysis from the previous subtask), resulting in $N \geq 49\,150$.

Note in particular that the bounds in this task are quite tight: already with the slightly weaker bound $N \leq 50\,000$ our strategy would not be able to guarantee a detour of 30. In fact, one can force *any* strategy to make a detour of at least 32 on the following tree with $N = 49\,150$ nodes: take three balanced binary trees of height 13 and add a new node connected precisely to their roots. This is actually the only instance with less than 65 535 nodes on which our strategy will lead to a detour larger than 30.

Solutions with more ties.

If we are allowed to mark the nodes with integers up to $N - 1$ (i.e. have our assistant leave behind up to $N - 1$ ties per room), a natural strategy is to simply mark each node with the distance to the special node. We will then immediately recognize the special node upon entering, and before that we can always simply walk to the unique neighbour whose marking is smaller than the current one. This solves the line case directly, while for the general case we again need the idea to check the larger subtree first.

For a solution with markings bounded by 2, we observe that the distance can only increase or decrease by 1 when we travel to a neighbouring node. Thus, to recognize whether the distance decreased, it is enough to remember the distance modulo 3. The secret node can be identified from this data as the unique node such that we cannot decrease the distance by walking to a neighbouring node.