The solution combines a dynamic programming approach with the Bellman-Ford algorithm to compute the minimum time cost of supply paths from a source station s to each bathhouse station $b_i \in B$ in the aqueduct grid and is based off of the following formula:

$$OPT(s, B) = min_{b_i \in B}\{|p_1| + |p_2|\}$$

s: $source\ station$
B: $set\ of\ water - supplying\ stations$
p_1: $path\ from\ s\ to\ closest\ b_i \in B$
p_2: $path\ from\ b_i \in B\ closest\ to\ s\ to\ every\ other\ b_i \in B$

The input aqueduct grid is represented as a graph via a dictionary of stations and a set of edges. A Station object has attributes for its x and y position as well as its height. An Edge object has attributes for each station involved as well as its time cost, derived from the following formula:

$$time((x, y), (x', y')) = max\{-1, 1 + height(x', y') - height((x, y))\}$$
(x, y): $position\ of\ origin\ station$
(x', y'): $position\ of\ destination\ station$

The program design involves finding the shortest path from the source station s to each bathhouse station $b_i \in B$, considering all possible intermediate stations for each step of the path. This is implemented recursively by exploring paths from s to the first bathhouse station and then to each subsequent station. Within each recursive call, the Bellman-Ford algorithm is executed to find the shortest path from the current station to the next station. Memoization is used to store the shortest path distances between stations, avoiding redundant calculations and optimizing performance.

The time complexity of the solution is$O(B(B-1)VE)$, where V is the number of stations, E is the number of edges, and B is the number of water-supplying stations. There is a recursive call for each node to every other node ($O(B(B-1))$) and in each recursive call, the Bellman-Ford algorithm ($O(VE)$) is called once.