Assignment 9 – Analysis

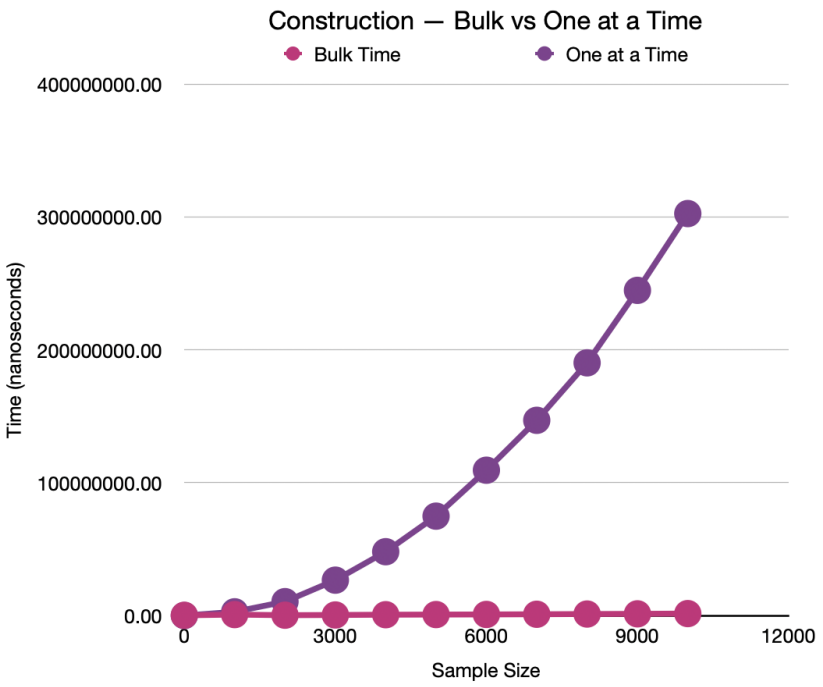
CS 6012

Lindsay Haslam

Bulk vs One at a Time

Construction

1	182097.62	195.84
1001	721025.4	2814412.54
2001	264629.56	1.046594834E+07
3001	428244.16	2.66538204E+07
4001	631520.04	4.807066042E+07
5001	735087.47	7.48538404E+07
6001	829059.15	1.094437858E+08
7001	971711.66	1.470220629E+08
8001	1143526.29	1.9029921876E+08
9001	1290526.64	2.4488341083E+08
10001	1477443.74	3.0270216628E+08

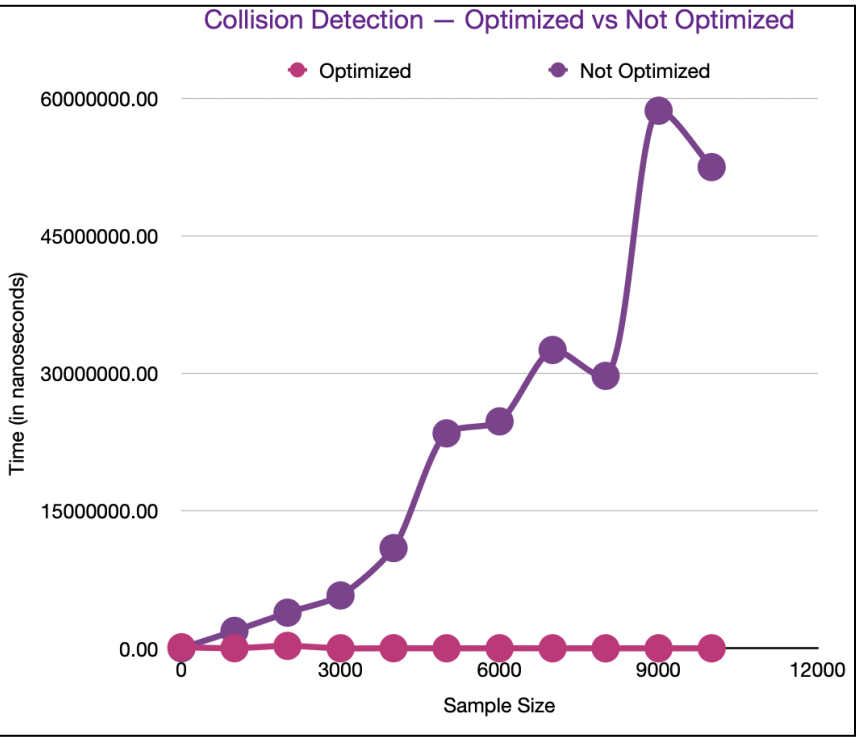


For constructing a tree, the “bulk” approach resembles an O(log n) runtime (even though it looks like it is O(1) on the graph. The Big O for “one at a time” is O(N^2), which makes sense, as it is the worst case runtime multiplied by the number of elements. The “worst case” order to insert them would be in an unbalanced order, like if it was sorted by the x coordinate or if you inserted elements in a way that would create a chain. I expected “one at a time” to match the time complexity it’s showing, and I expected “bulk” to be at O(log n) because Professor Jones said something along those lines during our lecture.

Collision Detection

CollisionDetection

1	122071.65	8347.55
1001	19124.61	1906952.43
2001	271996.31	3892262.04
3001	7513.75	5772593.74
4001	13827.5	1.094121542E+07
5001	5530.45	2.345038331E+07
6001	11728.34	2.476332127E+07
7001	8061.73	3.254829581E+07
8001	3463.68	2.972512955E+07
9001	8694.97	5.864749284E+07
10001	6937.51	5.249836541E+07



I was expecting the optimized collision detection to run at O(log n) like it does in the graph. For the optimized method, I checked for collisions between a query segment and the segments inside the BSP tree, then measured the time it took for collision detection. For the non-optimized method (provided by the professor), we defined a new collision detection method using the traverseFarToNear method and counted the collisions. For each experiment size, I generated a list of random segments between 0 and 1, then created a BSP tree from those segments. I then created a query segment with fixed coordinates, and timed how long it took to detect a collision for the optimized and non-optimized method.