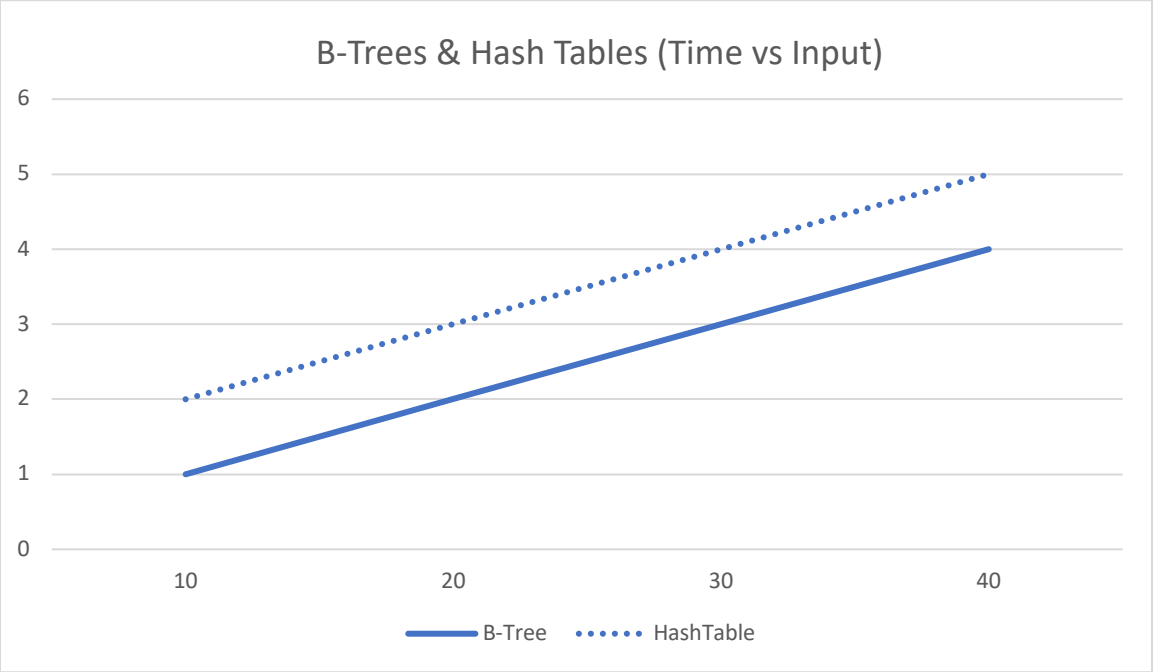
B-Trees vs. HashTables

Construct input files for B-Trees and HashTables to demonstrate that a B-Tree is asymptotically better than a HashTable. To do this, we expect you to show trends. You might consider fitting a curve to your results. Explain your intuition on why your results are what they are.

Bad Answer #1:



As you can see here, the B-Tree clearly has better run time at many different inputs. The graphs show that both run times increase linearly, but the B-Tree always has a slightly lower run time, so it's definitely the better data structure. All of our inputs are in the experiments file. We used different types of values: integers, strings and objects containing (x,y) coordinate pairs. We thought it was very interesting that both run times looked pretty linearly, we would have expected more differences, but data is data! For practical use, we don't see that there would be much of a difference in using either one of these since HashTable's run time is only slightly worse than B-Tree's.

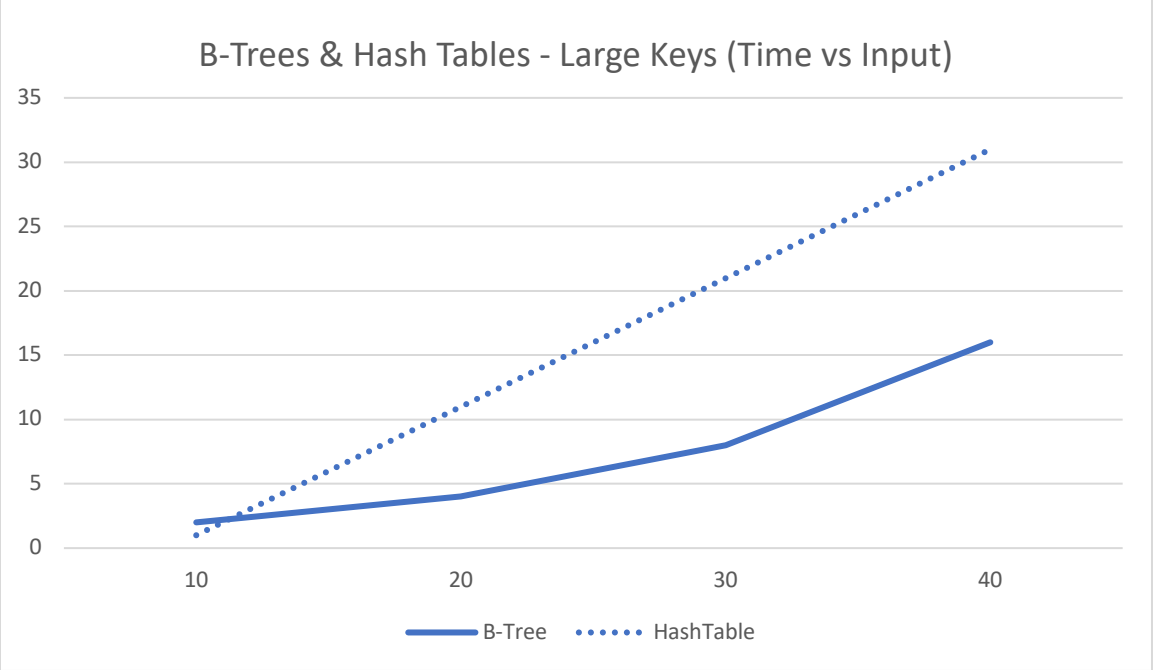
What's wrong:

