Во время измерения производительности было решено разбить время исполнения целого алгоритма еще на измерение скорости предварительного построения и получения результатов по запросам. Как мы можем увидеть из 2-го графика, самое быстрое построение происходило у наивного решения просто потому, что никаких предварительных действий там нет. В двух других случаях мы имеем O(n^3) и O(n\*logn) сложности для построения карты и дерева. Безусловно, алгоритм на карте имеет наихудшую кубическую асимптотику и работает очень медленно даже при небольшом количестве данных. Было принято решение не рассматривать его дальше и не брать в расчет в целях экономии времени, затрачиваемого на выполнение программы, но очевидно, что работать он будет долго.

Что касается последнего графика, то тут уже интереснее. Абсолютным лидером является алгоритм на карте, поскольку имеет очень маленькие константы, и фактически представляет собой эталонную логарифмическую сложность на запрос. (Уточнение: графики показаны не для единичного запроса, а для множества точек = количеству прямоугольников) Как раз из-за таких констант, наивный алгоритм лидировал по скорости изначально, поскольку входные данные там сравнительно малы. Однако довольно быстро уступил алгоритму на карте и алгоритму на дереве. К сожалению, алгоритм на дереве не может отразить идеальную логарифмическую сложность из-за большого количества констант и некоторой дополнительной работы, однако на порядок превосходит скорость работы наивного алгоритма, к чему мы и стремились при его написании.

Наконец, рассматривая полную скорость выполнения, учитывая и построение и запросы, можно сказать, что алгоритм на карте имеет худшую асимптотику. Наивный алгоритм сохранял лидерство в скорости выполнения чуть дольше(до n=1000 во входных данных) за счет того, что в алгоритму на карте прибавилось время его построения, однако, на больших данных наивный алгоритм снова уступил в скорости, что делает алгоритм на дереве лучшим в целом. Алгоритм на карте был бы идеален, если бы предварительная подготовка не входила бы в его работу. И, наконец, наивный алгоритм является средним между ними.