Jars Problem

solution description:

My approach is similar to the one we discussed at the office which is done by applying BFS implemented by a queue.
Visiting recurrent nodes is prevented by a map storing as a key the nodes already visited. As a key value I stored the father node and that would enable us to follow back the path to the root when the answer is found.

ways of enhancing the program's efficiency:

1) Using unordered_map instead of a map or a customized hash function that'd give an optimal solution.
2) Storing the reverse pour information that gets you back to the father node as a value to each key (we need to store 3 pieces of information to reverse the pour correctly and not just two).
3) Using better copy constructors.
4) Using map to determine recurrency right before pushing pour coordinations to the queue.