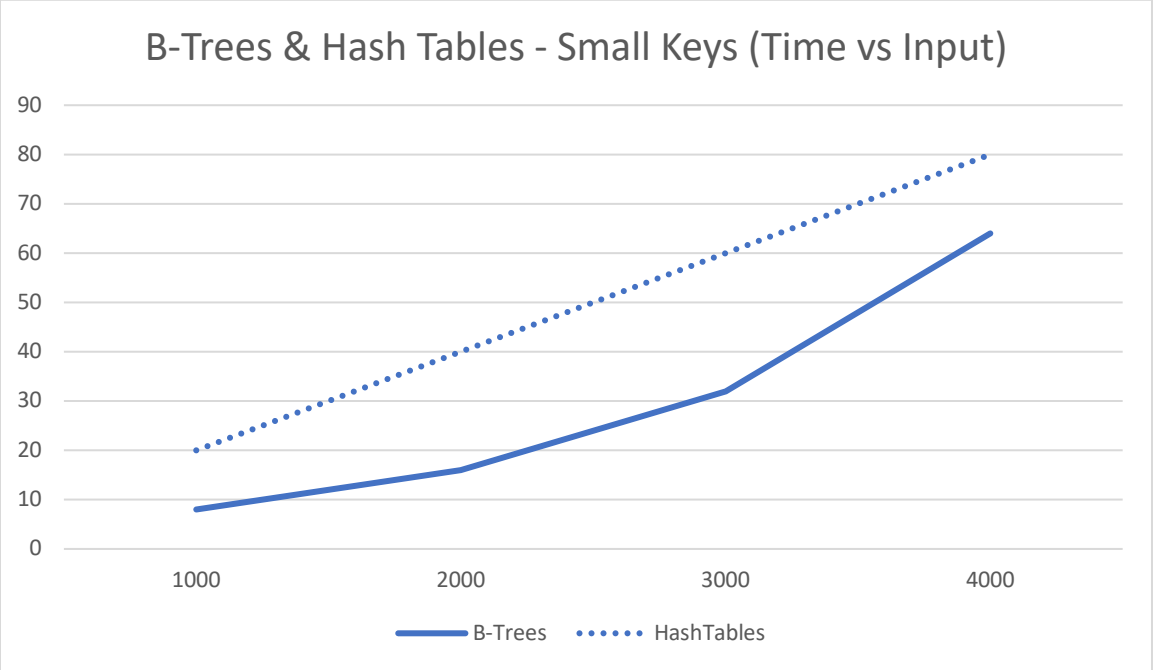
1. The question asks the student to show that one is asymptotically better than the other, but the data collected shows linear graphs, meaning these would both be O(n). Likely, bad data sets were chosen that did not demonstrate the differences between the data structures. For this example, you'd need to use very large keys/values and/or very many of them to make the B-Tree advantageous (the whole point is bringing in whole pages from disk, inserting 10 ints takes up very few bytes and won't require any paging to disk). They should have used data that would have allowed the B-Tree to be properly utilized.
2. There are no deep conclusions drawn, they just summarize the data that's already shown in the graph. The closest thing to a conclusion is the last sentence, but it's a very superficial evaluation of the graph above and doesn't show any deeper thought or consideration. Also, it goes against

the thing they were asked to show, which implies they probably gathered faulty data or used faulty test cases (see 1 above).

3. They say they used many types of input, but they don't explain how these various types of input relate to the graph (is this an average of all inputs? Or is it just one of the inputs? What are they showing here?)
4. Graph has no labels and no units.
5. The graph does not clearly show the data points the trendlines is approximating. This is an issue because then we have no idea how many data points they gathered and also is generally wrong because their data is not continuous. Trend lines are obviously fine, but these graphs don't give an honest representation of the data.

Bad Answer #2:





Our B-Tree did have better run time, as shown in the graph above. We think this is due to the fact that the B-Tree is paging to disk efficiently, but the HashTable is having to page to disk far more often. We tried two different kinds of input, one with very large keys with a lower M and L, and another with smaller keys and a larger M and L. We found that in both cases, the B-Tree performed better. This surprised us as HashTables usually work quite well with small values, but clearly if there are enough of them, having to page to disk multiple times really does make a difference. We did try with small numbers of small sized input as well, and as expected the HashTable performed better (we did not

include this graph, we just did it out of curiosity). This made sense to us as the whole advantage of a B-Tree is more efficient disk look-ups, so the overhead in the B-Tree makes it perform worse than a HashTable with small inputs. Up to this point we had wondered why we don't hear more about B-Trees as they seem like such a better data structure to use, but now we see that there are only specific cases where it is really advantageous to use a B-Tree. (We know you said this in class, but now we've seen it in the data!) We think this sort of data structure would be good for things like large file systems on servers, or perhaps an application like GoogleDocs where a company is storing a lot of files within a lot of nested directories.

What's wrong:

1. It's subtle, the conclusions and discussion are good, BUT the data they gathered totally conflicts with all of their conclusions! The B-Tree graph looks like it's going to be O(n^2) but the HashTables graph looks O(n). This makes it seem as if they just ran experiments to produce data and then just wrote what they thought the grader wanted to hear rather than actually using the data to draw conclusions. (keep in mind this isn't a real question and I'm just making stuff up, so don't take any of this as actual relations between B-Trees and HashTables!). The conclusion should have a direct reference to data in the graphs and tables to support their statements.
2. The graph still does not clearly show the data points the trendlines is approximating. This is an issue because then we have no idea how many data points they gathered and also is generally wrong because their data is not continuous. Trend lines are obviously fine, but these graphs don't give an honest representation of the data.
3. The graph still needs to include units and axis labels.