Joseph Barbati
CS323 - Fall 2016

A comparison between performance of Binary Heap and Ternary Heap as Priority Queue

	Binary Heap		Ternary Heap	
N	Add (ms)	Remove (ms)	Add (ms)	Remove (ms)
100	6.48	12.81	6.1	14.38
200	21.13	44.75	18.45	46.51
300	38.44	89.29	34.7	93.77
400	62.1	151.32	57.28	160.71
500	91.57	232.32	84.24	245.21
600	126.19	327.97	116.50	348.14
700	167.09	442.45	155.43	474.43
800	213.97	578.91	198.07	618.13
900	264.57	733.95	246.32	783.13
1000	322.46	907.32	299.81	970.04

Figure 1. Average time it took for each heap to add and remove N items in an array. Numbers are based on 200 tests.

Time differences between Binary Heap and Ternary Heap		
N	Addition	Removal
100	-5.9601%	12.282%
200	-12.65%	3.925%
300	-9.729%	5.021%
400	-7.825%	6.203%
500	-8.005%	5.55%
600	-7.676%	6.15%
700	-6.982%	7.229%
800	-7.43	6.774%
900	-6.898%	6.701%
1000	-7.023%	6.913%

Figure 2. % difference in addition and removal from respective heaps. Negative numbers mean ternary heap was faster by the absolute value of the %, positive numbers mean the binary heap was faster by the absolute value of the %.

Adding into Priority Queue

According to Figure 2, ternary heap was quicker at adding numbers into the priority queue. This makes perfect sense, because the swim method makes one comparison between parent and child in each level of the heap. Since the ternary heap has 3 children, this means it would have fewer levels (or less depth) than a binary heap of the same size. Specifically, for N items, a binary heap would have $\log_2(N)$ levels and the ternary heap would have $\log_3(N)$ levels.

Removing from Priority Queue

Ternary heap added items more quickly than the binary heap, but the binary heap tends to remove items more quickly. The reason the binary heap is slightly faster is because of the sink method. The sink method checks all children, finds the one with the greatest value and swaps that with the parent. In a binary heap, the method only has to check two children, while the ternary heap has to check three children. This fact is causing the remove method to be slower in the ternary heap priority queue when compared to the binary heap priority queue. It can be safely assumed that the binary and ternary heaps are checking two and three children, respectively, based on the fact that in order to be called a heap, the tree must be perfectly balanced.