Let $f[x][i][j]$ denote the number of subtrees rooted at x with height i and diameter j (a single node has height 0). For simplicity consider the binary tree case, where a node z has two children x and y, and we want to compute $f[z][\cdot][\cdot]$ from $f[x][\cdot][\cdot]$ and $f[y][\cdot][\cdot]$. Easy to see $f[x][i_1][j_1] \cdot f[y][i_2][j_2]$ will transit to $f[z][\max(i_1,i_2)+1][\max(j_1,j_2,i_1+i_2+2)]$ (there are three cases: the longest path within the subtree rooted at z is either contained in the subtree rooted at x, contained in the subtree rooted at y, or crosses z).

To efficiently compute the transition, we perform a case analysis:
1. $\max(i_1,i_2) = i_2$ and $\max(j_1,j_2,i_1+i_2+2) = j_2$. For a fixed $f[z][i_2][j_2]$, we need to compute

$$\sum_{i_1,j_1:i_1\leq i_2, j_1\leq j_2, i_1\leq j_2-i_2-2} f[x][i_1][j_1],$$

which can be computed in $O(1)$ time after precomputing the prefix sums. Then we take a product with $f[y][i_2][j_2]$.
2. $\max(i_1,i_2) = i_2$ and $\max(j_1,j_2,i_1+i_2+2) = j_1$. For a fixed $f[z][i_2][j_1]$, we need to compute

$$\sum_{i_1:i_1\leq i_2, i_1\leq j_1-i_2-2} f[x][i_1][j_1] \quad\text{and}\quad \sum_{j_2:j_2\leq j_1} f[y][i_2][j_2],$$

which can be computed in $O(1)$ time after precomputing the prefix sums. Then we take the product.
3. $\max(i_1,i_2) = i_2$ and $\max(j_1,j_2,i_1+i_2+2) = i_1+i_2+2 = j$. For a fixed $f[z][i_2][j]$, we know $i_1 = j-i_2-2$, and we need to compute

$$\sum_{j_1:j_1\leq j} f[x][i_1][j_1] \quad\text{and}\quad \sum_{j_2:j_2\leq j} f[y][i_2][j_2],$$

which can be computed in $O(1)$ time.
The other cases are symmetric. The total running time is $O(n^3)$.

To further improve the running time, let n_1 denote the size of the whole subtree rooted at x and n_2 denote the size of the whole subtree rooted at y. Notice that when $n_1 \ll n_2$, we only need to recompute $O(n_1 n_2)$ entries in $f[z][\cdot][\cdot]$ (namely, for each $i \geq n_1$, we only need to recompute $f[z][i][j]$ s.t. $i \leq j \leq i+n_1$), in $O(1)$ time per entry. Solving the recurrence $T(n) \leq \max_{n_1,n_2:n_1+n_2=n}(T(n_1) + T(n_2) + O(n_1 n_2))$ shows that the total running time is $T(n) = O(n^2)$.

References