Performance document:

std::vector<WayID> all_ways();

Estimate of performance: O(n)

Map keys are inserted to vector inside a for-loop, so time complexity is linear in the size of the container.

bool add_way(WayID id, std::vector<Coord> coords);

Estimate of performance: O(n)

Because of for loop depends on the amount of coords (n). Uses std::insert() that has average time complexity O(1) and worst case O(n) when using unordered_map.

std::vector<std::pair<WayID, Coord>> ways_from(Coord xy);

Estimate of performance: O(n)

Map keys are stored in a vector inside a for-loop, so time complexity is linear in the size of the container.

std::vector<Coord> get_way_coords(WayID id);

Estimate of performance: Average for unordered_map O(1), worst case O(n)

std::find() has the above time complexity with unordered_map

void clear_ways();

Estimate of performance: O(n)

std::clear() time complexity is linear in the size of the container. Used twice to clear both ways_ and crossroads_.

std::vector<std::tuple<Coord, WayID, Distance>> route_any(Coord fromxy, Coord toxy);

Estimate of performance: O(V+E) (DFS-algorithm)

Using DFS-algorithm because it is more efficient when the objective is to find any route. V is the number of vertices and E is the number of edges in the graph.

Maximum loop amount for While-loop is O(V). Maximum loop amount for For-loop is O(E). So the time complexity for the whole algorithm is O(V+E). Function also uses std::find that has average time complexity O(1) and worst case O(n) when using unordered_map. Crossroads are reset in a for loop with time complexity O(n) where n is the size of crossroads_-map.

bool remove_way(WayID id);

Estimate of performance: O(n)

Because of for loop depends on the amount of connections (n)

crossroad has. Function also uses std::find that has average time complexity O(1) and worst case O(n) when using unordered_map. Function also uses std::erase that has average time complexity O(1) and worst case O(n) when using unordered_map. Function uses std::at with average time complexity O(1), worst case O(n) for unordered_map.