Power Plant (Solution)

We shall interpret this task in terms of graph theory. The structure of the power plant becomes a tree, and the bases of the power plant become the vertices of the tree.

Subtask 1

Since the tree has a small number of vertices, we can try all possibilities of the status (ON/OFF) of the power generators' switches. Choose a vertex of a power generator whose status is ON, and consider a rooted tree whose root is the chosen vertex. By the depth-first search (DFS), we can calculate the profit in $O(N)$ time. Therefore, the total time complexity of this algorithm is $O(N2^N)$.

Since $N \leq 16$, we may calculate the profit using a slower algorithm which takes $O(N^2)$ or $O(N^3)$ time.

Subtask 1 (Another solution)

We consider the minimum subtree which contains all the power generators which are finally activated or broken. In such a subtree, all the leaves are activated power generators, and all power generators which are not leaves are broken. Therefore, considering all subtrees for which all leaves are power generators, we can solve this task by calculating the maximum of

(The number of leaves) − (The number of power generators which are not leaves).

Subtask 2

We fix a power generator whose status will be ON, and consider the problem of maximizing the profit when we turn the switch of the fixed power generator to ON. We consider the rooted tree whose root is the vertex of the fixed power generator. Then, a power generator other than the root is broken if and only if its descendants contain a power generator whose switch is ON. We may think that this task can be solved by dynamic programming.

For each vertex v, consider the subtree consisting of v and its descendants, and let dp_v be the maximum of

(The number of activated power generators) − (The number of broken power generators).

in the subtree when we choose the status (ON/OFF) of every power generator in the subtree. We calculate the value of dp_v by updating the values from the leaves successively.

Let us consider how to update the value of dp_v. For each vertex v, let E_v be the number of power generators at v ($E_v = 0, 1$). As explained above, when we determine the status of v, we consider the case where the descendants contain a power generator whose status is ON and the case where the descendants do not contain such a power generator, separately.

- (The case where the descendants of v do not contain a power generator whose status is ON)
  In this case, the status of every descendant of v is OFF. If there is a power generator at v, we should turn its switch to ON. Therefore, the maximum value of dp_v is E_v.
- (The case where the descendants of v may contain a power generator whose status is ON)
  Since we can freely choose the status (ON/OFF) of the descendants of v, we will calculate the sum of dp's of the descendants of the children of v. But, if there is a power generator at v, it will be broken.

Summarizing these two cases, we have the following recurrence formula:

$$dp_v = \max\left\{ E_v, \left(\sum_{c\,:\,\text{a child of } v} dp_c \right) - E_v \right\}.$$

We shall calculate the values of dp_v by the depth-first search (DFS). Note that, at the root vertex, even if one child or the descendants of one child contain a power generator whose status is ON, the power generator at the root vertex will not be broken.

By the above algorithm, when we fix a power generator whose status will be ON, we can calculate the maximum of the profit in $O(N)$ time. Therefore, considering all choices of a fixed power generator, the original problem will be solved in $O(N^2)$ time.

Subtask 3 (Full score solution)

In the algorithm of Solution 2, we consider the rooted tree whose root is the fixed power generator whose status will be ON. We can calculate the value of dp at a vertex since we can decide whether the power generator at the vertex is broken or not using the information of its descendants only. In the following full score solution, we will also calculate the values of dp by considering a rooted tree. For each vertex, assuming there is a power generator whose status is ON outside the subtree consisting of its descendants, we will calculate the maximum of the profit when we choose the status (ON/OFF) of the subtree.

In order to calculate the maximum value completely, we consider the minimum subtree which contains all the power generators which are finally activated or broken. For each vertex v, we shall calculate the maximum value for the minimum subtree where the vertex closest to the root is v. If we can calculate this value for every vertex, we can solve the task.

Let us consider the calculate for the vertex v.

- (The case where there is a power generator at the vertex v, and it will be activated)
  In this case, we can freely choose the status (ON/OFF) of a vertex in the subtree of one child only. The status of every vertex in the subtrees of other children should be OFF. Therefore, we shall calculate

$$\max \{ dp_c \mid c : \text{ a child of } v \} + 1.$$

- (The case where there is a power generator at the vertex v, and it may be broken)
  In this case, we can freely choose the status (ON/OFF) of a vertex of the descendants. We will calculate the sum of dp of the children of v. Assuming the power generator at v will be broken, we calculate

$$\left(\sum_{c : \text{a child of } v} dp_c \right) - 1.$$

- (The case where there is no power generator at the vertex v)
  Similarly, since we can freely choose the status (ON/OFF) of a vertex of the descendants, we calculate

$$\sum_{c : \text{a child of } v} dp_c.$$

Summarizing, for each vertex v,

- if $E_v = 1$, we calculate

$$\max \left\{ \max \{ dp_c \mid c : \text{ a child of } v \} + 1, \quad \left(\sum_{c : \text{a child of } v} dp_c \right) - 1 \right\}, \text{ and}$$

- if $E_v = 0$, we calculate

$$\sum_{c : \text{a child of } v} dp_c.$$

Then the maximum of these values will be the answer.

By the above algorithm, the answer can be calculated in $O(N)$ time.