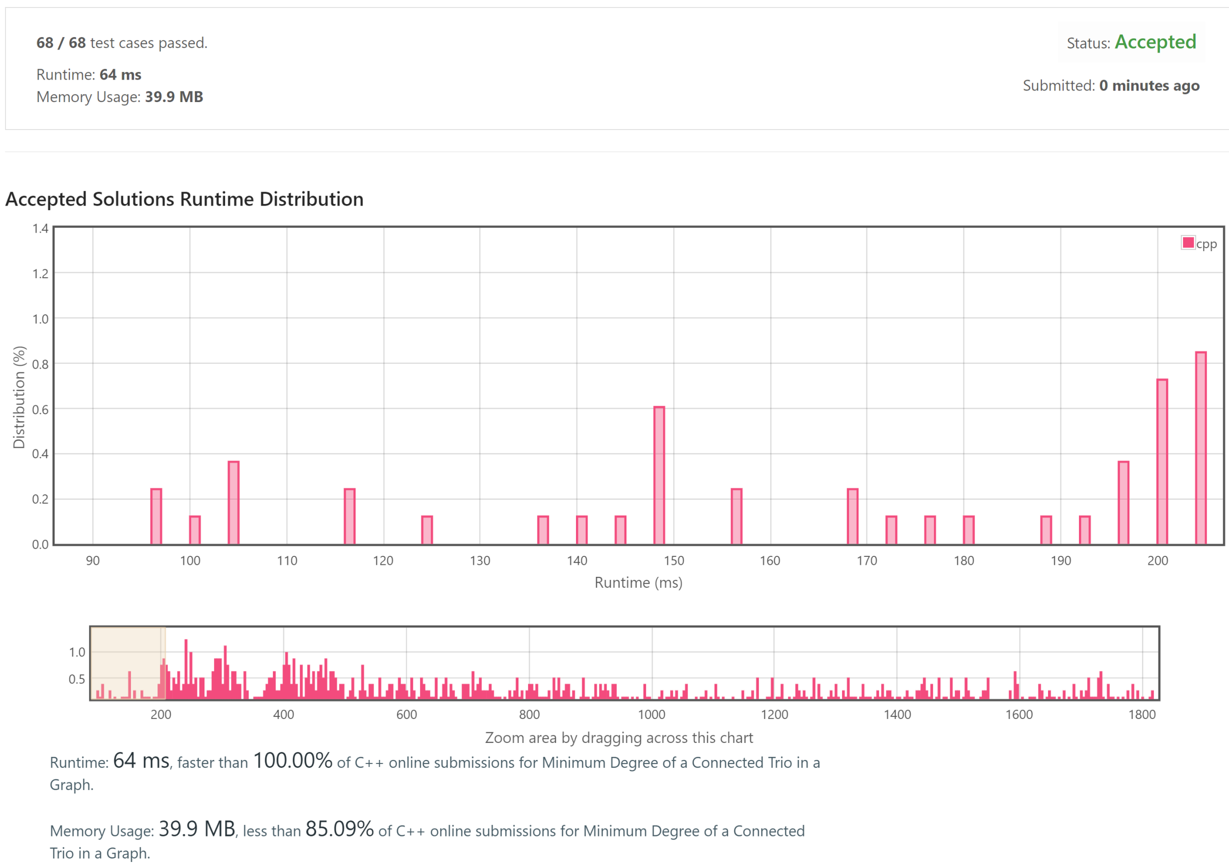
1. This is a special case of the min-weight triangle finding problem (on dense undirected graphs with node weights in $[n]$).
It is no harder than the unweighted triangle detection problem, which can be solved in $O(n^{\omega}) \approx O(n^{2.373})$ time [1]. (see [2]. By binary search we can reduce to node-weighted negative weight triangle detection.)
We can also solve in $O(n^{\frac{3+\omega}{2}}) \approx O(n^{2.69})$ time by dividing into blocks of size $O(n^{\frac{\omega-1}{2}})$ and using boolean matrix multiplication.
2. For each pair (i, j) of vertices, we can detect whether there exists k that together form a triangle with less weight than the current best solution, using bitset in $O(\frac{n}{w})$ time. The solution can change $O(n)$ times, and each time we use $O(n)$ time to find such k. $O(\frac{n^3}{w})$.



References

[1] Virginia Vassilevska and Ryan Williams. Finding, minimizing, and counting weighted subgraphs. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 455–464, 2009.

[2] Virginia Vassilevska Williams and Ryan Williams. Subcubic equivalences between path, matrix and triangle problems. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 645–654. IEEE, 2010.