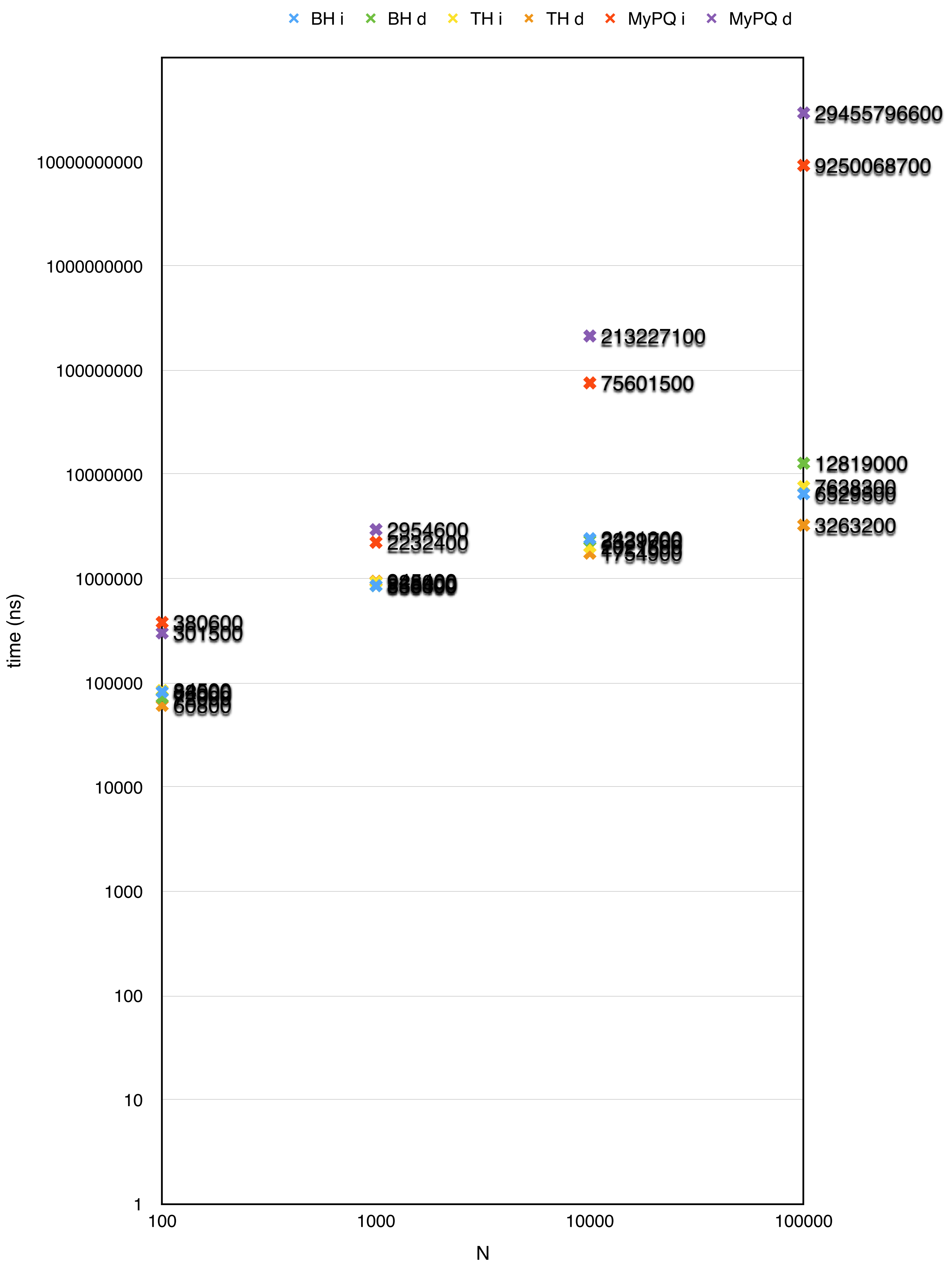
Timothy Ha
1367917
junkwan
CSE 373
04.29.15
HW 3 README

1. Worst case asymptotic running times

	isEmpty	size	insert	findMin	deleteMin
BinaryHeap	O(1)	O(1)	O(log(n))	O(1)	O(log(n))
ThreeHeap	O(1)	O(1)	O(log(n))	O(1)	O(log(n))
MyPQ	O(1)	O(1)	O(nlog(n))	O(1)	O(nlog(n))

2. Timing Code

nanoseconds	N = 100	N = 1000	N = 10000	N = 100000
BH insert	82000	856400	2421000	6529500
BH deleteMin	72000	855500	2339700	12819000
TH insert	84500	928000	2021600	7638300
TH deleteMin	60800	945400	1754900	3263200
MyPQ insert	380600	2232400	75601500	9250068700
MyPQ deleteMin	301500	2954600	213227100	29455796600



BH i
BH d
TH i
TH d
MyPQ i
MyPQ d
time (ns)
N
10000000000
1000000000
100000000
10000000
1000000
100000
10000
1000
100
10
1
100
1000
10000
100000
29455796600
9250068700
213227100
75601500
12819000
7628300
3263200
2954600
2232400
380600
301500

3. Comparison

I don't think the asymptotic analysis is very useful for me to see the time complexities of insert and deleteMin for my three implementations of the priority queue. I couldn't find out the analysis for MyPQ, the linkedlist implementation but I knew that compared to the BinaryHeap and TernaryHeap, MyPQ would be way more inefficient because it has to search through several parts of the list. Both Binary and Ternary heap acted logarithmically as I expected, although I'm sure Ternary heap isn't exactly O(logN) in the worst case, it will be less efficient. What I am surprised with is that deleteMin for Binary Heap seems to be a lot more taxing than insert, but both are supposed to be O(logN) time complexities. I'm not sure what could have caused the error, whether it be my computer or my code.

I definitely would not recommend MyPQ implementation for a priority queue because of how inefficient it is. I would recommend the TernaryHeap if the user wanted more "items" per level, it allows for pretty quick insert, but at the same time it will take longer to read in general. The BinaryHeap is very quick and suffices for most other applications if the user wants more access to what the insert.

4. Testing

I created the two required Binary and Ternary heaps and then my own version of a binary heap, a linked node list heap. In order to test for their functionality, I made sure first that their inserts worked correctly, by adding println statements within the code and adding toString methods so I could print the heaps out as a list on the client side. I checked what the console output was against my handwritten calculations for what the heaps should look like and compared and fixed anything that needed fixing. Everything was pretty simple to test except insert and deleteMin which both required a little bit more, as they are the more code heavy parts.

For deleteMin, I printed out each time it ran the method and ran it until there were no nodes left. This checked end points (having 0 nodes) as well as making sure that every time deletedMin was called, the heap was rearranged correctly according to my handwritten notes and calculations. For both insert and deleteMin, I was also able to test the BinaryHeap and MyPQ against my handwritten notes since they should give the same output, given the same input.

5. Heap Children

	Children at i
Binary Heap	i * 2, i * 2 + 1
Three Heap	i * 3 - 1, i * 3, i * 3 + 1
Four Heap	i * 4 - 2, i * 4 - 1, i * 4, i * 4 + 1
Five Heap	i * 5 - 3, i * 5 - 2, i * 5 - 1, i * 5, i * 5 + 1
d heap	LEFTMOST CHILD: i * d - (d - 2)